

reference manual

**graphics  
terminal**  
**2648A**



---

HEWLETT-PACKARD COMPANY  
19400 HOMESTEAD ROAD, CUPERTINO, CALIFORNIA, 95014

### **NOTICE**

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

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

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

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another program language without the prior written consent of Hewlett-Packard Company.

This manual provides detailed programming and accessory installation information for the HP 2648A Graphics Terminal. It is written to provide a system programmer with the information needed to use the terminal in a variety of applications. Extensive information explaining the operation of the terminal's data communication function is provided in a separate section.

This manual assumes that you are already familiar with operating the terminal from the keyboard. Operating information is given in the *HP 2648A Graphics Terminal User's Manual* (02648-90001). The *HP 2648A Service Manual* (02648-90003) provides a discussion of troubleshooting, repair, and theory of operation.

## HOW TO USE THIS MANUAL

This manual describes all of the terminal's programmable features. The various functional groups such as display control and communications are described in separate sections. If you have not used an HP terminal before, you should read Section I for a brief overview of the terminal and its capabilities. Information on the terminal's graphics functions is contained in Section III. If you are familiar with the HP 2645 series terminals you can use the index at the back of the manual to locate answers to specific questions.

This manual is made up of the following sections and appendices:

Section I. *General Description* — This section provides a brief description of the terminal, its architecture, and overall operation.

Section II. *Display Memory and Terminal Control Functions* — This section contains information for controlling the terminal's alphanumeric display. Included are cursor sensing and positioning, fields, edit operations, and display enhancements. This section also contains information for programmatically controlling the terminal's switch settings.

Section III. *Terminal Graphics Functions* — This section contains information for controlling the terminal's graphics modes and functions.

Section IV. *Device Control* — This section describes how to control operational input/output devices (cartridge tape drives, printers, etc.)

Section V. *Data Communications* — This section describes the terminal's communication function and gives procedures for configuring the terminal to meet various communication requirements.

Section VI. *Status* — This section describes how to obtain and interpret terminal status.

Section VII. *Installation* — This section contains step-by-step procedures for installing and configuring the terminal and its accessories.

Section VIII. *HP-IB Configurations* — This section contains examples of interfacing printers and plotters via the Hewlett-Packard Interface Bus (HP-IB).

Appendix A. *Applications* — This appendix contains examples of various terminal applications.

Appendix B. *Reference Tables* — This appendix contains condensed reference information for all of the terminal's features.

Appendix C. *Communications Flowcharts* — This appendix contains flowcharts of the communication function.




Appendix D. *Cartridge Tape Rethreading Procedure* — This appendix contains a procedure for rethreading cartridge tapes.

## TERMS AND CONVENTIONS






The descriptions in this manual use the following text conventions:

[< >] — The right and left bracket, less than, and greater than characters are used to set off variable parameters in some escape code sequences. These characters are added for descriptive purposes and are not a part of the escape sequence. They should not be transmitted.

Example:  $\text{f} \& \text{p} [\text{<“from” device code>}]$

<character><sup>c</sup> or  <character> — When a character is shown followed by a superscript c or is shown preceded by the  key, it indicates a control character. Control characters are normally generated from the keyboard by holding the  key down while pressing the character.

Example:  $G^c = \text{bell character} = \text{CTRL} \text{ G}$ .

Graphics control functions labeled on the sides of the graphics control keys are accessed by holding the  key down while pressing the function key. Normally the symbols for the shifted graphics keys are shown as trapezoids. For example, to make a change in the graphics text size from the keyboard, hold the  key down and press . This would be shown as:  .

# CONTENTS

Section I	Page	Soft Key Applications .....	2-14
<b>GENERAL DESCRIPTION</b>		Using Soft Key Labels .....	2-14
Introduction .....	1-1	Additional Control Functions .....	2-15
Mainframe .....	1-1	Bell .....	2-15
Microprocessor .....	1-1	Send Display .....	2-15
Terminal Memory .....	1-3	Wait .....	2-15
Input/Output Modules .....	1-3	Keyboard Disable/Enable .....	2-15
CRT Monitor .....	1-3	Reset Terminal .....	2-15
The Raster .....	1-3	Monitor Mode .....	2-16
Alphanumeric and Microvector Character Sets .....	1-4	Self-Test .....	2-16
Display Features and Alternate		Modem Disconnect .....	2-16
Character Sets .....	1-4	Program Down Load .....	2-16
Keyboard .....	1-4		
The Firmware .....	1-4	Section III	Page
System Monitor .....	1-4	<b>GRAPHICS CONTROL FUNCTIONS</b>	
Keyboard and I/O Subsystem .....	1-5	Introduction .....	3-1
Cursor Movement .....	1-5	Graphics Display .....	3-1
Display Memory Management .....	1-5	Keyboard Graphics Functions .....	3-1
Data Communications .....	1-5	Programmable Graphics Functions .....	3-1
The Tape Units .....	1-6	Control Codes .....	3-3
		Commands .....	3-3
		Parameters .....	3-3
Section II	Page	Graphics Display Control .....	3-4
<b>DISPLAY MEMORY FUNCTIONS</b>		Graphics Cursor Control .....	3-4
Introduction .....	2-1	Zoom .....	3-5
Display Memory Functions .....	2-1	Graphics Memory Control .....	3-5
Display Control .....	2-1	Inserting Delays In Graphics Operations .....	3-6
Memory Addressing Scheme .....	2-1	Graphics Drawing Mode Parameters .....	3-6
Cursor Sensing .....	2-2	Drawing Modes .....	3-6
Cursor Positioning .....	2-2	Drawing Patterns .....	3-8
Other Cursor Operations .....	2-3	Area Fill .....	3-12
Tabs .....	2-3	Relocatable Origin .....	3-13
Margins .....	2-3	Rubber Band Line .....	3-14
Edit Operations .....	2-5	Selecting the Graphics Default Parameters .....	3-14
Insert Character With Wraparound .....	2-5	Plotting Sequences .....	3-14
Delete Character With Wraparound .....	2-5	Pen Control .....	3-15
Moving Text Blocks .....	2-6	Vectors .....	3-15
Forms Mode (Format Mode) .....	2-6	ASCII Formats .....	3-15
Protected Fields .....	2-6	Binary Formats .....	3-16
Unprotected Fields .....	2-6	Recording Graphics Functions .....	3-19
Transmit Only Fields .....	2-7	Graphics Hardcopy Operations .....	3-19
Data Checking .....	2-7	Raster Data Transfers .....	3-19A
Tabbing .....	2-7	Raster Output Format .....	3-19A
Editing .....	2-7	Raster Dump Commands .....	3-19B
Building the Form .....	2-8	Speed .....	3-19B
Sending Data to the Computer .....	2-8	Tape Used .....	3-19B
Display Enhancements .....	2-8	Graphics Dump .....	3-19B
Alternate Character Sets .....	2-8	Graphics Restore .....	3-19B
Selecting Alternate Sets .....	2-8	Graphics Text .....	3-20
Using Alternate Sets .....	2-8	Program Control of Graphics Text .....	3-21
Terminal Control Functions .....	2-11	Autoplot .....	3-24
Latching Keys .....	2-11	Autoplot Menu .....	3-24
Keyboard Interface Switches .....	2-12	Local (User) Control .....	3-25
Programmable Soft Keys .....	2-12	Program Control .....	3-25
Controlling the Soft Key Display .....	2-12	Drawing the Axes .....	3-26
Defining Soft Keys .....	2-13	Plotting the Data .....	3-26
Triggering Soft Keys .....	2-14		

# CONTENTS (continued)

Using Autoplot With Other Graphics Functions .	3-31
Autoplot Errors .....	3-31
Compatibility Mode .....	3-32
Compatibility Mode Straps .....	3-33
Graphics Data .....	3-33
Graphics Data Format .....	3-35
Text .....	3-36
Cartridge Tape Operation In Compatibility Mode	3-36

## Section IV

### DEVICE CONTROL

Introduction .....	4-1
Device Control Completion Codes .....	4-2
Selecting "from" and "to" Devices .....	4-2
Cartridge Tape Control Operations .....	4-3
Rewind (0c) .....	4-3
Space "p" Records (1c) .....	4-3
Space "p" Files (2c) .....	4-4
Locate File "p" (2c) .....	4-4
Locate End-of-Data (3c) .....	4-4
Condition Tape (4c) .....	4-4
Write File Mark (5c) .....	4-4
Write End-of-Data Mark (6c) .....	4-4
Text CTU (7c) .....	4-4
Space "p" Records Immediately Without	
Writing End-of-Data Mark (8c) .....	4-5
Turn On Write-Backspace-Read Mode (9c) .....	4-5
Turn Off Write-Backspace-Read Mode (10c) .....	4-5
Device-to-Device Operations .....	4-5
Copy Record (b) .....	4-5
Copy File (f) .....	4-5
Copy All (m) .....	4-6
Compare Record (1b) .....	4-6
Compare File (1f) .....	4-7
Compare All (1m) .....	4-7
External Printer Control Operations .....	4-8
Device-to-Data Comm Data Transfers .....	4-8
ASCII (7-bit) Read Operations .....	4-9
Binary (8-bit) Read Operations .....	4-11
Fast Binary (Program Load) Read Operations .....	4-12
Data Comm-to-Device Data Transfers .....	4-12
ASCII (7-bit) Write Operations .....	4-12
Binary (8-bit) Write Operations .....	4-13
Graphics Hard Copy Operations .....	4-15
Graphics Transfer Operations .....	4-15
HP-IB Operations .....	4-15
Addressing .....	4-15
Transferring Data from Computer to	
HP-IB Device .....	4-16
Transferring Data from HP-IB Device	
to Computer .....	4-16
Transferring Data Between HP-IB Device	
and Terminal Devices .....	4-16
Timeout .....	4-16
Status .....	4-16
Initialization .....	4-16

## Section V

### DATA COMMUNICATIONS

Introduction .....	5-1
Connecting Terminals to a Computer .....	5-1
Networks .....	5-1
Interfaces .....	5-1
Interface Signals .....	5-3
Modems .....	5-5
Communication Protocols .....	5-6
Character Protocols .....	5-6
Block Protocols .....	5-6
Character Protocols .....	5-6
Operating at High Speeds .....	5-6
Character Mode .....	5-7
Multicharacter Transfers .....	5-7
Block Mode .....	5-9
Full Duplex Operation .....	5-9
Teletype Compatible Communications .....	5-9
Half Duplex Operation(202 Modem	
Compatibility) .....	5-9
Monitor Mode .....	5-14
Configuration .....	5-14
Block Protocols with 13260C or 13260D	
Communications Accessory .....	5-23
Character Mode Transfers .....	5-23
Multi Character Transfers .....	5-23
Message Blocks .....	5-23
Block Operation .....	5-23
Block Protocol Control Sequences .....	5-27
Multipoint Communications .....	5-34
Control Sequences .....	5-36
Configuration Procedure .....	5-38
Monitor Mode .....	5-44
Driver Mode .....	5-46

## Section VI

### STATUS

Introduction .....	6-1
Interpreting Status .....	6-1
Terminal Status .....	6-1
Primary Terminal Status .....	6-2
Secondary Terminal Status .....	6-4
Graphics Status .....	6-6
Read Device I.D. .....	6-7
Read Current Pen Position .....	6-7
Read Graphics Cursor Position .....	6-7
Read Cursor Position with Wait .....	6-7
Read Display Size .....	6-7
Read Device Capabilities .....	6-8
Read Graphics Text Status .....	6-8
Read Zoom Status .....	6-8
Read Relocatable Origin .....	6-8
Read Reset Status .....	6-9
Read Area Shading Capability .....	6-9
Read Graphic Modification Capabilities .....	6-9
Any Other Parameters .....	6-9
Device Status .....	6-6

# CONTENTS (continued)

Section VII	Page
<b>INSTALLATION</b>	
Introduction .....	7-1
Opening the Terminal .....	7-2
Grounding Requirements .....	7-5
Selecting Line Voltage .....	7-5
Accessory Installation Procedures .....	7-6
HP 13231A Display Enhancements .....	7-6
HP 13232 Cable Assemblies .....	7-8
HP 13234A (4K) Terminal Memory Module .....	7-10
HP 13236B Cartridge Tape Unit .....	7-12
HP 13238A Terminal Duplex Register .....	7-12
HP 13245A Character Set Generation Kit .....	7-12
HP 13246A/B Printer Subsystem (9866) .....	7-13
HP 13250B Serial Printer Interface .....	7-13
HP 13254A Video Interface .....	7-13
HP 13260A,B,C,D Data Communications Accessories .....	7-13
HP 13261A-003 Device Support Firmware .....	7-15
HP 13296A Shared Peripheral Interface .....	7-16B
HP 13349A Printer Subsystem (9871) .....	7-17
Power Supply Adjustment .....	7-17
Selecting Optional Operating Functions .....	7-17
Data Communications Cabling .....	7-26
Interface Signals .....	7-26
Logic Levels .....	7-26
Cable Types .....	7-26
Point-to-Point Communications Cabling .....	7-29
Multipoint Communications Cabling .....	7-29
Power Down Protect Cabling .....	7-29
Fabricating Your Own Data Communica- tions Cable .....	7-34
Self-Test .....	7-39
Basic Self-Test .....	7-39
Cartridge Tape Unit Self-Test .....	7-43
Data Communications Self-Test .....	7-43

Section VIII	
<b>HP-IB Configurations</b>	
Introduction .....	8-1
Cabling Considerations .....	8-2
Cabling Limitations .....	8-3
Device Considerations .....	8-3
Operation Considerations .....	8-3
Sample Configurations .....	8-3
Testing the HP-IB .....	8-4

Appendix A	Page
<b>APPLICATIONS</b>	
Sample HP 3000 BASIC Program to Control the HP 9872A Plotter .....	A-1
Multipoint Example .....	A-2
Forms Building .....	A-3
Programming the Soft Keys .....	A-3
Building the Form .....	A-5
Large Character Set Utility .....	A-7

Appendix B	Page
<b>REFERENCE TABLES</b>	
Introduction .....	B-1
Specifications .....	B-2
Programmer's Reference Table .....	B-3
Character Code Chart .....	B-8
Options and Accessories .....	B-9
Coding the Large Character Set .....	B-10

Appendix C	Page
<b>COMMUNICATIONS FLOWCHARTS</b> .....	
	C-1

Appendix D	Page
<b>CARTRIDGE TAPE RETHREADING PROCEDURE</b> .....	
	D-1

# ILLUSTRATIONS

<b>Title</b>	<b>Page</b>
Terminal Architecture .....	1-2
Character Cell .....	1-3
System Monitor Basic Loop .....	1-4
Display Memory Linked List Structure .....	1-6
Row Addressing .....	2-2
Column Addressing .....	2-2
Character Delete With Margins .....	2-5
Character Delete With Wraparound .....	2-5
Character Set Locations .....	2-9
Example Using the Math Set .....	2-9
Example Using the Large Character Set .....	2-9
Math Set Elements .....	2-10
Large Character Set Elements .....	2-10

<b>Title</b>	<b>Page</b>
Line Drawing Set .....	2-10
Sample Form .....	2-11
Location of Graphics Keys .....	3-1
Examples of Drawing Modes .....	3-7
Predefined Line Type Patterns .....	3-8
Using Area Patterns As Line Types .....	3-9
Examples of User Defined Line Patterns .....	3-9
Area Pattern Examples .....	3-11
Relocatable Origin .....	3-13
Current Pen Position and New End Point .....	3-15
Recording Graphics Sequences .....	3-19
Raster Data Format .....	3-19A
Graphics Text Characters .....	3-20

# ILLUSTRATIONS (continued)

Title	Page	Title	Page
Graphics Text Sizes .....	3-21	Top Plane Assembly Removal .....	7-7
Graphics Text Direction .....	3-21	Control Memory PCA Jumper Socket Location ....	7-10
Graphics Text Justification .....	3-23	4K Memory PCA Jumper Socket Location .....	7-10
Autoplot Menu .....	3-24	Typical Memory Map .....	7-11
Sample Financial Data .....	3-28	Configuring 4K Memory PCA Jumpers .....	7-11
Completed Menu for Multiplot Example .....	3-29	Terminal Duplex Register PCA	
Multiplot Results .....	3-30	Jumper Configuration .....	7-12
Autoplot Example Using Computer		Control Memory PCA Data Comm Firmware	
Generated Data .....	3-30	IC Locations .....	7-15
Curve Shading With Autoplot .....	3-31	Installing Keyboard Overlays .....	7-16
Turning on Compatibility Mode .....	3-34	13261A-003 Device Support Firmware ROM IC	
Comparison of a Terminal With 1024 × 780		Locations on Control Memory PCAs .....	7-16A
Display and the HP 2648A .....	3-35	HP-IB Interface Switch Settings .....	7-16B
Scaled Data .....	3-35	Typical Strapping Option Switch Assembly .....	7-18
Unscaled Data .....	3-35	Keyboard Interface PCA Strapping Options .....	7-18
Terminal Network Configurations .....	5-2	Extended Asynchronous Communications	
Block Transfer Enabled by the ENTER Key .....	5-8	PCA Strapping Options .....	7-22
Block Transfer Enabled by the Computer .....	5-8	Asynchronous Multipoint Communications	
Block Mode Operation .....	5-10	PCA Strapping Options .....	7-26
Example of Format Mode with Page Strapping ....	5-11	Synchronous Multipoint Communications	
Main Channel Protocol .....	5-15	PCA Strapping Options .....	7-26
Sample Data Transfers Using Main		Point-to-Point Communications Cabling .....	7-29
Channel Protocol .....	5-16	Current Loop Cabling .....	7-29
Reverse Channel Protocol .....	5-17	Modem By-Pass Cabling .....	7-29
Sample Data Transfers Using		Asynchronous Multipoint Cabling .....	7-30
Reverse Channel Protocol .....	5-18	Synchronous Multipoint Cabling .....	7-31
Point-to-Point Data Communications		Power-Down-Protect Cabling for Asynchronous	
Configuration Flowchart .....	5-19	Multipoint Configuration .....	7-32
Examples of Block Transmissions .....	5-23	Power-Down-Protect Cabling for Synchronous	
Operation of Block Protocol		Multipoint Configuration .....	7-33
Control Characters .....	5-29	Assembling the PCA Hood Connector .....	7-36
Block Keyboard Data Communication Switches ....	5-33	Assembling the RS232C Connector .....	7-37
Terminal Addressing .....	5-36	Assembling the Multipoint Connector .....	7-38
Typical Configuration Status Request		Basic Self-Test Patterns .....	7-39
and Response Sequence .....	5-38	Basic Terminal Self-Test Flowchart .....	7-40
Configuration Status Byte Contents .....	5-39	Basic Data Comm Self-Test Flowchart .....	7-45
Multipoint Data Communications		Multipoint Data Comm Self-Test Flowchart .....	7-48
Configuration Flowchart .....	5-40	HP-IB Interconnecting Cable .....	8-1
Communication Line Using a Monitor .....	5-44	HP-IB Interface Adapter .....	8-1
Sample Data Transfers Displayed in		Selecting Additional Device Loads .....	8-3
Monitor Mode .....	5-45	Terminal-to-Plotter Configuration .....	8-3
Character Distortion in Group Poll .....	5-45	Plotter/Line Printer Configuration .....	8-4
Data Overflow Indication .....	5-45	Maximum Cabling Length Configuration .....	8-5
Driver Mode Configuration .....	5-46	Soft Key Programming and Soft Key Overlay	
Sample Select Sequence Using Driver Mode .....	5-47	for File 1 .....	A-4
Control Character Display On Driver Terminal ....	5-47	Soft Key Programming and Soft Key Overlay	
Terminal Input .....	5-47	for File 2 .....	A-4
Primary Terminal Status Example .....	6-2	Soft Key Programming and Soft Key Overlay	
Secondary Terminal Status Example .....	6-4	for File 3 .....	A-4
Device Status Example .....	6-10	Sample Form .....	A-5
Opening the Terminal .....	7-2	Building a Form — Phase 1 .....	A-6
Mainframe Bottom Part Locations .....	7-3	Building a Form — Phase 2 .....	A-6
Mainframe Top Part Locations .....	7-4	Building a Form — Phase 3 .....	A-6
Fuse Positions for 115 VAC and 230 VAC		Large Character Set Utility Routine .....	A-7
Line Voltage .....	7-5	Point-to-Point Communications Flowcharts .....	C-5
Sample Character Set Configuration .....	7-6	Keyboard Communication Switches Flowcharts .....	C-9
Display Enhancement PCA Jumper and ROM		Tape Cartridge Rethreading .....	D-1
Socket Locations .....	7-7		



# TABLES

Title	Page	Title	Page
Cursor/Display Operations .....	2-4	ASCII Status Characters .....	6-1
Edit Operations .....	2-5	Graphics Status Requests .....	6-6
Keyboard Interface Switch Summary .....	2-12	Display Enhancement PCA Jumper Protocol .....	7-7
Graphics Control Keys .....	3-2	HP 13232 Cable Assemblies .....	7-8
Summary of Graphics Sequence Types .....	3-3	Contents of HP 13260 Data Communications Accessories .....	7-14
Graphics Display Control Functions .....	3-4	HP-IB Interface Switch Settings .....	7-16B
Graphics Mode Commands .....	3-6	Keyboard Interface PCA Strapping Options for Point-to-Point .....	7-19
Graphics Parameter Default Values .....	3-14	Keyboard Interface PCA Straps for Block Operating Using 13260C or 13260D Communications Accessory .....	7-21
Graphics Plotting Control Functions .....	3-14	Extended Asynchronous Communications Interface Strapping Options .....	7-23
Characters Used In Packed Data Formats .....	3-17	Asynchronous Multipoint Communications Interface Strapping Options .....	7-24
Absolute Format Addressing Bytes .....	3-18	Synchronous Multipoint Communications Interface Strapping Options .....	7-25
Incremental (Short) Vector Bytes .....	3-19	EIA RS232C and CCITTV24 Interface Data and Control Signals .....	7-27
Graphics Control Sequences Used in Record Operations .....	3-19	Data Communications Signal Levels .....	7-27
Raster Dump Command Summary .....	3-19B	Multipoint Data Communication Cables .....	7-28
Raster Dump Transfer Rates .....	3-19B	Parts for Fabricating Your Custom Data Communications Cable .....	7-34
Graphics Text Keyboard Functions .....	3-20	13260 Series Data Communications PCA Signal Names .....	7-35
Autoplot Commands .....	3-26	Data Communications Self-Test Connectors .....	7-44
Draw Axes and Autoplot Parameters .....	3-31	Point-to-Point Data Communications Self-Test Procedure .....	7-44
Compatibility Mode Control Sequences .....	3-32	Multipoint Data Communications Self-Test Procedure .....	7-47
Commands for Selecting Compatibility Mode .....	3-33	Specifications .....	B-2
Coding of Compatibility Mode Graphics Data .....	3-37	Programmer's Reference Table .....	B-3
Device Control Command Characters .....	4-1	Character Code Chart .....	B-8
Data Communication Interfaces .....	5-1	Options and Accessories .....	B-9
Data Communication Interface Capabilities .....	5-3	Coding the Large Character Set .....	B-10
Keyboard Interface Switch Summary .....	5-3	ASCII Character Set .....	C-2
Keyboard Communications Switches .....	5-4	ASCII to EBCDIC Code Conversion Table .....	C-3
Modems .....	5-5		
Protocol Characteristics .....	5-6		
Terminal Functions .....	5-7		
Keyboard Interface PCA Strapping Options for Point-to-Point .....	5-12		
Parities Available with ASCII Data .....	5-24		
Block Protocol Control Characters .....	5-27		
Summary of Block Protocol Control Characters .....	5-31		
Keyboard Interface Straps for Block Operation Using 13260C or 13260D Communications Accessory .....	5-32		
Terminal Address Characters .....	5-35		

# GENERAL DESCRIPTION

SECTION

I

## INTRODUCTION

The Graphics Terminal uses a microprocessor under firmware control. The terminal provides interactive graphics features available under user or program control. It operates in character or block mode, with full editing capability. The terminal is designed for such applications as data entry and preparation, information display and editing, interactive programming, data communications, and time-sharing operation.

In addition to the features of the basic terminal, option 007 provides an integrated mass storage capability of up to 220 kilobytes of data using two tape cartridges. This allows the terminal to be used for either stand-alone or on-line operation. For example, forms designed at the terminal using the line drawing or other character sets can be stored on a tape cartridge and selectively retrieved from the keyboard or through commands from a remote computer.

Data communications accessories are also available to provide a choice of communications capability. The standard terminal is teletypewriter compatible (EIA RS232-C serial asynchronous, ASCII, half or full duplex). It operates at speeds up to 9600 bits per second, and transmits either character-by-character as a fully interactive terminal or operates on variable length blocks of information. Optional capabilities include 20mA current loop; and either asynchronous or synchronous polling for multipoint communications networks. Also, the terminal can be used with a wide selection of modems over dialed or leased lines.

A block diagram of the terminal is shown in figure 1-1. The terminal has three major and mechanically independent sections: keyboard, CRT monitor, and mainframe. The specific functional properties of the terminal are determined by firmware programs resident in ROM (read-only-memory). It is these programs that make it possible for the terminal to have many powerful features such as self test, dynamic memory allocation, transparent control codes, and off-screen storage.

## MAINFRAME

The heart of the system is the mainframe section, which can be considered a microcomputer system. In the mainframe is the power supply and a bus-oriented logic system containing the microprocessor, program and alphanumeric memory, graphics control and graphics memory, video display subsystem, keyboard interface, and data communications interface. The basic terminal contains

four slots for options and accessories. All mainframe modules are functionally, mechanically, and electrically independent, giving a high degree of flexibility and reducing service time.

## Microprocessor

The terminal uses an 8-bit microprocessor to control most of the terminal's operation. The microprocessor executes code that may be in ROM or RAM memory. It controls the alphanumeric display, programmable functions, and I/O devices.

## Display

The terminal uses two separate display memories. An alphanumeric memory is used to hold up to 6,000 characters. Graphic data is stored in a separate graphics memory. The graphics data is stored as a dot pattern containing 259,200 points (720 × 360). Both the alphanumeric and the graphic memories are accessed by the same display circuitry during the refresh cycle.

The display subsystem has two functions that use the bus. Cursor control is an output function and the DMA refresh is a bus requestor for memory read operations.

**ALPHANUMERIC DISPLAY.** The alphanumeric display is maintained by the Display Control module. It reads ASCII characters and display commands from the display memory. These characters are converted to dot patterns and passed to the display circuitry during the refresh cycle. The Display Timing module provides the timing signals for the display circuitry.

**GRAPHICS DISPLAY.** The graphic display is maintained by the Graphics Microcontroller and Graphics Display Memory PCA. Graphic data received from the keyboard, data communications interface, or cartridge tape is processed by the Graphics Microcontroller before being stored in the Graphics Memory. Input data is normally in the form of vector end points. The Microcontroller uses the previous vector end point together with the new end point to generate a line of display dots that approximate the line. These dots are then stored in the Graphics Memory.

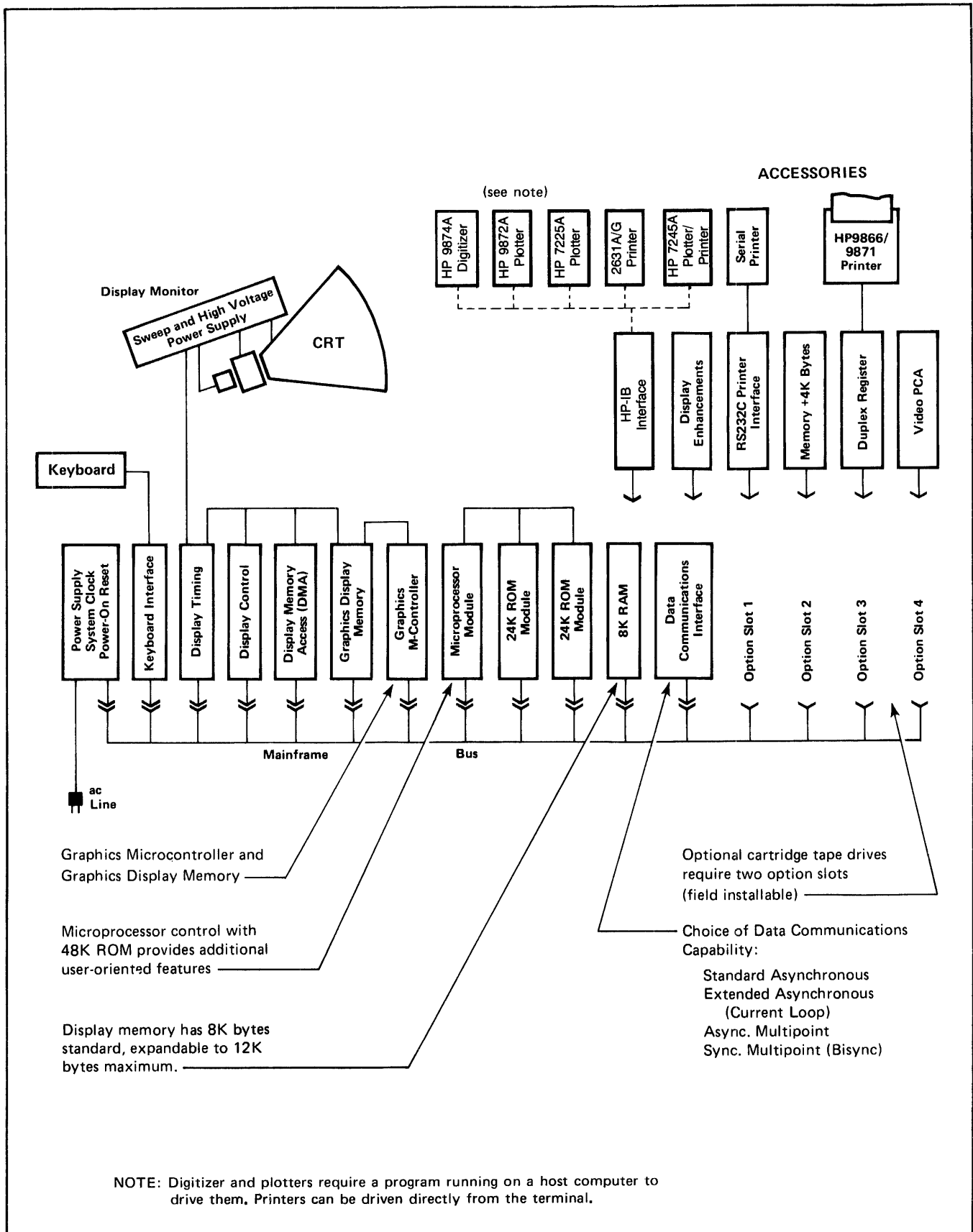


Figure 1-1. Terminal Architecture

## Terminal Bus

A major element of the logic system is the terminal bus (a printed circuit board with connectors) which is attached to the bottom of the mainframe and to the power supply. The bus distributes power to the individual modules and provides data, address, and control lines for communication between the various logic functions. The terminal bus provides communication paths between processor, memory, input/output, and display refresh on a shared basis.

All modules are slot-independent and carry their own select code or memory address. The only requirements are that the four display modules must be grouped together, and that no empty slots be left between the power supply and the last module. However, the two cartridge tape modules (part of option 007) can be plugged into the connectors on the far end, opposite the power supply.

A bus access cycle begins when a requesting module determines it needs the terminal bus for instruction or data fetch or input/output. If the terminal bus is busy, the requesting module must wait until it is available. To determine who gets the bus next, a priority chain has been incorporated. The modules nearest the power supply are first in the priority chain, and a module wanting the bus next breaks the chain for modules farther away from the power supply.

## Terminal Memory

Like any other computer system, the microcomputer module is useful only if it has a program to execute and memory in which to store data. This is the function of the terminal memory modules, which are two types, read/write or random-access memory (RAM) and read-only memory (ROM). The RAM stores display characters and data; the ROM stores terminal programs (firmware). Terminal programs are called firmware because the ROM makes them more permanent than software but less permanent than hardware. In the terminal three-quarters of the available memory is dedicated to ROM or program memory and the remaining memory locations are RAM and can be used for data. All of the terminal memory is MOS semiconductor memory. A separate RAM memory is used to store the  $720 \times 360$  graphics data.

Optional RAM modules are added when more random-access memory is required for alphanumeric display and data storage. A separate graphics display RAM memory is used to hold graphic data.

## Input/Output Modules

Also a part of the logic system are several terminal input/output modules: the keyboard interface PCA, Graphics Microcontroller PCA, Graphics Memory PCA, the data communications PCA, the optional eight-bit duplex register PCA (used for the HP 9866A and HP 9871A Printer interface), the optional serial printer interface PCA, and the HP-IB Interface PCA (used for interfacing the HP 2631G Printer and the HP 7245A-001 Plotter Printer).

These never request control of the bus, but all must respond to commands from the microcomputer and its programs.

The basic I/O commands output data or control codes from the microprocessor and input data or status from the interface module. Each of the I/O cards has different data and control formats, but all are controlled by the microcomputer. Each I/O module has a rear edge connector for the attachment of a connector hood and cable assembly to carry the signals out the back of the terminal.

## CRT MONITOR

The CRT monitor section contains sweep and high voltage circuits, the high-resolution, low-profile cathode-ray tube, and fan.

## The Raster

The terminal uses raster scan deflection method, similar to that used in television sets. In a raster scan display, the electron beam traverses the screen in a series of closely spaced horizontal lines, starting from the top. Characters are formed from line segments and dots produced by turning the beam intensity on and off at appropriate times.

There are  $720 \times 360$  dots on the screen. The alphanumeric display makes use of characters that fill an entire  $7 \times 9$  character cell while the graphics display allows you to access individual dots.

The terminal uses a low-profile CRT to keep overall height to a minimum while maintaining a screen capacity of 1920 characters, partitioned into 24 rows of 80 characters each. All of the character positions are fundamentally rectangles 7 dots wide by 9 scan lines high. Four additional scan lines beneath the  $7 \times 9$  matrix are used for the descender areas of lower-case characters, for underlining, and for the blinking underscore cursor. One other dot is used on either side for character-to-character spacing, and one scan line is reserved at the top and bottom for row-to-row spacing. This results in a character cell of 9 dots by 15 scan lines replicated over the entire screen area (see figure 1-2).

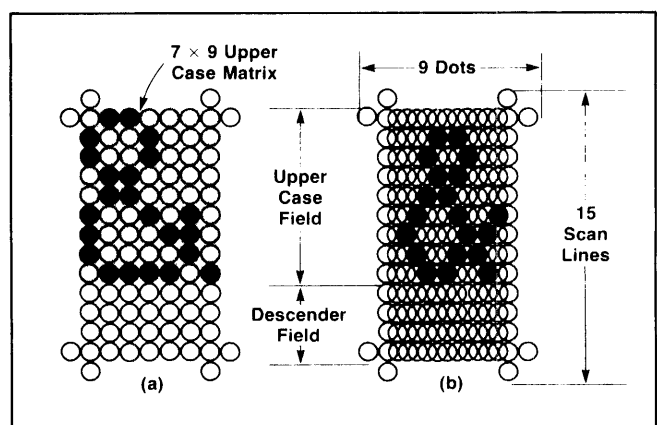


Figure 1-2. Character Cell

## Alphanumeric and Microvector Character Sets

Two types of character sets can be stored within the terminal: alphanumeric sets and microvector sets. Alphanumeric sets support the primary use of the terminal, displaying textual and numeric information. Characters are designed around a basic 7x9 dot matrix with provision for lower-case descenders. The characters are embellished by use of the half-shift. With this type of set the character-to-character spacing of two dots is hardwired. This prevents the design of characters that would form continuous horizontal lines. However, all 15 scan lines of the row are available so that vertically contiguous symbol segments can be designed. An example of this is the three-row-high integral sign found in the math symbol set.

Microvector sets use the entire 9-dot-by-15-scan-line character cell without the half-shift. This allows characters to be designed with both horizontal and vertical continuity. This type of set finds its greatest application where a minimal set of graphic kernels is needed to represent more complex pictorial information. The data entry forms shown in Section II illustrate the use of the line drawing set in representing a form.

## Display Features and Alternate Character Sets

The basic terminal uses 128 alphanumeric characters and one display feature, inverse video fields (black characters on white backgrounds). With the addition of the display enhancement board, up to three additional 128-character sets can be stored within the terminal. Three display features are also added: half-bright, underline, and blinking fields.

All sixteen possible combinations of the four display features can be applied to any character or characters on the screen. No displayable character positions are required to start, stop, or modify either the features or the character sets. Therefore, consecutive characters on the screen may be from different sets or have different display features.

## KEYBOARD

The processor scans the keyboard at discrete intervals for a depressed key. Each key is assigned a position in a matrix of 14 columns and 8 rows. This matrix provides a reference to a look-up table that the firmware uses to display the character and/or send the character code over the data communications line.

## THE FIRMWARE

The firmware contains the operating system or main terminal code modules and the various input/output, data communications, and utility routines that control the terminal. The firmware is stored in read-only memory (ROM) circuits on the Control Memory assemblies.

## System Monitor

The system monitor is a section of the firmware that dispatches data within the terminal. The processor normally executes a basic loop, in which it scans the keyboard and the data communications interface and waits for something to happen (see figure 1-3). When a character is received from either the keyboard or the data communications interface, a general character interpretation routine is executed to determine the action to be taken. The monitor then performs the specified functions, such as putting a character on the display, transmitting a character over the data communications interface, or moving the cursor. When this has been completed, the monitor returns to the basic scan loop to look for the next input.

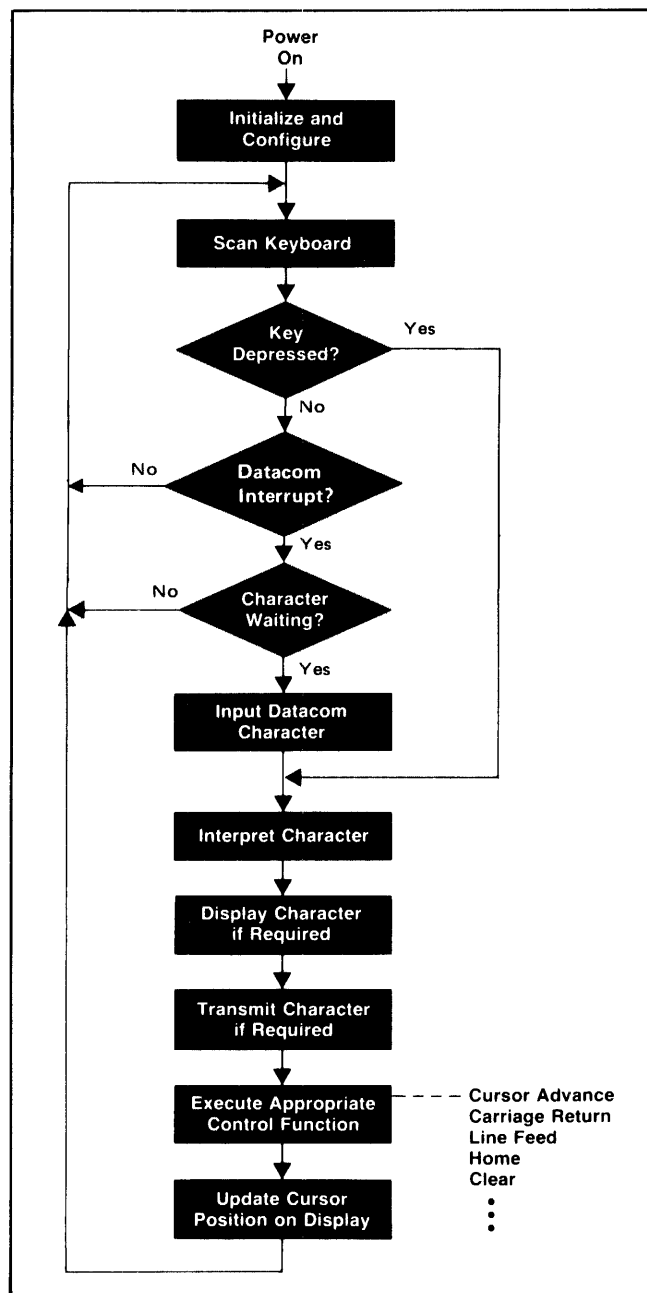


Figure 1-3. System Monitor Basic Loop

## Keyboard and I/O Subsystem

The I/O subsystem contains the firmware required for performing all input/output functions. The firmware operates using both scan and interrupt methods. The keyboard is scanned at regular intervals, while inputs from the devices such as the data communications interface and cartridge tape units are interrupt driven. If a new key depression is detected, the key number associated with this key is calculated and used as an index into a table that assigns a code to the key. If the key is one of the ASCII keys, the proper code is determined based on the state of the CNTL, SHIFT, and CAPS LOCK keys.

If the key in question is not one of the ASCII keys, the firmware may be required to generate a multiple character sequence consisting of an ASCII escape character followed by one or more characters that define the escape sequence. Keys in a third group do not generate codes at all, but simply perform internal terminal functions, such as BLOCK MODE, REMOTE, and CAPS LOCK.

I/O associated with the display is minimal because the display memory access module (DMA) causes the display to be refreshed without processor intervention. Display I/O control mainly involves transmitting the cursor coordinates to the display whenever necessary.

## Cursor Movement

The terminal firmware contains many subroutines for moving the cursor on the display. All cursor movement is handled by the firmware. When a character is typed on the keyboard and appears on the display, the cursor moves to the next column position because a cursor advance subroutine has been executed and has calculated a new cursor position. Similar subroutines exist for moving the cursor up, down, right, left, and home. The tab function is also a firmware routine; it uses a one-bit-per-column table to determine the next tab stop.

A separate graphics cursor can be displayed. The graphics cursor can be controlled from the keyboard or from a program. It functions independent of the alphanumeric data and can be used to aid in the input of graphics data. A set of graphics cursor control keys (similar to those used for the alphanumeric cursor) can be used by the operator to position the graphics cursor.

## Display Memory Management

A large part of the terminal firmware is devoted to management of the display memory. Most conventional terminals use a byte of display memory for every displayable position on the CRT screen. If there are many short lines, as is frequently the case, there is a substantial amount of unused memory. The terminal does not allocate memory for character positions to the right of the last character entered, so this memory is available for other purposes. Turn on and turn off of the various display enhancements

and character set selections, or start and end of unprotected fields between individual characters can be accomplished without an intermediate blank character position. With these features, the address of a character occupying a given row and column cannot be directly computed without some sort of scanning process.

The display memory consists basically of a linked list of fixed-sized blocks of RAM (see figure 1-4). This list is set up in such a way that the DMA can start at the first address on the screen and follow the list to produce an entire screen of information. All memory not currently allocated for display use is kept on a free-storage link list. Individual rows are linked with next and preceding rows, while blocks within a row are linked only in a forward direction. The storage allocated for a row may be as little as one block (16 bytes), or much larger than 80 characters, depending upon the number of displayable and non-displayable characters needed to create the row on the CRT.

The firmware finds the address corresponding to a given character position by starting at the last known position and moving through the list either backward or forward until it finds the new address. If the end of the list is found before the row in question has been found, blocks are removed from the free-storage list and used to create new rows. Once the correct row has been found, the firmware searches for the cursor column. If the end of the row is found before the column has been found, additional blocks are removed from the free-storage list and used to build the length of the row out to the column required. Whenever a block is required and free list is empty, an existing row must be released from display memory. This row is the first row of memory if the addition is at the end of memory, and is the last row of memory if a row other than the last row is being lengthened.

## DATA COMMUNICATIONS

Data communications in the terminal is both a hardware and a firmware function. The data communications interface is a basic terminal module. This module has the necessary logic to interface the terminal bus to the communication line.

The communication interface accepts parallel data from the terminal, serializes it, and adds framing or synchronizing bits (start and stop). It performs the reverse process on incoming data, converting serial data to parallel and removing start and stop bits. The interface can generate and check parity and can also detect data overruns. A status word keeps the processor informed of the status of the interface.

The terminal firmware for communications has three main functions. First, the program reads the control settings on the keyboard, keyboard interface, and the data communications interface. Second, it processes input characters and transmits output characters. Many decisions are made on incoming characters, especially on control characters. The third function is modem control using

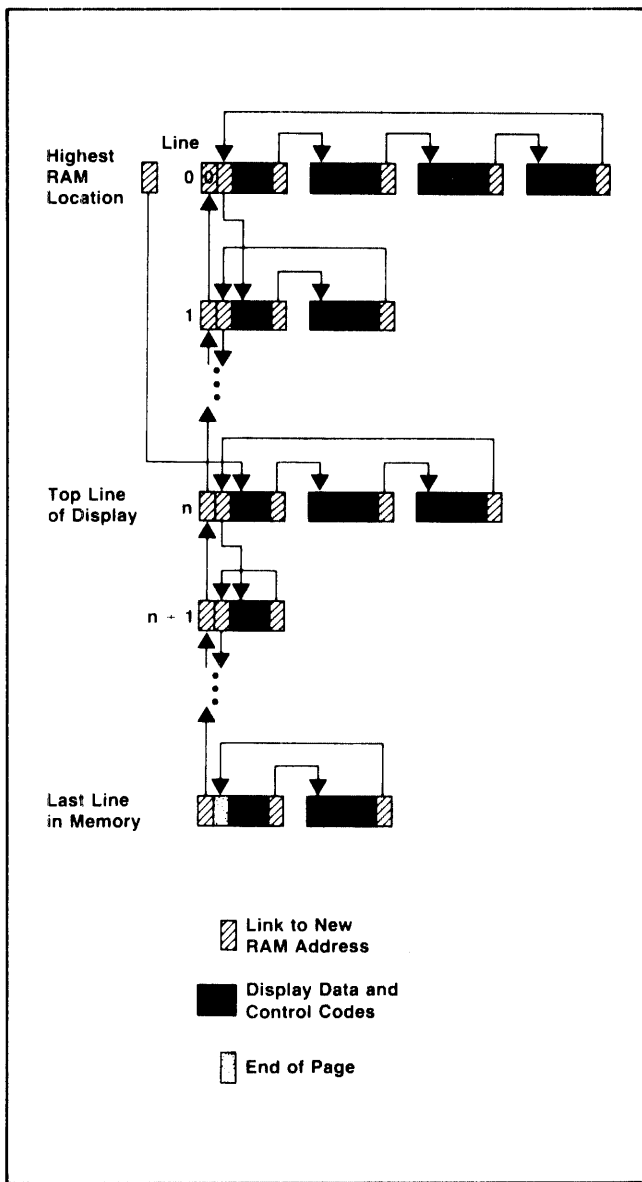


Figure 1-4. Display Memory Linked List Structure

control lines on the communications interface. Direct connections to a computer and Bell 103 type modems require a minimum of firmware control while Bell 202 type modems require more control. A more detailed description of the data communications operation is given in Section V.

## THE TAPE UNITS

Cartridge tape operations are divided between the cartridge tape hardware and firmware. The hardware maintains constant speed and provides for selection of fast or slow speed and the direction of tape motion.

A line of data on the screen is recorded on the tape as a record. The information is stored serially on the tape at a recording density of 800 bits per inch and a read/write speed of 10 inches per second. Data is organized into files, records, bytes, and bits. Separating the files are file marks, special-length gaps (areas of unidirectional magnetization) with a file identification record between them.

Holes are punched at each end of the tape to provide reference for beginning of tape and end of tape. If a hole is detected in the tape, indicating tape location, tape motion is stopped until the firmware commands it to start again.

The hardware encodes and decodes data bytes into bit patterns on the tape and records interrecord gaps. The hardware reacts to commands given by the firmware and presents status information to the firmware.

The firmware controls all tape motion and maintains tape-position information. The firmware dictates whether the hardware is reading, recording, or writing gaps. It formats data into records and generates special tape marks that have significance in organizing the tape into records and files.

## Error Messages

Two types of error messages are displayed by the 264X terminals; self test errors and operating-time errors.

**SELF TEST ERRORS** .....

Self test errors and their meanings can be found in the paragraph titled "Basic Terminal Self Test."

**OPERATING-TIME ERRORS** .....

The errors which might occur during operation are listed in the table below.

MESSAGE	MEANING
<b>BUFFER OVERFLOW</b>  <b>NO DEVICE DRIVER I/O ERROR X</b>	<p>There is insufficient RAM memory available for data communications buffer space.</p> <p>An I/O operation has been attempted for which the necessary accessory is not installed.</p> <p>An invalid device interrupt has occurred. Specific meanings are listed below. (Most I/O error messages imply execution of bad code being attempted or bad code or data being addressed.)</p> <p><b>X=0</b> Undefined interrupt. Probably a bad Processor PCA, or ROM error or bad CTU.</p> <p><b>X=A</b> ROM containing CTU code missing or bad.</p> <p><b>X=B</b> ROM which should contain code for the alternate I/O device missing.</p> <p><b>X=1</b> Probably malfunctioning Processor PCA or ROM or RAM error.</p> <p><b>X=2</b> Same as X=1.</p> <p><b>X=3</b> Timer error.</p> <p><b>X=4</b> Data communications error.</p> <p><b>X=6</b> Probable ROM or RAM error. (All 1's, except for bit 0, in word being analyzed.)</p> <p><b>X=7</b> Probable ROM or RAM error. (All 1's in word being analyzed.)</p>



# DISPLAY MEMORY AND TERMINAL CONTROL FUNCTIONS

SECTION

II

## INTRODUCTION

This section contains information for using the display memory and terminal control functions. The display memory functions change the position of display data or assign special attributes to blocks or fields of display data. The special attributes alter the way data is displayed or transmitted. The terminal control functions allow you to programmatically set terminal straps or use the terminal's soft keys.

## DISPLAY MEMORY FUNCTIONS

The following paragraphs describe the display memory functions. These functions consist of the following groups:

- Display Control
- Edit Operations
- Forms Mode
- Display Enhancements
- Alternate Character Sets

The following paragraphs describe how to control the alphanumeric display memory functions from a computer program. Each of the display functions can also be entered from the terminal keyboard or read from a cartridge tape. In addition to escape sequences, most of the display memory functions have been assigned to special keys on the keyboard. Refer to the *User's Manual* for a description of keyboard functions.

### Display Control

The display control functions are made up of cursor and display positioning operations. Only the alphanumeric display functions are described here. Refer to Section III for a description of the graphic display functions. The individual functions available are as follows:

- Cursor Sensing
  - Absolute
  - Relative
- Cursor Positioning
  - Absolute Addressing
  - Screen Relative Addressing
  - Cursor Relative Addressing
  - Space
  - Backspace
  - Set Tab
  - Clear Tab
  - Tab
  - Backtab

Set Margins  
Home Up  
Home Down

- Display Positioning
  - Roll Up
  - Roll Down
  - Next Page
  - Previous Page
  - Display Lock (Memory Lock)

**MEMORY ADDRESSING SCHEME.** Display memory positions can be addressed using absolute or relative coordinate values. Display memory is made up of 80 columns (0-79) and a number of rows determined by the memory options installed in the terminal. There can be as many as 150 lines of 80 characters (6 screens). The amount of memory in the terminal can be determined from byte 0 of the primary terminal status (refer to Section VI). The types of addressing available are:

- Absolute
- Screen Relative
- Cursor Relative

**Row Addressing.** Figure 2-1 illustrates the way the three types of addressing affect row or line numbers. The cursor is shown positioned in the fourth row on the screen. Screen row 0 is currently at row 6 of display memory. In order to reposition the cursor to the first line of the screen the following three destination rows could be used:

- a. Absolute: row 6
- b. Screen Relative: row 0
- c. Cursor Relative: row 3

**Column Addressing.** Column addressing is accomplished in a manner similar to row addressing. There is no difference between screen and cursor relative column addressing. Figure 2-2 illustrates the difference between absolute and relative addressing. The cursor is shown in column 5.

Whenever the row or column addresses exceed those available, the largest possible value is substituted. In screen relative addressing, the cursor cannot be moved to a row position that is not currently displayed. For example, in figure 2-1c a relative row address of -10 would cause the cursor to be positioned at the top of the current screen (relative row -3). Column positions are limited to the available screen positions (0 to 79 in figure 2-2a and -5 to 74 in figure 2-2b). The cursor cannot be wrapped around from column 0 to column 79 by specifying large negative values for relative column positions.

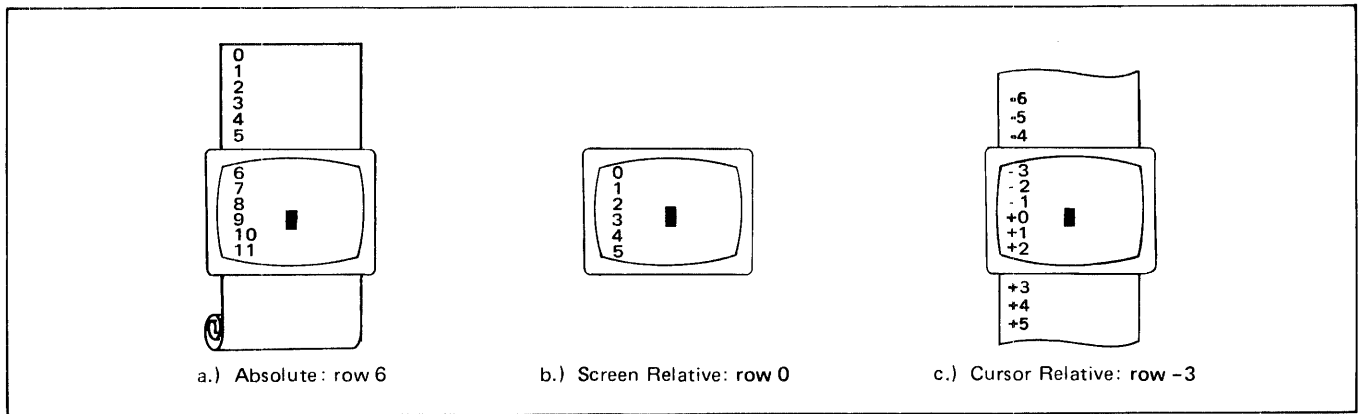


Figure 2-1. Row Addressing

**CURSOR SENSING.** The current position of the screen cursor can be sensed. The position returned can be the absolute position in display memory or the location relative to the current screen position. (Absolute and relative addresses are discussed under Memory Addressing.)

**Absolute Sensing:**

```
ESC a
```

**Example:** The cursor is at column 20, row 40.

```
computer: ESC a
terminal: ESC & a 020c 040R
```

**Relative Sensing:**

```
ESC `
```

**Example:** The cursor is again at column 20, row 40, but screen row 0 begins at row 35 of display memory.

```
computer: ESC `
terminal: ESC & a 020c 005Y
```

**CURSOR POSITIONING.** The cursor can be positioned directly by giving memory or screen coordinates, or by sending the escape codes for any of the keyboard cursor positioning operations.

**Absolute Addressing.** The cursor can be positioned to any displayable position using absolute coordinates. Absolute cursor positioning is accomplished using the following sequence:

```
ESC & a <row number> r <column number> C
```

where: row number is 0 to 255.

**Example:** Position the cursor at row 35, column 6.

```
ESC & a 35r 6C
or
ESC & a 6c 35R
```

Absolute addressing cannot be used while Memory Lock is on.

**Screen Relative Addressing.** The cursor can be positioned to any position currently displayed on the screen by using screen relative coordinates:

```
ESC & a <screen row number> y <column number> C
```

where: the top row of the screen is now 0.

**Example:** Position the cursor to screen relative row 15, column 53.

```
ESC & a 15y 53C
or
ESC & a 53c 15Y
```

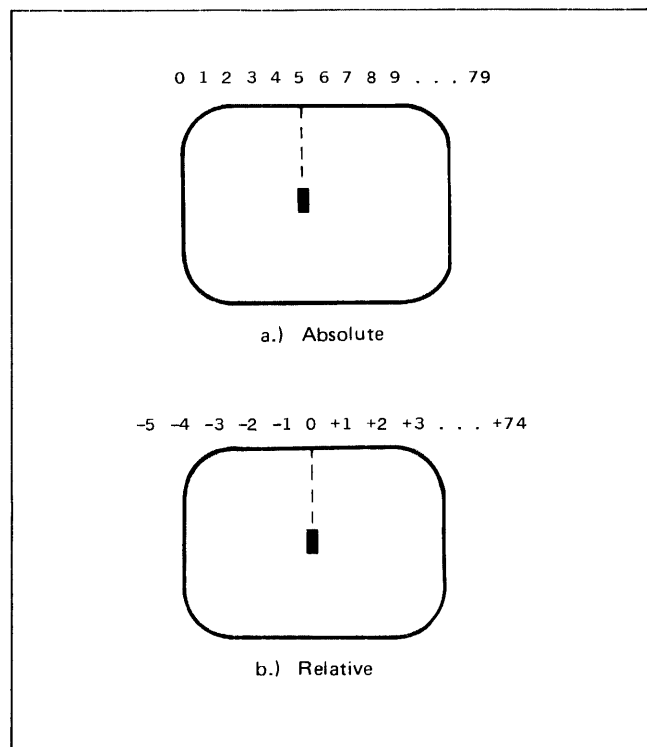


Figure 2-2. Column Addressing

**Cursor Relative Addressing.** The cursor can be positioned to any displayable position by using cursor relative coordinates. ( $\pm$ row,  $\pm$ column). Cursor relative addressing cannot be used while Memory Lock is on.

**Example:** The cursor is currently at row 7 and column 10 of the screen. Move the cursor to row 9 column 6.

```
ESC & a +2r -4C
      or
ESC & a -4c +2R
```

**Combinations of Absolute and Relative Addressing.** Relative and absolute coordinates can be combined in the same sequence.

**Example.** Move cursor from its current row down 8 rows and to column 60.

```
ESC & a +8r 60C
      or
ESC & a 60c +8R
```

**Example:** Move cursor from its current position 15 columns left, and to relative screen row 4.

```
ESC & a 4y -15C
      or
ESC & a -15c 4Y
```

**Alphanumeric Cursor On/Off.** The alphanumeric cursor is turned on or off whenever the alphanumeric display is turned on or off. The cursor can also be controlled separately using the following commands:

Turn Alphanumeric  
Cursor On:

```
ESC * d q
```

Turn Alphanumeric  
Cursor Off:

```
ESC * d r
```

**OTHER CURSOR OPERATIONS.** In addition to positioning the cursor using coordinates, you can use a variety of keyboard equivalent operations. These operations normally require only one or two characters to be sent to the terminal. Table 2-1 lists each of the operations together with its code and a brief description.

**TABS.** You can programmatically set tabs, tab, and clear tabs.

**Setting Tabs.** To set a tab, move the cursor to the desired column and send ESC 1. Once a tab is set, the tab function or **TAB** key can be used to move the cursor to the next tab setting. In Forms Mode, previously set tabs are ignored; however, when Forms Mode is turned off, previously set tabs are in effect.

**Using Tabs.** Once tab positions have been set you can tab in the same manner that you would on a typewriter. You can even tab backwards to the previous tab position by sending ESC i. When you are at the first tab position in a

line and you backtab, the cursor moves to the last tab position in the previous line. Once the cursor has reached the first tab position in the first line of memory, no further backtabbing movement can be made.

**Clearing Tabs.** You can clear individual tabs by moving the cursor to the tab position and sending ESC 2. All of the tabs can be cleared at once without having to position the cursor by sending ESC 3.

**MARGINS.** You can set the left and right margins to make the entry of data easier. When the terminal is turned on or a full reset performed, the margins are set at columns 0 and 79. This gives a full 80 character line. You can define new margins as follows:

**Left Margin.** Move the cursor to the desired left margin setting. Send ESC 4.

Set Left Margin: 

```
ESC 4
```

**Right Margin:** Move the cursor to the desired right margin setting. Send ESC 5.

Set Right Margin: 

```
ESC 5
```

The terminal will beep when you are eight characters from the right margin. When the right margin is reached, the cursor will move to the left margin of the next line.

**Example:** Set the margins for a 40 column page centered on the screen.

Move the cursor to column 20 and set the left margin. Move the cursor to column 59 and set the right margin. Place the cursor back at column 20 and begin sending data.

ESC & a 20C	Esc 4	Esc & a 59C	Esc 5
\ /	\ /	\ /	\ /
position	set	position	set
cursor	margin	cursor	margin
column numbers			
2	3	4	5
0	0	0	0

This is an example using margins to control data entry.

Margins are cleared or changed by setting new margins (or a full reset) or by entering forms mode, where the margins are reset to columns 0 and 79.

**Alphanumeric Display On/Off.** The entire alphanumeric display, including the alphanumeric cursor, can be turned on and off. The alphanumeric data is not lost when the display is turned off. The alphanumeric cursor can also be controlled independent of the display (refer to Alphanumeric Cursor).






Turn Alphanumeric  
Display On:

```
ESC * d e
```

Turn Alphanumeric  
Display Off:

```
ESC * d f
```

Table 2-1. Cursor/Display Operations

FUNCTION	CODE	DESCRIPTION
<b>Cursor</b>		
	ESC*me	Turn cursor on.
	ESC*mf	Turn cursor off.
Line Feed	LF (J <sup>c</sup> )	Move the cursor to the next line.
Return	CR (M <sup>c</sup> )	Return the cursor to the left margin, halt I/O operations, and clear messages.
	ESC G	Move cursor to first column of current row.
Backspace	BS (H <sup>c</sup> )	Move the cursor one column to the left. If the cursor is in column 0, it remains there.
	ESC A	Move the cursor up one row. If the cursor is in row 0, it wraps around to row 23.
	ESC B	Move the cursor down one row. If the cursor is in row 23, it wraps around to row 0.
	ESC C	Move the cursor right one column. If the cursor is in column 79, it wraps around to column 0 of the next row. If the cursor is in row 23, column 79, it wraps around to row 0, column 0.
	ESC D	Move the cursor left one column. If the cursor is in column 0, it wraps around to column 79 of the previous row. If the cursor is in row 0, column 0, it wraps around to row 23, column 79.
	ESC h	Move the cursor to the beginning of the first line of memory (excluding transmit-only fields in Format Mode.)
Home Up	ESC H	Move cursor to the beginning of the first line in display memory (including transmit only fields in Format Mode).
Home Down	ESC F	Move the cursor to the beginning of the line following the last data in display memory.
<b>Tabs</b>		
Tab	HT (I <sup>c</sup> ) or ESC I	Move the cursor forward to the next tab position.
Back Tab	ESC i	Move the cursor back to the previous tab position.
Set Tab	ESC 1	Place a tab at the current cursor column.


FUNCTION	CODE	DESCRIPTION
Clear Tab	ESC 2	Clear the tab at the current cursor column.
Clear All Tabs	ESC 3	Clear all tabs in display memory.
<b>Margins</b>		
Set Left	ESC 4	Set the left margin at the current cursor column.
Set Right	ESC 5	Set the right margin at the current cursor column.
<b>Display</b>		
	ESC*mq	Turn display on.
	ESC*mr	Turn display off.
Clear Display	ESC J	Clear display memory from the cursor position to the end of memory.
Clear to End of Line	ESC K	Clear current line beginning at the column containing the cursor.
Roll Up	ESC S	Roll the screen up one row (until the last row of memory is located at the top of the display). Cursor is stationary.
Roll Down	ESC T	Roll the screen down one row (until the first row of memory is located at the top of the display). Cursor is stationary.
Next Page	ESC U	Display the next 24 rows of memory (until the last row of memory is located at the top of the display). The cursor is moved to the first unprotected location on the new page when in forms mode.
Prev Page	ESC V	Displays the previous 24 rows of memory (until the first row of memory is located at the top of the display). The cursor is moved to the first unprotected location on the new page when in forms mode.
Memory Lock	ESC l	Turn on memory lock (overflow protect). Note that when Memory Lock is on, only screen relative addressing can be used.
	ESC m	Turn off memory lock. Refer to the User Manual for additional information on Memory Lock.

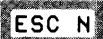
## Edit Operations


The terminal allows you to edit data displayed on the screen. This can be done by simply overstriking the old data. In addition, several edit operations are available. These edit operations are listed in table 2-2.

Table 2-2. Edit Operations

FUNCTION	CODE	DESCRIPTION
Insert Line	ESC L	The line containing the cursor and all lines below it are rolled down one line. A blank line is inserted where the line containing the cursor was. The cursor is moved to the left margin of the blank line.
Delete Line	ESC M	The line containing the cursor is deleted. The lines below the cursor are rolled up and the cursor is positioned at the left margin.
Insert Character	ESC Q	Turn on Insert Character Mode (and indicator). New characters will be inserted in the line at the current cursor position. Characters that are moved past the right margin are lost.
Insert Character with Wraparound	ESC N	Turn on Insert Character with Wraparound (indicator blinks). Characters extending beyond the right margin are wrapped to the next line. Refer to the next line. Refer to the <i>User Manual</i> for additional information.
	ESC R	Turn off Insert Character Mode (and indicator).
Delete Character	ESC P	The character at the current cursor position is deleted. Characters to the right of the cursor are moved one column to the left (see figure 2-2).
Delete Character with Wraparound	ESC O	The character at the cursor is deleted. Characters from the next lines are wrapped to the end of the current line (see figure 2-3).

**INSERT CHARACTER WITH WRAPAROUND.** You can insert characters with wraparound by sending ESC N. This will cause the  indicator to blink. While in this mode characters that overflow a line due to insertion are moved to the next line. Sending ESC R returns the terminal to normal operation.

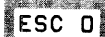
Turn on Insert with Wrap: 

Turn off Insert with Wrap: 

If the current line is full and additional characters are inserted, they will push characters from the end of the first line to the beginning of the next line. If the second line becomes full while the cursor is still in the first line, a blank line will be inserted between line one and two. The characters overflowing line one will then be entered on the new line.

## DELETE CHARACTER WITH WRAPAROUND.

When characters are deleted using ESC O, one character from the left margin on the next line is moved up to the right margin of the line containing the cursor. If the next line is blank, no wraparound is performed.

Delete Character with Wrap: 

When margins are used together with the Delete Character and wraparound operations, the characters to the right of the cursor are moved as shown in figures 2-3 and 2-4.

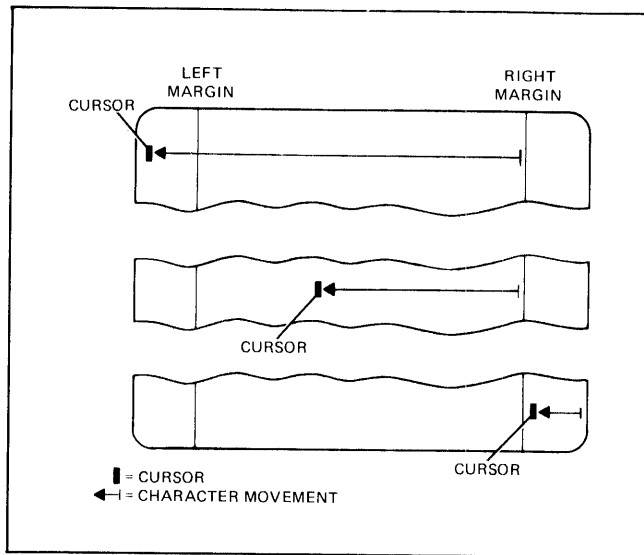


Figure 2-3. Character Delete With Margins

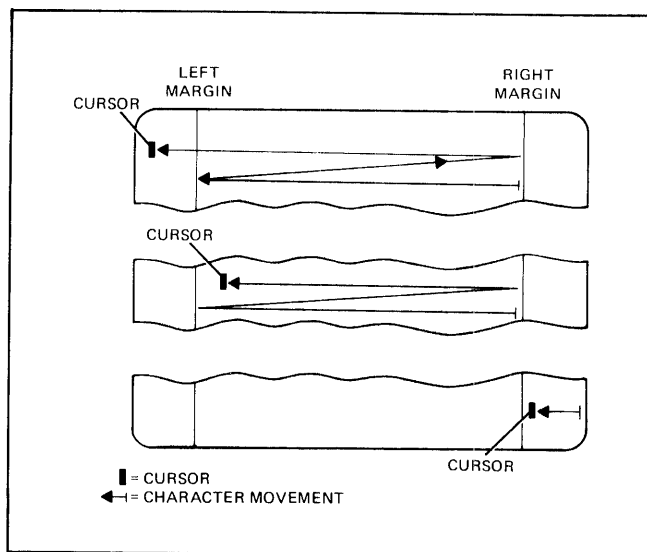


Figure 2-4. Character Delete With Wraparound

**MOVING TEXT BLOCK.** You can move blocks of text or data using Memory Lock.

**Example:** In the following text, move the paragraphs into the proper order. The current top of screen is row 1 of display memory.

Initial order:

- (Top of screen)
3. This is paragraph 3. It should be last in the group.
  2. This is paragraph 2. It should be second.
  1. This is paragraph 1. It should be first.
- (blank line)

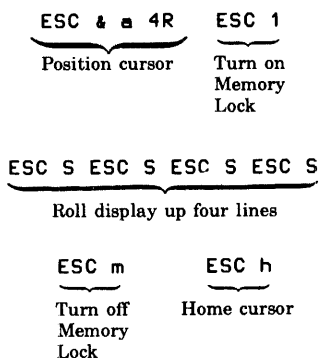
**Step 1.** Position the cursor in the first line of paragraph 2.

**Step 2.** Turn on Memory Lock.

**Step 3.** Roll up the display until the remaining paragraphs have rolled up under the cursor position and off the screen (4 lines).

**Step 4.** Turn off Memory Lock.

**Step 5.** Home the cursor.



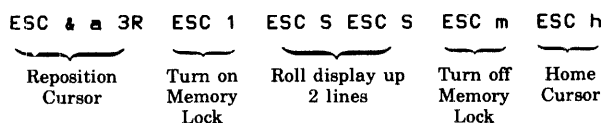
The display should appear as follows:

- (Top of screen)
2. This is paragraph 2. It should be second.
  1. This is paragraph 1. It should be first.
  3. This is paragraph 3. It should be last in the group.

**Step 6.** Now move paragraph 1 by positioning the cursor in the first line of paragraph 1 and turning on Memory Lock.

**Step 7.** Roll up the display until the cursor is in the first line of paragraph 3.

**Step 8.** Turn off Memory Lock and home the cursor. The paragraphs should now be in order.



## Forms Mode (Format Mode)

In Forms Mode the terminal prevents you from overwriting or transmitting data in protected fields. Forms Mode is normally entered under control of the computer or through commands recorded on a cartridge tape. Forms Mode is turned on by sending ESC W (the cursor is homed to the beginning of the first unprotected field). Normal operation is returned with ESC X (the cursor remains in its present position).

**PROTECTED FIELDS.** Fields can be protected so that displayed data cannot be overwritten or sent to a computer. When the terminal is placed in "Forms Mode" (Format Mode) all character positions on the screen are protected except those fields that have been specifically defined as "unprotected" or "transmit only".

**UNPROTECTED FIELDS.** Data can be written into unprotected fields in the normal manner. After reaching the end of an unprotected field, the cursor moves to the beginning of the next unprotected field. The tab functions can be used to move from one unprotected field to the beginning of the next unprotected field. ESC i causes the cursor to be positioned at the beginning of the previous unprotected field. Fields are defined as "unprotected" by using ESC [ at the start of the field. ESC ] or the end of the line is used to end the field.

In the following figure only the fields shown in white are protected or transmit only. Even if the operator moves the cursor to a protected field and types a character the cursor will move to the nearest unprotected field before displaying the character.

FORM #1876R					
Vendor Name	Address	City	State	Zip	
PACIFIC TOOL INC	1273 CRECENT WAY	SAN JOSE	CALIFORNIA	95131	
Voucher Date	Units	Purchase And Assembly Details		Post Ref.	Cost
07 16 1976	98	FINISHED STEEL CASTINGS		874738	65.88
03 19 1976	749	TAPE TRANSPORT BACKPLATES		875483	9753.88
02 28 1976	13	MILLED FLANGE ASSEMBLY		748563	877.44
19					
19					
INITIATED BY: H.C. DOUGLAS		DATE: 04 14 1976			

**Example:** Define columns 1 through 9 of line 3 as "Unprotected".

**Step 1.** Position the cursor at column 1 in line 3.

**Step 2.** Send ESC [.

**Step 3.** Move the cursor to column 10 of line 3.

**Step 4.** Send ESC ].

Now try turning on Forms Mode (ESC W) and sending data. Note that data can only be entered into the unprotected field. (Remember to turn off Forms Mode with ESC X.)

**TRANSMIT ONLY FIELDS.** It is often desirable to be able to return fixed data used as labels or headings to the computer. Transmit only fields are similar to protected fields except that they are sent to a computer along with the data that you enter. Normally data can only be entered in unprotected fields. But by positioning the cursor in the transmit only fields (using the cursor functions), you can also enter data into transmit only fields. The tab functions skip over transmit only fields. After reaching the end of the transmit only field the cursor moves to the beginning of the next unprotected field. Fields are defined as "transmit only" by using ESC { at the beginning of the field. ESC ] or the end of the line will end the field.

**Example:** Continue the previous example and define column 11 through 14 of line 3 as transmit only.

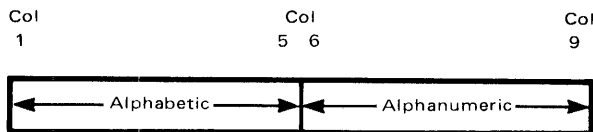
- Step 1.** Turn forms mode off (ESC X).
- Step 2.** Position the cursor at column 11 line 3.
- Step 3.** Send ESC {.
- Step 4.** Move the cursor to column 15 in line 3.
- Step 5.** Send ESC ].

Position the cursor in column 0 and turn on Forms Mode (ESC W). Try typing data. Note that the cursor moves over the Transmit Only field without entering data.

**DATA CHECKING.** While in Format Mode the terminal can test data in unprotected and transmit-only fields to make sure that it is numeric or alphabetic. If a field is defined as numeric and an alphabetic character is entered the terminal will beep and the keyboard will lock. This condition can be cleared by pressing **RETURN**. You can then continue entering data. Data checking fields are defined by beginning the field with one of the following sequences:

- ESC 6 — begin alphabetic field (A thru Z, a thru z, space only).
- ESC 7 — begin numeric field (space, 0 thru 9, -, +, ., and ,)
- ESC 8 — alphanumeric field (all keyboard characters)

**Example:** Define columns 1 through 5 on line 4 to be alphabetic and columns 6 through 9 to be alphanumeric.



- Step 1.** Turn off forms mode (ESC X).
- Step 2.** Position the cursor at column 1 of line 4.
- Step 3.** Send ESC [ to define the beginning of an unprotected field.
- Step 4.** Send ESC 6 to define an alphabetic field.

- Step 5.** Move the cursor to column 6 of line 4.
- Step 6.** Send ESC 8 to define normal alphanumeric data.
- Step 7.** Move the cursor to column 10 of line 4.
- Step 8.** Send ESC ] to end the unprotected field.
- Step 9.** Turn on forms mode (ESC W).

When a number is typed in the alphabetic field the keyboard is locked, the terminal beeps, and the cursor remains under the invalid character. Press the **RETURN** key to unlock the keyboard. The operator can then make the correct entry.

The numeric character may be left in the alphabetic field by moving the cursor to the next character position using the cursor control keys. This has the effect of overriding the data check.

**CAUTION**

Deleting characters while in Forms Mode can destroy the data checking field or alter the unprotected field length. Use **CLEAR DISPLAY** (ESC J) or **ENTER CLEAR DISPLAY** (ESC K) to delete information.

**TABBING.** Tabs are automatically set at the beginning of each unprotected field when Forms Mode is turned on; any tabs set previously are ignored. When Forms Mode is turned off, any previously set tabs are reinstated. Tabs cannot be set within any unprotected field.

**EDITING.** While in Forms Mode, the unprotected fields can be edited (inserting and deleting characters). The INSERT LINE and DELETE LINE functions are disabled in Forms Mode.

**Inserting Characters.** Characters may be inserted in any unprotected field by turning on Insert Character Mode (ESC Q). Characters received or typed are inserted at the cursor position. Characters moved out of the end of the unprotected field are lost.

Insert Character with Wraparound acts the same as insert character without wraparound explained above (the wraparound function has no effect).

**Deleting Characters.** Characters may be deleted in any unprotected field by the Delete Character function (ESC P). The character at the cursor position is deleted and all characters to the right of the deleted character in the field are moved left one column.

Delete Character with Wraparound acts the same as delete character without wraparound explained above (the wraparound function has no effect).

Characters may be deleted from the current cursor position to the end of the field by sending ESC K (clear to end of line). Also, characters may be deleted from the current cursor position to the end of the last field by sending ESC J (clear display).

**BUILDING THE FORM.** Appendix A contains a method for building forms using the cartridge tapes and soft keys.

**SENDING DATA TO THE COMPUTER.** Refer to "Block Mode" in Section V.

### Display Enhancements

The standard terminal can display data using inverse video (black on white). In addition, if your terminal has the 13231A Display Enhancement accessory you can also use half bright, underline, and blinking characters. Each character position on the screen can be displayed with various combinations of these features.

- **Half Bright** — characters are displayed at half intensity (grey).
- **Underline** — an underline is displayed below the normal character.
- **Inverse Video** — the screen is white and characters are black.
- **Blinking** — characters including the inverse video, underline, and half bright features blink.

The display enhancements are used by assigning one or more of them to a field. The selection sequence is:

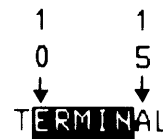
```
ESC & d <enhancement character>
```

The enhancement character (@, A through O) is used to select the combination of display enhancements to be assigned to the field. The following table lists the enhancement character for each of the combinations. The field is ended by selecting another enhancement, the end of the current line, or by ESC & d @.

	@	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
Underline					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 Enhancement	X															

**Example:** Define columns 10 through 14 of line 5 to be inverse video and blinking.

- Step 1.** Position the cursor at column 10 in line 5.
- Step 2.** Send ESC & d C
- Step 3.** Move the cursor to column 15 in line 5.
- Step 4.** Send ESC & d @ (this ends the enhancements). The field should be white.
- Step 5.** Send the word TERMINAL beginning in column 9 of line 5. It should appear as shown below. (If your terminal does not have the 13231A accessory installed the characters will not blink.)



### Alternate Character Sets

The terminal can display up to four different character sets. Each character set can contain up to 128 characters or symbols. In addition to the Math, Line Drawing, and Large Character sets available as options, you can create character sets tailored for special applications. Contact your nearest Hewlett-Packard Sales Office for additional information on special character sets.

Switching from one character set to another can be done on a character-by-character basis. For example, a character from the Math Symbol Set can be displayed next to characters from the Roman set. This is done by defining one or more character positions in a line to be from alternate character sets. (Each group of characters can be thought of as a field.)

#### NOTE


The following discussion assumes that the Math and Line Drawing character sets are present and are installed as alternate sets A and B respectively.

**SELECTING ALTERNATE SETS.** To use optional character sets, first select the character set to be used as the alternate. (With the terminal in its initial state, character set A is defined to be the alternate.) An alternate set is selected with the following sequence:

```
ESC ) <set>
```

where: set = @, A, B, or C



Note that if @ is used, the Roman or basic terminal set would be selected as the alternate. To find out which character set corresponds to @, A, B, or C, generate the test pattern by pressing the  key). This displays the order of the character sets as shown.

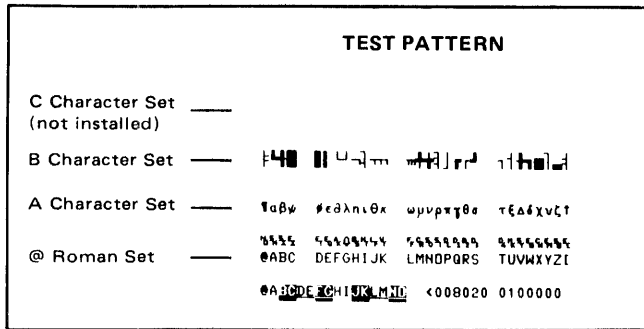


Figure 2-5. Character Set Locations

**USING ALTERNATE SETS.** Once the alternate character set is defined, you can switch from the Roman to the alternate set with a N<sup>c</sup> (SO).

The terminal automatically returns to the base or Roman set at the end of a line. To return to the base set within a line, send a O<sup>c</sup> (SI). This means that you must send another N<sup>c</sup> to turn on the alternate set if it extends to the next line.

**Example:** Define the Math Set as the alternate character set.

From the test pattern the Math Symbol Set is found to be the A alternate character set.

An alternate set is selected with the following sequence:

```
ESC ) A
```

To display AαBβ send the following sequence:

```
A Nc A Oc B Nc B
```

The screen should display AαBβ:

**AαBβ**

Once a field has been defined as from the alternate set the field moves with the display. To change to a different alternate character set another ESC ) <set> sequence must be sent.

Once a field in display memory has been defined as an alternate character field, it will continue to display alternate characters whenever data is written in the field until the terminal is reset or the portion of the line containing the alternate character set control code (N<sup>c</sup>) is deleted.

The Math Set is useful for applications requiring the use of equations or formulas. The elements of the optional Math Symbol Set are shown in figure 2-8. An example of the use of the Math Set is shown in figure 2-6.

$$\int_0^{\infty} \int_0^{\infty} \int_0^{\infty} \Psi * \begin{bmatrix} -h & \partial \Psi \\ 2\pi i & \partial t \end{bmatrix} dv = \int_0^{\infty} \int_0^{\infty} \int_0^{\infty} \Psi * E \Psi dv$$

Figure 2-6. Example Using the Math Set

The Large Character set allows you to create alphabetic characters that are three times the size of normal characters. The elements of the Large Character set are shown in figure 2-9. An example of how to use the Large Character Set to build the character "B" is shown in figure 2-7. Table B-3 in Appendix B shows the keys required to build each character. Appendix A gives a program to build each character.

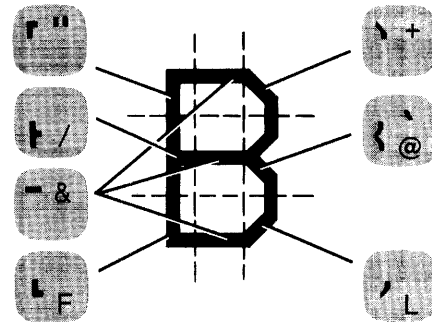


Figure 2-7. Example Using the Large Character Set

The Line Drawing Set provides a limited graphics capability. Simple line drawings and fairly complex forms for data entry applications can be generated. The elements of the optional Line Drawing Set are shown in figure 2-10. Figure 2-11 shows how the line drawing set can be used to build a data entry form.

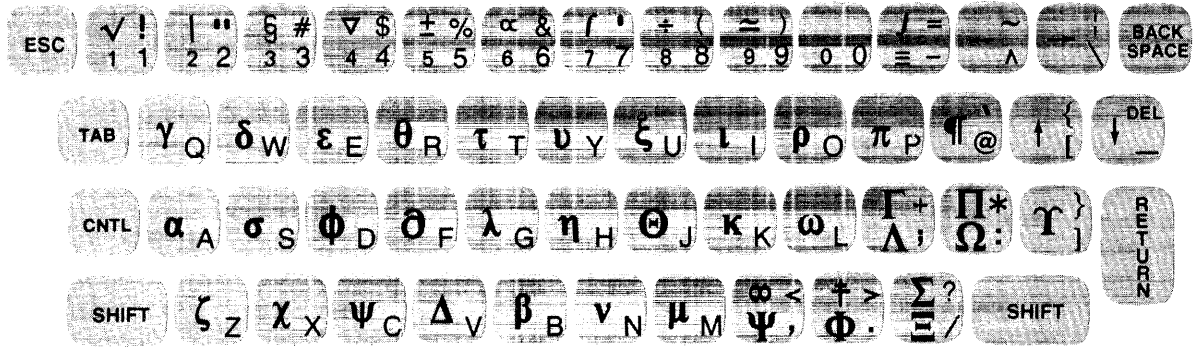


Figure 2-8. Math Set Elements

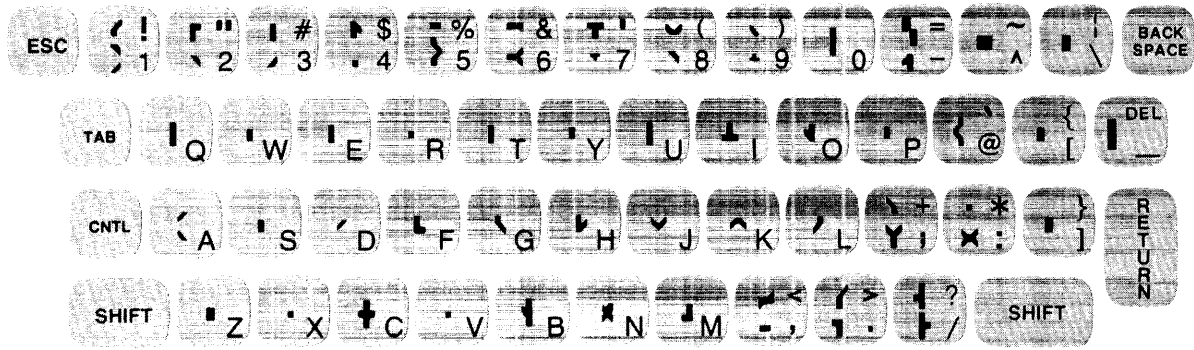


Figure 2-9. Large Character Set Elements

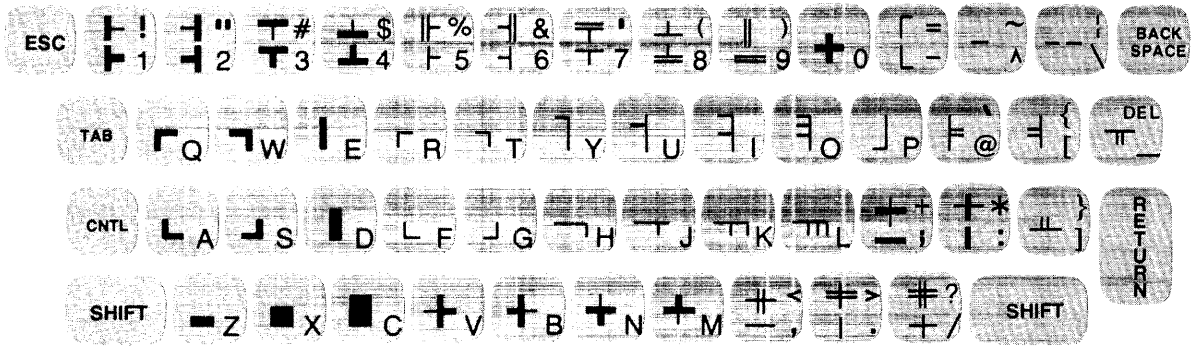


Figure 2-10. Line Drawing Set

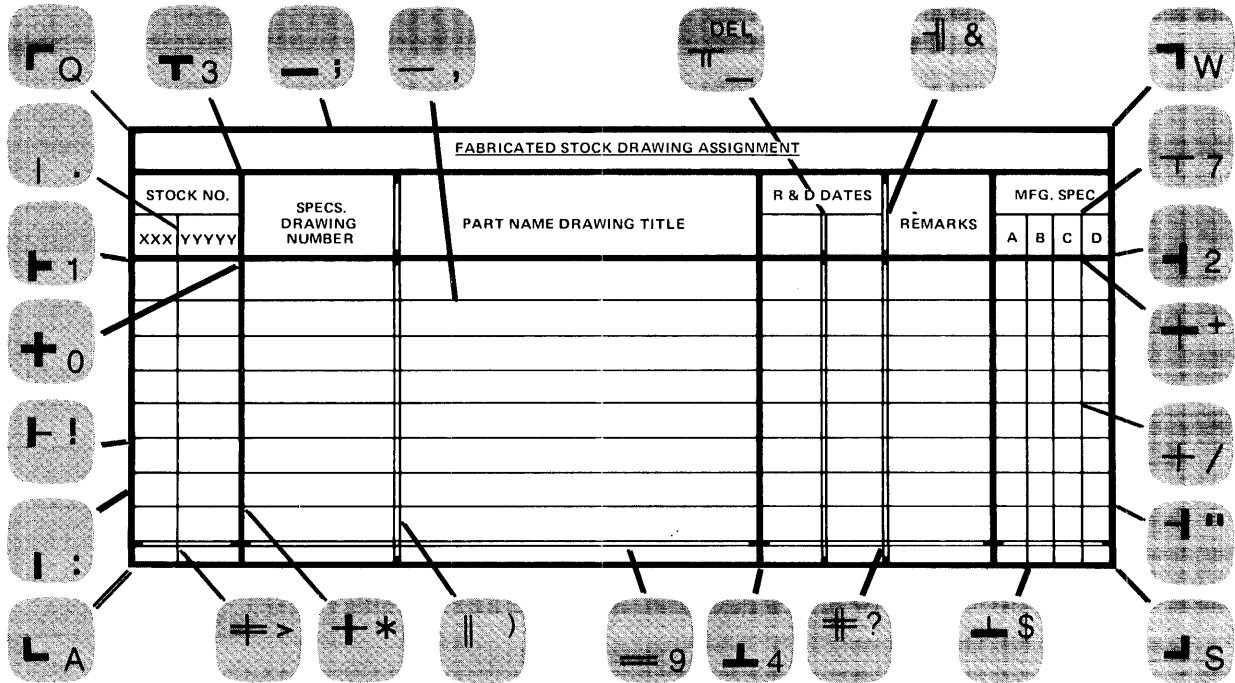


Figure 2-11. Sample Form

### TERMINAL CONTROL FUNCTIONS

The following paragraphs describe how to programmatically change most of the terminal's control settings and perform various other control functions. Refer to Section III, *Graphics Functions* for additional terminal settings. The settings consist of the following:

- Latching Keys **AUTO LF**, **BLOCK MODE**, **CAPS LOCK** and **REMOTE**
- Keyboard Interface Switch Settings (A-Z)

When the terminal is powered on or reset, the states of the various keyboard and internal switches are stored in memory. Most of these stored settings can be changed programmatically without physically changing the switch setting. (Note that if the terminal is reset the terminal will return to the physical setting.)

You can select a specific operating configuration from within the application program. This eliminates the problem of requiring the terminal operator to make the settings before continuing with the application program. It also allows individual programs or even subroutines to change terminal configuration for a specific function and then return the terminal to the original state before passing control back to the main program.

### Latching Keys

Four of the mechanically latching keys, **AUTO LF**, **BLOCK MODE**, **CAPS LOCK**, and **REMOTE** can have their electronic state changed programmatically. The escape code sequence is as follows:

```
ESC & k <state> <key> . . .
```

where:  
 <state> is { 0 (up)  
 or  
 1 (down)

<key> is { a (AUTO LF)  
 or  
 b (BLOCK MODE)  
 or  
 c (CAPS LOCK)  
 or  
 r (REMOTE)

The ESC & k is followed by one or more groups of state and key parameters. The state is a 0 or 1 to indicate that the key is to be up or down respectively. The key is a single letter a, b, c, or r. The groups can be in any order (**CAPS LOCK** can come before **BLOCK MODE**). The last key letter in the sequence must be capitalized to indicate the end of the sequence.

**Example:** Set **AUTO LF** up and **CAPS LOCK** down.

```
ESC & k 0 a 1 C
```

An invalid character (any character other than 0, 1, a, b, c, or r) will cause the entire sequence to be ignored. An improper setting, (0 b when the terminal is operating in a multipoint configuration) will cause only the invalid setting to be ignored. The reset of the sequence will be accepted. If the terminal is initialized while configured for multipoint operation, the **LOCK MODE** key will be read as down regardless of the switch's physical setting.

### Keyboard Interface Switches

The switches on the Keyboard Interface allow you to alter terminal operation for specific applications. Table 2-3 contains a summary of the switches and their function. A more complete description of the switches is given in Section VII.

The switch settings are made using the following sequence:

```
ESC & s <state> <switch> . . .
```

where:

<state> is  $\left\{ \begin{array}{l} 0 \text{ (closed)} \\ \text{or} \\ 1 \text{ (open)} \end{array} \right.$

<switch> is  $\left\{ \begin{array}{l} \text{A through Z} \\ \text{less I and O} \end{array} \right.$

An invalid character in the sequence will cause the entire sequence to be ignored. The sequence must be terminated with an upper case switch character. A full reset will cause the terminal to return to the physical settings of the switches. Switches S and T cannot be changed if the terminal is configured for Main Channel protocol.

**Example:** Set switches A, B, and D open and switch C closed.

```
ESC & s 1 a 1 b 1 d 0 C
```

In certain operating configurations (i.e. multipoint), some switch settings cannot be changed. If attempted, the new setting for the switch will be ignored.

### PROGRAMMABLE SOFT KEYS

The terminal has 8 programmable keys **f1** — **f8**. In addition, the **RETURN** key can also be assigned a string value. The **RETURN** key is addressed as the f0 key in escape sequences. All 9 keys can be used by the operator or triggered from a program. Each key can be assigned a string of up to 80 characters. The keys can be defined to be used at the terminal only (L), transmitted to the computer only (T), or to be treated as normal keyboard input (N). The keys can be programmed with escape code sequences to control or modify terminal operation. The keys can be used in appli-

Table 2-3. Keyboard Interface Switch Summary

SWITCH	POINT-TO-POINT FUNCTION	MULTIPOINT FUNCTION
A	Function key transmission	same
B	Space overwrite latch	same
C	Cursor end-of-line wraparound	same
D	Block mode (Line/Page)	same
E	Paper tape mode	same
F	Fast binary read	(not used)
G	Block transfer handshake	(not used)
H	Inhibit DC2	(not used)
J	Auto terminate	same
K	Clear terminator	same
L	Self-test inhibit	same
M	Reverse CNTL key effect on INSERT CHAR and DELETE CHAR keys	same
N	Escape code transfer to printer	same
P	Compatibility Mode (scaled)	same
Q	Compatibility Mode (unscaled)	same
R	Circuit Assurance	Set trailing PAD
S	Main Channel Protocol	(not used)
T	Main Channel Protocol	Output block size
U	CPU break	Output block size
V	Carrier Detect	Continuous carrier
W	Data Comm self-test enable	same
X	Data speed select	same
Y	Transmit LED	same
Z	Parity	same

cation programs to create "menu" lists of special commands or in the case of **RETURN** to create a terminator for communications protocols.

### Controlling the Soft Key Display

You can cause the current soft key assignments to be displayed using the following escape sequence:

Display soft keys: **ESC j**

This will also allow the terminal operator to enter new key assignments from the keyboard. Procedures for entering new soft key assignments by escape sequences are given next in this section. The soft key display is in the following format:

```
F # type
string
```

where: "#" is the key number (0-8)

"type" is  $\left\{ \begin{array}{l} \text{L (local only)} \\ \text{N (normal keyboard operation)} \\ \text{T (transmit only)} \end{array} \right.$

"string" is any series of up to 80 characters

The soft key assignments are displayed in place of the normal screen display. Data in display memory is not lost. (Note — if display memory is full, assigning characters to soft keys may cause some display data to be lost.) When the key assignment is completed and the terminal returned to normal operation, the old display is returned to the screen. Normal operation is restored using the following escape sequence:

Remove soft key display:



## Defining Soft Keys

The key assignment operation displays the current key assignments in format mode. The attribute and string fields are unprotected allowing the operator to enter new values. In addition, the values are tested during input to make sure that only valid parameter values are used. When the terminal is initialized the soft keys are assigned default values as follows:

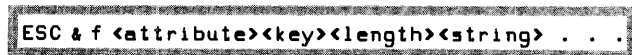
	=	↵ (normal)
	=	ESC p (transmit only)
	=	ESC q (transmit only)
	=	ESC r (transmit only)
	=	ESC s (transmit only)
	=	ESC t (transmit only)
	=	ESC u (transmit only)
	=	ESC v (transmit only)
	=	ESC w (transmit only)

### NOTE

If the memory is full, adding characters to the soft key string may delete lines in normal display memory.

The soft keys can be loaded by the terminal operator (refer to the *User Manual*) or under program control. It is not necessary to display the current key assignments to enter new ones. Soft Key assignments can be made directly using the following escape sequence:

Soft key assignment sequence:



where:

$$\langle \text{attribute} \rangle = \left. \begin{array}{l} 0 \text{ (normal)} \\ 1 \text{ (local only)} \\ 2 \text{ (transmit only)} \end{array} \right\} \text{ a (0 is default)}$$

$\langle \text{key} \rangle$  is 0-8 k (1 is default)

$\langle \text{length} \rangle$  is 1-80 l (1 is default)

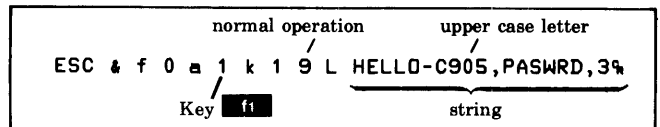
$\langle \text{string} \rangle$  is the character sequence to be assigned

If a carriage return (CR) is included in the string portion of the soft key definition, the CR will be translated to a CR,LF if the key is down. The CR used in assigning values to keys f1—f8 is unaffected by the string assignment made to the CR key. The CR key will only generate the CR character while the soft key assignments are displayed on the screen.

If the transmit only attribute (2) is used, the key will have no effect unless the terminal is set for remote operation.

Also, it may invoke a Block transfer handshake and append the appropriate terminator to the string. (See Appendix C, figure C-2, sheet 2.) The key assignment escape sequence must be terminated with an upper case character.

**Example:** Assign "HELLO-C905,PASWRD,3" to the key.



After the key assignment in the previous example has been made, a display of the key assignments would appear as follows:

```

 N

 N
HELLO-C905 , PASWRD , 3
 T

 T

 T

 T

 T

 T

 T

    
```

### CAUTION

Do not include line feeds (LF's) or block terminators (RS/GS) in soft key string (other than at the end of the string) if the soft key assignment is to be stored on cartridge tape from the soft key display.

### Triggering Soft Keys

Soft keys can be triggered from a program. A soft key cannot be assigned a value and then triggered in the same sequence. Only one soft key can be triggered in the sequence.

Soft key trigger sequence: `ESC & f <0-8> E`

where: <0-8> is the soft key number

RETURN	= 0		
f1	= 1	f5	= 5
f2	= 2	f6	= 6
f3	= 3	f7	= 7
f4	= 4	f8	= 8

For example, to trigger the **f1** key the following sequence would be used:

`ESC & f 1 E`

### Soft Key Applications

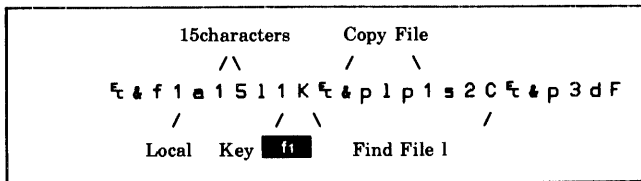
There are many applications of the soft keys. One application is the creation of "menu" operations. For example, the keys can be loaded with the escape sequences to find and read various files on a tape cartridge.

**Example:** Program the soft keys to find a file on the left tape drive and copy the file to the display. This can be done using the following technique:

`ESC & pnp1=2C ESC & p3dF`

where: n = the number of the file to be read

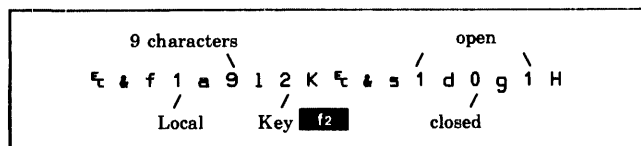
To program the **f1** key to find and display file number 1:



The same procedure could be used for the remaining function keys.

Another application would be to change the control settings on the Keyboard Interface to configure the terminal for use with different computer systems.

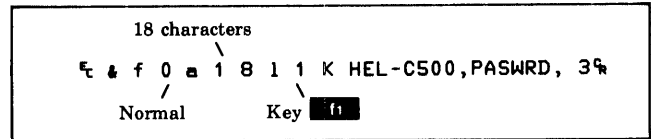
**Example:** To program the **f2** key to change the settings of Switches D, G, and H to open, closed, and open, the following sequence could be used:



The soft keys can be used to hold log-on, program control, and log-off messages. This makes it easier for the terminal operator to use unfamiliar computer systems.

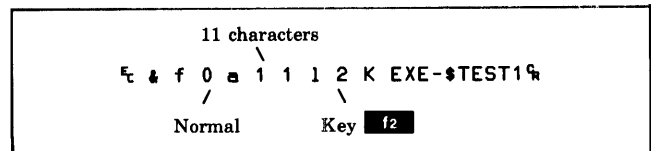
**Example:** A sample key assignment to log-on to the Hewlett-Packard 2000 Series Timeshare System might be as follows:

`f1 = HEL-C500,PASWRD,3`



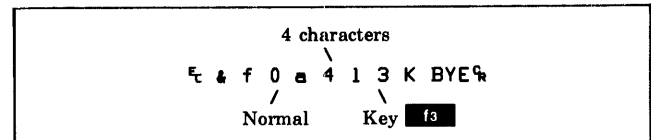
A second key could be used to call and execute a library program:

`f2 = EXE-$TEST1`



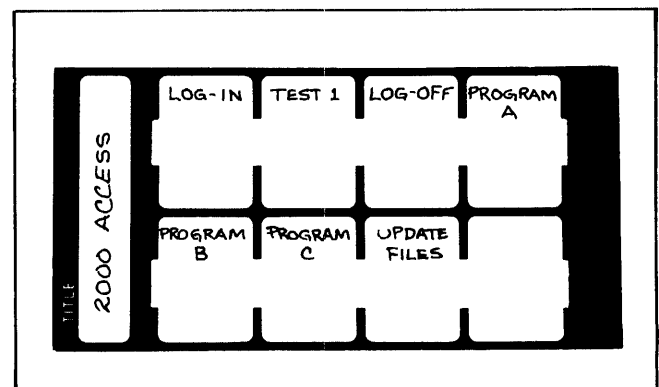
A third key could be used to log off of the system:

`f3 = BYE`



### Using Soft Key Labels

The terminal comes with 5 blank soft key templates. These templates can be labeled with the functions assigned to the soft keys. The template can be dropped over the function key group for easy reference.



## ADDITIONAL CONTROL FUNCTIONS

In addition to the control settings there are several control operations that can be controlled programmatically. These control functions are as follows:

- Bell — G<sup>c</sup>
- Send Display — ESC d
- Wait — ESC @
- Keyboard Disable — ESC c
- Keyboard Enable — ESC b
- Reset Terminal (Soft) — ESC g
- Reset Terminal (Full) — ESC E
- Turn On Monitor Mode — ESC y
- Terminal Self-Test — ESC z
- Data Comm Self-Test — ESC x
- Modem Disconnect — ESC f
- Program Down Loading — ESC & b or ESC & c

### Bell

The G<sup>c</sup> character causes the terminal to “beep”. A beep is automatically generated at the end of each unprotected field in format mode and during normal operation as the cursor passes within eight positions of the right margin.

### Send Display

The ESC d sequence causes the terminal to send a block of display memory data to the computer. The data sent depends on the Line/Page setting of Keyboard Interface switch D and whether the terminal is in format mode or not.

Data is transmitted beginning at the current cursor position. If the terminal is strapped for page, data is transmitted until the end of the current display. If strapped for line, transmission stops at the end of the current line for non format mode or at the end of the current field if in format mode.

### Wait

The terminal can be made to pause for approximately 1 second by sending it an ESC @. Multiple commands can be used to obtain any desired time period.

### Keyboard Disable/Enable

The terminal keyboard can be locked by sending an ESC c. It must then be unlocked by sending an ESC b or by pressing the RESET TERMINAL key.

## Reset Terminal

A programmatic “Soft Reset” can be made by sending an ESC g to the terminal. A “Full Reset” can be made using ESC E.

**Soft Reset (ESC g).** A soft reset results in the following:

1. Any error messages present are cleared, the normal display is returned and the keyboard is unlocked.
2. If DISPLAY FUNCTIONS is enabled, it is turned off.
3. Incomplete device selections are cleared. (Previous selections are retained.)
4. If the terminal is set for REMOTE, RECORD operations are ended.
5. Device operations (tape or printer) are stopped. If a tape drive was moving at the time the command was received, the tape will be rewound. In addition, if the tape drive was recording data, an end-of-data mark will be recorded before the tape is rewound.
6. Current transmission of data stops. Data waiting to be sent to the computer is not sent. Partial messages from the computer are lost. The data communications facility is re-initialized.
7. All keyboard lights are turned on for 0.5 seconds.

**Full Reset (ESC E).** A full reset has the same effect as turning power on and consists of the following:

1. The screen and memory are cleared, then TERMINAL READY is displayed. Format mode, display functions, and all programmable functions including the function keys (f0 — f8) are turned off or set to their default values.
2. Device assignments are set to their default values, tapes (if present) are rewound to their load point. (An end-of-data mark is not written.)
3. All graphics functions are set to their default values.

### NOTE

The CPU must wait 200 milliseconds after issuing ESC E before sending additional data.

## Monitor Mode

Monitor Mode can be turned on by sending ESC y. Refer to Section V for a discussion of Monitor Mode.

## Self-Test

The Terminal Self-Test can be executed by sending an ESC z. The Data Comm Self-Test can be executed by sending an ESC x. Descriptions of the self-tests are given in Section VII.

## Modem Disconnect

The terminal can be directed to "hang up" the modem by sending an ESC f. The terminal does this by lowering the CD (Data Terminal Ready) line for 1 second if 13260B data comm is used or 10 seconds if the terminal is configured for multipoint.

## Program Down Load

The ESC & b and ESC & c sequences allow special diagnostic programs to be loaded into the terminal and executed. The escape sequence must precede the program to be loaded. This function can be used by HP diagnostics only. The ESC & c functions the same as ESC & b except that the LOADER message is not displayed.



# GRAPHICS CONTROL FUNCTIONS

SECTION

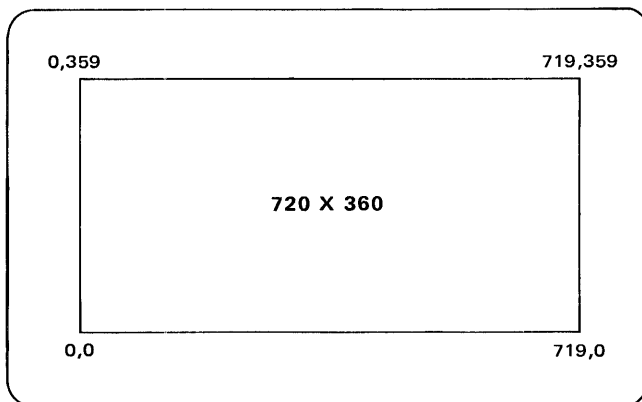
III

## INTRODUCTION

This section contains a description of the terminal's graphics functions and how they are used. The information and examples are intended for use in developing programs to control the graphics functions. Additional information on how to use "stand alone" graphics features such as AUTO PLOT is contained in the User's Manual.

## GRAPHICS DISPLAY

You can display graphics data by addressing points in a 720 by 360 array.



The graphics and alphanumeric data are displayed in the same area on the screen but are stored in separate RAM memories. This allows you to read or modify graphics and alphanumeric data separately.

### NOTE

Display enhancements (blinking, inverse video, and half-bright) will affect the display of graphics data. The display enhancements affect alphanumeric character positions on the screen. Since the graphics data is displayed using the same screen dots as the alphanumeric data, it is also affected by the enhancements. The data in the graphics display memory is unchanged.

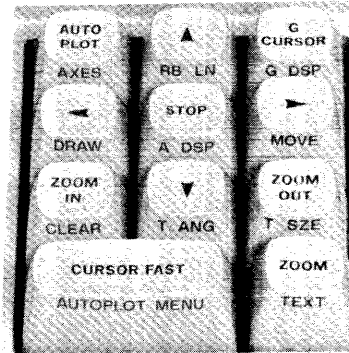


Figure 3-1. Location Of Graphics Keys

## KEYBOARD GRAPHICS FUNCTIONS

All of the graphics function commands can be entered from the terminal keyboard by the operator. Most of the functions are available through a special set of graphics control keys located to the right of the normal ASCII character set (see figure 3-1). Table 3-1 contains a list of the keys and a description of their functions. These keys can be used in both local and remote operation. This allows a combination of operator and program control of graphics functions to be used to make maximum use of the terminal's capabilities. Detailed information for using the graphics control keys is contained in the User's Manual.

Each key controls two functions. Pressing the key alone executes the command labeled on top of the key. Pressing both the function and shift keys executes the command labeled on the front of the key. The functions labeled on the front of the keys may permanently alter the display (ie. CLEAR). These functions are selected by pressing the SHIFT key and the function key. This makes a mistake less likely. The only graphics keys that repeat are those controlling zoom and the graphics cursor.

## PROGRAMMABLE GRAPHICS FUNCTIONS

Graphics functions are controlled by parameterized escape sequences. All graphics escape sequences begin with ESC \*. The third character, always lower case, selects the type of graphics sequence. Table 3-2 lists the types of graphics sequences. For example, ESC \* p specifies a plotting sequence.

Table 3-1. Graphics Control Keys


























KEY	DESCRIPTION	KEY	DESCRIPTION
	Turns autoplot on. If plot from display has been selected, or a point count specified, autoplot will turn itself off. Otherwise, the STOP key must be used to terminate autoplot. (See Autoplot.)		Toggles the graphics display, to inhibit the graphics image without erasing.
	Toggles the graphics cursor on and off.		Toggles the alphanumeric display.
	Move the graphics cursor. More than one can be pressed for diagonal motion.		Erases the graphics image memory.
	Speeds up the graphics cursor if pressed in conjunction with the cursor keys. The rate returns to normal when released.		Draws a vector from the current pen position to the graphics cursor. Only works if the cursor is on.
	Toggles zoom mode. When zoom is turned on, the area about the graphics cursor is magnified by the amount set by the   keys. Moving the cursor changes the zoomed area.		Moves the pen to the graphics cursor without drawing a vector. The graphics cursor must be on.
	Increments the zoom magnification.		Selects the graphics image memory as the destination for all text. Characters entering from the keyboard, datacomm, or tape are drawn as vectors in the graphics memory, using the current size and angle as specified by the  and  keys. If the terminal is not in Compatibility Mode the drawing mode is set to jam pattern to allow for backspacing and retyping of characters. If the terminal is in scaled Compatibility Mode the drawing mode is unchanged. (Both Drawing Mode and Compatibility Mode are discussed later in this section.)
	Decrements the zoom magnification.		The graphics cursor indicates the position of the next character. Moving the graphics cursor resets the start of line point. Carriage return, line feed, and backspace work as expected even on inverted text. The  key terminates this mode.
	Turns autoplot and graphics text mode off. (See Autoplot.)		
	Key held down selects the alternate key functions.		Increases the character size from 1 to 8X. The smallest character is a 5 by 7 matrix in a 7 by 9 cell. Increasing the size makes the dots bigger; the character is still drawn as a 5 by 7 matrix.
	Draws the axes, tic marks, and labels specified in the autoplot menu. (See Autoplot.)		Sets the character orientation (multiples of 90 degrees) and turns slant on or off.
	Toggles the menu for autoplot parameters. (See Autoplot.)		
	Toggles the rubber band line connecting the current pen position and the graphics cursor.		

Table 3-2. Summary of Graphics Sequence Types

ESCAPE SEQUENCE	DESCRIPTION
Esc * a	Autoplot
Esc * b	Raster Dump (data transfer)
Esc * d	Display Control
Esc * l	Labeling
Esc * m	Drawing Mode
Esc * p	Vector Plotting
Esc * r	Raster Dump (device control)
Esc * s	Graphics Status
Esc * t	Compatibility Mode

Subsequent characters in the control sequence are read as either parameters or commands, depending on the location of the character in the ASCII table.

BIT 7 6 5 4 3 2 1	0 0 0 0	0 0 0 1	0 1 0 0	0 1 1 0	1 0 1 0	1 0 1 1	1 1 1 1
0000	NUC NUL	DLE DLE	SP 0	@ @	P P	p p	
0001	SOH SOH	BC1 BC1	! !	A A	Q Q	a a	q q
0010	STX STX	DC2 DC2	" "	B B	R R	b b	r r
0011	ETX ETX	DC3 DC3	# #	C C	S S	c c	s s
0100	EDT EDT	DC4 DC4	\$ \$	D D	T T	d d	t t
0101	ENG ENG	NAK NAK	% %	E E	U U	e e	u u
0110	ACK ACK	SYN SYN	& &	F F	V V	f f	v v
0111	BEL BEL	ETB ETB	' '	G G	W W	g g	w w
1000	BS BS	CAN CAN	( (	H H	X X	h h	x x
1001	HT HT	EM EM	) )	I I	Y Y	i i	y y
1010	LF LF	SUB SUB	* *	J J	Z Z	j j	z z
1011	VT VT	ESC ESC	+ +	; ;	K K	 	k k
1100	FF FF	FS FS	, ,	L L	 	 	 
1101	CR CR	OS OS	- -	= =	M M	] ]	m m
1110	SQ SQ	RS RS	. .	> >	N N	^ ^	n n
1111	SI SI	US US	/ /	? ?	O O	o o	DEL DEL

		Parameters	Commands
BIT	7 6 5 4 3 2 1		
	0 0	Control Code	
	0 1	Parameter	
	1 0	Command and Terminate Sequence	
	1 1	Command and Continue Sequence	

## Control Codes

Control codes are generally ignored, with the exception of ESC. If an ESC character is detected and the previous graphics control sequence has not been properly terminated with a "Z" or some other valid capital character, the ESC will cause the execution of the previous sequence to be terminated. The new escape sequence will then be executed.

## Commands

Graphics commands come from columns 4-7 of the ASCII table, the upper and lower case letters (A-Z and ^). Both upper and lower case commands execute the same function. Upper case letters terminate the sequence and cause it to be executed. You can use more than one command in a sequence.

Graphics sequences can be any length. (The terminal ignores CR and LF characters in the middle of graphics sequences.) For example, to plot a figure containing 100 points the escape sequence could appear as follows:

Esc \* p a <x1,y1> . . . <x100,y100>Z

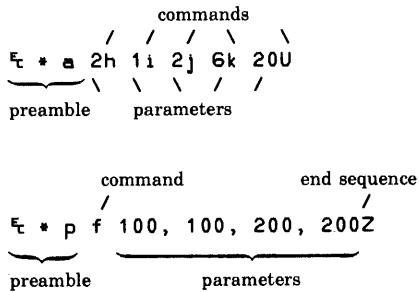
This could cause problems if an error occurs and the system tries to report it in the middle of a long sequence. Since most systems use upper case characters for messages, the first character of the message would end any graphics sequence that might be in progress. Letters that have not been assigned a function for a particular graphics sequence are treated as NOPs and if they are lower case, are ignored. If upper case, they will end the sequence. The letter z has been defined as a NOP in all sequences so that a capital Z can always be used to end a graphics escape sequence.

## Parameters

Parameters come from columns 2 and 3 of the ASCII table (SPACE through ?). Most parameters are simply the ASCII numeric characters used to represent data coordinates or to select one of several settings. Binary formatted data is generated by appending the bits 0 1 to five bits of binary data. Note that in binary formats, spaces are treated as data and are not ignored or used as delimiters. Both ASCII and binary data formats are described later in this section.

Parameters precede their associated commands (postfix notation). The most frequently used parameters are vector data. Refer to the discussion of Vectors for additional information on parameters used to define vector operations.

Examples:



The programmable graphics functions are organized into five major groups.

- Graphics Display Control
- Plotting
- Graphics Text
- Autoplot
- Compatibility Mode

The remainder of this section contains descriptions of each of these functional groups.

### GRAPHICS DISPLAY CONTROL

Graphics display control is made up of the functions used to control the graphics cursor, the portion of the graphics memory that is currently being displayed, or the state of the graphics memory. These functions are as follows:

- Graphics Cursor Control
- Zoom
- Graphics Memory Control

Table 3-3 lists the escape sequences for each of the graphics display control functions.

Table 3-3. Graphics Display Control Functions

FUNCTION	CODE	DESCRIPTION
<b>Graphics Cursor Control</b>		
Cursor On	ESC * d k	Turn on the graphics cursor.
Cursor Off	ESC * d l	Turn off the graphics cursor.
Move Absolute	ESC * d < x , y > o	Position the graphics cursor.
Move Relative	ESC * d < x , y > p	Position the graphics cursor.
<b>Zoom</b>		
Zoom On	ESC * d g	Turn on the zoom function.
Zoom Off	ESC * d h	Turn off the zoom function.
Zoom Size	ESC * d < s i z e > i	Set the zoom size.
Zoom Position	ESC * d < x , y > j	Set the zoom position.
<b>Graphics Memory Control</b>		
Clear Memory	ESC * d a	Turn off all dots in graphics memory.
Set Memory	ESC * d b	Turn on all dots in graphics memory.
Display On	ESC * d c	Enable the graphics display.
Display Off	ESC * d d	Inhibit the graphics display.

### Graphics Cursor Control

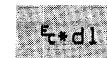
A separate graphics cursor is available for use in locating points in the graphics display. The graphics cursor is used by the terminal operator to input position data or to interact with a graphics application program.

**GRAPHICS CURSOR ON/OFF.** The graphics cursor is initially off (power on or full reset). Turning the cursor on or off does not effect the data in graphics memory. (The graphics cursor is also turned on when graphics text operation is enabled.)

Graphics Cursor On:

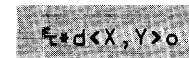


Graphics Cursor Off:

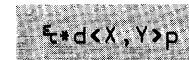


**GRAPHICS CURSOR POSITIONING.** The graphics cursor is initially at position (0,0) after power on or a full reset. The cursor can be positioned (even if it is not turned on) using either absolute or relative coordinates. In the following sequences X and Y give the new cursor position. Refer to Vectors for a discussion of absolute and relative coordinates.

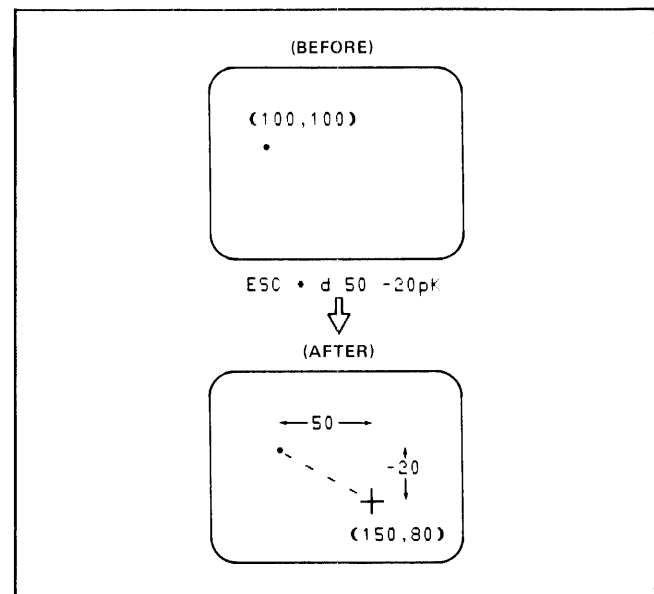
Position Graphics Cursor Absolute:



Position Graphics Cursor Relative:



**Example:** The cursor is currently at position 100,100 and off. Move it 50 units to the right and 20 units down from its current position and turn it on.



## Zoom

You can display a portion of the graphics memory data at increased size. The zoom size (magnification) settings are from 1 to 16 times. The data in the graphics memory is not lost or changed. As much of the graphics memory as will fit on the screen at the new size is displayed. The effect is similar to "windowing". The portion of the graphics memory to be centered on the screen can be changed allowing you to "pan" through the entire graphics memory at the magnified setting.

**ZOOM SIZE.** The zoom size is initially set to 1 after power on or a full reset. The size can be set to one of sixteen sizes (1-16). The magnified data is not displayed until the zoom function is turned on.

Set Zoom Size:

```
Ⓢ * d <size> i
```

where: **<size>** is 1-16

**ZOOM POSITION.** The zoom position is initially the graphics cursor position. It can be set to any position in graphics memory using ASCII absolute coordinates. The selected data is not displayed until the zoom function is turned on.

Set Zoom Position:

```
Ⓢ * d <X, Y> j
```

If multiple images are being displayed by changing the zoom position to display only a portion of the screen at a time, it is possible to insert a delay of approximately 16 ms between the images (frames) by inserting the "zoom on" command between positioning commands. (Refer to Inserting Delays In Graphics Operations.)

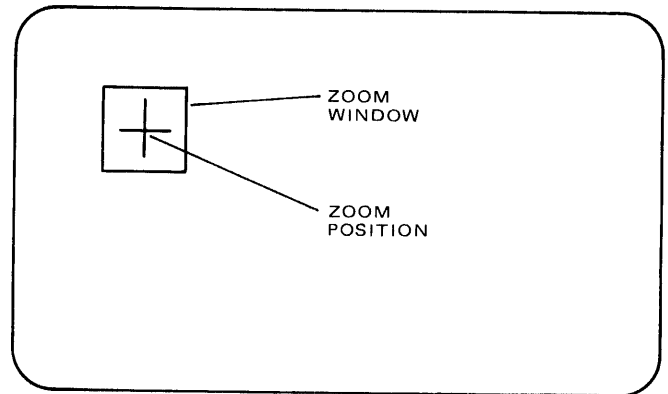
**Example:** Step the zoom position between (50,50), (50,250), (250,250), and (250,50) at 16 ms intervals.

```
Ⓢ * d 50,50 j g 50,250 j g 250,250 j g 250,50 j g 50,50 j
```

**ZOOM ON/OFF.** Once a zoom size and position are selected the data is displayed by turning on the zoom function. If the cursor is outside of the zoom window, turning on the zoom function causes the graphics cursor to be moved to the zoom position.

Data displayed on the screen while the zoom function is on can be modified using the graphics memory set and clear commands. Most of the data in the graphics memory that is not being displayed is unaffected by the set and clear memory commands (see the note in the following example) but can be modified using the plotting commands described later in this section (refer to Plotting Commands).

Moving the graphics cursor while zoom is turned on will cause the display to "pan" across the data in the graphics memory. If the graphics cursor is moved beyond the edge of the "zoom window" the window will move with the cursor. This causes the zoom position to change accordingly.



Note: Up to 16 points to the left and right of data displayed in the zoom window may be affected by the set or clear commands.

**Example:** Set the zoom size to 8, center the zoom window at 100,200 and turn on the zoom function.

```
Ⓢ * d 8 i 100 200 j g
```

## Graphics Memory Control

The graphics display can be turned on or off or the entire memory can be set to all ones (dots on) or all zeros (dots off).

**GRAPHICS DISPLAY ON/OFF.** The graphics display and graphics cursor can be turned on or off. The data in the graphics memory is unaffected.

Graphics Display On:

```
Ⓢ * dc
```

Graphics Display Off:

```
Ⓢ * dd
```

**GRAPHICS DISPLAY SET/CLEAR.** The graphics data currently displayed on the screen can be set to all ones or cleared to all zeros. When used together with the zoom function this function can be used to clear or set blocks of the graphics memory.

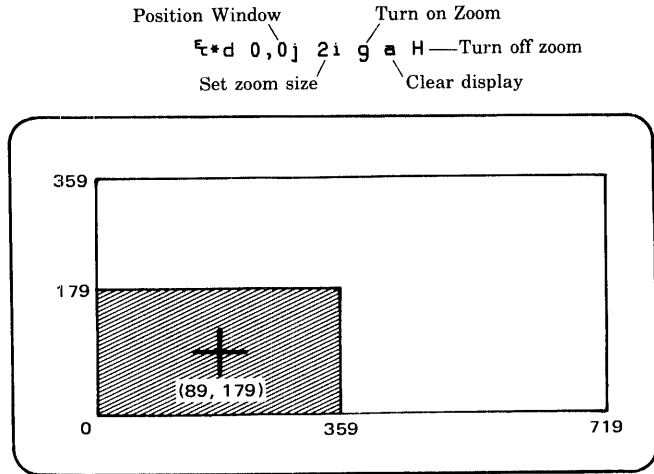
Clear Graphics Memory:

```
Ⓢ * da
```

Set Graphics Memory:

```
Ⓢ * db
```

**Example:** Clear the lower left portion of the graphics display.



### Inserting Delays In Graphics Operations

Certain graphics operations are executed only at the end of a frame (drawing of the display). These operations take approximately 16 ms and can be used as delays to slow down or synchronize changes in the display. Refer to Set Zoom Position for an example.

OPERATION	CODE
Graphics Display On	E * dc
Graphics Display Off	E * dd
Alphanumeric Display On	E * de
Alphanumeric Display Off	E * df
Zoom On	E * dg
Zoom Off	E * dh
Set Zoom Size	E * di
Set Zoom Position	E * dj

### GRAPHICS DRAWING MODE PARAMETERS

There are several drawing parameters that can be set to allow a wide variety of drawing capabilities. These parameters select whether data will be stored in the graphics memory as 1's or 0's, define line or area patterns to be used when drawing vectors, position the relocatable origin, and define graphics text settings.

Graphics drawing control sequences begin with E \* m followed by one or more of the drawing parameters. Table 3-4 lists the mode control commands.

### Drawing Modes

Vectors can be drawn by setting, clearing, or complementing the data in the graphics memory. Normally the memory is cleared and vectors are drawn by setting selected bits to make white lines on a dark screen. If instead you want black vectors on a white screen, you can begin by

setting memory (refer to the Set Memory command), select a clear or complement line type and draw dark vectors (refer to the example that follows). Figure 3-2 illustrates the various drawing modes.

Table 3-4. Graphics Mode Commands

E * m <parameters>	
PARAMETERS	DESCRIPTION
a	select drawing mode
b	select line type
c	define line pattern
d	define area pattern
e	area fill, absolute
f	area fill, relocatable
j	set relocatable origin
k	set relocatable origin to pen position
l	set relocatable origin to cursor position
m	set graphics text size
n	set graphics text direction
o	turn on character slant
p	turn off character slant
q	set text origin
r	set graphics defaults
z	NOP

Set Drawing Mode:

```
E * m <parameter> a
```

where: <parameter> is

- 0 Graphics memory not changed.
- 1 Clear (turn off graphics bits).
- 2 Set (turn on graphics bits).
- 3 Complement (toggle the graphics bits).
- 4 Jam (turn bits on or off according to the data).

**CLEAR MODE.** Clear mode causes selected display bits to be turned off. The "selected bits" are those that are "on" in the line pattern. If a solid line type (the default) has been selected, all of the bits in a vector will be selected. In clear mode this means that all of the dots making up a vector will be turned off. This allows you to draw dark vectors on a white background. Only those bits that are on in the pattern are cleared. Bits that are off in the pattern do not affect the display.

**SET MODE.** Set mode is similar to clear mode except that the selected bits are turned on instead of off. Only the bits that are on in the line type are affected.

**COMPLEMENT MODE.** Complement mode causes the selected display bits to change state (on to off, off to on). Again only those bits that are on in the line type or pattern are affected.

**JAM MODE.** Jam mode differs from the other modes in that both the bits that are on in the line type or pattern and the bits in the pattern that are off affect the display. Jam mode has the effect of overlaying the display with the pattern.

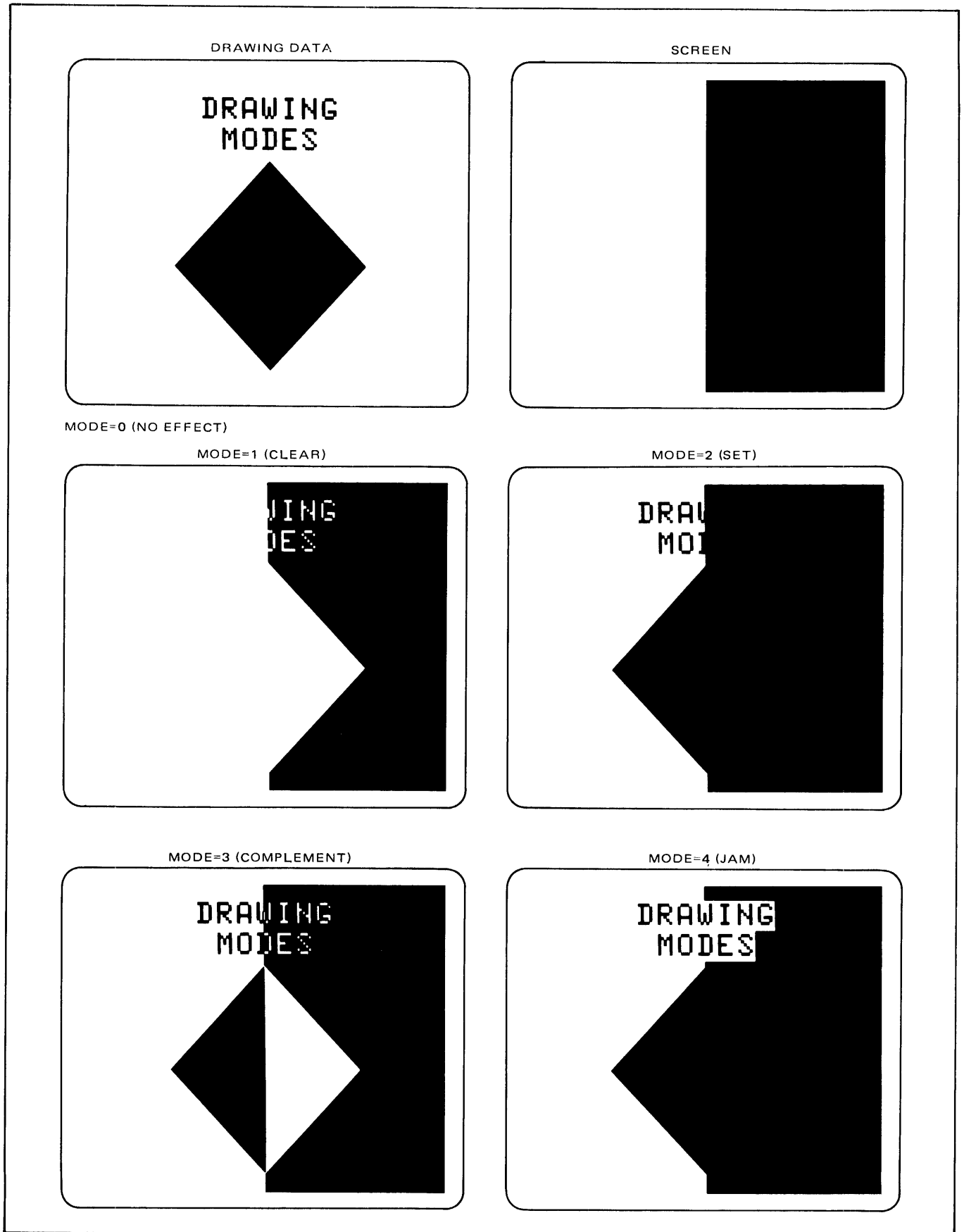


Figure 3-2. Examples Of Drawing Modes

**Selective Erase.** A vector drawn in set mode can be selectively erased by redrawing it in clear mode. This will cause gaps to occur if the erased line is intersected by other lines. This problem can be overcome by initially drawing the line in complement mode and then redrawing it in complement mode to erase the line. This technique will preserve the original display. Complement mode is useful for drawing and erasing temporary figures.

**Example:** Select complement mode, draw a vector, and then erase the vector by redrawing.

```

⌘ * m 3A      (select complement mode)
⌘ * p a f 100,300 300,300Z  (draw vector)
⌘ * p a f 100,300 300,300Z  (erase vector)
    
```

### Drawing Patterns

You can select the dot pattern used when drawing vectors or filling rectangular areas. Dotted and dashed lines can be drawn by selecting one of nine predefined line patterns or a user defined line or area pattern. This allows you to use different line patterns to distinguish between groups of plotted data or easily generate shading and cross hatching for use in engineering drawings, graphs or fabric patterns.

**LINE TYPE.** One of eleven line types can be selected. Once a line type has been selected all drawing vectors are drawn using that line type. The patterns for the predefined line types are shown in figure 3-3. Refer to the Define Line Pattern command for additional information.

Select Line Type: `⌘ * m <line type> b`

- where: `<line type>` is
- 1 Solid line (default)
  - 2 User defined line pattern
  - 3 User defined area pattern
  - 4 Predefined pattern #1
  - 5 Predefined pattern #2
  - 6 Predefined pattern #3
  - 7 Predefined pattern #4
  - 8 Predefined pattern #5
  - 9 Predefined pattern #6
  - 10 Predefined pattern #7
  - 11 Point plot

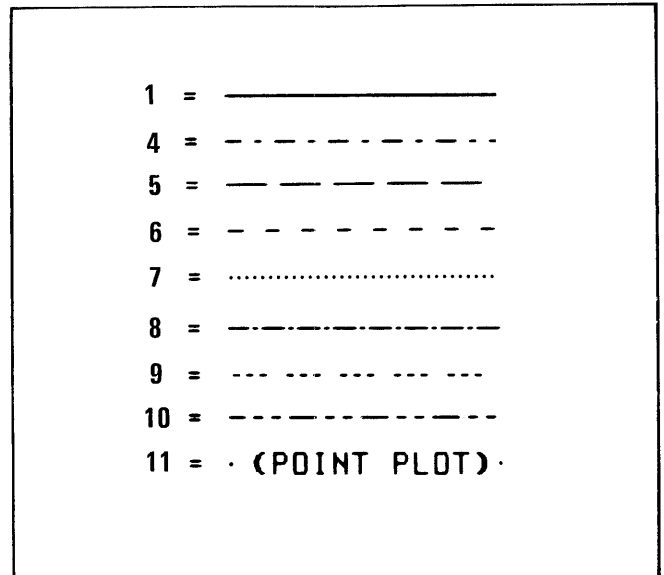


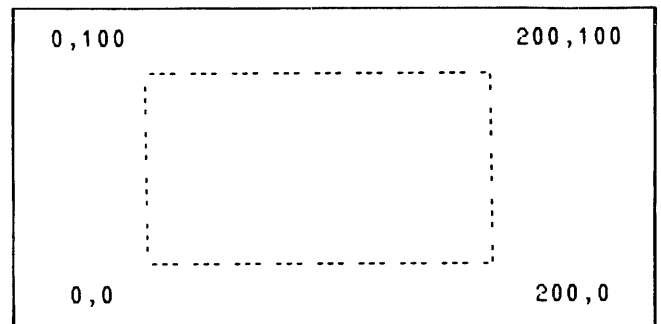
Figure 3-3. Predefined Line Type Patterns

Point plot causes a single point to be plotted at the coordinates specified by the data. This line type is useful for generating "scattergram" type graphs. If user defined area shading is selected (type = 3) the line patterns used are selected from the eight lines making up the area fill pattern (refer to Define Area Pattern). The display is divided into groups of eight rows and eight columns. Horizontal and vertical lines are drawn using the appropriate row or column from the area pattern. Diagonal lines are drawn using a solid vector.

**Example:** Select line type 9 and draw a figure using the new line type.

```

⌘ * m 9 B
⌘ * p a 0,0 200,0 200,100 0,100 0,0Z
    
```



**Example:** Select the area pattern as the line type.

```

⌘ * m 3 B
    
```



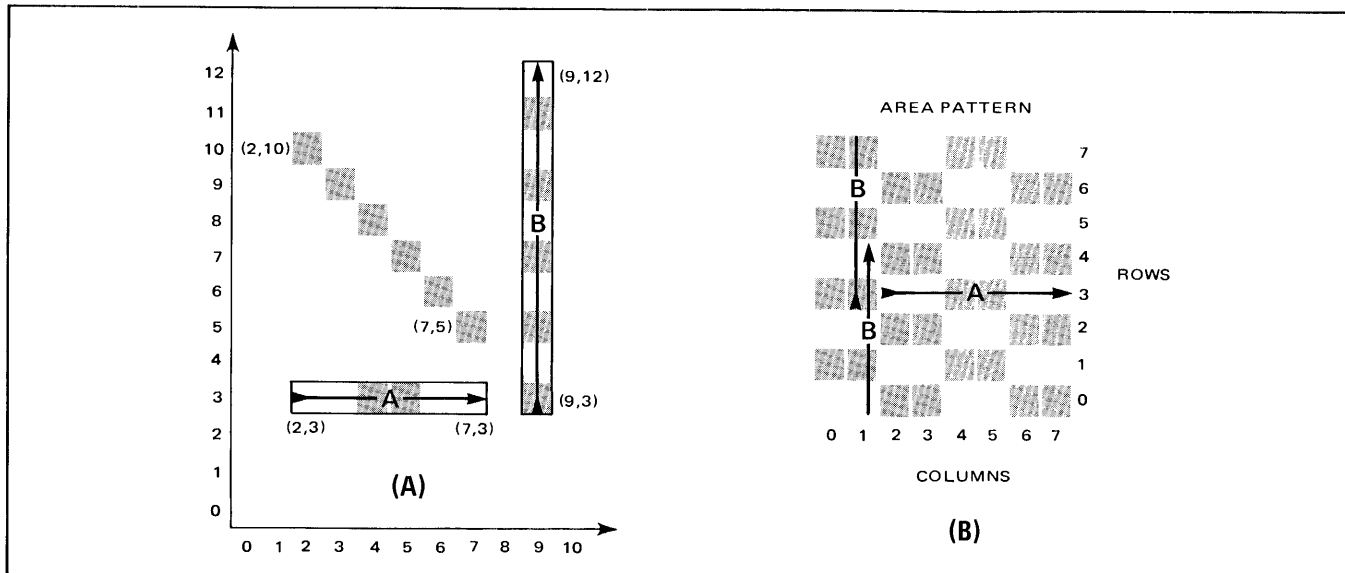


Figure 3-4. Using Area Patterns As Line Types

Drawing vectors (2,3)-7,3), (9,3)-(9-12), and (3,7)-(7,3) using the area pattern shown in figure 3-4b would result in the drawing shown in figure 3-4a.

Adjacent horizontal or vertical lines using the user defined line type (type = 2) can be used to create patterns more complicated than those available in an 8x8 area pattern. User defined line and area patterns are described in the following paragraphs.

**DEFINE LINE PATTERN.** The dot pattern used to draw vectors can be defined programmatically. Once a pattern is defined you must select the user defined line type (type = 2) using the Select Line Type command. Figure 3-5 gives examples of line patterns.

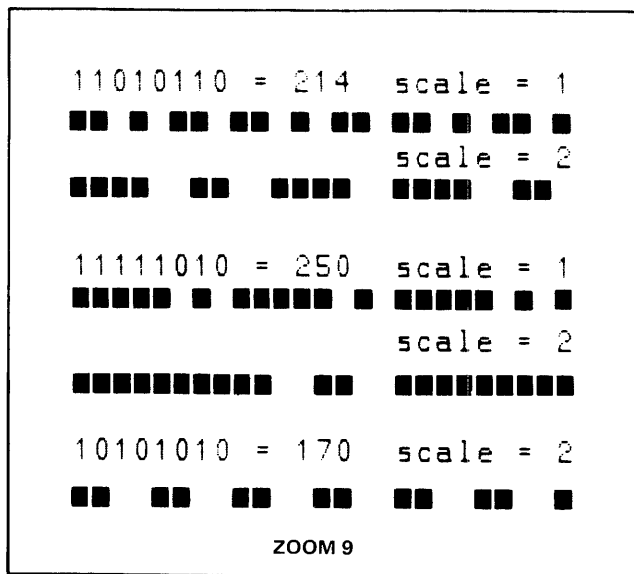


Figure 3-5. Examples of User Defined Line Patterns

Define Line Pattern: `␣ * m <pattern> <scale> c`

where: **<pattern>** is the decimal value 0 to 255) defining an 8-bit binary pattern. For example, . . . . = 10101010 = 170.

**<scale>** is a scale factor (1 to 16) to be applied to the pattern. For example, with a scale factor of 3 the pattern defined above would be as follows: ... ..

Line patterns too complex to be obtained from an 8x8 area pattern can be generated by plotting a series of lines and varying the patterns used for successive lines. Complex patterns such as those used in weaving can be generated easily using this technique.

**Example:** Define a pattern to generate the following vector:

```
*****oo**oo*****oo**oo
pattern = 11111010 = 250
scale = 2
␣ * m 250 2 c
```

**DEFINE AREA PATTERN.** An 8x8 pattern can be defined for use in filling rectangular areas. The pattern can also be used to provide line patterns for horizontal or vertical lines when the area pattern is selected as a line type (type = 3). (Refer to Define Line Type.) Irregular shapes can also be built up by selecting the area shading pattern and then using successive lines.

The area pattern is defined using 8 parameters, one for each of the rows in the pattern. Each parameter is a decimal number (0 to 255) representing an 8-bit binary pattern. Refer to Define Line Pattern for additional information. The display is divided up into 8x8 cells. Every point on the display is mapped to a corresponding bit in the pattern. Drawing horizontal or vertical lines causes the corresponding row or column of the pattern to be used as the line pattern. Diagonal vectors will always be drawn using a solid line. Figure 3-6 contains sample area fill patterns.

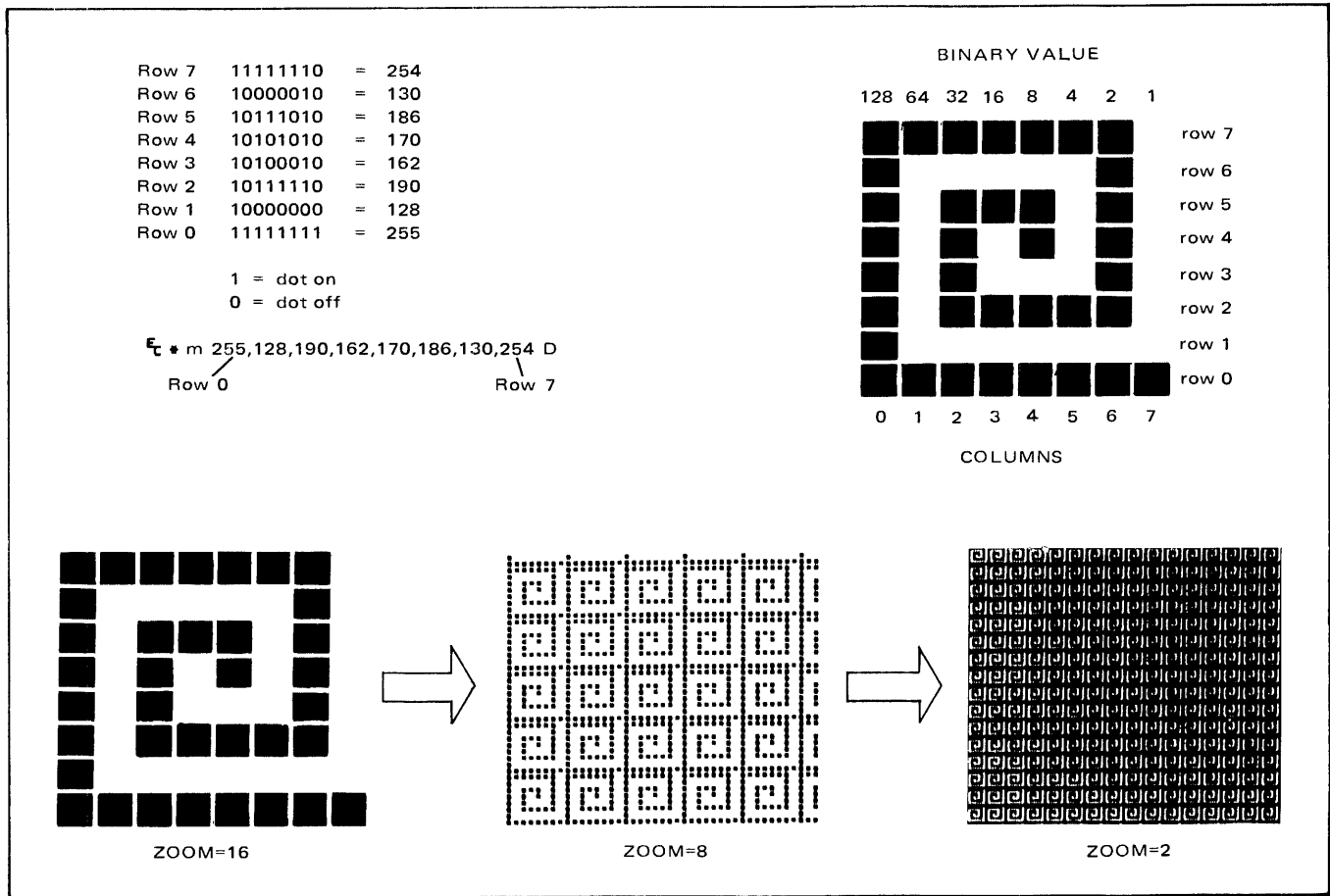
Define Area Pattern:



where: <row 0> is the 8-bit pattern for row 0  
 :  
 :  
 <row 7> is the 8-bit pattern for row 7

A simple checkerboard pattern would be defined as follows:

ε \* m 170 85 170 85 170 85 170 85 D  
 Row 0 Row 7



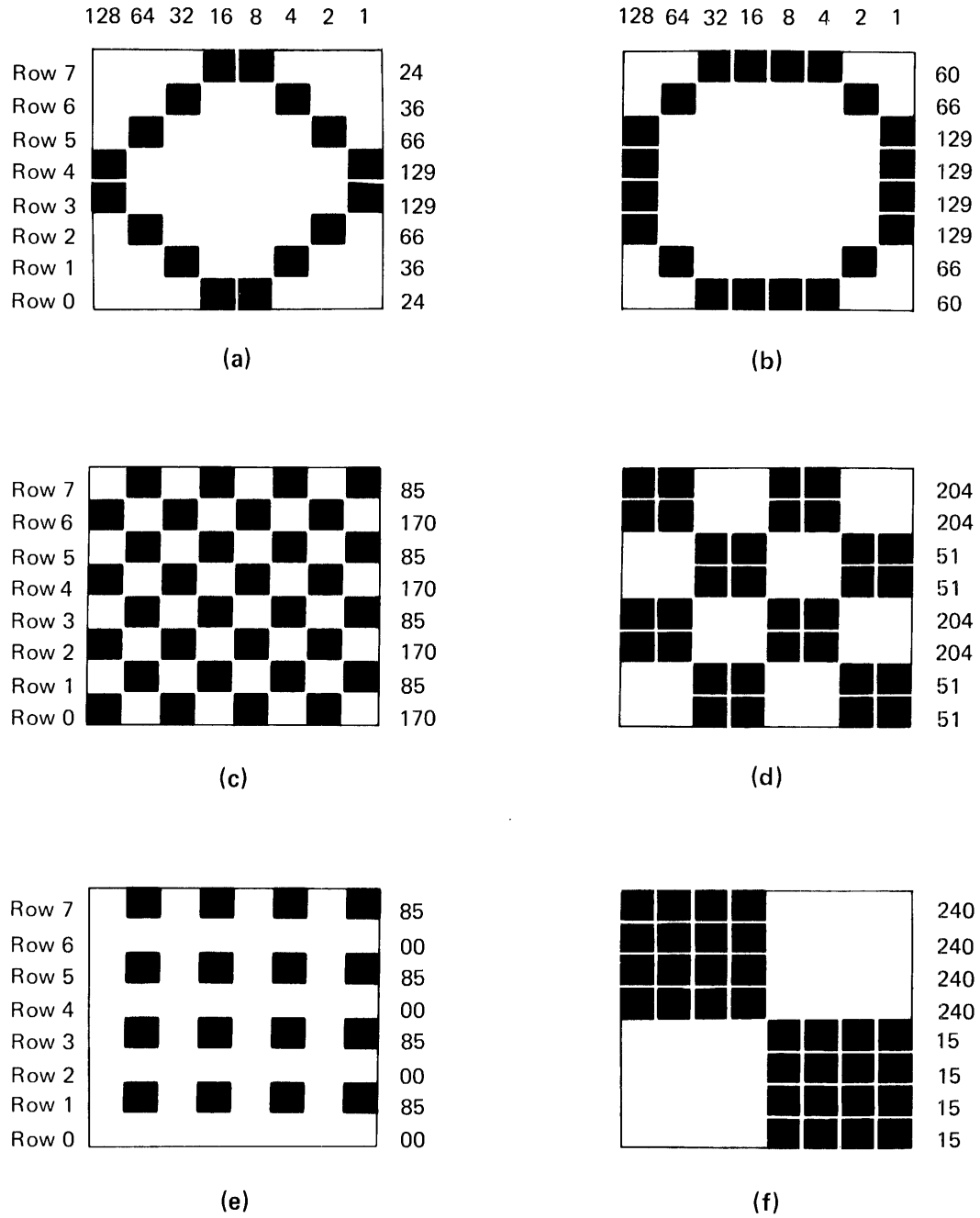


Figure 3-6. Area Pattern Examples

### Area Fill

A rectangular area can be filled with a pattern by simply sending the lower left (LL) and upper right (UR) coordinates of the rectangle. The coordinates can be in either absolute or relocatable format. The pattern used is selected by the Line Type command. This allows a choice of predefined and user defined line patterns as well as an 8x8 bit area pattern (refer to Define Area Pattern).

An easy way to selectively erase a portion of the graphics display would be to set the drawing mode to clear, select the solid vector line type (type = 1), and then use the area fill command to select the area to be cleared. Area fill is also useful for shading bar graphs or engineering drawings. The soft keys can be loaded with the proper escape sequences and then triggered to generate area patterns locally using either the current cursor or pen position as a coordinate.

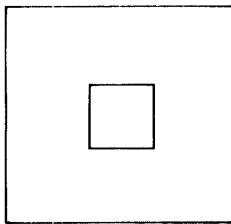
**AREA FILL, ABSOLUTE.** The absolute area fill command uses the absolute coordinates of the area.

Area Fill, Absolute: `␣ * m <XLL, YLL> <XUR, YUR> e`

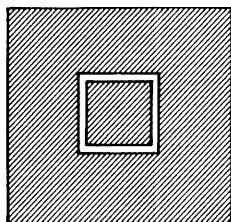
where: <XLL, YLL> and <XUR, YUR> are the absolute coordinates of the lower left and upper right corners of the area to be filled.

**Example:** Draw a box and then complement the entire graphics display. Note that repeating the ␣ \* m sequence would restore the original display.

`␣ * p a 150 150 200 150 200 200 150 200 150 150 F`



`␣ * m 3 a 1 b 0 0 719 359 E`



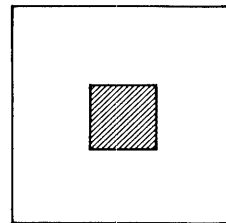
**Example:** Clear the display area with lower left (LL) coordinates 0,0 and upper right (UR) coordinates 100,100.

`␣ * m 1 a 1 b 0,0 100,100 E`  
 ← clear mode    ← solid line    ← area to be cleared

If a predefined or user defined line pattern is selected, the area fill command can be used to provide area shading or complex patterns. If the user defined area pattern is selected, the area will be filled with the 8x8 area pattern (refer to Define Area Pattern).

**Example:** Using the area fill pattern shown in figure 3-6a, shade the area with XLL, YLL = 50,50 and XUR, YUR = 100,100.

`␣ * m 2 a 3 b 24 36 66 129 129 66 36 24 d 50 50 100 100 E`



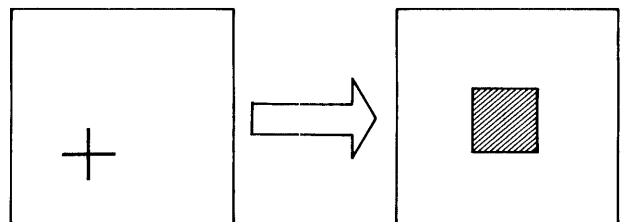
**AREA FILL, RELOCATABLE.** The relocatable area fill command uses area coordinates in relocatable ASCII format.

Area Fill, Relocatable: `␣ * m <XLL, YLL> <XUR, YUR> f`

where: <XLL, YLL> and <XUR, YUR> are the relocatable coordinates for the lower left and upper right corners of the area to be filled.

**Example:** Using the area fill pattern shown in figure 3-6c, shade a 50x50 unit area with the lower left corner at the current cursor position.

`␣ * m 2 a 3 b 85 170 85 170 85 170 85 170 d 1 0 0 50 50 F`



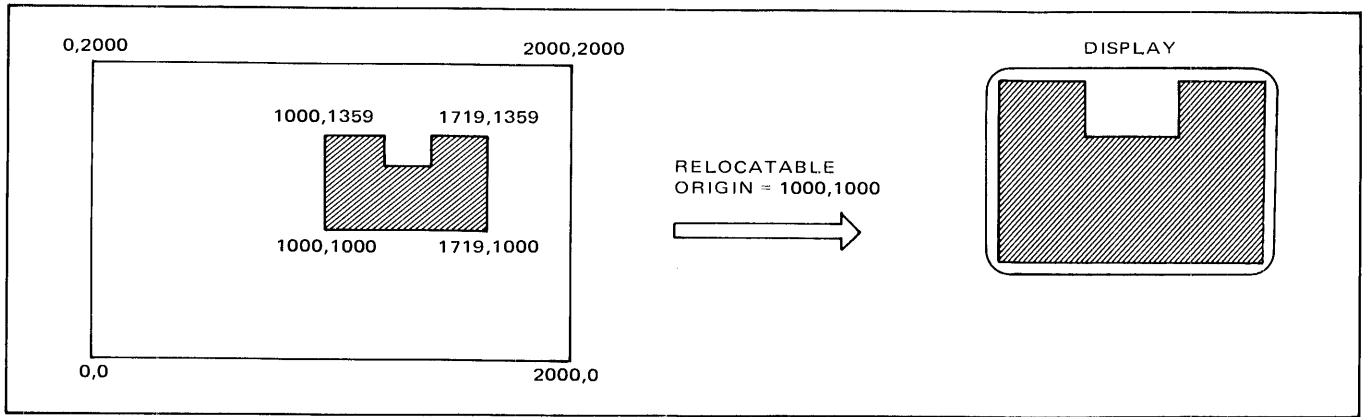


Figure 3-7. Relocatable Origin

### Relocatable Origin

The relocatable origin allows you to use one set of data and drawing commands to display a figure at several different positions on the screen. (See the resistor example under ASCII Relocatable Format.)

You can also display portions of a figure that is too large to fit on the screen. You can create a "window" that can be positioned to display any 720 by 360 unit portion of the figure. The value of the relocatable origin is added to the relocatable data to obtain the coordinates used to draw the data. Figure 3-7 illustrates the effect of a Relocatable Origin on the display.

This technique eliminates the need to check boundary conditions or compute new data in order to display the desired portion of the figure. Simply set the relocatable origin to the proper value to display the desired portion of

the figure and then send the unchanged figure data to the terminal. The terminal will then automatically select and adjust the "window" data.

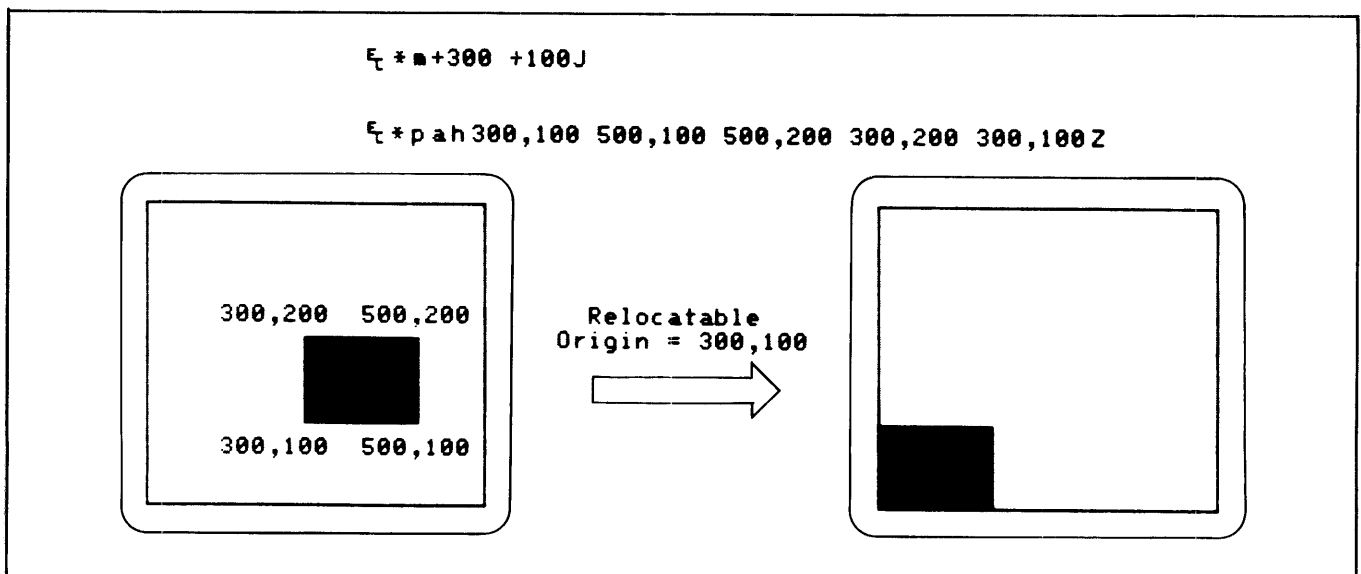
**SET RELOCATABLE ORIGIN ABSOLUTE.** The relocatable origin can be set to any absolute coordinates using ASCII absolute format (-16384 to 16383).

Set Relocatable Origin Absolute:

```
␣ * m <X,Y> j
```

where: <X,Y> are the x and y coordinates in ASCII absolute format.

**Example:** Set the relocatable origin to display the box in the figure so that the box is positioned at the lower left corner of the display.



**SET RELOCATABLE ORIGIN TO CURRENT PEN POSITION.** The relocatable origin can be set to the current pen position.

Set Relocatable Origin  
To Current Pen Position:



**SET RELOCATABLE ORIGIN TO GRAPHICS CURSOR POSITION.** The relocatable origin can be set to the current graphics cursor position.

Set Relocatable Origin  
To Graphics Cursor Position:



### Rubber Band Line

The "rubber band" feature can be used to preview vectors before they are stored in the graphics memory or sent to the computer. This feature is normally used by the operator when inputting graphics data from the keyboard. When the rubber band feature is enabled, a temporary line is displayed in the current line type (refer to Line Type). The line is shown extending from the current pen position to the graphics cursor. The line is dynamic and will follow the cursor as it is moved. The line is not stored until a draw command is entered.

Rubber Band Line On:



Rubber Band Line Off:



### Selecting The Graphics Default Parameters

Graphics parameters can be set to their default (power on or full reset) values. Table 3-5 lists the various parameters and their default values. Additional information can be found under the discussions of the individual parameters.

Set Graphics  
Default Parameters:



The current graphics mode and settings can be obtained with graphics status requests. Graphics status requests are described in Section VI, Status. It may be desirable to reselect graphics settings before you send graphics data to the terminal.

Table 3-5. Graphics Parameter Default Values

PARAMETER	DEFAULT VALUE
Pen Condition	up
Line Type	1 (solid)
Drawing Mode	set
Relocatable Origin	0,0
Text Size	1
Text Direction	1
Text Origin	1 (left, bottom justified)
Text Slant	0 (off)
Graphics Text	off
Graphics Video	on
Alphanumeric Video	on
Graphics Cursor	off
Alphanumeric Cursor	on
Rubber Band Line	off
Zoom	off
Zoom Size	1
Autoplot	off
Autoplot Menu	clear
Compatibility Mode	
Page Full Straps	0 (out)
GIN Strap	0 (CR only)

### PLOTTING SEQUENCES

All vector plotting sequences are initiated by ESC \* p. Table 3-6 lists the commands that can be used within a plotting sequence.

Table 3-6. Graphics Plotting Control Functions

ESC * p <parameters and data>	
PARAMETER	DESCRIPTION
a	lift the pen
b	lower the pen
c	use graphics cursor position as new point
d	draw a single dot at the current pen position
e	set relocatable origin = current pen position
f	use ASCII absolute format
g	use ASCII incremental format
h	use ASCII relocatable format
i	use binary absolute format
j	use binary short incremental format
k	use binary incremental format
l	use binary relocatable format
z	NOP/synch

After ESC \* p has been sent, the drawing format is normally specified before data is sent.

If no format is specified, ASCII absolute is assumed. There is no explicit draw vector command. When enough parameter bytes to specify a single end point have been received (the number depends on the format used), the pen is moved from its current position to the new end point. (See figure 3-8.) If the pen is down, a vector will be drawn. If

the pen is up, the pen is moved to the new point (without drawing a vector) and lowered. The new end point becomes the current pen position.

Note that if a parameter byte is lost or garbled in transmission, all following end points will be improperly read. To minimize data errors caused by the loss of a data byte, any command can be used to reset the parameter count and restore synchronization. Nops (z), redundant format, or pen down commands can also be inserted to insure synchronization if necessary.

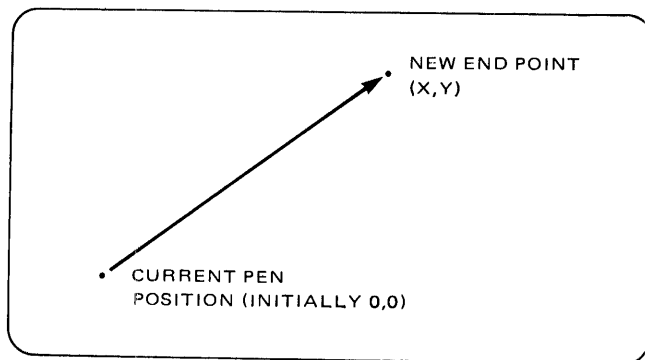


Figure 3-8. Current Pen Position And New End Point

Graphics sequences can extend indefinitely. In general, longer sequences are preferred as they minimize the overhead necessary for a plot sequence. ESC \* p <format> must be sent for each series of vectors. As the sequence length decreases, the percentage of preamble characters increases, and the vector drawing rate goes down. The worst possible case would be to send ESC \* p <format> for each vector; approximately 50% of the characters sent would be overhead, reducing vector speed by a factor of 2.

The general format for an absolute plotting sequence is:

```
ESC * p i a <byte1> <byte2> <byte3> <byte4> (z)
<byte1> <byte2> <byte3> <byte4> ...
... <byte1> <byte2> <byte3> <byte4> Z (or any
capital command)
```

Each block of 4 bytes specifies a single point. The "i" indicates that absolute format is to be used. The "a" raises the pen before it is moved to the point specified by the next four bytes and lowered. A NOP (z) can be added to insure synchronization, if necessary. The lowered pen draws a vector as it moves to the next point, and so on. The capital "Z" terminates the plotting sequence.

The vector end point formats allow the pen to be moved completely off the screen (an absolute coordinate of 1000, for example). The actual range of the pen position can be from -16384 to 16383. Vectors that extend beyond the screen are clipped so that they will not wrap around.

## Pen Control

The terminal uses the concept of a "pen" in drawing vector data. The pen can be lifted or lowered as well as be positioned using absolute or relative coordinates. For example, the pen is lifted, moved to a starting coordinate, lowered and moved to an endpoint to draw a line. The pen is initially in the up state and positioned at absolute coordinates 0,0 following power up or a full reset. If the pen is raised and coordinates given, the pen is moved to the coordinates and then lowered. The pen is normally left in the down position.

Raise Pen:



Lower Pen:



## Vectors

Graphic data is made up of vectors. Each vector is specified by the current graphic starting point and an end point. The current graphic starting point is one of the following:

0,0 Initial starting point

Last point defined by the user with the MOVE key

Last point defined by the user with the DRAW key

Last point defined by the graphics cursor (ESC \* p c)

Last point defined by data in a draw or move command (ESC \* p f/g/h/i/j/k/l)

Graphic points are specified in one of following formats:

- ASCII Absolute
- ASCII Incremental
- ASCII Relocatable
- Binary Absolute
- Binary Incremental
- Binary Short Incremental
- Binary Relocatable

If no format is specified in the graphic command, ASCII absolute format is assumed. More than one point can be given in a command. This minimizes communications overhead. Tables 3-7, 3-8 and 3-9 provide a reference for computing data bytes used in the various vector formats.

## ASCII Formats

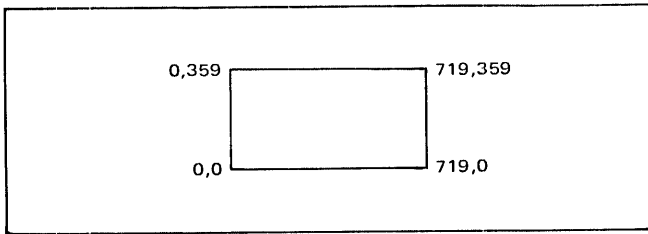
In the ASCII formats, coordinates are specified with ASCII characters 0 through 9. This means that numeric characters generated by a simple print statement can be used to specify X,Y pairs. The first value is used as the X coordinate, and the second as the Y coordinate.

Spaces or commas must be used to delimit the X and Y values. Excess delimiters are ignored. Digits following a decimal point are ignored (i.e. 123.456 is read as 123).

Exponential notation cannot be used. Consequently, the values must be in integer form. The number of bytes necessary to specify a single end point depends on the magnitude of the values.

**ASCII ABSOLUTE FORMAT.** The values used in the ASCII absolute format can range between -16384 and 16383. Note that only points where X is in the range 0 to 719 and Y is in the range 0 to 359 will be visible on the screen. The following example draws vectors around the perimeter of the screen:

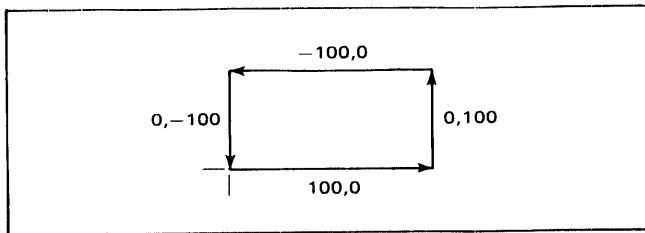
```
⌘ * p a 0,0 719,0 719,359,0,359,0,0Z
```



Since no format is indicated, ASCII absolute is assumed. The "a" raises the pen, which is moved to (0,0) and lowered. Vectors are then drawn to (719,0), (719,359), (0,359), and back to (0,0). (Note that the values are delimited by spaces or commas. The capital Z (a nop) terminates the sequence. Imbedded carriage return and line feed characters are ignored.

**ASCII INCREMENTAL FORMAT.** In the ASCII incremental format you can specify a delta X and a delta Y. These values are added to the current pen position to obtain a new end point. The first value is read as delta X and the second as delta Y. For example to draw a square 100 units on a side, the following sequence could be used:

```
⌘ * p g 100 0 0 100 -100 0 0 -100 Z
```



Beginning at the current pen position, a series of vectors is drawn by moving the pen 100 units to the right, up 100 units, left 100 units, and finally down 100 units. The same figure could have been drawn at any screen location by first positioning the pen to the desired starting point before sending the drawing sequence.

**ASCII RELOCATABLE FORMAT.** The ASCII relocatable format allows you to use a relocatable origin to be added to the incoming X and Y coordinate values. The resultant values are then treated as absolute coordinates by the terminal. The relocatable format allows you to use absolute data as if it were incremental by merely changing the relocatable origin. For example, symbol elements specified in absolute coordinates can be drawn in different locations as shown in the following example.

**Example:** Draw a resistor symbol stored in absolute coordinates at screen locations 50,100 and 200,100.

```
Resistor Data = 0,10
                10,10
                15,15
                25,5
                35,15
                45,5
                50,10
                60,10
                0,20
                60,0
                0,0
                60,0
```

```
⌘ * m 50,100J
⌘ * p a h 0,10 10,10 15,15 25,5 35,15
                45,5 50,10 60,10Z
```

```
⌘ * m 200,100J
⌘ * p a h 0,10 10,10 15,15 25,5 35,15
                45,5 50,10 60,10Z
```

### Binary Format

In binary format all points are sent in a packed binary format. The coordinate values are sent using the bit patterns of the ASCII characters listed in table 3-7. The number of characters required to specify a coordinate depends on the format used. The values for X and Y coordinates can be from -16384 to 16383.

**BINARY ABSOLUTE FORMAT.** Binary absolute data is plotted with respect to an origin at 0,0. Four bytes are required to specify a single end point. A 10 bit coordinate in the range 0-1023, is sent for both x and y.

The bytes are ordered as follows:

BIT	7	6	5	4	3	2	1	
BYTE 1	0	1	X9	X8	X7	X6	X5	HI X
BYTE 2	0	1	X4	X3	X2	X1	X0	LOW X
BYTE 3	0	1	Y9	Y8	Y7	Y6	Y5	HI Y
BYTE 4	0	1	Y4	Y3	Y2	Y1	Y0	LOW Y



Although it is possible to send coordinates in the range 0 to 1023, only points in the range 0-719 for X, and 0-359 for Y are visible on the screen. Vectors going off the screen are clipped. If the data requires scaling, this must be done before the data is sent to the terminal.

The following example shows how the 4 data bytes are computed. The numbers are converted to the 10 bit binary equivalent. Bits 7 and 6 are set to 01 to indicate a parameter.

```

X = 0 = 00000 00000      Y = 0 00000 00000
      HI X  LOW X          HI Y  LOW Y

      BYTE 1 = 01 00000 = SPACE HI X
      BYTE 2 = 01 00000 = SPACE LOW X
      BYTE 3 = 01 00000 = SPACE HI Y
      BYTE 4 = 01 00000 = SPACE LOW Y

X = 360 = 01011 01000     Y = 180 = 00101 10100
      HI X  LOW X          HI Y  LOW Y

      BYTE 1 = 01 01011 = + HI X
      BYTE 2 = 01 01000 = ( LOW X
      BYTE 3 = 01 00101 = % HI Y
      BYTE 4 = 01 10100 = 4 LOW Y
    
```

An escape sequence to draw a vector from 0,0 to 360,180 is as follows:

```

Esc * p i a SP SP SP SP + ( % 4 Z
      \X=0/  \Y=0/ \X=360/ \Y=180/
    
```

ESC \* p selects a plotting sequence. The "i" specifies absolute format. The "a" raises the pen up. The first 4 bytes (all spaces) move the raised pen to 0,0, where it is lowered. The next 4 bytes specify the point 360,180. After the 4th byte is received, the pen is moved to that point, drawing a vector. The capital "Z" terminates the escape sequence. Note that if spaces are used in the data sequence they are interpreted as data resulting in an improper plot.

**BINARY SHORT INCREMENTAL FORMAT.** The short incremental format uses two bytes to specify a delta X and a delta Y in the range -16 to +15. The five least significant bits are interpreted as a signed, two's complement number. This number is added to the current pen position to obtain the new end point. The data bytes are ordered as follows:

```

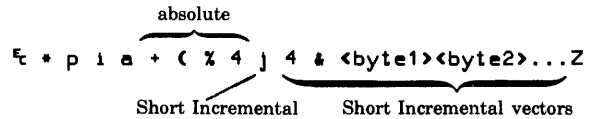
      BIT 7 6 5 4 3 2 1
      BYTE 1 0 1 < DELTA X >
      BYTE 2 0 1 < DELTA Y >
    
```

The following example illustrates the computation and use of the short incremental format:

```

DELTA X = -12 = 10100  DELTA Y = 6 = 00110
BYTE1 = 01 10100 = 4 DELTA X
BYTE2 = 01 00110 = & DELTA Y
    
```

The following sequence moves the pen to 360,180 in absolute format, then draws a vector to X = 360-12 = 350, y = 180+6 = 186.



**BINARY INCREMENTAL FORMAT.** Incremental is similar to short incremental, but with a larger range. Using six bytes, delta X and Y can range from -16384 to +16383.

```

      BIT 7 6 5 4 3 2 1
      BYTE1 0 1 DX14 DX13 DX12 DX11 DX10 HI DELTA X
      BYTE2 0 1 DX9 DX8 DX7 DX6 DX5 MID DELTA X
      BYTE3 0 1 DX4 DX3 DX2 DX1 DX0 LOW DELTA X

      BYTE4 0 1 DY14 DY13 DY12 DY11 DY10 HI DELTA Y
      BYTE5 0 1 DY9 DY8 DY7 DY6 DY5 MID DELTA Y
      BYTE6 0 1 DY4 DY3 DY2 DY1 DY0 LOW DELTA Y
    
```

The following example shows how incremental data bytes are generated.

```

DELTA X = -400 = 11111 10011 10000
              HI DX  MID DX  LO DX

DELTA Y = 100 = 00000 00011 00100
              HI DY  MID DY  LO DY

      BYTE 1 = 01 11111 = ? HI DELTA X
      BYTE 2 = 01 10011 = 3 MID DELTA X
      BYTE 3 = 01 10000 = 0 LO DELTA X

      BYTE 4 = 01 00000 = space HI DELTA Y
      BYTE 5 = 01 01001 = # MID DELTA Y
      BYTE 6 = 01 00100 = $ LO DELTA Y
    
```

Table 3-7. Characters Used in Packed Data Formats

ASCII Character	Bit Pattern	ASCII Character	Bit Pattern
SP	01 0 0000	0	01 1 0000
!	01 0 0001	1	01 1 0001
"	01 0 0010	2	01 1 0010
#	01 0 0011	3	01 1 0011
\$	01 0 0100	4	01 1 0100
%	01 0 0101	5	01 1 0101
&	01 0 0110	6	01 1 0110
'	01 0 0111	7	01 1 0111
(	01 0 1000	8	01 1 1000
)	01 0 1001	9	01 1 1001
*	01 0 1010	:	01 1 1010
+	01 0 1011	;	01 1 1011
,	01 0 1100	<	01 1 1100
-	01 0 1101	=	01 1 1101
.	01 0 1110	>	01 1 1110
/	01 0 1111	?	01 1 1111

**BINARY RELOCATABLE FORMAT.** Binary relocatable format specifies absolute X and Y coordinates in the range -16384 to +16383 using 6 bytes. The value specified in the relocatable origin command is taken to be the 0,0 point. The actual screen address is computed by the terminal by adding the relocatable origin to the X,Y pair.

BIT	7	6	5	4	3	2	1		
BYTE 1	0	1	X14	X13	X12	X11	X10	HI	X
BYTE 2	0	1	X9	X8	X7	X6	X5	MID	X
BYTE 3	0	1	X4	X3	X2	X1	X0	LOW	X
BYTE 4	0	1	Y14	Y13	Y12	Y11	Y10	HI	Y
BYTE 5	0	1	Y9	Y8	Y7	Y6	Y5	MID	Y
BYTE 6	0	1	Y4	Y3	Y2	Y1	Y0	LOW	Y

The following example shows how relocatable data bytes are computed.

RELOC X = -600	=	11111	01101	01000					
		HI X	MID X	LOW X					
RELOC Y = 200	=	00000	00110	01000					
		HI Y	MID Y	LOW Y					
BYTE 1	=	01 11111	=	?				HI	X
BYTE 2	=	01 01101	=	-				MID	X
BYTE 3	=	01 01000	=	(				LOW	X
BYTE 4	=	01 00000	=	space				HI	Y
BYTE 5	=	01 00110	=	&				MID	Y
BYTE 6	=	01 01000	=	(				LOW	Y

Table 3-8. Absolute Format Addressing Bytes

	0	1	2	3	4	5	6	7	8	9
0	##	#!	##	##	##	##	##	##	##	##
10	##	##	##	##	##	##	##	##	##	##
20	##	##	##	##	##	##	##	##	##	##
30	##	##	##	##	##	##	##	##	##	##
40	!(	!)	!+	!+	!,	!-	!.	!/	!0	!1
50	!2	!3	!4	!5	!6	!7	!8	!9	!:	!;
60	!<	!=	!>	!?	!"	!"	!"	!"	!"	!"
70	!"	!"	!"	!"	!"	!"	!"	!"	!"	!"
80	"0	"1	"2	"3	"4	"5	"6	"7	"8	"9
90	":	";	"<	"=	">	"?	"#	"!	""	"#
100	#\$	#\$	#\$	#\$	\$(	\$(	\$(	\$(	\$(	\$(
110	##	##	##	##	##	##	##	##	##	##
120	##	##	##	##	##	##	##	##	##	##
130	##	##	##	##	##	##	##	##	##	##
140	##	##	##	##	##	##	##	##	##	##
150	##	##	##	##	##	##	##	##	##	##
160	##	##	##	##	##	##	##	##	##	##
170	##	##	##	##	##	##	##	##	##	##
180	##	##	##	##	##	##	##	##	##	##
190	##	##	##	##	##	##	##	##	##	##
200	##	##	##	##	##	##	##	##	##	##
210	##	##	##	##	##	##	##	##	##	##
220	##	##	##	##	##	##	##	##	##	##
230	##	##	##	##	##	##	##	##	##	##
240	'0	'1	'2	'3	'4	'5	'6	'7	'8	'9
250	':	';	'<	'=	'>	'?	'#	'!	'"	'#
260	(\$	(\$	(\$	(\$	(\$	(\$	(\$	(\$	(\$	(\$
270	(.	(/	(0	(1	(2	(3	(4	(5	(6	(7
280	(8	(9	(:	(;	(<	(=	(>	(?)	(#	(!
290	)"	)#	)\$	)%	)&	)'	)("	)#	)!	)"
300	)	)-	).	)/	)0	)1	)2	)3	)4	)5
310	)6	)7	)8	)9	):	);	)<	)=	)>	)?
320	##	##	##	##	##	##	##	##	##	##
330	##	##	##	##	##	##	##	##	##	##
340	##	##	##	##	##	##	##	##	##	##

	0	1	2	3	4	5	6	7	8	9
350	*>	*?	*#	*!	*"	*#	*\$	*%	*&	*'
360	+(<	+)	++	++	+,	+-	+.	+/	+0	+1
370	+2	+3	+4	+5	+6	+7	+8	+9	+:	+
380	+<	+ =	+>	+?	,"	,"	,"	,"	,"	,"
390	,&	,'	,("	,)	,*	,+	,,	,-	,.	,/
400	,0	,1	,2	,3	,4	,5	,6	,7	,8	,9
410	,:	,;	,<	,=	,>	,?	,#	,!	,"	,#
420	-\$	-\$	-\$	-\$	-\$	-\$	-\$	-\$	-\$	-\$
430	-.	-/	-0	-1	-2	-3	-4	-5	-6	-7
440	-8	-9	-:	-;	-<	-=	->	-?	-#	-!
450	."	."	."	."	."	."	."	."	."	."
460	..	..	..	..	..	..	..	..	..	..
470	.6	.7	.8	.9	.:	.;	.<	.=	.>	.?
480	/#	/!	/"	/#	/\$	/\$	/\$	/\$	/\$	/\$
490	/*	/+	/,	/-	/.	//	/0	/1	/2	/3
500	/4	/5	/6	/7	/8	/9	/:	/;	/<	/=
510	/>	/?	/#	0!	0"	0#	0\$	0%	0&	0'
520	0(	0)	0*	0+	0,	0-	0.	0/	00	01
530	02	03	04	05	06	07	08	09	0:	0;
540	<#	0=	0>	0?	1#	1!	1"	1#	1\$	1%
550	1&	1'	1("	1)	1*	1+	1,	1-	1.	1/
560	10	11	12	13	14	15	16	17	18	19
570	1:	1;	1<	1=	1>	1?	2#	2!	2"	2#
580	2\$	2%	2&	2'	2("	2)	2*	2+	2,	2-
590	2.	2/	20	21	22	23	24	25	26	27
600	28	29	2:	2;	2<	2=	2>	2?	3#	3!
610	3"	3#	3\$	3%	3&	3'	3("	3)	3*	3+
620	3,	3-	3.	3/	30	31	32	33	34	35
630	36	37	38	39	3:	3;	3<	3=	3>	3?
640	4#	4!	4"	4#	4\$	4%	4&	4'	4("	4)
650	4*	4+	4,	4-	4.	4/	40	41	42	43
660	44	45	46	47	48	49	4:	4;	4<	4=
670	4>	4?	5#	5!	5"	5#	5\$	5%	5&	5'
680	5(	5)	5*	5+	5,	5-	5.	5/	50	51
690	52	53	54	55	56	57	58	59	5:	5;
700	5<	5=	5>	5?	6#	6!	6"	6#	6\$	6%
710	6&	6'	6("	6)	6*	6+	6,	6-	6.	6/

Note: # indicates a "space" character; every coordinate address must consist of the two characters shown in the table.

## RASTER DATA TRANSFERS

Image data can be transferred directly from the graphics memory to cartridge tape and/or compatible printer (such as the HP 2631G or HP 2745A). A remote CPU can read the graphics memory by first directing the terminal to dump the image data to cartridge tape, then reading the tape to the CPU. Also, graphics memory can be restored from cartridge tape. A remote CPU can transfer image data to the graphics memory by first sending the data to the terminal's cartridge tape, then directing the terminal to read the data into graphics memory. The following paragraphs discuss the data format for such transfers, as well as the speed of the transfers and the amount of tape required for storage of the image data.

### Raster Output Format

The 2648 always outputs raster data in the same format, regardless of the destination. This format consists of a start record, 360 data records, and an ending record. The start record initializes the transfer and may dimension the display. Data records consist of a byte count followed by the proper number of binary data bytes. The data bytes contain the data for one image line, 720 dots, scanned from

left to right. (See figure 3-9A.) The display is scanned from top to bottom, so that the first data record represents the topmost raster line of the display. A data record will either contain 90 bytes (720 dots) or, in the case of a blank raster line, will not contain any data. The ending record terminates the transfer. Escape sequence preambles are used to differentiate the different types of records.

```

ε * r 720 = 360 t A           (Start Transfer)

ε * b 90 W <90 data bytes>
or
ε * b 0 W <blank line, no data follows>
.
.
.
.
ε * r B                       (End Transfer)
    
```

} (Repeated for 360 Lines)

Note that there is no CR/LF at the end of a raster escape sequence. When a store to tape is selected, the raster sequences are stored one raster line per record. Note that when copying directly from tape to printer, it is possible to position the tape so that the initial ε \* r A preamble will not be sent.

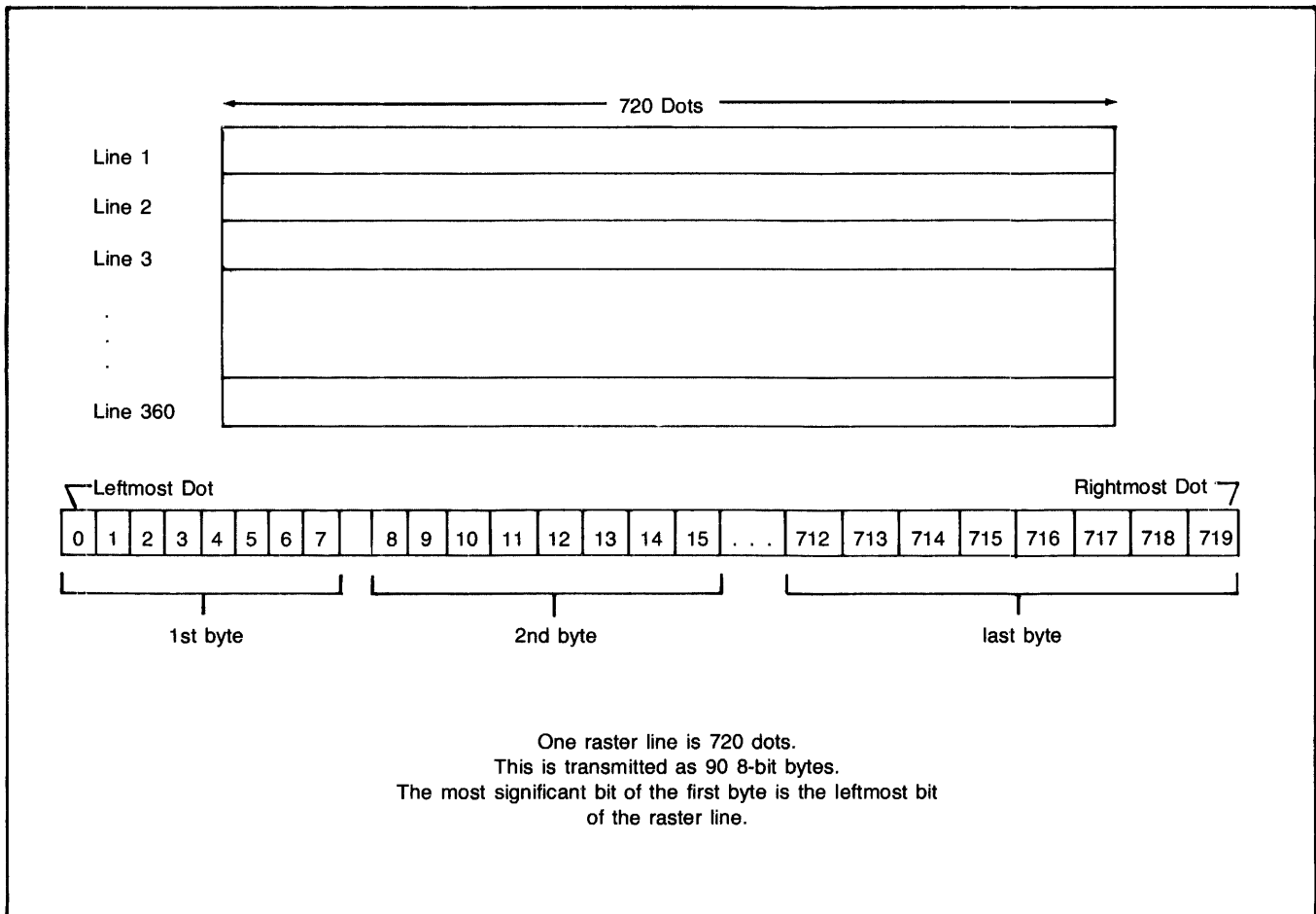


Figure 3-9A. Raster Data Format

## Raster Dump Commands

When restoring the graphics image memory from tape, the 2648A will understand the subset of the raster protocol given in table 3-10A. Any other raster command received will be ignored unless it is upper case, which will terminate the escape sequence.

Table 3-10A. Raster Dump Command Summary

ESCAPE SEQUENCE	FUNCTION
⌘ * r A	(start of transfer, a nop)
⌘ * r B	(end of transfer, a nop)
⌘ * r C	(erase screen)
⌘ * r D	(turn video on)
⌘ * b < COUNT > W	(count is limited to the range 0 to 254)

## Speed

Table 3-10B lists the times at which image data may be transferred from one device to another.

Table 3-10B. Raster Dump Transfer Rates

Graphics to tape	70 sec (tape speed limited)
Graphics to HP 7245	65 sec (HP 7245 limited)
Graphics to HP 2631G	58 sec (HP 2631 limited)
Graphics to tape and HP 7245	98 sec
Tape to graphics	70 sec (tape limited)
Tape to HP 7245	70 sec (tape limited)
Graphics to HP 1602 Logic Analyzer	31 sec (HP 2648A limited)

## Tape Used

A worst case picture (no blank lines) uses slightly less than 1/2 of a tape.

Full tape = 1679 inches

After dump 979 inches remaining, 700 inches used  
(No file mark after data)

### NOTE

The examples that follow assume that the reader is familiar with the generalized device control sequences explained in section 4, including the discussion of HP-IB.

## Graphics Dump

A "REWIND" command to Alternate I/O (device code 5) causes the entire graphics image to be sent to the current destination device(s). The default device is the right tape.

To print graphics, Alternate I/O (a compatible printer connected to the HP-IB) must be selected as the destination. If the destination is a tape, no file mark is written after the dump. An invalid destination device (DISPLAY) results in an error.

Graphics image data stored on tape may also be copied to a compatible printer connected to the HP-IB.

The entire image memory will always be dumped, even if the display is being 'zoomed'.

### Example — GRAPHICS TO TAPE

1. Select left or right tape as destination

```
⌘(GOLDI), f5 (left tape)      ⌘&p1D
⌘(GOLDI), f6 (right tape)     ⌘&p2D
```

2. Rewind Alternate I/O

```
⌘(GREENI), f5, ⌘(INSERT CHAR) ⌘&p5u0C
```

### Example — GRAPHICS TO HPIB PRINTER

1. Select Alternate I/O as destination

```
⌘(GOLDI), ⌘(INSERT CHAR)      ⌘&p5D
```

2. Rewind Alternate I/O

```
⌘(GREENI), f5, ⌘(INSERT CHAR) ⌘&p5u0C
```

### Example — MULTIPLE DESTINATIONS

1. Select left tape and HPIB printer as destinations

```
⌘(GOLDI), f5, ⌘(INSERT CHAR) ⌘&p1d5D
```

## Graphics Restore

A 'MARK FILE' operation to Alternate I/O (device code 5) causes data to be read from the current source device to graphics memory. An attempt is made to interpret this data as raster format escape sequences. Alpha data which is not part of a raster escape sequence is printed on the display. Control codes not part of a raster sequence are ignored. Valid raster sequences will be executed. The image is drawn using the current drawing mode. SET mode turns bits on where a data bit is 1, CLEAR mode turns those same bits off. JAM mode, which causes both 1's and 0's to be copied directly from the data byte into the image memory, does not always work as expected. A blank line will be indicated by 0 data bytes; consequently, that line in the image memory will be unchanged.

The only valid source devices for this operation are the left and right tapes. The default is the left tape.

### Example — TAPE TO GRAPHICS

1. Select left or right tape as source

```
⌘(GOLDI), f1 (left tape)      ⌘&p1S
⌘(GOLDI), f2 (right tape)     ⌘&p2S
```

2. Mark file on Alternate I/O

```
⌘(GREENI), f6, ⌘(INSERT CHAR) ⌘&p5u5C
```

Table 3-9. Incremental (Short) Vector Bytes

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?

## RECORDING GRAPHICS FUNCTIONS

The **DISPLAY FUNCTIONS** key can be used to display and record the graphics escape sequences or the action of graphics control keys. The control sequences are entered into the alphanumeric display each time a command is executed. The sequences can then be stored on cartridge tape using Edit Mode or a Record or Copy command. Table 3-10 lists the graphics control sequences that are generated when DISPLAY FUNCTIONS is on. Additional information on recording graphics functions is given in the User's Manual.

Table 3-10. Graphics Control Sequences Used in Record Operations

Key	Sequence	Description
<b>A</b> <b>→</b> <b>←</b> <b>↑</b> <b>↓</b>	none	Graphics cursor controls
<b>CURSOR FAST</b>	none	Graphics cursor fast
<b>SHIFT</b> <b>RB LN</b>	⌘ * dM ⌘ * dN	
<b>STOP</b>	⌘ * aB ⌘ * dT	
<b>ZOOM IN</b>	⌘ * dnI	"n" is the new zoom size
<b>ZOOM OUT</b>	⌘ * dnI	"n" is the new zoom size
<b>ZOOM</b>	⌘ * dH ⌘ * dG	
<b>G CURSOR</b>	⌘ * dL ⌘ * dK	
<b>AUTO PLOT</b>	⌘ * aA	
<b>SHIFT</b> <b>AXES</b>	⌘ * aC	
<b>SHIFT</b> <b>AUTOPLT MENU</b>	⌘ * aF ⌘ * aG	
<b>SHIFT</b> <b>G DSP</b>	⌘ * dC ⌘ * dD	
<b>SHIFT</b> <b>CLEAR</b>	⌘ * dA	
<b>SHIFT</b> <b>A DSP</b>	⌘ * dF ⌘ * dE	
<b>SHIFT</b> <b>TEXT</b>	⌘ * dS ⌘ * dK ⌘ * m4A	Turns on cursor, jam pattern (if not in scaled Compatibility Mode), and turns on Graphics Text Mode.
<b>SHIFT</b> <b>T SIZE</b>	⌘ * mnM	"n" is the text size
<b>SHIFT</b> <b>T ANG</b>	⌘ * mnN ⌘ * mO ⌘ * mnN ⌘ * mP	"n" is the text angle

Note that the DRAW and MOVE commands do not execute unless the graphics cursor is on.

Figure 3-9 shows the sequences generated when drawing a simple box. The graphics cursor is initially on and positioned at 0,0.

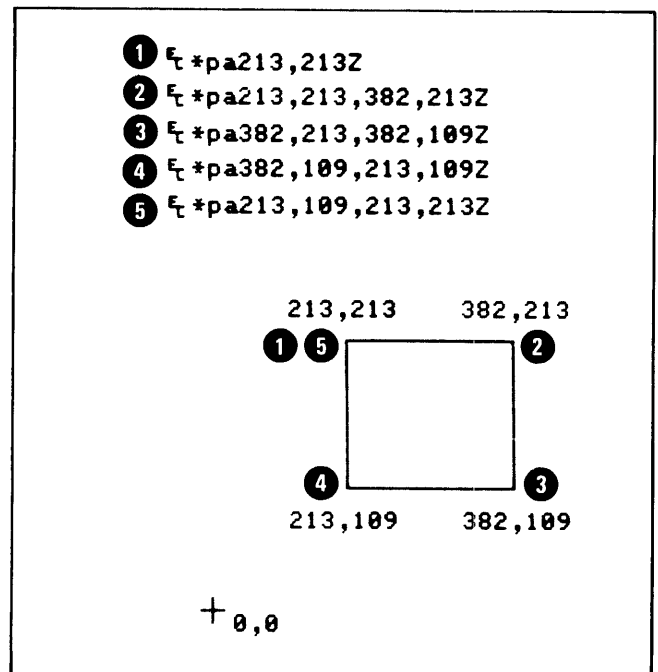


Figure 3-9. Recording Graphics Sequences

## Graphics Hardcopy Operations

A video hardcopy subsystem is available to make printed copies of the graphics display. The video hardcopy subsystem uses the HP 13254A Video Interface. Instructions for installing and configuring the interface are given in Section VII, Installation. Procedures for making video copies are given in Section IV, Device Control. If the interface is configured as address 04 (refer to Section VII and the 13254A Installation and Service Manual part no. 13254-90001), a PRINT command (**IGREEN**, **f6**, **f8**) can be entered locally at the terminal or sent from a computer. Copies can also be initiated manually from the hard copy unit itself.

## GRAPHICS TEXT

Text strings can be written directly into the graphics image memory. An internal character generator converts the ASCII codes into a dot matrix representation which is drawn as vectors. The character set includes upper and lower case (95 characters) and will be drawn as a 5 by 7 matrix in a 7 by 10 cell, with descenders for lower case. This character set is in addition to the normal alphanumeric character set. While this character set may seem redundant, it offers the following advantages:
















- Characters can be drawn at any dot position, rather than the 24 by 80 alphanumeric character positions.
- Characters can be rotated in multiples of 90 degrees.
- Characters can be scaled in size, from 1 to 8 times.
- Characters can be slanted 45 degrees for an italics-like effect.
- Lines of characters can be right, left, or center justified.
- In zoom mode, characters in the graphics memory are magnified.

Figure 3-10 shows the graphics character set.

### Keyboard Control Of Graphics Text

Graphics text can be entered directly from the keyboard. The backspace, carriage return, and line feed functions work as expected (even on inverted text), making it easy to add or edit titles and labels. A summary of keyboard operations affecting Graphics Text Mode is given in table 3-11. Additional information on keyboard text entry is contained in the User's Manual.

Table 3-11. Graphics Text Keyboard Functions

Key	Description
	Selects the graphics image memory as the destination for all text. Characters entering from the keyboard, datacom, or tape are drawn as vectors in the graphics memory using the current text size and angle (see the  and  keys). The drawing mode is initially set to jam pattern to allow for backspacing and retyping of characters. The graphics cursor indicates the position of the next character. Moving the graphics cursor will cause the next text line to begin at the new cursor position. The carriage return, line feed, and backspace functions work normally.
	Terminates Text Mode.
	Increases the character size from 1 to 8X. The smallest character is a 5 by 7 matrix in a 7 by 10 cell. Increasing the size makes the dots bigger while the character is still drawn as a 5 by 7 matrix.
	Sets the character orientation (multiples of 90 degrees) and turns slant on or off.
	Spaces one graphics text character to the right. (The actual direction of movement will depend on the text orientation.)
	(Vertical Tab). Spaces one graphics text line up. (The actual direction of movement will depend on the text orientation.)
In addition, the following keys function in the same manner as for alphanumeric text characters:  ,  ,  ,  ,  ,  , 	

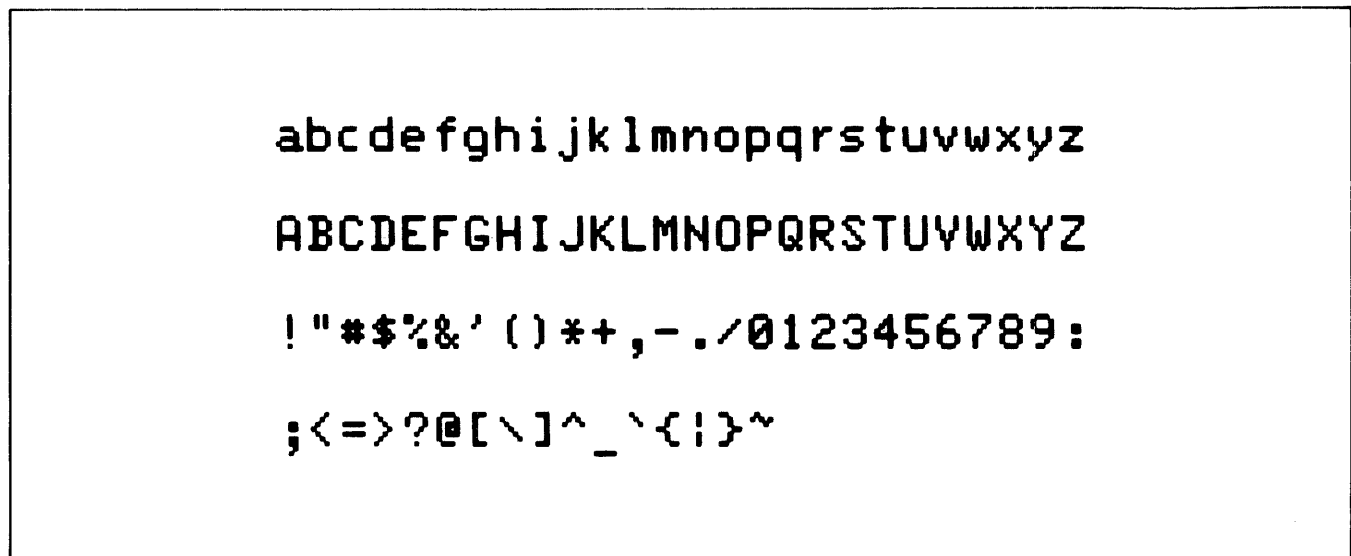


Figure 3-10. Graphics Text Characters

### Program Control of Graphics Text

All of the parameters for graphics text can be set programmatically. Commands are of the form: ESC \* m <parameter> <command>. The command can be alone or part of another ESC \* m sequence.

**SIZE.** The ASCII characters 1 through 8 specify the character size for graphics text. A "1" indicates the smallest character, a 5 by 7 dot matrix character in a 7 by 10 cell. Increasing the size increases the size of the dots. If a text size of 1 is specified, each dot in the cell is one dot on the screen. A size of 2 uses 4 screen dots for each character dot (2 X 2), and so on (see figure 3-11). A size of "1" is the default.

Set Graphics  
Text Size:

```
ESC * m <size> m
```

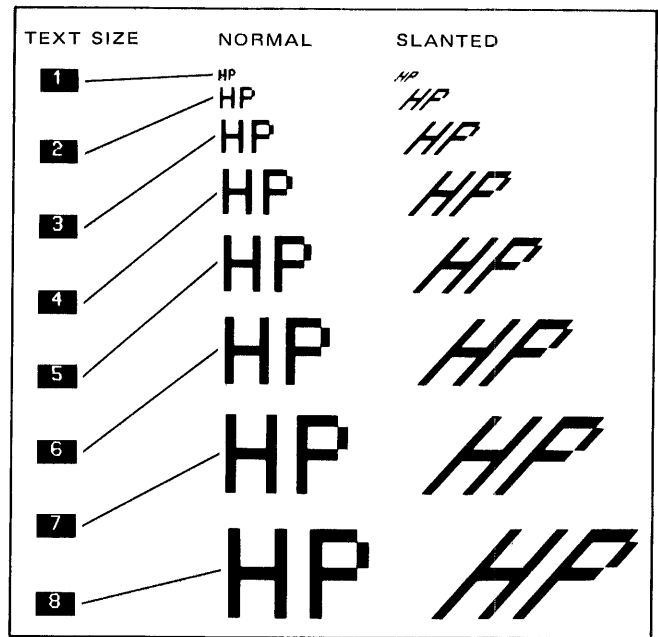


Figure 3-11. Graphics Text Sizes

**TEXT DIRECTION.** This command uses the ASCII characters 1 through 4 to specify the text orientation (see figure 3-12). This also changes the direction of line feed, carriage return, and backspace.

- 1 — Normal (upright, the default)
- 2 — Rotated 90 degrees counter clockwise
- 3 — Rotated 180 degrees counterclockwise (inverted)
- 4 — Rotated 270 degrees counter clockwise

Set Graphics  
Text Orientation:

```
ESC * m <orientation> n
```

**SLANT.** The graphics text characters can be slanted 45 degrees for an italics effect.

Turn On Graphics  
Text Slant:

```
ESC * m o
```

Turn Off Graphics  
Text Slant:

```
ESC * m p
```

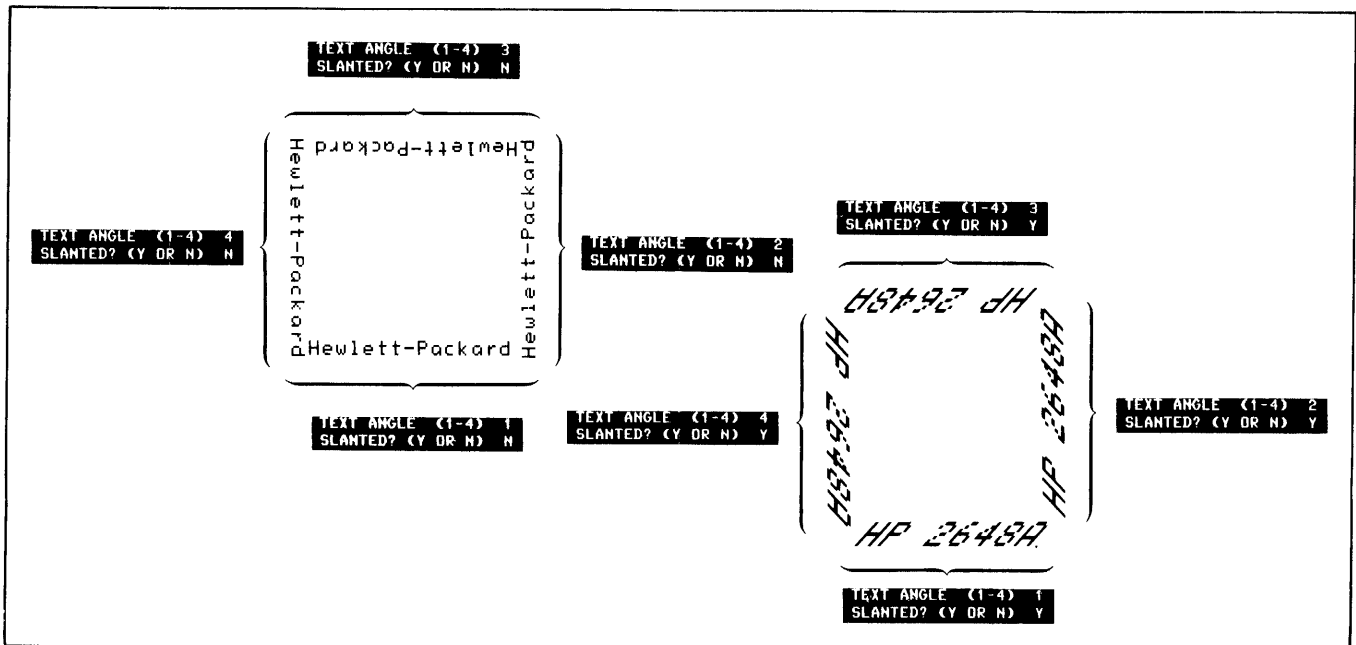


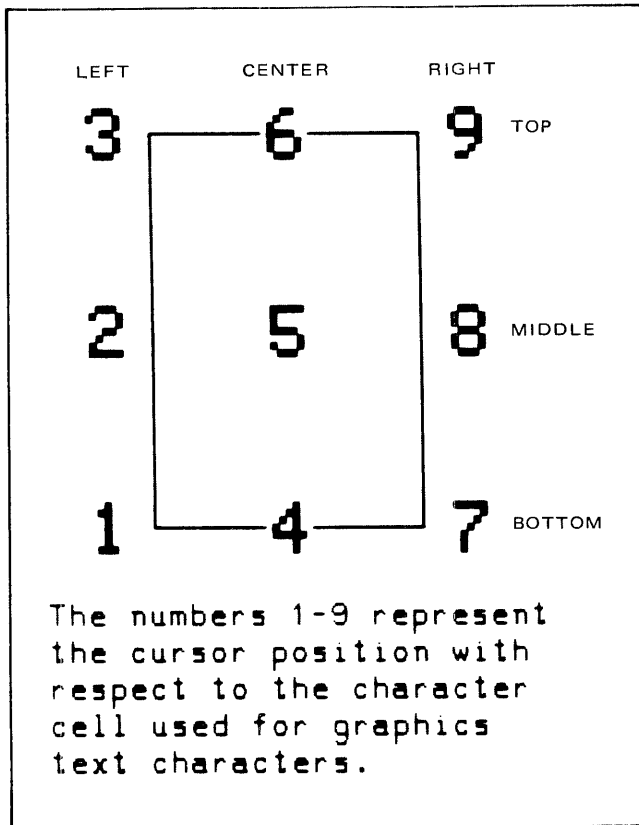
Figure 3-12. Graphics Text Direction

**JUSTIFICATION/ORIGIN.** Text strings can be automatically right or left justified, or centered about a specified point. An ASCII character 1 through 9 indicates the origin (justification and base line) for characters with respect to the current pen position. This function is useful when drawing labels. (Refer to the Label command.)

Set Graphics  
Text Justification:

```
ESC * m <origin> q
```

If text is left justified, the current pen position is the left margin. Center causes the label to be centered on the pen position. Right justify selects the pen position as the right margin. Bottom, middle, and top select the base line for the line of text.



For example, if text was to be right justified and set with a base line on top of the normal character position, the number "9" would be used. Figure 3-13 illustrates the various text positions.

When centering or right justification is used, the text strings are buffered (stored) until all of the characters in the string have been received. The string end is detected by a CR or LF. The string is not displayed until the CR or LF is received. This may be confusing when entering text from the keyboard. The maximum length of a string when center or right justifying is 80 characters (not including the CR(LF)). In all cases, data written beyond the edge of the screen is lost. There is no automatic RETURN when the screen boundry is reached.

**TURNING GRAPHICS TEXT ON AND OFF.** Graphics text mode can be turned on or off from a program. These two commands use the ESC \* d sequence but are discussed here under graphics text for completeness.

**On.** This command will cause Graphics Text Mode to be turned on. All displayable characters will be stored in the graphics memory. If the command is entered from the keyboard using the **SHIFT** **TEXT** keys the graphics cursor is turned on to indicate the position where the next character will be displayed. The drawing mode is initially set to jam mode to permit overstrike replacement of characters. A different mode, such as set or complement, can be selected at any time.

Text is drawn using the current text assignments for size and orientation. Graphics text mode accepts CR, LF, BS, HT, and VT as control characters. The **←**, **→**, **↑**, and **↓** keys can be used to position the graphics cursor in character increments. Graphics text cannot be used in AUTOPLLOT mode. If the Graphics Text On command is received it will be ignored.

Turn On Graphics  
Text Mode:

```
ESC * d s
```

If the graphics cursor is moved or a DRAW or MOVE command is executed, the graphics text margin is moved to the new cursor or pen position.

Characters are drawn using the current drawing mode (set, clear, or jam). If set mode is used, entering a character, backspacing, and entering a second character causes an overstrike. If jam mode is used, the new character will replace the old character.

If a lower case "s" is used, additional escape parameters can be appended to the sequence. Otherwise the next characters will be routed to the graphics memory.

**Examples:**

```
ESC * d s k 100,100 o B — set graphics memory
                               position cursor at 100,100
                               turn on cursor
```

```
ESC * d S This is a text string
```

**Off.** This sequence turns off graphics text mode and restores normal alphanumeric operation.

Turn Off Graphics  
Text Mode:

```
ESC * d t
```

**GRAPHICS TEXT STATUS.** You can check the current text settings with a graphics text status request. Refer to Section VI, Status for additional information.



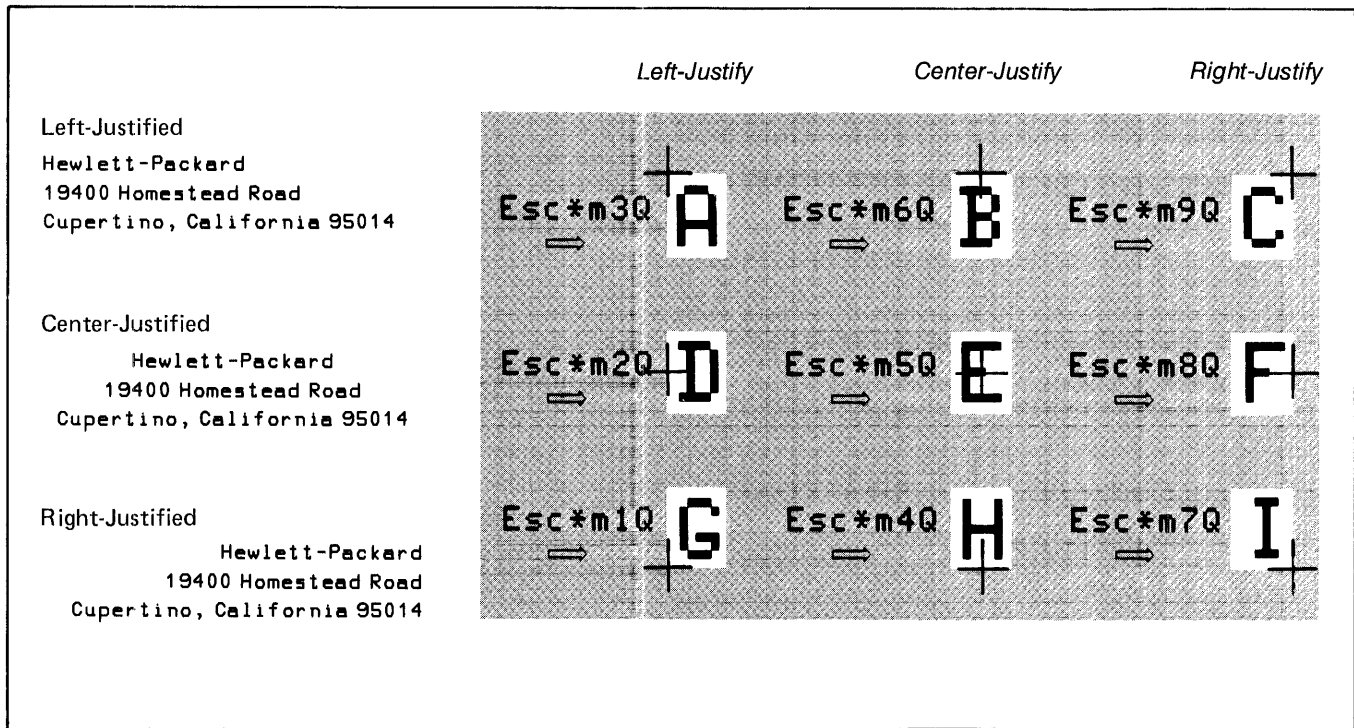


Figure 3-13. Graphics Text Justification

**LABEL.** This sequence is used to send a single record of graphics text to the terminal. The characters are stored in the graphics memory using the current text size, angle, slant, and justification. The label is drawn beginning at the current pen position.

Graphics  
Text Label: `Esc * 1 <text string> Esc (LF)`

The record must end with a CR, LF, or both. A CR moves the pen to its original position when the label command was first received. An LF moves the pen down one line (character spacing). Note that the actual directions moved following a CR or LF depend on the text orientation selected.

The maximum record length is 80 characters, not including the ESC \* 1 preamble or the CR(LF).

**Example:** `Esc * 1 This is a sample label Esc`

## AUTO PLOT

Autoplot is a feature which allows the terminal to plot tabular data without any graphics support in the host computer, and with minimal knowledge of graphics by the user. It produces immediate graphics output from any of the following:

- Programs that generate numeric data in tabular form
- Data stored on a cartridge tape
- Data entered locally from the keyboard and stored in display memory

Autoplot makes the terminal very useful in business applications where statistical and accounting data already exists in tabular form. It eliminates the need to develop and support special graphics programs.

To use Autoplot you must tell the terminal:

1. How many columns of data there are.
2. Which column contains the X (abscissa) data, and which contains the Y (ordinate) data.
3. What the minimum and maximum values are, so that the plot can be scaled.

In addition, there are several optional capabilities that can be selected. These options are selected by entering format parameters in an Autoplot Menu. The Autoplot Menu then controls the drawing of axes and the scaling and plotting of data. The menu can be loaded programmatically or from the keyboard or cartridge tapes.

### AUTO PLOT Menu

Sending an ESC \* a f to the terminal (or pressing **SHIFT** **AUTO PLOT MENU**) displays the Autoplot Menu. A blank menu is shown in figure 3-14. You enter the parameters for the plot by completing the specifications in the menu. It is not necessary to display the menu in order to enter plot parameters.

When loading the menu, only numeric type characters are accepted, (0-9, +, -, ., and E). Short fields in the menu (A1-A4, and C1-C4) indicate integer values only. The longer fields (A5-A8 and B1-B4) accept integer, floating point, and exponential notation.

The menu is divided into three parts: Plot Specification, Axes Specification, and Plot Options.

#### A. Plot Specification.

The Plot Specification specifies the data parameters and type of line used for the plot. Items 1 thru 3 specify the data parameters, item 4 specifies the line type, and items 5 thru 8 specify scale limits of the graph.

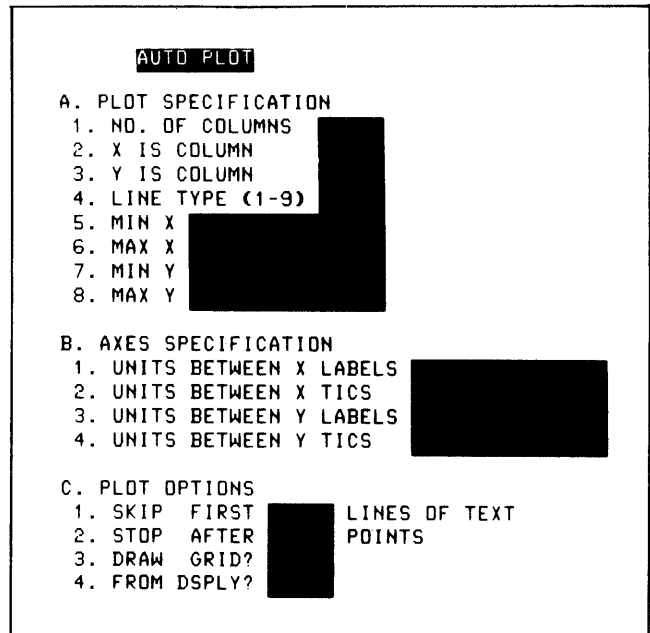
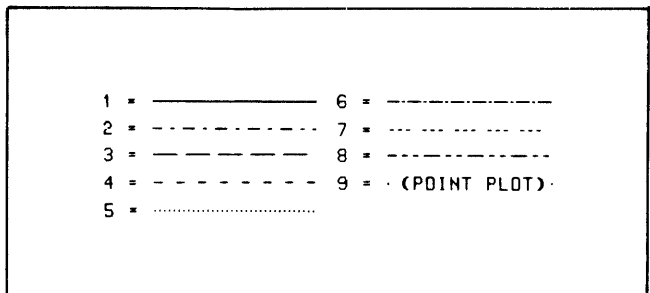


Figure 3-14. Autoplot Menu

1. NO. OF COLUMNS. Specifies the number of columns used to contain the data. (In the example, the number of columns is 5.) It is possible to force the autoplot scanner to plot selected pairs of data points by changing the number of columns. If twice the number of columns is selected for instance, then every other line of data would be plotted.
2. X IS COLUMN. Specifies which column no. will be used for the X values. (In the example, column 1 is used for the X values.)
3. Y IS COLUMN. Specifies which column no. will be used for the Y values. (In the multiplot example, columns 2 thru 5 are used in succession for each plot.
4. LINE TYPE (1-9). Specifies the type of line to be used to show the plotted data. The different line types allow you to distinguish between multiple plots on the same graph.



5. MIN X. Specifies the minimum value to be assigned to the X-axis (the left side of the graph). (In the example, 1965 is the minimum value.)

6. **MAX X.** Specifies the maximum value to be assigned to the X-axis (the right side of the graph). (In the example, 1975 is the maximum value.)
7. **MIN Y.** Specifies the minimum value to be assigned to the Y-axis (the bottom of the graph). (In the example, 0 is the minimum value.)
8. **MAX Y.** Specifies the maximum value to be assigned to the Y-axis (the top of the graph). (In the example, 100 is the maximum value.)

## B. Axes Specification

The Axes Specification tells where the "tic" marks will appear on the graph and which tic marks will have labels (i.e., numeric values assigned to certain tic marks).

1. **UNITS BETWEEN X LABELS.** Specifies units between labels on the x-axis. (In the example, a "1" is used to place a label at each tic mark; 1965, 1966, 1967, etc.)
2. **UNITS BETWEEN X TICS.** Specifies how many tic marks will be used on the x-axis (i.e., how many units each tic mark will represent.) (In the example, tic marks are desired at every point for the year.)
3. **UNITS BETWEEN Y LABELS.** Specifies units between labels on the y-axis. (In the example, "10" is entered for labels at 10 units, 20 units, 30 units, etc.)
4. **UNITS BETWEEN Y TICS.** Specifies how many tic marks will be used on the y-axis (i.e., how many units each tic mark will represent.) In the example, tic marks are desired at every 5 units, therefore, "5" is specified.

The formats for the tic labels are determined by the values given for Units Between Labels (menu items B-1 and B-3). If the tic spacing is an integer, the label value is rounded to the nearest integer, and displayed without a decimal point. If the value given uses a decimal point, the label value is displayed with the same number of decimal places. If the label value is too small (less than .000001) or too large (greater than 9999999) exponential format is used.

## C. Plot Options



The Plot Options specify optional parameters that can be entered to accommodate special data or format requirements.

1. **SKIP FIRST      LINES OF TEXT.** If the data to be received has header lines (i.e., the first lines of data do not contain plot values), you must specify how many lines contain non-plot numeric data so that these lines will be skipped over before accepting valid data. Skip lines does not work when plotting from the display. In

this case you must position the cursor to the first data column. Note that the terminal will normally skip over alphabetic data automatically.







2. **STOP AFTER      POINTS.** If you do not want to plot all of the of data, you can specify how many points to plot before terminating.
3. **DRAW GRID?** You can specify a grid pattern connecting labeled tics in the Axes Specification to make reading points in the center of the plot pattern easier. Enter a non-zero number in the space provided (any number will be interpreted as "yes").
4. **PLOT FROM DISPLAY?** If the data to be plotted is contained in display memory, you must enter a non-zero number in the space provided (any number will be interpreted as "yes"). Entering a blank or zero causes the data read from the data comm or a cartridge tape to be plotted. If the data is from a tape, you must use a read tape command (from the computer or the keyboard) to enter the data.

The only keyboard functions that work when the menu is displayed are the alphanumeric cursor keys, home up, home down, tab, back tab, CR, LF, and clear display (clears field from cursor), record, and enter.

If the menu was displayed it must be removed before autoplot can be enabled. Remove the menu with ESC \* a g (or press  ). A completed menu can be cleared with ESC \* a d.

## Local (User) Control

To use Autoplot locally a user would normally perform the following:

- Select the Autoplot Menu (by pressing  ), and enter the parameters for the plot.
- Draw the axes (by pressing  ). (The data can be plotted without drawing the axes, but the plotted values would be difficult to interpret.)
- Plot the data (by pressing  .

A more detailed explanation of user control of Autoplot is given in the User's Manual.

## Program Control

The following paragraphs describe the commands used to control the Autoplot function from a program. A summary of the commands is given in table 3-12. These escape sequences load the Autoplot Menu, turn Autoplot on or off, and cause the axes to be drawn. ASCII data, as generated

Table 3-12. Autoplot Commands

Command	Code	Comments	
⌘ * a <commands>			
START AUTO PLOT	a	(no parameters). If an error condition occurs because a maximum value is less than or equal to a minimum, an error message will be displayed.	
STOP AUTO PLOT	b	(no parameters).	
DRAW AUTO PLOT AXES	c	(no parameters). If an error occurs because too many tics were specified, an error message will be displayed.	
CLEAR MENU	d	(no parameters). All menu fields are cleared.	
DISPLAY AUTO PLOT MENU	f	(no parameters).	
TURN OFF AUTO PLOT MENU	g	(no parameters).	
NUMBER OF COLUMNS	<byte1>...<byte5> h	Not all of the indicated bytes need be entered. The commands indicate the maximum number of bytes allowed for each menu item.	
COLUMN FOR X DATA	<byte1>...<byte5> i		
COLUMN FOR Y DATA	<byte1>...<byte5> j		
LINE TYPE	<byte1>...<byte5> k		
MIN X	<byte1>...<byte15> l		
MAX X	<byte1>...<byte15> m		
MIN Y	<byte1>...<byte15> n		
MAX Y	<byte1>...<byte15> o		
UNITS BETWEEN X LABELS	<byte1>...<byte15> p		
UNITS BETWEEN X TICS	<byte1>...<byte5> q		
UNITS BETWEEN Y LABELS	<byte1>...<byte15> r		
UNITS BETWEEN Y TICS	<byte1>...<byte5> s		
SKIP HEADING LINES	<byte1>...<byte5> t		
NUMBER OF POINTS	<byte1>...<byte5> u		
DRAW GRID	<byte1>...<byte5> v		
PLOT FROM DISPLAY	<byte1>...<byte5> w		
NOP	z		(no parameters).

by a simple print statement, is accepted for loading menu fields. Blanks are ignored. In order to accept exponential notation, capital E is not treated as a command, and does not terminate the escape sequence. Parameters for integer fields cannot be longer than five characters, disregarding blanks. Floating point fields can be up to 15 characters long, and accept integer, floating point, or exponential notation. Commands which load menu fields clear the field before loading the new value. (A command without any parameters simply clears the selected field.)

The following escape sequence loads the menu in figure 3-16.

```
⌘*a5h1i2j1k1965l1975m0n100o1p1q10r5s1u1v1w
```

### Drawing The Axes

To draw the axes specified in the Autoplot Menu send ESC \* a C (or press **SHIFT** **AXES**). This causes the terminal to draw the axes, tics, and labels. If a non-zero entry is made for item c-3 in the menu, a grid will be drawn for all labeled tics.

### Plotting The Data

To plot the source data, send ESC \* a A (or press **AUTO PLOT**). If the data is to be plotted from the display, position the alphanumeric cursor to the first data entry to be plotted before giving the AUTO PLOT command. The terminal will begin scanning through the data, reading the X and Y fields, and plotting the data. Data in graphics memory is

not scanned by Autoplot (i.e. numeric data stored as graphics text will not be plotted). Any Autoplot commands (ESC \* a ) will terminate Autoplot.

If a large number of data points are to be plotted from a computer, it may be necessary to use pad characters or a handshake to prevent a data overrun. Refer to Section V, Data Communications for information on high speed communications.

**INPUT DATA FORMAT.** When Autoplot is turned on, a scan is made for valid numeric values as characters enter the terminal. The first data found is assumed to be column one, the second column two, and so on until the limit specified (NUMBER OF COLUMNS) is reached. Then column one is again assumed, and so on. As the columns designated X and Y are found, they are highlighted in inverse video, the values are scaled, and the data point is plotted.

The plot data is highlighted in inverse video as it is plotted. Since the display enhancement uses up memory space some of the data may be rolled out of memory to make room for the new data. If you need to retain the data in display memory you can use "Data Logging" to record data to a cartridge tape or a printer.

Note that only the relative position in the data stream determines which column the terminal thinks a number is in, not its location on the screen. This means that titles, blank lines, text, even numbers longer than 80 characters which wrap around to the next line, do not affect the Autoplot scan. However, dates, page numbers, and other short numbers will be interpreted as belonging in data columns. Once Autoplot is turned on, any numbers are assumed to be data. It will be necessary to edit such extraneous values out to prevent them from being read as erroneous data and invalidating the plot. There are several ways to do this:

1. Edit with the Cartridge Tapes. Record the data on tape, use edit mode to remove the extraneous values from the tape, then plot the data from the edited tape.
2. Skip heading lines. This Autoplot Menu parameter allows you to specify a number of lines to be initially skipped before the Autoplot scan starts. (However, you cannot use this in cases where there are heading numbers on every page.)

3. Plot from the display memory. The normal alphanumeric display memory can be selected as the source of data. Once it contains the desired data, it can be edited using the local editing functions (Insert/Delete Character, Insert/Delete Line, etc.).

When plotting from the display you must insure that all of the plot data will fit in the available display memory.

Keyboard entries are not plotted directly by Autoplot. When Autoplot is initially turned on, and no numbers have been plotted yet, any keyboard entry is ignored by the Autoplot scanner. However, once the first point has been plotted, any keyboard entry terminates Autoplot.

When plotting from the display, a number cannot be broken across two lines, or it will be interpreted as two separate numbers. For example, 125.3 on two lines might be read as follows:

```
125 = 125 and 0.3
.3
```

This is not a problem when plotting data from the data comm interface or a cartridge tape, since the Autoplot scanner does not care where on the screen the characters are entered.

To facilitate the plotting of financial data, "\$" and "," imbedded in numbers are ignored. In addition, trailing - or + signs, generated by some COBOL programs, are recognized.

Any nonnumeric displayable character delimits a number. Carriage return also acts as a delimiter. Numbers in escape sequences are executed as a part of the escape sequence and are not plotted (unless they are displayed using display functions). The following examples show how Autoplot scans numeric strings. To avoid problems, you must separate data with blanks or alphabetic characters to be sure of getting the proper result.

The precision of Autoplot is limited to five significant figures although numbers containing up to 20 characters can be used. If a number is longer, the entire number will be scanned, but only the first 20 characters will be used. The largest allowable number is approximately  $1.0 \times 10^{**30}$ . The smallest is approximately  $1.0 \times 10^{**-30}$ .

Input Data	Result	Comments
123x456#78 9	→ 123 456 78 9	(non numeric character delimits data)
123- 456	→ -123 456	(trailing - assumed)
123 -456	→ 123 -456	(- goes with second number)
-\$123,456	→ -123456	(\$ and , ignored)
\$123,456,\$789	→ 123456 789	(as expected)
- 123	→ 123	(can't have blanks after sign)
123E 02	→ 123 2	(can't have blanks between E and exponent)
123E+02	→ 123E02	(correct result)

**MULTIPLE PLOTS.** Multiple plots can be made on the same set of axes by changing the appropriate data columns and running the data through as many times as necessary. (This was accomplished in the multiplot example by changing the column number in the menu. Different dot-dash patterns can also be selected to differentiate the plots. This is easiest to do when plotting from the display, as all that is necessary to reposition the cursor and start Autoplot (instead of rewinding and rereading a tape file, etc).

**MULTIPLIER EXAMPLE.** This example will illustrate the use of Autoplot to make several plots of the data in display memory. Figure 3-15 shows some typical financial data. A plot of total sales, polarizers, mashers and converters versus year is desired.

1. Display the Menu. ESC \* a f ( **SHIFT** **AUTOPLOT MENU** ) brings up the menu shown in figure 3-14. This is optional and is not required to enter menu parameters programmatically. It is useful if you require operator input for plot parameters.
2. Enter the Plot Specifications.
  - a. Specify the Data Columns (lines 1-3). There are five columns of data to be plotted; therefore, enter "5". To plot calendar time (first column) versus total sales (fifth column), set the X column to "1" and the Y column to "5".
  - b. Select the Line Type (line 4). For the first plot, enter "1".
  - c. Set the plot range (lines 5-8). X ranges from 1965 to 1975. Y actually ranges from 5.1 to 64.5, but the range specified is from 0 to 100. In general, the min

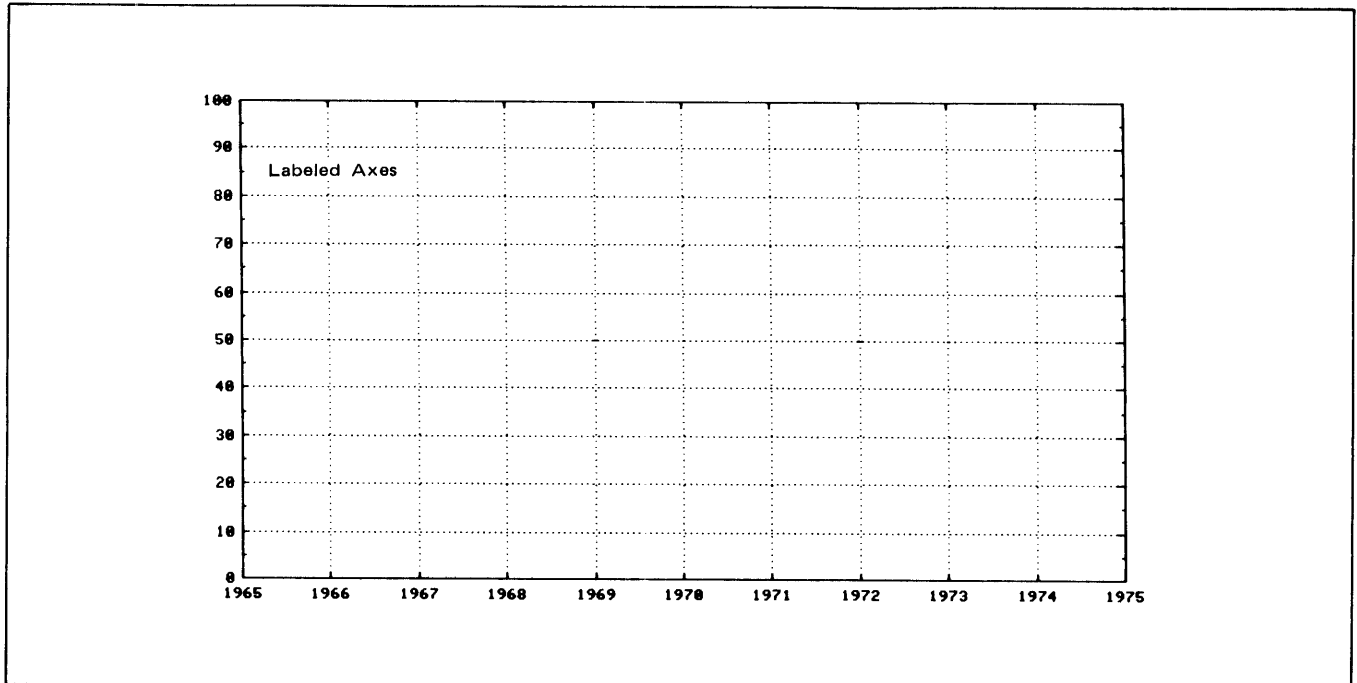
and max values should be chosen as a reasonable range, not necessarily the actual min and max values in the data column. In this example, the Y range was chosen to allow all four data columns to be plotted against the same scale, so that the plots would be in the proper proportion.

3. Select the Axes Specification (Tic Spacing).
  - a. Set the unlabeled (minor) tic interval. Tics always start at 0. The value specified in lines 2 and 4 give the intervals between tics. The x tic spacing of "1" places tics at 0,1,2, . . . 1965,1966, . . . 1974, and 1975. (Only those tics within the minimum to maximum range are actually drawn.) The Y tic spacing of 5 units puts tics at 0,5,10, . . . 90,95, and 100. To further illustrate, say that the range was -1 to 1, and the tic spacing was 0.1. This would put tics at -1., -0.9, . . . -0.1, 0, .1, .2, . . . .9, and 1.0.
  - b. Set the labeling interval. The values in lines B-1 and B-3 tell how many units separate labels (and major tics). The x label interval of "1" says to space labels one unit apart. The Y label interval of "2" says to place labels every two units. Labeled (major) tics are twice as long as unlabeled (minor) tics. If a grid is specified (line C-3), the grid lines are placed at the labeled (major) tic marks.
4. Select the Plot Options.
  - a. Skip first X lines of text. Since we are plotting from the display and can position the cursor at the beginning of the data columns, we can leave this blank.
  - b. Stop After X Points. Since we are plotting all points in the data columns, we can leave this blank, also.

XYZ WIDGET COMPANY				
EARNINGS BY PRODUCT LINE, 1965-1975				
YEAR	POLARIZERS	HELICAL CONVERTERS	AUTOMATIC MASHERS	TOTAL
1965	\$ 5.1	-0-	-0-	\$ 5.1
1966	9.3	-0-	-0-	9.3
1967	13.2	3.0	-0-	16.2
1968	16.9	5.9	-0-	22.8
1969	15.2	6.3	1.3	22.8
1970	19.7	10.1	4.5	34.3
1971	23.6	12.9	9.3	45.8
1972	15.2	16.8	12.7	44.7
1973	8.3	20.9	19.0	48.2
1974	-0-	27.5	22.4	49.9
1975	-0-	35.4	29.1	64.5

Figure 3-15. Sample Financial Data

- c. Draw Grid. Entering a non-zero integer specifies that we want a grid at the labeled tic marks.
- d. From Display?. Enter a "1" to plot the data in display memory.
- 5. Remove the Menu. If the menu was displayed, return the normal display with ESC \* a g.
- 6. Draw the Axes. ESC \* a c draws the axes, tics, labels, and grid (if specified in the menu) as shown in figure 3-16.



- 7. Plot the data. Position the cursor above the first data line, and send ESC \* a a. Since plot from display was selected, the cursor immediately moves through the data, picks out the X and Y fields, and plots the data.

The total sales column has now been plotted. To plot the remaining data, as shown in figure 3-17, it is necessary to run Autoplot three more times. Only the Y IS COLUMN (line A-3) and the LINE TYPE (line A-4) fields in the menu need be changed. ESC \* a 3j 2k A etc.

```

AUTO PLOT
A. PLOT SPECIFICATION
1. NO. OF COLUMNS      5
2. X IS COLUMN          1
3. Y IS COLUMN          2
4. LINE TYPE (1-9)     1
5. MIN X                1965
6. MAX X                1975
7. MIN Y                0
8. MAX Y                100

B. AXES SPECIFICATION
1. UNITS BETWEEN X LABELS  1
2. UNITS BETWEEN X TICS   1
3. UNITS BETWEEN Y LABELS 10
4. UNITS BETWEEN Y TICS   5

C. PLOT OPTIONS
1. SKIP FIRST          LINES OF TEXT
2. STOP AFTER          POINTS
3. DRAW GRID?         1
4. FROM DSPLY?        1
    
```

**LABELING PLOTTED DATA.** After the data is plotted, labels can be added either by the operator or under program control. The labels can be used to identify individual curves or add notes. Refer to the discussion of Graphics Text for additional information.

**PLOTTING FROM DATA COMM INTERFACE OR CARTRIDGE TAPE.** To further illustrate the use of autoplot, assume that the data in figure 3-15 is to be plotted as it arrives from the datacom (or cartridge tape). The menu is loaded as before, with blanks or "0" entered in the PLOT FROM DISPLAY field (line C-4). The start AUTO PLOT command is sent. To indicate that autoplot is active, the yellow LED above the gold key will blink. The proper command sequence to run a program (if from data comm) or read a tape (if from a cartridge tape) is then entered locally or sent from the computer. As the lines of data enter the terminal they will be scanned and plotted.

Figure 3-16. Completed Menu For Multiplot Example

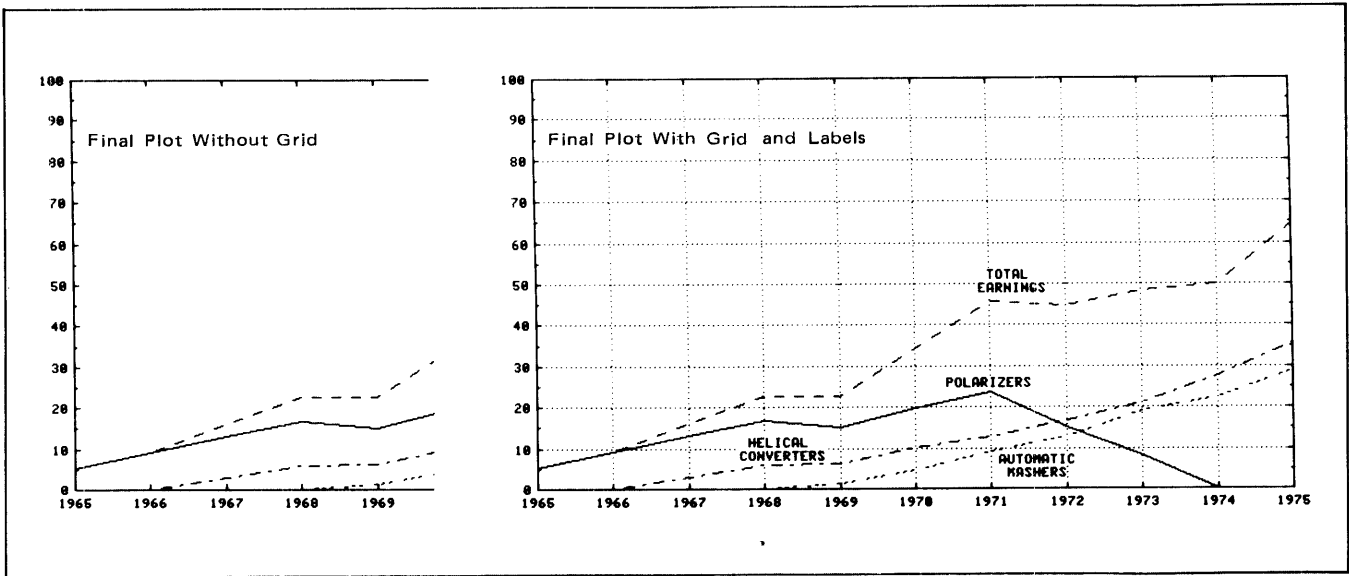


Figure 3-17. Multiplot Results

The STOP command or number of plot points terminate the plot. To plot all four columns it will be necessary to stop autoplot after each plot, change the menu, and rerun the data to plot each column. A program or soft key could be used to automatically update the menu and rerun the data. Figure 3-18 illustrates how a function might be plotted from a computer.

**CURVE SHADING WITH AUTO PLOT.** You can approximate curve shading while in Autoplot Mode by selecting an appropriate drawing scale and plotting additional data points. Figure 3-19 illustrates a technique for shading curves with Autoplot.

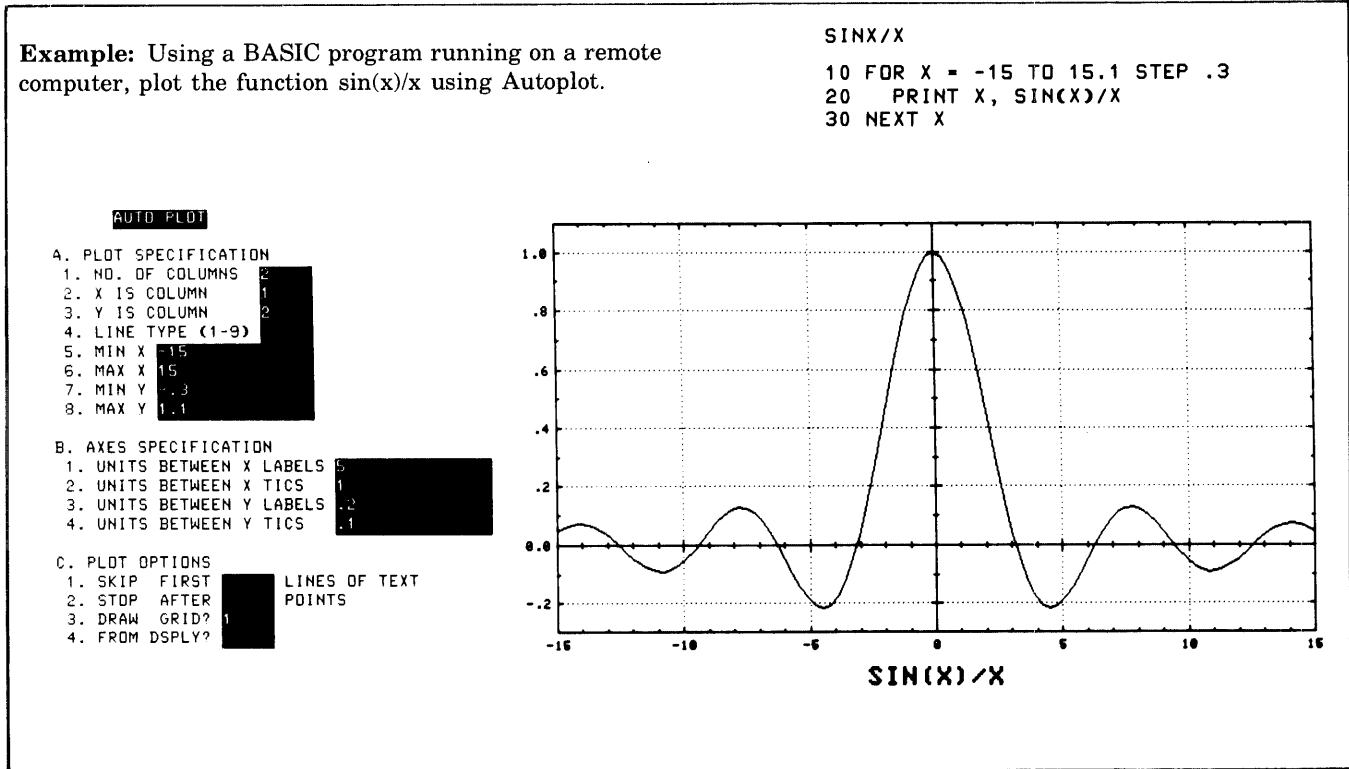


Figure 3-18. Autoplot Example Using Computer Generated Data



**Example:** Using a program running on a remote computer, plot the SIN(X)/X function from the previous example, shading the area under the curve can be accomplished by plotting two additional vectors following each data point. The first is a vector back to the X-axis, the other is a horizontal increment of delta X to insure that a vertical line will be drawn to the next data point.

```
10 FOR X=-15 TO 15.1 STEP .05
20 PRINT X, SIN(X)/X; X,0; X+.05,0
30 NEXT X
```

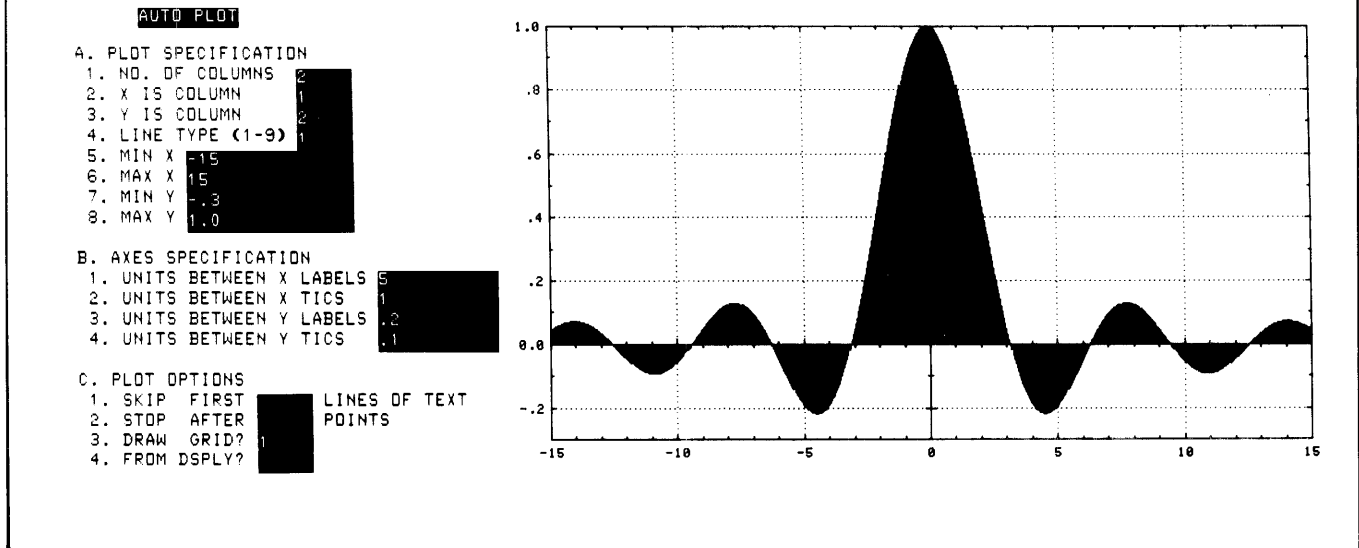


Figure 3-19. Curve Shading With Autoplot

### Using Autoplot With Other Graphics Functions

When the Draw Axes or Autoplot functions are executed they change some of the graphics parameters. These new values remain in effect after the Draw Axes and Autoplot functions have completed execution. Table 3-13 lists these new parameter values. It may be necessary to reset these parameters before performing other graphics functions.

#### NOTE

The Draw Axes and Autoplot functions are terminated when an ESC \* a, ESC \* p, or ESC \* s command sequence is received.

In addition, Autoplot turns Graphics Text Mode off. If Autoplot is on, Graphics Text Mode cannot be turned on. Data in graphics memory is not scanned by Autoplot (i.e. numeric data stored as graphics text will not be plotted).

Table 3-13. Draw Axes and Autoplot Parameters

PARAMETER	VALUE
Draw Axes	
Drawing Mode	set (1)
Graphics Text	off
Graphics Video	on
Alphanumeric Video	on
Autoplot	
Text Origin	left, bottom (1)
Text Angle	0° (1)
Text Slant	Off
Text Size	1
Line Type	menu value (unchanged if menu 0 or blank)

### Autoplot Errors

There are three error conditions. The first occurs if a maximum X or Y is less than or equal to the minimum. This is detected when autoplot is enabled or axes are drawn. The second occurs if more than 1000 tic marks (either X or Y) are requested. The third occurs if more than 100 X or Y labels are requested. These conditions are detected when the terminal draws the axes. If an error is detected, an error message is displayed at the terminal and the keyboard is locked until cleared by the operator pressing the **RETURN** key.

There may be variations in labeling due to roundoff errors. These variations do not generate an error message and can usually be corrected by changing the MIN and MAX values in the plot specification.

### COMPATIBILITY MODE

Compatibility Mode allows the terminal to plot data intended for a terminal using a display with 1024 by 1024 addressable points. This mode makes it possible to use graphics programs developed for use with other graphics terminals with a minimum of reprogramming.

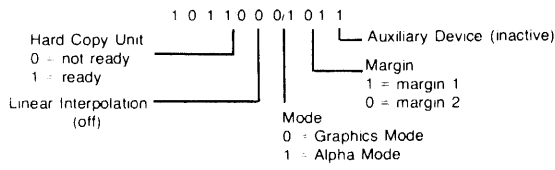
The terminal operates in two submodes while in Compatibility Mode. In Alphanumeric mode the terminal simply displays alphanumeric data on the screen as in normal operation. In Graphics mode the terminal responds to alphanumeric data as vector coordinates. Normally the terminal will be switched between these modes to display messages, plot graphics figures, and then display additional messages. These modes are controlled with several control sequences. (These sequences are ignored or acted on differently if the terminal is not set for Compatibility Mode.) Table 3-16 lists the terminal's responses to Compatibility Mode control sequences.

If delays are required, the baud rate can be lowered or fill characters added to prevent data loss when operating the terminal at high speeds. Refer to Section V, Data Communications.

Vectors are drawn using the current line type and line drawing mode. This gives you the capability of drawing dotted and dashed lines, etc. by changing the program to send the additional escape sequences. In general, all of the normal features of the terminal (display enhancements, tape control, etc.) are available only in the Alphanumeric mode.

Compatibility Mode is turned on by selecting either scaled or unscaled operation. Escape sequences controlling Compatibility Mode begin with  $\text{ESC}^*t$ . This preamble is then followed with one or more commands. These commands are listed in table 3-17. As in all other escape sequences, a capital letter ends the sequence. Figure 3-24 contains examples of typical escape sequences.

Table 3-16. Compatibility Mode Control Sequences

CONTROL SEQUENCE	DESCRIPTION	RESPONSE
$\text{ESC}^*s$	Read status and alpha cursor position	$\langle \text{status byte} \rangle \langle \text{HI X} \rangle \langle \text{LO X} \rangle$ $\langle \text{HI Y} \rangle \langle \text{LO Y} \rangle \langle \text{terminator} \rangle$
<div style="text-align: center;">  </div> <p>The terminal will return one of the following characters as the status byte:</p> <ul style="list-style-type: none"> <li>1 - Margin 2, Graphics Mode</li> <li>3 - Margin 1, Graphics Mode</li> <li>5 - Margin 2, Alpha Mode</li> <li>7 - Margin 1, Alpha Mode</li> </ul>		
$\text{ESC}^*S$ (20 ms delay) $\text{ESC}^*s$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*F$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*S$ $\text{ESC}^*S$	Read graphics cursor position Read graphics cursor position when key struck Make hardcopy End graphics mode, clear screen, and home cursor Go into graphics mode (draw vectors) Go into alpha mode Backspace (H <sup>c</sup> ). Moves 1 space left (14 units) Horizontal Tab (I <sup>c</sup> ). Moves 1 space right (14 units) End graphics mode Line Feed (J <sup>c</sup> ). Moves 1 line down (22 units) Vertical Tab (K <sup>c</sup> ). Moves 1 line up (22 units)	$\langle \text{HI X} \rangle \langle \text{LO X} \rangle \langle \text{HI Y} \rangle \langle \text{LO Y} \rangle \langle \text{terminator} \rangle$ $\langle \text{KEY} \rangle \langle \text{HI X} \rangle \langle \text{LO X} \rangle \langle \text{HI Y} \rangle \langle \text{LO Y} \rangle \langle \text{terminator} \rangle$
<p style="text-align: center;"><b>NOTES</b></p> <p>The terminal will normally respond with an <math>\text{ESC}^*k</math> character when an <math>\text{ESC}^*s</math> character is received. Compatibility Mode disables the terminal's <math>\text{ESC}^*k</math> handshake. Compatibility Mode causes most control codes to be ignored.</p> <p>The Read Status, alpha cursor position, and graphic cursor position cause block transfers to the computer system. If the computer system does not use the DC1/DC2 handshake, straps G and H on the Keyboard Interface PCA must be OPEN for these transfers to occur. (Refer to "Multicharacter Transfers" in Section V.)</p>		


## Compatibility Mode Straps

Compatibility Mode operation is controlled by Keyboard Interface switches P and Q. These switches can be set manually or programmatically using the "ESC & s . . ." sequence described in Section II. The P and Q switches determine the terminal's mode of operation after being initialized (power up or full reset). The switches are interpreted as follows:

SWITCHES		DESCRIPTION
(Open=1, Closed=0)		
<b>P</b>	<b>Q</b>	
0	0	Normal graphics operation
0	1	Unscaled Compatibility Mode (expanded data comm buffer) <sup>1</sup>
1	0	Scaled Compatibility Mode (expanded data comm buffer) <sup>1</sup>
1	1	Normal graphics operation (expanded data comm buffer) <sup>1</sup>
<sup>1</sup> To obtain the larger buffer, the P and Q switches must be set physically. Refer to Section V.		

In addition, when in Compatibility Mode, you can select the following optional capabilities:

**GRAPHIC INPUT TERMINATOR.** You can select the terminator sent by the terminal following the input of cursor address information. The terminator can be a CR, CR and EOT, or no terminator.

**PAGE FULL BUSY.** When this strap is in, the keyboard will be locked after the 35th line of text is received from the computer. The terminal can be cleared by pressing the . This strap is ignored in Unscaled Mode.

**PAGE FULL BREAK.** When this strap is in, the terminal will send a 200ms break signal to the computer after the 35th line of text is displayed. The terminal may also be set to BUSY (see Page Full Busy). When out, the strap will cause the cursor to home and the next 35 lines of text to be set with a left margin at x = 256. This strap is ignored in Unscaled Mode.

The commands to control these strap options are listed in table 3-15. Refer to the manual for the replaced graphics terminal for additional information on the operation of these straps and how they should be set.

## Graphic Data

There are differences in display size (720 × 360 for the HP 2648A versus 1024 × 780 for other terminals) and line length (24 lines of 80 characters for the HP 2648A versus 35 lines of 74 characters for other terminals). See figure 3-21.

Table 3-15. Commands for Selecting Compatibility Mode

COMMAND	CODE
TURN SCALED COMPATIBILITY MODE ON (P open)	⌘ & s 1 p 0 Q
TURN UNSCALED COMPATIBILITY MODE ON (Q open)	⌘ & s 0 p 1 Q
TURN COMPATIBILITY MODE OFF (P,Q closed)	⌘ & s 0 p 0 Q
The following commands simulate straps used on other graphics terminals:	
SET GRAPHICS INPUT TERMINATOR STRAP 0 — Carriage return only (Normal position) 1 — Carriage return and EOT 2 — No carriage return, no EOT	⌘ * t <byte1> a
SET PAGE FULL BREAK STRAP 0 — Out (Normal position) 1 — In	⌘ * t <byte1> b
SET PAGE FULL BUSY STRAP 0 — Out (Normal position) 1 — In	⌘ * t <byte1> c
NOP	z

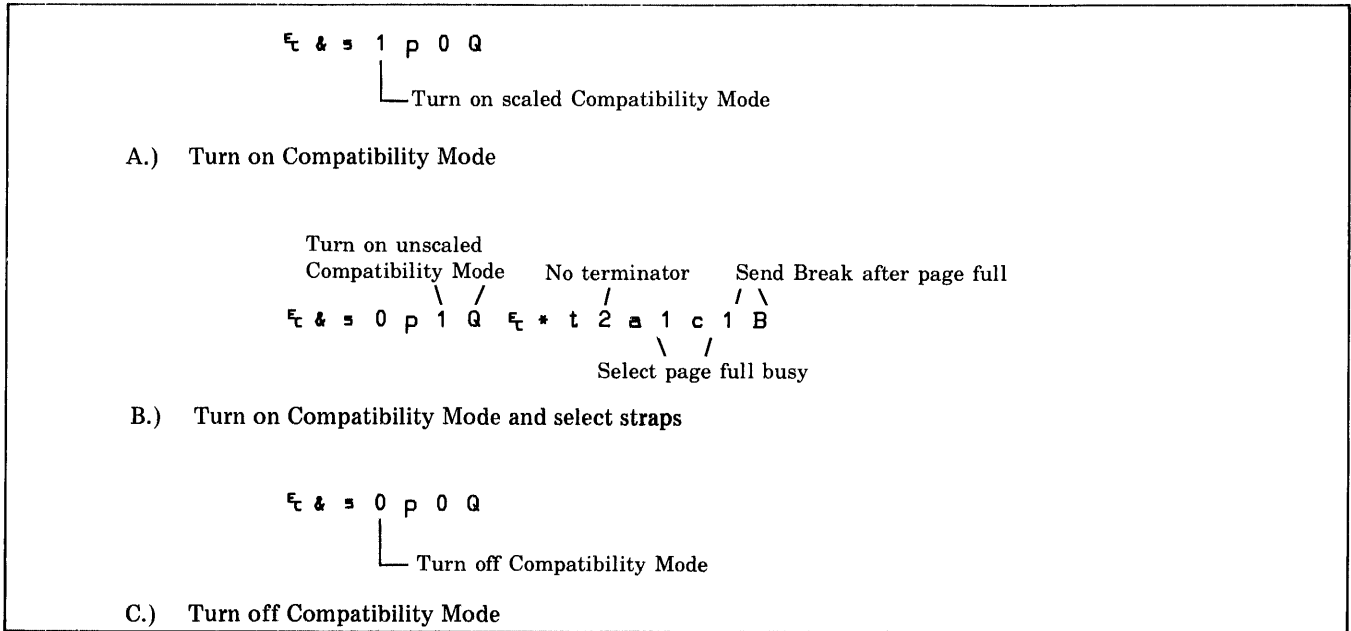


Figure 3-20. Turning on Compatibility Mode

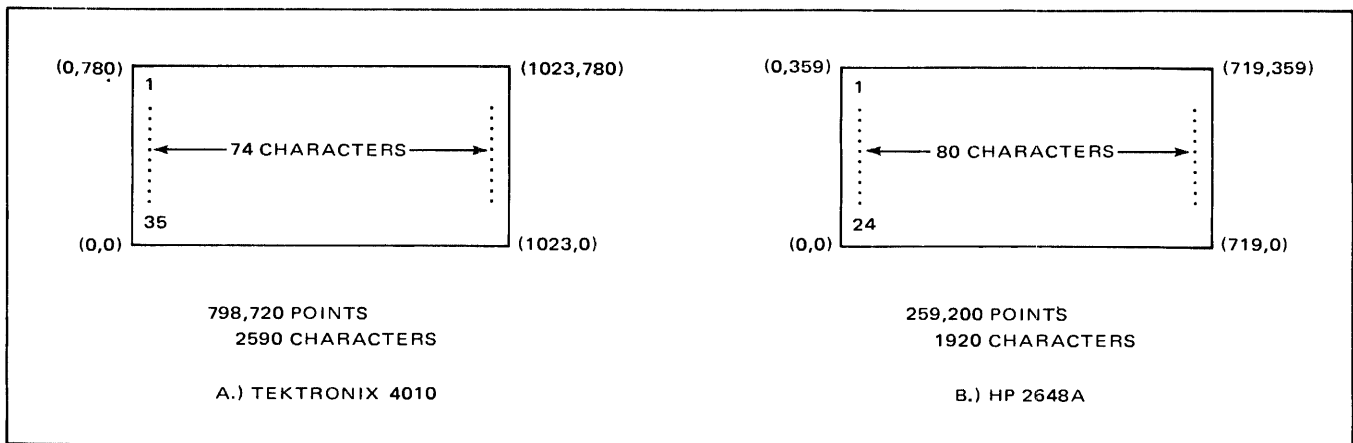


Figure 3-21. Comparison of a Terminal with 1024 × 780 Display and the HP 2648A

Graphic data can be drawn either scaled or unscaled. Scaling divides X coordinates by 2, and Y coordinates by 128/59. This maps the 1024 × 780 display into 512 by 360. This allows a program written for the 1024 × 780 terminal to run unchanged, and still display the entire picture (with some loss in resolution). The image doesn't cover the entire screen (only going to X = 512). The remainder can be used as a dialog area for alphanumeric text (see figure 3-22).

Unscaled mode shows a 720 by 360 subset of the 1024 × 780 picture. The area this covers can be changed by modifying the value of the relocatable origin (and redrawing the picture). The relocatable origin is subtracted from all incoming coordinates in unscaled mode. If this is set to 0,0 (the default) the range X = 0 to 719, Y = 0 to 359 will be displayed (see figure 3-23a.)

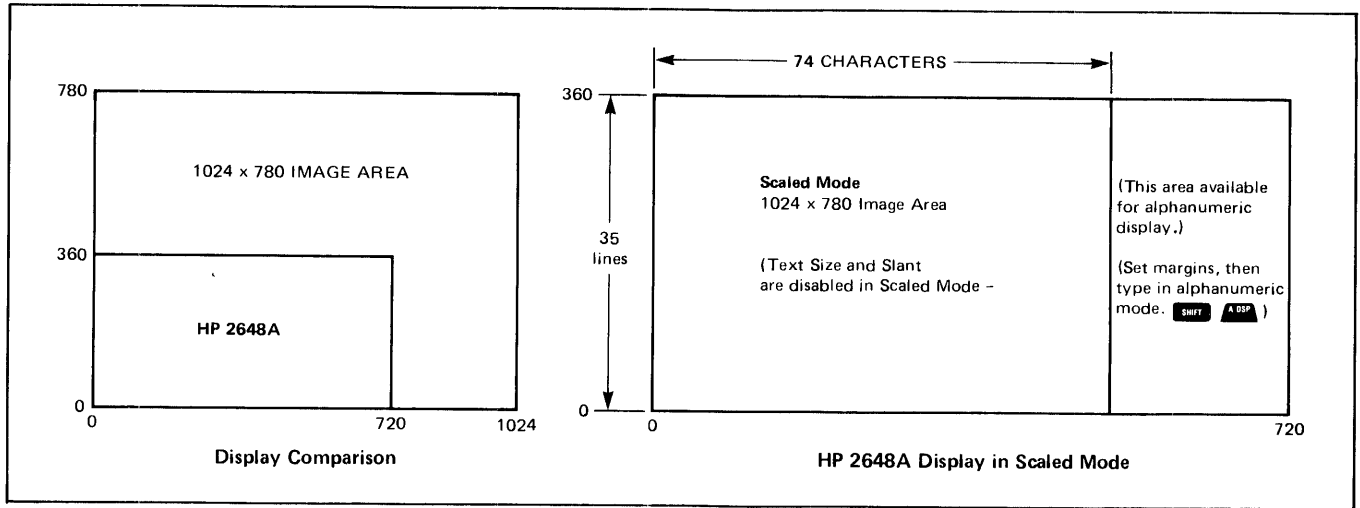


Figure 3-22. Scaled Data

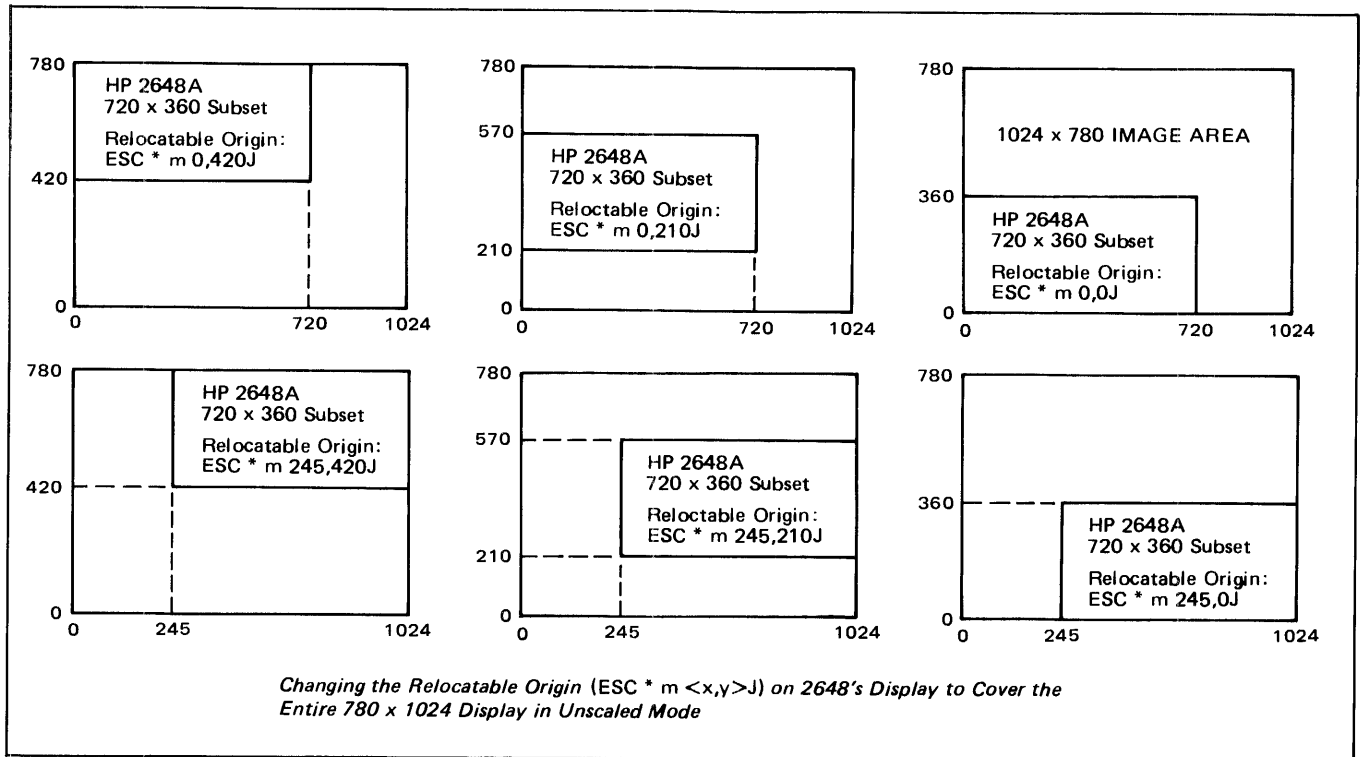


Figure 3-23. Unscaled Data

Setting the origin to 0,360 would cover the area X = 0 to 719, Y = 360 to 719. To display an area larger than 720 x 360, you must change the scaling statements in the program. The advantage of unscaled mode over scaled mode is that unscaled allows you to use the entire available display area (see figure 3-23b).

### Graphics Data Format

In Compatibility mode the graphics data is formatted as two-byte coordinate values. The lower five bits of each byte are used to make a 10 bit (0-1023) coordinate. Data sent to the terminal must have the "Y" coordinate sent first; <Upper Y> <Lower Y> <Upper X> <Lower X>.

When data is returned to the computer (cursor position, etc.), the X coordinate is returned first; <Upper X> <Lower X> <Upper Y> <Lower Y>.

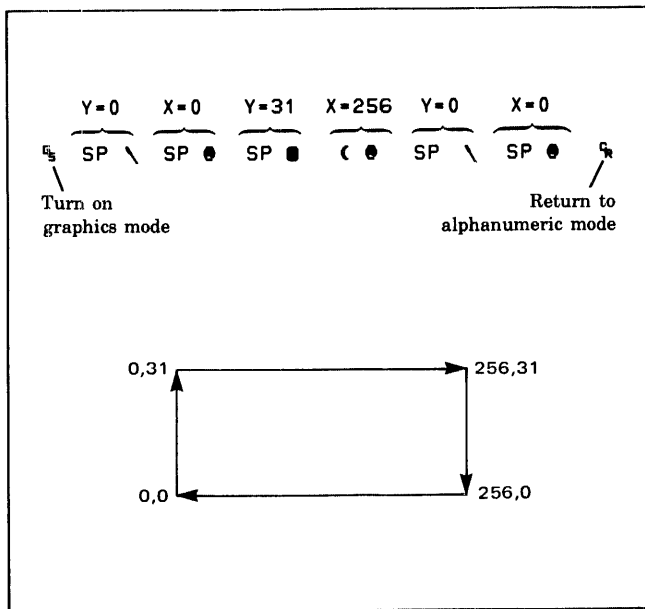
Data bytes sent to the terminal use bits 6 and 7 to indicate the byte is an Upper byte, a lower Y, or a lower X. Bit 8 (parity) is not used.

Bits		
7	6	
0	1	Upper X or Y byte
1	0	Lower X byte
1	1	Lower Y byte

These identifying bits allow you to send only the changed portion of a four byte address. The following data bytes must always be sent:

- Lower X byte
- Any changed byte
- Lower Y byte if the Upper X byte has changed

Table 3-16 can be used to determine address bytes. For example, to plot the points (0,0), (0,31), (256,31), (256,0) the following sequence would be used:



## Text

Text can be placed in either the alphanumeric memory or in the graphics memory. If the terminal is set for alphanumeric text, the text will be sent to the alphanumeric memory. This is generally the most useful, as text can be scrolled, edited, erased, etc. without affecting the graphics image. If you select graphics text (Esc \* d s), text will go into the graphics memory. Text to be written to the graphics memory can be scaled or rotated. (Refer to Graphics Text for additional information.)

When text is written to the graphics memory, the graphics cursor is moved to indicate where the next character will be stored. (The alphanumeric cursor is only used when data is stored in the alphanumeric memory.) This differs from terminals that have only one mode for text and display the graphics cursor only when waiting for graphic input from the user.

**SCALED MODE GRAPHICS TEXT.** In Scaled Mode, text is initially written into the graphics memory, the size is fixed to allow for 35 lines of text. The text angle is set at 0 degrees and unslanted. The text origin is set to the left and bottom. These settings allow the "Page Full" feature to work properly and existing software to run without changes. If you do not require the Page Full feature, you can not change the text settings. You can redirect the text to the alphanumeric memory.

**UNSCALED MODE GRAPHICS TEXT.** In Unscaled Mode, the text size is unchanged and graphics text mode is not initially turned on. Text is stored in the alphanumeric memory unless the graphics text mode is specifically enabled.

## Cartridge Tape Operation In Compatibility Mode

When operating in Compatibility Mode, local cartridge tape READ operations do not automatically append a CR(LF) at the end of each record. This prevents this extra CR from turning off graphic mode while reading graphic data. CR(LF) characters actually recorded on the cartridge tape are read normally. When operating in remote a CR(LF) is appended (if needed) at the end of each record read.



## INTRODUCTION

The terminal's display, cartridge tapé units, and external printer can be controlled by a program executing in a host computer through the use of an escape sequence of the following general form:

`^t&p <commands>`

where each command consists of an optional numeric parameter followed by one of the command characters shown in Table 4-1. The final command in the sequence is identified by an uppercase command character; all prior commands in the sequence use lowercase command characters. The characters "`^t&p`" must not be separated by intervening blanks; blanks occurring subsequently in the escape sequence are ignored.

Within each device control escape sequence, you may specify only one physical operation (such as rewinding, spacing, reading, writing, copying, or comparing). For example, if you wish to programmatically rewind the left tape, rewind the right tape, and then copy a file from one tape to the other, you would have to issue three separate escape sequences. You may, however, include source (s),

unit (u), number of records/files (p), and multiple destination (d) specification commands within an escape sequence containing one of the physical operation commands.

You should not initiate a device control operation (other than a status request) before a previous operation has been completed. For example, after initiating a read operation, the data record must be read by your program before initiating another device control operation; otherwise the read operation may not be executed properly.

Some of the device control operations can be interrupted and prematurely terminated by pressing the **RETURN** key. If a control operation is user-interruptable, it has the following general characteristics:

- Once the operation is initiated the terminal's keyboard is disabled except for the **RETURN** key.
- Pressing the **RETURN** key terminates the operation, returns a completion code to your program indicating that a user interrupt occurred, and reenables the keyboard.
- The completion code (if any) is transmitted when the operation is completed.

Table 4-1. Device Control Command Characters

Command Character		Meaning	Command Character		Meaning
Upper-case	Lower-case		Upper-case	Lower-case	
S	s	Defines the source ("from") device.	^	~	Reads device status (see section VI).
D	d	Defines the destination ("to") device.	R	r	Reads a record or file from the current source device.
C	c	Specifies the desired cartridge tape or external printer control operation.	W	w	Writes a record to the current destination device(s).
U	u	Specifies on which terminal device (left CTU, right CTU, or printer) the specified control operation is to be performed.	B	b	Using the current source and destination device(s), copies or compares one record.
P	p	For cartridge tape spacing operations, specifies the number of records or files to be spaced over. For external printer "skip lines" operations, specifies the number of lines to be skipped over.	F	f	Using the current source and destination device(s), copies or compares one file.
			M	m	Using the current source and destination device(s), copies or compares to the end of medium (copy all or compare all).



- Completion of the operation is indicated by receipt of the completion code or (for read operations) by receipt of all expected data.

Non-interruptable control operations have the following general characteristics:

- The terminal's keyboard is enabled throughout the entire operation.
- The completion code (if any) is transmitted when the operation is initiated.
- Completion of the operation is indicated by the "busy" device status bit (bit 1, byte 1) being cleared to "0".

The following device control operations are always interruptable:

Locate end-of-data (3c)  
 Test CTU (7c)  
 Turn on write-backspace-read mode (9c)  
 Turn off write-backspace-read mode (10c)  
 Copy record (b)  
 Copy file (f)  
 Copy all (m)  
 Compare record (1b)  
 Compare file (1f)  
 Compare all (1m)  
 Device-to-datacomm data transfers  
 Datacomm-to-device data transfers  
 External printer control operations

The following operations are always non-interruptable:

Rewind (0c)  
 Condition tape (4c)  
 Write file mark (5c)  
 Write end-of-data mark (6c)


The following operations are non-interruptable when initiated locally at the terminal but are interruptable when initiated remotely over the datacomm line:




Space "p" records (1c)  
 Space "p" files (2c)  
 Locate file "p" (2c)  
 Space "p" records immediately without writing end-of-data mark (8c)

Any errors in a device control escape sequence cause the entire sequence to be ignored by the terminal. This could cause the CPU to go into a wait loop if your program is expecting a response upon completion of the escape sequence. You can use a programmed time-out to avoid this problem.

## DEVICE CONTROL COMPLETION CODES

For cartridge tape control operations, device-to-device data transfer operations, and ASCII or binary write operations, you determine whether or not the operation was

performed successfully by executing an INPUT or similar instruction which requests one ASCII character from the terminal. The terminal responds by sending an S, F, or U. An "S" indicates successful completion, an "F" indicates that the operation failed, and a "U" indicates that a device-to-device data transfer operation was interrupted by the terminal operator pressing the  key. Note that these completion codes cannot be suppressed by strap settings or any other means. They are always transmitted and your programs should include input commands explicitly for accepting them.

For ASCII or binary read operations, successful completion is indicated by receipt of all expected data. An I/O failure or end-of-file is indicated by receipt of just a , , or  code (depending upon the protocol employed). Datacomm errors are reported by way of the Primary Terminal Status Bytes (see Section VI).

Once your program issues a device control escape sequence to the terminal, the terminal queues further data received from the host computer in a buffer. The queued data is not acted upon until the terminal has transmitted the completion code (if any) for the current device control operation back to the host computer.

## SELECTING "from" AND "to" DEVICES

To select source ("from") and destination ("to") devices, use the following commands in a device control escape sequence:


```
< "from" device code > s
< "to" device code > d
```

where the device codes are as follows:


- 1 = left cartridge tape unit
- 2 = right cartridge tape unit
- 3 = display
- 4 = external printer
- 5 = HP-IB device

Note that if the "s" or "d" command character is not preceded by a numeric code, the existing "from" or "to" device assignment (respectively) remains in effect.

The following escape sequence selects the right cartridge tape unit as the "from" device and the external printer as the "to" device:

```
p2s4D
```

Within a single device control escape sequence, you may select only one source device but you may select multiple destination devices. For example, the following escape sequence selects the left tape cartridge as the source device and both the display and the right tape cartridge as destination devices:

```
p1s2d3D
```

Once you select source and destination devices, the designated device assignments remain in effect until:

1. They are changed by a subsequent device control escape sequence;
2. They are changed by a keyboard entry (a command channel entry for the HP 2647 or a GOLD key selection for all other 2640 series terminals).
3. The terminal operator manually initiates a "hard" reset or a program executing in the host computer initiates a "hard" reset by transmitting an  $\text{␣E}$  sequence.
4. The terminal's power is turned off and then back on.

After a hard reset or after the terminal's power is first turned on, the "to" and "from" device assignments are as follows:

Source ("from") device = left tape cartridge

Destination ("to") device = right tape cartridge

## CARTRIDGE TAPE CONTROL OPERATIONS

To perform cartridge tape control operations (such as rewinding, record spacing, locating files, and tape conditioning), use the c, u, and p commands in a device control escape sequence.

The "c" command specifies which physical operation is to be performed, as follows:

Control Code	Operation	Default Device
0	Rewind.	"from"
1	Space "p" records.	"from"
2	Space "p" files or locate file "p".	"from"
3	Locate end-of-data mark.	"from"
4	Condition tape.	"to"
5	Write file mark.	"to"
6	Write end-of-data mark.	"to"
7	Test cartridge tape unit.	"to"
8	Space "p" records immediately without writing end-of-data mark.	"to"
9	Turn on write-backspace-read mode.	(none)
10	Turn off write-backspace-read mode.	(none)

If no control code precedes the "c", code zero (rewind) is the default.

The "u" command specifies on which cartridge tape unit the specified operation is to be performed, as follows:

- 1 = left
- 2 = right

If the "u" command is omitted or if the "u" is not preceded by a "1" or "2", the current "from" or "to" device is used (as designated above). Control codes "9" and "10" always apply to both the left and right cartridge tape unit regardless of what is specified by "s", "d", or "u" commands.

For record spacing operations (1c or 1C), the "p" command specifies the number of records to be spaced over. A positive integer preceding the "p" specifies forward spacing while a negative integer specifies backspacing.

For file spacing operations (2c or 2C), the "p" command specifies either the number of files to be spaced over or the number of the file to be located. A positive integer preceding the "p" specifies forward spacing, a negative integer specifies backspacing, and an unsigned integer specifies the number of the file to be located.

If the "p" command is omitted or if no integer precedes the "p", the value +1 is used by default.

## Rewind (0c)

The following escape sequence rewinds the left cartridge tape:

$\text{␣t&p1u0C}$

The following escape sequence rewinds the right cartridge tape:

$\text{␣t&p2u0C}$

## Space "p" Records (1c)

The following escape sequence spaces the right cartridge tape forward 12 records:

$\text{␣t&p+12p2u1C}$

The following escape sequence spaces the left cartridge tape backward four records:

$\text{␣t&p-4p1u1C}$

If a file mark is the last record encountered while backspacing, the tape is automatically spaced forward so that it is positioned immediately following the tape mark (i.e., just before the first record of the file). In addition, the end-of-file mark status bit (bit 4 of cartridge tape status byte 0) is set. When the tape is located at the start of a file, to reposition it just before the preceding tape mark you must backspace two records (the tape mark being one of the records) and then forwardspace one record.

## Space "p" Files (2c)

The following escape sequence spaces the left cartridge tape forward three files:

$\text{\&#224;p+3p1u2C}$

The following escape sequence spaces the right cartridge tape backward one file:

$\text{\&#224;p-1p2u2C}$

For backspacing, if the tape is located immediately following a tape mark the relative file number works as you might expect: namely, backspace one file (-1p) positions the tape at the start of the preceding file. If the tape is positioned somewhere within a file the relative file number includes the current file, as follows: backspace one file (-1p) positions the tape at the start of the current file, backspace two files (-2p) positions the tape at the start of the preceding file, and so forth.

You may also use +0p or -0p to locate the start of the current file.

## Locate File "p" (2c)

The following escape sequence locates the start of the third file on the left cartridge tape:

$\text{\&#224;p3p1u2C}$

The following escape sequence locates the start of the fifth file on the right cartridge tape:

$\text{\&#224;p5p2u2C}$

You may use either 0p or 1p to locate the load point on a cartridge tape.

## Locate End-of-Data (3c)

The following escape sequence locates the end-of-data mark on the left cartridge tape:

$\text{\&#224;p1u3C}$

The following escape sequence locates the end-of-data mark on the right cartridge tape:

$\text{\&#224;p2u3C}$

After executing either of the above escape sequences, the tape on the selected unit is positioned at the end-of-data mark. You do NOT need to reposition (backspace) the tape to add data to it. If you copy data to a tape that is positioned at the end-of-data mark, the data is appended to the last file on the tape. To add a new file to a tape that is positioned at the end-of-data mark, you must first write a file mark and then copy the data to the tape.

## Condition Tape (4c)

After a tape cartridge has been used for a while, it develops varying degrees of tension throughout the tape. The HP 2640 series terminals include a control function that forward spaces a tape to the physical end-of-tape and then rewinds it to the load point. This procedure, which is referred to as "conditioning" a tape, reestablishes uniform tension throughout the tape.

The following escape sequence conditions a tape on the left cartridge tape unit:

$\text{\&#224;p1u4C}$

The following escape sequence conditions a tape on the right cartridge tape unit:

$\text{\&#224;p2u4C}$

## Write File Mark (5c)

The following escape sequence writes a file mark on the left cartridge tape:

$\text{\&#224;p1u5C}$

The following escape sequence writes a file mark on the right cartridge tape:

$\text{\&#224;p2u5C}$

## Write End-of-Data Mark (6c)

The following escape sequence writes an end-of-data mark on the left cartridge tape:

$\text{\&#224;p1u6C}$

The following escape sequence writes an end-of-data mark on the right cartridge tape:

$\text{\&#224;p2u6C}$

## Test CTU (7c)

The following escape sequence initiates the cartridge tape test on the left cartridge tape unit:

$\text{\&#224;p1u7C}$

The following escape sequence initiates the cartridge tape test on the right cartridge tape unit:

$\text{\&#224;p2u7C}$

The cartridge tape test is described under "Self Test" at the end of Section VII, Installation, of this manual.

## Space “p” Records Immediately Without Writing End-of-Data Mark (8c)

Normally an end-of-data mark is written before a skip, locate, rewind, or condition tape operation (0c–4c) if the last operation performed on the selected cartridge tape unit was a write operation. The “skip records immediately” operation (8c) inhibits the writing of end-of-data mark and is intended primarily for write verification in a write-backspace-read operation sequence. After using the “skip records immediately” operation, you must explicitly write a file mark on the tape before rewinding it. This control operation should not be used to skip forward on a cartridge tape on which a write operation was the last operation performed.

The following escape sequence backspaces the left cartridge tape one record immediately without writing an end-of-data mark:

```
␣&p-1p1u8C
```

The following escape sequence backspaces the right cartridge tape one record immediately without writing an end-of-data mark:

```
␣&p-1p2u8C
```

## Turn On Write-Backspace-Read Mode (9c)

The HP 2640 series terminals include a write-backspace-read capability which provides automatic verification of data written to cartridge tape. When this capability is turned on, each time a record is written to cartridge tape the tape is automatically backspaced and the record is read and compared against the original data. If a discrepancy is detected, the terminal tries to write the record again. If the record cannot be successfully written in nine retries, both the write error (bit 1, byte 0) and hard error (bit 3, byte 2) cartridge tape status bits are set. If the record is successfully written during one of the retries, the soft error (bit 4, byte 2) cartridge tape status byte is set.

The following escape sequence turns on the write-backspace-read mode:

```
␣&p9C
```

## Turn Off Write-Backspace-Read Mode (10c)

The following escape sequence turns off the write-backspace-read mode:

```
␣&p10C
```

## DEVICE-TO-DEVICE OPERATIONS

You can use device control escape sequences to copy or compare data between terminal devices.

## Copy Record (b)

To copy a record from a terminal device to one or more other terminal devices, use the command character “b” in a device control escape sequence. Within the same escape sequence you may also specify the source and destination devices by using the “s” and “d” command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current “from” and “to” device assignments are used.

Some examples are as follows:

␣&p3s2dB      Copy one record (the line containing the cursor) from the display to the right cartridge tape.

␣&pB            Copy one record from the current “from” device to the current “to” device(s).

␣&p1s2d3dB    Copy one record from the left cartridge tape to both the display and the right cartridge tape.

␣&p1sB          Copy one record from the left cartridge tape to the current “to” device(s).

␣&p3dB          Copy one record from the current “from” device to the display.

␣&p3s4dB       Copy one record (the line containing the cursor) from the display to the external printer.

Any file or end-of-data marks on the “from” device are copied to the “to” devices and count as one record each. No file marks are transferred, however, when the display is the “from” device and format mode is off.

Both of the following are considered to be errors:

1. The “from” device is located at the end-of-data when the copy operation is initiated.
2. An attempt is made to copy a record beyond the available data space of a “to” device (beyond the end of a cartridge tape, for example).

## Copy File (f)

To copy a file from a terminal device to one or more other terminal devices, use the command character “f” in a device control escape sequence. Within the same escape sequence you may also specify the source and destination devices by using the “s” and “d” command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current “from” and “to” device assignments are used.

When a non-formatted display is the source device, the copy file operation starts at the first character position of the line containing the cursor and copies through the last character position currently displayed on the screen. No file mark is copied.

When a formatted display is the source device, the copy file operation starts at the current cursor position and copies all "unprotected" and "transmit only" fields through the end of display memory, at which time a file mark is also copied.

When a cartridge tape is the source device, the copy file operation starts at the current tape position and copies until either a file mark or an end-of-data mark is detected. Upon completion, the mark itself is copied.

When a non-formatted display is the destination device, the copy file operation starts at the current cursor position. File marks copied from the source device are discarded.

When a formatted display is the destination device, the copy file operation starts at the current cursor position. Data is copied only into "unprotected" and "transmit only" fields. File marks copied from the source device are discarded.

When a cartridge tape is the destination device, the copy file operation starts at the current tape position. Any file marks or end-of-data marks copied from the source device are recorded on the tape.

When an external printer is the destination device, any file marks or end-of-data marks copied from the source device cause a form feed to occur.

Some examples are as follows:

<code>␣&amp;p3=2dF</code>	Copy a file from the display to the right cartridge tape.
<code>␣&amp;pF</code>	Copy a file from the current "from" device to the current "to" device(s).
<code>␣&amp;p1=2d3dF</code>	Copy a file from the left cartridge tape to both the display and the right cartridge tape.
<code>␣&amp;p1=F</code>	Copy a file from the left cartridge tape to the current "to" device(s).
<code>␣&amp;p3dF</code>	Copy a file from the current "from" device to the display.
<code>␣&amp;p3=4dF</code>	Copy a file from the display to the external printer.

Both of the following are considered to be errors:

1. The "from" device is located at the end-of-data when the copy operation is initiated.
2. An attempt is made to copy beyond the available data space of a "to" device (beyond the end of a cartridge tape, for example).

## Copy All (m)

To copy all from a terminal device to one or more other terminal devices, use the command character "m" in a device control escape sequence. Within the same escape sequence you may also specify the source and destination devices by using the "s" and "d" command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current "from" and "to" device assignments are used.

The copy all operation starts at the current cursor or tape position of the "from" device and copies everything through the end of medium (end of display memory or end-of-data mark).

Some examples are as follows:

<code>␣&amp;p3=2dM</code>	Copy all (all characters from the current cursor position through the end of display memory) from the display to the right cartridge tape.
<code>␣&amp;pM</code>	Copy all from the current "from" device to the current "to" device(s).
<code>␣&amp;p1=2d3dM</code>	Copy all (all records from the current tape position through the end-of-data mark) from the left cartridge tape to both the display and the right cartridge tape.
<code>␣&amp;p1=M</code>	Copy all from the left cartridge tape to the current "to" device(s).
<code>␣&amp;p3dM</code>	Copy all from the current "from" device to the display.
<code>␣&amp;p3=4dM</code>	Copy all (all characters from the current cursor position through the end of display memory) from the display to the external printer.

Both of the following are considered to be errors:

1. The "from" device is located at the end-of-data when the copy operation is initiated.
2. An attempt is made to copy beyond the available data space of a "to" device (beyond the end of a cartridge tape, for example).

## Compare Record (1b)

To compare a record between one terminal device and another, use the command character "b" preceded by a "1" in a device control sequence. Within the same escape sequence you may also specify the source and destination devices by using the "s" and "d" command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current "from" and "to" device assignments are

used. Note that, unlike the copy record function, the compare record function requires that there be only one destination device.

Some examples are as follows:

- `␣p1s2d1B` Compare the current record on the left cartridge tape with the current record on the right cartridge tape.
- `␣p3s1d1B` Compare the line containing the cursor on the display with the current record on the left cartridge tape.
- `␣p2s3d1B` Compare the current record on the right cartridge tape with the line containing the cursor on the display.

If the current record on the “to” device does not exactly match the current record on the “from” device, the compare operation fails and the “from” and “to” devices are left positioned immediately following the records that do not match.

### Compare File (1f)

To compare a file between one terminal device and another, use the command character “f” preceded by a “1” in a device control sequence. Within the same escape sequence you may also specify the source and destination devices by using the “s” and “d” command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current “from” and “to” device assignments are used. Note that, unlike the copy file function, the compare file function requires that there be only one destination device.

See the description of the “Copy File (f)” function earlier in this section for the definition of what constitutes the current file on each possible “from” and “to” device.

Some examples are as follows:

- `␣p1s2d1F` Compare the current file on the left cartridge tape with the current file on the right cartridge tape.
- `␣p3s1d1F` Compare the current file on the display with the current file on the left cartridge tape.
- `␣p2s3d1F` Compare the current file on the right cartridge tape with the current file on the display.
- `␣p1d1F` Compare the current file on the current “from” device with the current file on the left cartridge tape.
- `␣p2s1F` Compare the current file on the right cartridge tape with the current file on the current “to” device.

- `␣p1F` Compare the current file on the current “from” device with the current file on the current “to” device.

The two files are compared record-by-record. If the records do not exactly match, the compare operation fails and is terminated and the “from” and “to” devices are left positioned immediately following the records that do not match.

### Compare All (1m)

To compare all data between one terminal device and another, use the command character “m” preceded by a “1” in a device control escape sequence. Within the same escape sequence you may also specify the source and destination devices by using the “s” and “d” command characters in conjunction with the appropriate parameter values. If you omit a source and destination device specification, the current “from” and “to” device assignments are used. Note that, unlike the copy all function, the compare all function requires that there be only one destination device.

Some examples are as follows:

- `␣p3s2d1M` Compare all on the display (all characters from the current cursor position through the end of display memory) with all on the right cartridge tape (all records from the current tape position through the end-of-data mark).
- `␣p1s2d1M` Compare all on the left cartridge tape with all on the right cartridge tape. The compare operation starts at the current tape position (for both tapes) and proceeds through the end-of-data mark.
- `␣p1d1M` Compare all on the current “from” device with all on the left cartridge tape (all records from the current tape position through the end-of-data mark).
- `␣p1M` Compare all on the current “from” device with all on the current “to” device.
- `␣p2s1M` Compare all on the right cartridge tape (all records from the current tape position through the end-of-data mark) with all on the current “to” device.

The data on the two devices are compared record-by-record. If the records do not exactly match, the compare operation fails and is terminated and the “from” and “to” devices are left positioned immediately following the records that do not match.

## EXTERNAL PRINTER CONTROL OPERATIONS

To cause line feeds or form feeds on an external printer, use the c, u, and p commands in a device control sequence.

When directing a device control escape sequence an external printer, you should explicitly use the command string "4u" or "4U" in the escape sequence.

The "c" command specifies which physical operation is to be performed. When the escape sequence is directed to an external printer, the "c" control codes have the following meanings:

Control Code	Operation
0	Form feed.
1	Skip "p" lines.
2-10	Form feed.

For the skip lines function (1c or 1C), the "p" command specifies how many lines are to be skipped. The terminal uses the absolute value of the integer immediately preceding the "p" and sends that many ASCII line feed (LF) control codes to the external printer.

Some examples are as follows:

ESC & p 0 c 4 U	Initiate a form feed on the external printer.
ESC & p 1 c 4 u 6 P	Initiate six consecutive line feeds on the external printer.
ESC & p 4 u 8 C	Initiate a form feed on the external printer.

## DEVICE-TO-DATACOMM DATA TRANSFERS

You can use device control escape sequences to initiate a "read operation" that transfers data from a terminal device (the display or the left or right cartridge tape unit) to your program over a data communications link.

### Terminology

The following terminology is used subsequently in this chapter to describe the format of the data being passed between terminal devices and the datacomm link.

**Record:** A line of text from the display or a physical record on tape. Records from the display, and records copied from the display to tape, are terminated by a CR LF which are counted as two bytes. The only exception is display records which contain an explicit LF in any column other than column zero;

in such a case no CR is added and the LF designates the end of the record. A record may contain from one to 256 bytes of data (there are no null records).

A record from a formatted display (possibly copied to tape) contains one or more fields, each field copied from a single unprotected or transmit-only field in the form. Such data should be read using an ASCII read operation, in which case each field is treated as a unit of data.

**File:**

On an unformatted display, a file consists of all records starting with the one containing the cursor and continuing through the end of the data currently displayed on the screen (zero to 24 records). On a formatted display, a file consists of all "unprotected" and "transmit only" data from the current cursor position through the end of display memory. In both cases, a block terminator (either an ASCII CR or LF, as described below) or a non-displaying terminator is considered the end of display memory.

On a cartridge tape, a file consists of all records from the current tape position up to and including the next file mark. The number of records within a tape file is limited only by the physical capacity of the tape.

**Handshake:**

In a point-to-point configuration, the type of handshake protocol used depends upon how switches D, G, and H on the Keyboard Interface PCA are set.

If switches G and H are both open, no handshaking occurs.

If switch G is closed, the handshake merely consists of an ASCII CR transmitted from the computer to the terminal.

If switch G is open and switch H is closed, the handshaking protocol is as follows:

1. Computer sends CR.
2. Terminal responds with LF (followed by CR [LF] if switch D is closed. The LF is included if the AUTO LF switch on the keyboard is latched down).
3. Computer responds with another CR.

In a multipoint configuration, the terminal sends data when it is polled. Refer to Section V, Data Communications, of this manual.

**End of Text**

**<end>:** In a point-to-point, half duplex configuration, the line is turned around. If main channel protocol is being used, an end-of-text character is sent by the terminal (an ASCII  $\text{5x}$  or  $\text{57}$  character, depending upon the setting of switch T on the terminal's Keyboard Interface PCA).

In a point-to-point, full duplex configuration, nothing happens as the result of an end-of-text condition.

In a multipoint configuration, transmission stops and an ASCII  $\text{5x}$  is sent at the end of the block.

**Block Terminator**

**< $\text{55}$  or  $\text{56}$ >:** In a point-to-point configuration, the block terminator is an ASCII  $\text{55}$ .

In a multipoint configuration, the block terminator is an ASCII  $\text{55}$ .

### ASCII (7-bit) Read Operations

To initiate an ASCII (7-bit) read operation, use the following device control escape sequence:

$\text{5c}$ &p [**<device code>** s] **<read control byte>** R

where:

**device code** is an optional parameter which, if present, specifies the "from" device as follows:

- 1 = left tape
- 2 = right tape
- 3 = display
- 5 = HP-IB device

**read control byte** specifies the type of read operation, as follows:

- 0 = ASCII, read a record
- 1 = ASCII, repeat last record
- 4 = ASCII, read a file

This escape sequence is recognized as valid only when received over the datacomm line. It is ignored if issued locally at the terminal.

The source ("from") device may be selected by way of a separate device control sequence. If the source device specification is omitted from the ASCII read device control escape sequence, the current source device is used.

The ASCII read device control escape sequence merely tells the terminal's maincode what to do in response to the next handshake; the read operation itself is initiated by the handshake. With the HP 3000 Computer System, for example, the handshake is automatically initiated by the device driver within the operating system in response to the next input command executed in your computer program.

For an ASCII (7-bit) read operation, data is sent to the datacomm as seven-bit ASCII characters. Eight-bit internal display codes (such as "start unprotected field" or "start inverse video") are translated into their equivalent escape sequences (such as  $\text{5c}$  [ or  $\text{5c}$ &dB). Unrecognized eight-bit codes are translated into DEL codes. If a device record ends with  $\text{5h}$   $\text{5f}$ , the  $\text{5h}$   $\text{5f}$  is stripped prior to transmission of the record ( $\text{5h}$  and  $\text{5f}$  codes are, however, added as shown later in this topic). In addition, any  $\text{5f}$  codes within a record are stripped. Parity is added according to the settings of the keyboard parity switch and switch Z of the terminal's keyboard interface PCA.

For multipoint, all data is passed in non-transparent mode blocks. If EBCDIC is being used, the ASCII data is translated just prior to transmission.

The "repeat last record" operation (read control byte = 1) is only valid following a "read record" or "repeat last record" operation (read control byte = 0 or 1). In such a case the previously-transmitted record or field is retransmitted. Any intervening data transfer, such as a local copy between devices or an escape sequence write operation, renders the "repeat last record" request invalid.

The other ASCII read operations (read control byte = 0 or 4) are described below. In each case the escape sequence is issued explicitly (via a PRINT or equivalent output statement) by the program executing in the host computer. The handshake and end of text sequences (denoted by **<handshake>** and **<end>**, respectively) are as described under "Terminology" earlier in this section. The optional  $\text{5f}$  code is transmitted by the terminal if the AUTO LF key on the keyboard is latched down.

#### Character or Block Line Mode, Read Normal Record:

Computer	Terminal
$\text{5c}$ &p0R	
	<b>&lt;handshake&gt;</b>
	<b>&lt;record&gt;</b> $\text{5h}$ [ $\text{5f}$ ] <b>&lt;end&gt;</b>

#### Character or Block Line Mode, Read Format-Mode Record:

Computer	Terminal
$\text{5c}$ &p0R	
	<b>&lt;handshake&gt;</b>
	<b>&lt;field&gt;</b> $\text{5h}$ [ $\text{5f}$ ] <b>&lt;end&gt;</b>



Character or Block Line Mode, Read End-of-File Record:

Computer	Terminal
␣p0R	
	<handshake>
	<␣ or ␣> ␣[␣] <end>

Character or Block Line Mode, Read Normal File:

Computer	Terminal
␣p4R	
	<handshake>
	<record 1> ␣[␣] <end>
	<handshake>
	<record 2> ␣[␣] <end>
	.
	.
	<handshake>
	<record n> ␣[␣] <end>
	<handshake>
	<␣ or ␣> ␣[␣] <end>

Character or Block Line Mode, Read Format-Mode File:

Computer	Terminal
␣p4R	
	<handshake>
	<field 1> ␣[␣] <end>
	<handshake>
	<field 2> ␣[␣] <end>
	.
	.
	<handshake>
	<field n> ␣[␣] <end>
	<handshake>
	<␣ or ␣> ␣[␣] <end>

Block Page Mode, Read Normal Record:

Computer	Terminal
␣p0R	
	<handshake>
	<record> ␣[␣] <␣ or ␣> <end>

Block Page Mode, Read Format-Mode Record:

Computer	Terminal
␣p0R	
	<handshake>
	<field> ␣[␣] <␣ or ␣> <end>

Block Page Mode, Read End-of-File Record:

Computer	Terminal
␣p0R	
	<handshake>
	<␣ or ␣> <end>

Block Page Mode, Read Normal File:

Computer	Terminal
␣p4R	
	<handshake>
	<record 1> ␣[␣] <␣ or ␣>
	<record 2> ␣[␣] <␣ or ␣>
	.
	.
	<record n> ␣[␣] <␣ or ␣>
	<␣ or ␣> <end>

Block Page Mode, Read Format-Mode File:

Computer	Terminal
␣p4R	
	<handshake>
	<field 1> ␣[␣] <␣ or ␣>
	<field 2> ␣[␣] <␣ or ␣>
	.
	.
	<field n> ␣[␣] <␣ or ␣>
	<␣ or ␣> <end>

## Binary (8-bit) Read Operations

To initiate a binary (8-bit) read operation, use the following device control escape sequence:

```
⌘&p [<device code> s] <read control byte> R
```

where:

**device code** is an optional parameter which, if present, specifies the "from" device as follows:

- 1 = left tape
- 2 = right tape
- 3 = display
- 5 = HP-IB device

**read control byte** specifies the type of read operation, as follows:

- 2 = binary, read a record
- 3 = binary, repeat last record
- 6 = binary, read a file

This escape sequence is recognized as valid only when received over the datacomm line. It is ignored if issued locally at the terminal.

The source ("from") device may be selected by way of a separate device control sequence. If the source device specification is omitted from the binary read device control escape sequence, the most recently specified source device is used.

The binary read device control escape sequence merely tells the terminal's maincode what to do in response to the next handshake; the read operation itself is initiated by the handshake. With the HP 3000 Computer System, for example, the handshake is automatically initiated by the device driver within the operating system in response to the next input command executed in your computer program.

For a binary (8-bit) read operation, data is sent to the datacomm without interpretation. No characters are added or deleted. The eighth (high-order) bit of each byte may or may not be replaced with parity, depending upon the factors detailed below.

A byte count is sent prior to transmission of each record. The byte count consists of four bytes of mapped hexadecimal, using the following mapping (the most significant byte is sent first):

```
Normal Hex: 0 1 2 3 4 5 6 7 8 9 A B C D E F
264x Hex:   0 1 2 3 4 5 6 7 8 9 : ; < = > ?
```

For example, if the record contains sixty bytes (3C hex), the terminal transmits a byte count of 003<.

For point-to-point configurations, if 8-bit data is desired (i.e., no parity), the keyboard parity switch must be set to "NONE" and switch Z on the terminal's keyboard interface PCA must be closed to defeat parity generation.

For multipoint configurations, the byte count is sent in a non-transparent mode block. All data is sent in transparent mode blocks. If any of the following conditions are true, parity is NOT generated for transparent mode blocks:

- EBCDIC data codes are selected instead of ASCII (switch J07 on the multipoint board is open).
- CRC-16 block checking is selected (switch J06 on the multipoint board is open).
- The keyboard parity switch is set to "NONE".
- Switch Z on the terminal's keyboard interface board is open.

If none of the above conditions are true, parity is generated for all data.

The "repeat last record" operation (read control byte = 3) is only valid following a "read record" or "repeat last record" operation (read control byte = 2 or 3). In such a case the previously-transmitted record is retransmitted. Any intervening data transfer, such as a local copy between devices or an escape sequence write operation, renders the "repeat last record" request invalid.

The other binary read operations (read control byte = 2 or 6) are described below. In each case the escape sequence is issued explicitly (via a PRINT or equivalent output statement) by the program executing in the host computer. The handshake and end of text sequences (denoted by <handshake> and <end>, respectively) are as described under "Terminology" earlier in this section. The optional ⌘ code is transmitted by the terminal if the AUTO LF key on the keyboard is latched down.

Character or Block Line Mode, Read Normal or Format-Mode Record:

Computer	Terminal
⌘&p2R	
	<handshake>
	<count> %[%] <end>
	<handshake>
	<record> <end>

Character or Block Line Mode, Error or Read End-of-File Record:

Computer	Terminal
⌘&p2R	
	<handshake>
	<% or %> %[%] <end>

Character or Block Line Mode, Read File:

Computer	Terminal
⌘&p6R	
	<handshake>
	<count 1> ⌘[Lr] <end>
	<handshake>
	<record 1> <end>
	:
	:
	<handshake>
	<count n> ⌘[Lr] <end>
	<handshake>
	<record n> <end>
	<handshake>
	<⌘ or ⌘> ⌘[Lr] <end>

Block Page Mode, Read Normal or Format-Mode Record:

Computer	Terminal
⌘&p2R	
	<handshake>
	<count> <⌘ or ⌘> <end>
	<handshake>
	<record> <end>

Block Page Mode, Error or Read End-of-File Record:

Computer	Terminal
⌘&p2R	
	<handshake>
	<⌘ or ⌘> <end>

Block Page Mode, Read File:

Computer	Terminal
⌘&p6R	
	<handshake>
	<count 1> <⌘ or ⌘> <record 1>
	<count 2> <⌘ or ⌘> <record 2>
	:
	:
	<count n> <⌘ or ⌘> <record n>
	<⌘ or ⌘> <end>

## Fast Binary (Program Load) Read Operations

To pass binary data from a terminal device directly into a host computer without any handshake protocol, use the following escape sequence:

⌘e

This escape sequence may be issued either locally or from a program executing in the host computer. In response to this escape sequence, a file from the current source ("from") device is copied to the datacomm without interpretation.

The entire file is transmitted without byte counts, delimiters, or any other characters inserted into the data stream. There is no handshake, except for acknowledgements required by the multipoint protocol. At the end of the transmission, two bytes of all zeros (000 octal) are sent if there were no errors; otherwise two bytes of all ones (377 octal) are sent.

For point-to-point configurations, parity generation is automatically suppressed. If switch F on the terminal's keyboard interface PCA is closed, the transmission occurs at the baud rate specified by the keyboard BAUD RATE switch. If switch F is open, the transmission rate is 9600 baud.

For multipoint configurations, all data is transmitted in transparent mode blocks. If any of the following conditions are true, parity is NOT generated:

- EBCDIC data codes are selected instead of ASCII (switch J07 on the multipoint board is open).
- CRC-16 block checking is selected (switch J06 on the multipoint board is open).
- The keyboard parity switch is set to "NONE".
- Switch Z on the terminal's keyboard interface board is open.

If none of the above conditions are true, parity is generated for all data.

## DATACOMM-TO-DEVICE DATA TRANSFERS

You can use device control escape sequences to initiate a "write" operation that transfers data from your program to a terminal device (the display or the left or right cartridge tape unit) over a data communications link.

### ASCII (7-bit) Write Operations

To initiate an ASCII (7-bit) write operation, use the following device control escape sequence:

⌘&p [ <device code> d] W <record>

where:

device code is an optional parameter which, if present, specifies the "to" device as follows:

- 1 = left tape
- 2 = right tape
- 3 = display
- 4 = external printer
- 5 = HP-IB device

record is the data to be transmitted.

This escape sequence is recognized as valid only when received over the datacomm line. It is ignored if issued locally at the terminal.

The record being transmitted is terminated by the 256th data byte after the "W" or by the first LF code, whichever comes first. On some terminals, one-byte records (containing a single LF code) cannot successfully be written to the display.

The data is sent from the datacomm to one or more terminal devices as seven-bit characters. The eighth (high-order) bit may be checked for parity, depending upon the settings of the keyboard PARITY switch, switch Z on the terminal's keyboard interface PCA, and (for multipoint configurations) whether ASCII or EBCDIC code is being transmitted. The eighth bit is then cleared.

If any transmission errors occur, the entire record being transmitted is discarded and the operation aborted.

The destination ("to") device(s) may be selected by way of a separate device control sequence. If a destination device specification is omitted from the ASCII write device control escape sequence, the current destination device(s) is used.

If a handshake (initiated by an INPUT or similar command) is performed following a write escape sequence, the terminal transmits an ASCII "S" or "F" to indicate whether or not the write operation was successfully performed. For interrupt-driven devices (such as tapes), the "S" indicates that the write was successfully initiated. For other devices, the "S" indicates successful completion. The "F" indicates failure. If a datacomm error occurs during transmission, the completion code is unpredictable. Datacomm errors are reported by way of the terminal status bytes.

For point-to-point configurations, NUL and DEL codes are normally stripped from the incoming data. In addition, the terminal responds to an incoming 8 code by transmitting an 8; the 8 code is not considered part of the incoming data.

For multipoint configurations, the data may be passed in either transparent or non-transparent mode blocks; after the multipoint protocol has interpreted the block, the

data is handled identically in both cases. If EBCDIC is being used, the terminal translates the characters to ASCII as they are received.

In the following definitions, the optional LF code is transmitted by the terminal if the AUTO LF key on the keyboard is latched down.

Character or Block Line Mode:

Computer	Terminal
ESC & p W <record>	
	<handshake>
	<S or F> 8[LF]

Block Page Mode:

Computer	Terminal
ESC & p W <record>	
	<handshake>
	<S or F> <8 or 8>

## Binary (8-bit) Write Operations

To initiate a binary (8-bit) write operation, use the following device control escape sequence:

ESC & p [<device code> d] <byte count> W <record>

where:

device code is an optional parameter which, if present, specifies the "to" device as follows:

- 1 = left tape
- 2 = right tape
- 3 = display
- 4 = external printer
- 5 = HP-IB device

byte count is a decimal integer within the range 1-256 which specifies the number of bytes to be transmitted. If the byte count is zero, an ASCII (7-bit) write operation is performed as described above.

record is the data to be transmitted.

This escape sequence is recognized as valid only when received over the datacomm line. It is ignored if issued locally at the terminal.

The record is transmitted from the datacomm to all "to" devices. The destination ("to") device(s) may be selected by way of a separate device control sequence. If a destination device specification is omitted from the binary

write device control escape sequence, the current destination device(s) is used. See the discussions of the individual protocols below to determine whether seven-bit or eight-bit data is passed.

If the data is passed to the display or an external printer, bytes with the high-order bit set are interpreted by the terminal as internal display codes (such as "start unprotected field" or "start inverse video"). Transmission of display code escape sequences to an external printer is enabled by opening switch N on the terminal's keyboard interface PCA.

Data passed to the cartridge tape units is copied exactly as received.

If any transmission errors occur, the entire record being transmitted is discarded and the operation aborted.

If a handshake (associated with an INPUT or similar command) is performed following a write escape sequence, the terminal transmits an ASCII "S" or "F" to indicate whether or not the write operation was successfully performed. For interrupt-driven devices (such as tapes), the "S" indicates that the write was successfully initiated. For other devices, the "S" indicates successful completion. The "F" indicates failure. If a datacomm error occurs during transmission, the completion code is unpredictable. Datacomm errors are reported by way of the terminal status bytes.

For point-to-point configurations, you must transmit an  $\text{E}_b$  to the terminal, and receive back an  $\text{A}_k$ , after transmitting the "W" in order to allow the terminal enough time to switch into binary mode. When the terminal is in binary mode, parity checking is automatically disabled. If the keyboard PARITY switch is set to "EVEN" or "ODD", the high-order bit of each data byte is cleared and seven-bit data is passed to the "to" device(s). If the keyboard PARITY switch is set to "NONE", all eight bits of each data byte are passed to the "to" device(s). During transmission of the record,  $\text{H}_b$ ,  $\text{E}_b$ , and DEL codes are passed as data (the  $\text{E}_b$ - $\text{A}_k$  handshake is disabled) until the byte count has been satisfied.

For multipoint configurations, the terminal does not care whether the data is sent in transparent or non-transparent mode blocks. Transparent mode blocks should be used in most cases, however, to avoid parity checks and protocol checks for special characters. Transparent mode blocks are not checked for parity if any of the following conditions is true:

- EBCDIC data codes are selected instead of ASCII (switch J07 on the multipoint board is open).
- CRC-16 block checking is selected (switch J06 on the multipoint board is open).
- The keyboard parity switch is set to "NONE".
- Switch Z on the terminal's keyboard interface board is open.

If none of the above conditions are true, all data is checked for proper parity.

In the following definitions, the optional  $\text{L}_F$  code is transmitted by the terminal if the AUTO LF key on the keyboard is latched down.

#### Point-to-Point, Character or Block Line Mode:

Computer	Terminal
$\text{E}_t\&p$ <count> W $\text{E}_b$	$\text{A}_k$ <end>
<record>	<handshake>
	<S or F> $\text{R}[\text{L}_F]$
	<end>

#### Point-to-Point, Block Page Mode:

Computer	Terminal
$\text{E}_t\&p$ <count> W $\text{E}_b$	$\text{A}_k$ <end>
<record>	<handshake>
	<S or F> $\text{C}_S$ or $\text{C}_F$
	<end>

#### Multipoint, Character or Block Line Mode:

Computer	Terminal
$\text{E}_t\&p$ <count> W <record>	
	<handshake>
	<S or F> $\text{R}[\text{L}_F]$
	<end>

#### Multipoint, Block Page Mode:

Computer	Terminal
$\text{E}_t\&p$ <count> W <record>	
	<handshake>
	<S or F> $\text{C}_S$ or $\text{C}_F$
	<end>

## GRAPHICS HARD COPY OPERATIONS

Certain hard copy peripheral devices, such as the Versatec 1640, can be used to make printed copies of graphics displays. The hard copy units are used with the HP 13254A Video Interface. If the interface is configured as address 04 (refer to the HP 13254A Installation and Service Manual, part no. 13254-90001), a print command can be entered locally at the terminal (GREEN f6 f8, for 2645 or 2648) or be sent from a program executing in a host computer (`&p4u5C`). You can also initiate copies from the hard copy unit itself.

## GRAPHICS TRANSFER OPERATIONS

You may copy the contents of graphics memory to cartridge tapes and compatible printers (such as the HP 2631G and the HP 7245A). The printers are connected to the terminal via the Hewlett-Packard Interface Bus (HP-IB). In addition, you may also restore the graphics memory from cartridge tape.

The graphics transfer to cartridge tape requires approximately 70 seconds and slightly less than 1/2 of tape capacity.

To copy the content of graphics memory to another medium, use the following escape sequence:

`&p5u0C`

where "5u" specifies graphics memory and "0C" specifies a graphics dump operation. For a graphics dump operation the terminal always copies the data to the currently assigned destination ("to") devices. Therefore, to be safe you should make it a practice to immediately precede the graphics dump escape sequence with an escape sequence that specifically defines the desired destination device(s).

To restore graphics memory from a cartridge tape, use the following escape sequence:

`&p5u5C`

where "5u" specifies graphics memory and "5C" specifies a graphics restore operation. For a graphics restore operation the terminal always copies the data from the currently assigned source ("from") device. Therefore, to be safe you should make it a practice to immediately precede the graphics restore escape sequence with an escape sequence that specifically defines the desired CTU as the source device.

Example: Copy the contents of graphics memory to the right tape and to the HP-IB printer.

ESC & p 2d 5D      Assign the right tape and printer as destination devices

ESC & p 5u 0C      Copy graphics memory to the destination devices

Example: Copy the contents of the left tape to graphics memory.

ESC & p 1S          Assign the left tape as the source device

ESC & p 5u 5C      Copy the contents of the source device to graphics memory.

## HP-IB OPERATIONS

To specify an HP-IB device as the source ("from") device, use the parameter "5s" or "5S" in an escape sequence.

To specify an HP-IB device as the destination ("to") device, use the parameter "5d" or "5D" in an escape sequence.

To direct a device command escape sequence (P,U,C) to an HP-IB device, use the parameter "5u" or "5U". In such an escape sequence, the control code parameter (c or C) has the following interpretation:

- 1      Select HP-IB talk address
- 2      Select HP-IB listen address
- 3      Enable HP-IB timeout
- 4      Disable HP-IB timeout
- 7      HP-IB byte count
- 8      Initialize HP-IB

## Addressing

When a data transfer to the HP-IB (5u) is initiated, the data is sent to the current LISTEN address. The default value is 6. If the address switches on the HP-IB device are not set to this address, then the LISTEN address may be changed by using the following escape sequence:

Assign LISTEN address:

ESC & p <listen address> p 5u 2C

When a data transfer from the HP-IB (5u) is initiated, the data is received from the current TALK address. The default value is 6. If the address switches on the HP-IB device are not set to this address, then the TALK address may be changed by using the following escape sequence:

Assign TALK address:

ESC & p <talk address> p 5u 1C

## Transferring Data from Computer to HP-IB Device

**ASCII DATA.** A record of ASCII data may be transferred from the computer to an HP-IB device (with the current LISTEN address) by the following escape sequence:

ASCII data:

ESC & p 5d W <data> LF

Up to 256 bytes may be sent; a line feed character (LF) terminates the sequence. If no LF character is specified, 256 bytes are accepted by the terminal.

**BINARY DATA.** A record of binary data may be transferred from the computer to an HP-IB device (with the current LISTEN address) by the following escape sequence.

Binary data:

ESC & p 5d <byte count> W ENQ

The byte count must consist of ASCII numerals. An ENquiry character (octal 5) must be sent before the data bytes. When an ACKnowledge character (octal 6) is received from the terminal, then the specified number of bytes of data may be sent.

## Transferring Data From HP-IB Device to Computer

Data may be transferred from the HP-IB device (with the current TALK address) to the computer by the following escape sequences:

For ASCII data terminated with an LF:

ESC & p 5s R

For ASCII data not terminated with an LF and binary data, send a byte count before requesting data. The terminal will not respond to any further program control until the specified number of bytes have been transferred.

Send Byte Count: ESC & p 5u <byte count> p 6C  
Send Data: ESC & p 5s R

## Transferring Data Between the HP-IB and Terminal Devices

Data may be transferred between the HP-IB and alphanumeric display memory, graphics display memory,

and cartridge tapes. (The graphics transfer is discussed under "Graphics Transfer Operations".) Assume that the LISTEN and TALK addresses have been assigned previously.

**Example:** Transfer the contents of alphanumeric memory to the printer on the HP-IB.

ESC & p 3s 5d M Assign display as source and HP-IB as destination. Copy all.

**Example:** Transfer a file on the right tape to the printer on the HP-IB.

ESC & p 2s 5d F Assign right tape as source and HP-IB as destination. Copy File.

## Timeout

If an HP-IB device does not respond within 10 seconds (devices busy, power off, etc.), "HP-IB DOWN" message will appear on the display. To disable or reenable the timeout function, use the following escape sequences:

Disable Timeout: ESC & p 5u 4C

Enable Timeout: ESC & p 5u 3C

### NOTE

If the HP-IB does not respond and the 10-second timer is running, a soft reset should not be performed to abort the operation. You should wait for the timer to timeout, then re-initialize the HP-IB. (Refer to "Initialization".)

## Status

If HP-IB status is requested, all zeros will always be returned.

Request Status: ESC & p 5<sup>^</sup>(DC1)

Terminal response: 000CR(LF)

Your program should monitor the "S" or "F" that is returned by terminal after the command is sent. (Refer to "Indicating Successful Completion of a Program-Controlled Function".)

## Initialization

The HP-IB is initialized at power-on and hard reset. Initialization includes setting the TALK and LISTEN addresses to 6, enabling the timeout function, etc. You can programmatically initialize the HP-IB by the following escape sequence:

ESC & p 5u 7C

## INTRODUCTION

This section describes the terminal's data communications capabilities and operating requirements. The topics include interface specifications, network considerations, point-to-point operation, multipoint operation, and communication configuration status.

## CONNECTING TERMINALS TO A COMPUTER

The terminal can be configured to work in a variety of computer applications. Your communication needs can be met by selecting a particular interface, modem, and protocol (communication control program). Refer to the remainder of this section for configuration information.

### Networks

The terminal can be connected in a variety of network configurations. Figure 5-1 illustrates the following configurations:

- Hardwired to a computer (figure 5-1A).
- Hardwired through other terminals to a computer (figure 5-1B).

- Connected to a computer through a modem (figure 5-1C).
- Connected through other terminals to a modem (figure 5-1D).

### Interfaces

The terminal can be used with a variety of communication interfaces. A list of available interfaces and a brief description of each is given in table 5-1. The interfaces are the 13260A Asynchronous, 13260B Extended Asynchronous, 13260C Multipoint Asynchronous, and the 13260D Multipoint Synchronous. A list of some of the capabilities of these interfaces is given in table 5-2.

Once the interface has been selected, the terminal can be configured to operate with a variety of protocols, parities, and data formats. This is done by setting switches or jumpers on the interfaces.

Section VII, Installation, contains complete lists of the possible switch settings for each of the interfaces together with brief descriptions of the switches. Also included in the Installation section are procedures for setting these switches.

Table 5-1. Data Communication Interfaces

<b>Basic Communications (Point-to-Point)</b>	
13260A	Standard Asynchronous Communications Interface Standard RS232C communications interface.
13260B	Extended Asynchronous Communications Interface provides either standard RS232C or 20 mA current loop communications. It allows split speed and custom baud rates.
<b>Multipoint Communications</b>	
13260C	Asynchronous Multipoint Communications Interface provides asynchronous multipoint communications. It allows several terminals to share the same communication line.
13260D	Synchronous Multipoint Communications interface provides synchronous multipoint communications. It allows several terminals to share the same communication line.



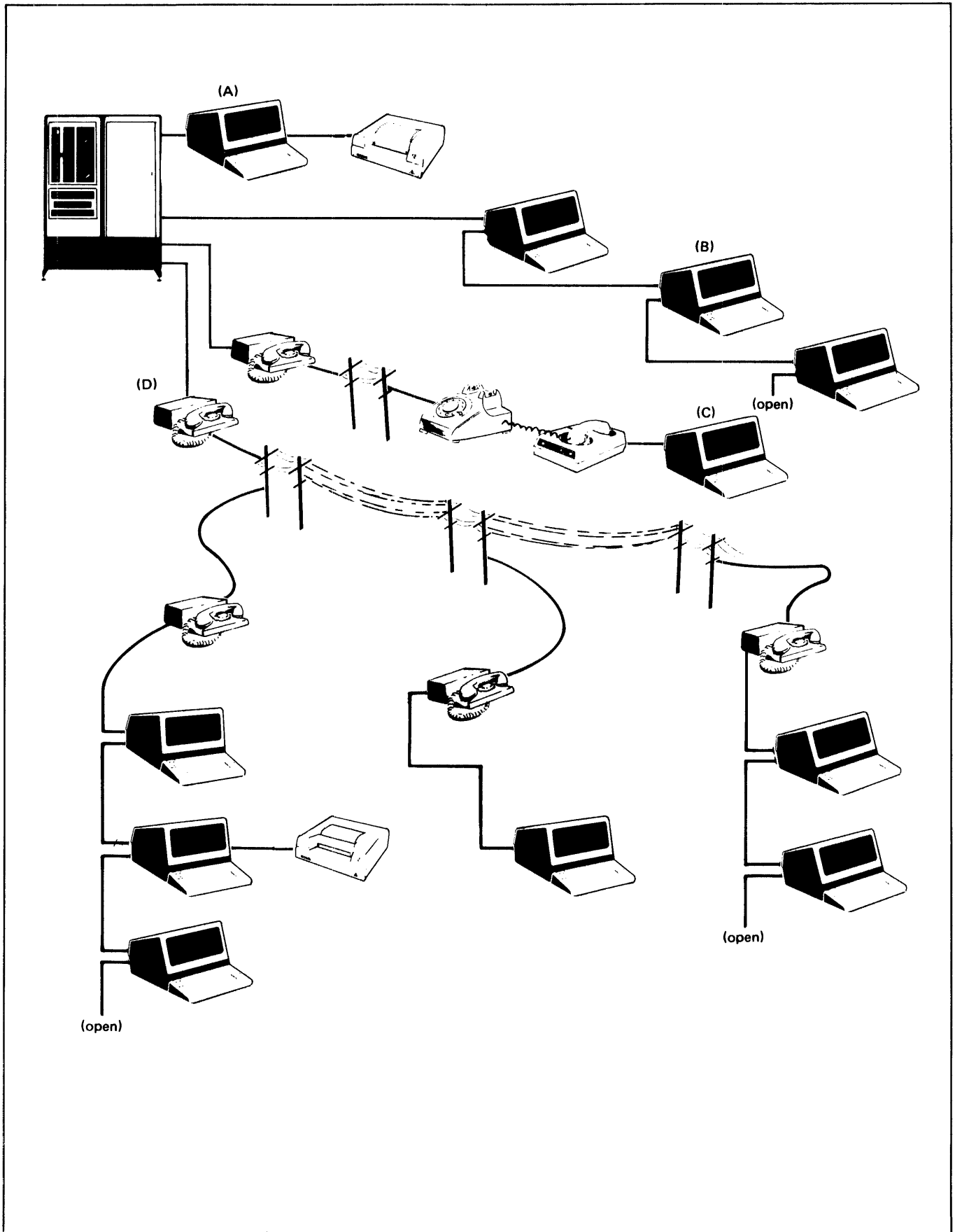


Figure 5-1. Terminal Network Configurations

Table 5-2. Data Communication Interface Capabilities

DATA COMMUNICATIONS FEATURES	13260			
	A	B	C	D
Transfer Rate:				
110, 150, 300, 1200, 2400, 4800, 9600 bits per second and external clocking (110-9600)	X	X		
300, 600, 1200, 1800, 3600, 4800, 7200, 9600 bits per second			X	
2400, 4800, 9600 bits per second and external clocking (300-9600)				X
Custom transfer rates within 1% from 37.5 to 2400 bits per second		X		
Split speed transmit/receive capability		X		
EIA RS 232-C	X	X	X	X
Teletypewriter compatible	X	X		
ASCII	X	X	X	X
EBCDIC			X	X
20mA DC Current Loop		X		
Transmission Modes:				
Character Transfer	X	X		
Block Transfer	X	X	X	X
Half-duplex	X	X	X	X
Full-duplex	X	X		
Asynchronous	X	X	X	
Synchronous				X
Hardwired to computer; dialed (switched) or leased line	X	X	X	X
Modem Compatibility:				
Bell 103A, 202D, 202C, 202S, 202T (Asynchronous)	X	X	X	
Vadic 3400 (Asynchronous/Synchronous)	X	X	X	X
Bell 201A, 201B, 201C, 208A, 208B, 209A (Synchronous)				X
Choice of main channel or reverse channel line turnaround for 202 modems	X	X		
Auto-Answer/Disconnect		X	X	X
Transparency	X	X	X	X
Data Comm. Self-Test	X	X	X	X
Error Checking:				
VRC, choice of parity generation/checking	X	X	X	X
LRC			X	X
CRC-16			X	X
Additional polling protocol features:				
Daisy-chained/multipoint line and modem sharing (up to 32 terminals/line)			X	X
Synchronous polling (IBM Binary Synchronous Multipoint Communication, Bisync)				X
Asynchronous polling (modeled after IBM Bisync)			X	
Group and device addressing; group poll; broadcast			X	X
Variable I/O buffer sizes			X	X
Configuration status			X	X
Monitor Mode	X	X	X	X
Driver Mode (option)			X	X

Some of the communication features can be selected from the Keyboard and the Keyboard Interface PCA. Tables 5-3 and 5-4 provide lists of these switches together with brief descriptions.

Table 5-3. Keyboard Interface (PCA) Switch Summary

SWITCH	CHARACTER PROTOCOL	BLOCK PROTOCOL
A	Function key transmission	(not used)
B	Space overwrite latch	same
C	Cursor end-of-line wraparound	same
D	Line/Page mode	same
E	Paper tape mode	same
F	Fast binary read	(not used)
G	Block transfer handshake	(not used)
H	Inhibit DC2	(not used)
J	Auto terminate	same
K	Clear terminator	same
L	Self-test inhibit	same
M	Reverse action of CNTL key with INSERT CHAR and DELETE CHAR keys (wrap function)	same
N	Escape code transfer to printer	same
P	Compatibility Mode (scaled)	same
Q	Compatibility Mode (unscaled)	same
R	Circuit Assurance	Internal Data Set Ready
S	Main/Reverse Channel configuration. Switches S and T cannot be modified programmatically.	Space Compression
T		Output block size. (Switches T and U cannot be modified programmatically.)
U	CPU break	Synch Mode for Asynchronous Operation
V	Carrier detect	
W	Data Comm self-test enable	same
X	Data speed select	same
Y	Transmit LED	same
Z	Force Parity	Transparency

## Interface Signals

The signals available on each of the communication interfaces are listed in the Installation section. This information can be used to verify interface compatibility or to fabricate special interface cables.

Table 5-4 Keyboard Communications Switches

<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p><b>DUPLEX</b></p> <p>HALF </p> <p>FULL </p> </div> <div style="text-align: center;"> <p><b>PARITY</b></p> <p>EVEN </p> <p>ODD </p> <p>NONE </p> </div> </div> <div style="text-align: center; margin-top: 20px;"> <p><b>BAUD RATE</b></p> </div> <p style="text-align: center; margin-top: 20px;"><b>BASIC COMMUNICATIONS DATA COMM SWITCHES</b></p> <p><b>DUPLEX Switch.</b> HALF: Typed characters are processed by the terminal and transmitted to the computer. FULL: Typed characters are transmitted to the computer and not processed by the terminal until returned from the computer. (This function is ignored in Block Mode.) Not present on terminals with multipoint interfaces.</p> <p><b>RANGE Switch.</b> This switch is used to select ranges for the BAUD RATE switch (multipoint only).</p> <p><b>PARITY Switch.</b> When set to EVEN/ODD/NONE, even/odd/no parity is transmitted for each character. Incorrect parity: a "■" is displayed.</p> <p><b>BAUD RATE Switch.</b> Selects data transmission rate of 110, 150, 300, 1200, 2400, 4800, or 9600 baud. EXT: any rate between 110 and 9600 can be selected from an external source. The 110 baud rate uses two stop bits per character; all others use one stop bit. In Multipoint configurations, the following additional speeds are available: 600, 1800, 3600, and 7200.</p> <p> When down, the terminal is in Remote (on-line) operation. Otherwise, the terminal is in local (off-line) operation.</p> <p> When the terminal is in Block Mode, typed data is displayed but not transmitted to the computer until requested by the computer or until after the  key has been pressed and the computer has responded. Otherwise, the terminal is in Character Mode and data is transmitted as typed. (See "Block Mode". In multipoint configurations the terminal is always in Block Mode, regardless of key position.</p>	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p><b>RANGE</b></p> <p>HI </p> <p>LO </p> </div> <div style="text-align: center;"> <p><b>PARITY</b></p> <p>EVEN </p> <p>ODD </p> <p>NONE </p> </div> </div> <div style="text-align: center; margin-top: 20px;"> <p><b>BAUD RATE</b></p> </div> <p style="text-align: center; margin-top: 20px;"><b>MULTIPOINT DATA COMM SWITCHES</b></p> <p> In basic communications, transmits a BREAK signal to interrupt computer operation. (Transmits a 200 ms space on the asynchronous data communication line and sets secondary channel low for 200 ms.)</p> <p>In multipoint an RVI is transmitted instead of ACK0 or ACK1 if  is pressed while the terminal is receiving text (Text-In). In other multipoint modes the  key clears the data comm output buffers and sends a CN (Cancel) to the computer. (Refer to the BREAK KEY description under multipoint.)</p> <p style="text-align: center;">○ TRANSMIT</p> <p>The indicator will be lighted when a data link exists for transmission between the terminal and the computer.</p> <p> <b>On-Line Mode</b></p> <ul style="list-style-type: none"> <li>• Character Mode, Format Off. The entire line containing the cursor is transmitted as a block.</li> <li>• Character Mode, Format On. Unprotected characters from the cursor position to the end of the unprotected field are block transmitted. The cursor is left at the first character position after the end of the field.</li> <li>• Block Mode, Format Off. After receiving a DC1 from the computer, the terminal informs the computer by transmitting a DC2 control character (or DC2 CR(LF) with Line Strapping — see "Strapping Options") that the terminal is ready to transmit characters from the cursor to the end of the line of memory (dependent on Line or Page strapping).<sup>1</sup></li> <li>• Block Mode, Format On. After receiving a DC1 from the computer, informs the computer by transmitting a DC2 (or DC2 CR(LF) with Line Strapping) that the terminal is ready to transmit the current field, or all unprotected fields from the cursor to the end of memory, each delimited by a unit separator, US (dependent on Line/Page strapping).<sup>1</sup></li> </ul>
---	---

<sup>1</sup>Basic Data Communications only

## Modems

The terminal can be used with a variety of modems depending on the requirements of the given configuration or network. Table 5-5 contains a list of modems and the configurations in which they can be used.

Table 5-5. Modems

MODEM	DATA RATE (BITS/SEC)	LINE TYPE: DIALED/LEASED	DUPLEX FULL/HALF	WIRES 2/4	REV. CHAN.
<b>Asynchronous</b>					
Bell 103A	300	D/L	H/F	2	No
Bell 202S Bell 202C ITT GH 2052 Nokia DS 9320	1200	D	H	2	Option
Bell 202T Bell 202D	1200 (3)	L	H/F	2/4	Option
Vadic VA3400	1200	D	F	2	No
<b>Synchronous (1)</b>					
Vadic VA3400	1200	D	F	2	No
Bell 201C Bell 201A Milgo 2200 Milgo 2400	2400	D/L (2)	H/F	2/4	No
Bell 208A	4800	L (2)	H/F	4	No
Bell 208B	4800	D	H	2	No
Bell 209A	9600	L (2) (4)	F	4	No
Notes:					
1. Synchronous modems require the internal clock modem option.					
2. Synchronous operation on a leased line requires the switched carrier modem option.					
3. C2 line conditioning allows operation at 1800 bits/sec.					
4. Requires D2 line conditioning.					

## COMMUNICATION PROTOCOLS

Control of computer-terminal communications is required for the orderly transfer of data. This control is provided in the form of a protocol or a set of rules and procedures. The protocol used determines who sends and who receives during each phase of communication. In addition the protocol normally provides for an orderly recovery from communication errors.

The protocols available with the terminal allow operation ranging from simple full duplex teleprinter compatibility to bisynchronous multipoint communications. The various protocols can be selected by installing the proper interface and ROM modules. The terminal and the interface can then be configured to meet your specific requirements.

The major characteristics of the available protocols are listed in table 5-6. The following paragraphs discuss each of these protocols.

### Character Protocols

Character protocols transmit a single character at a time. Data checking, if present, is done on individual characters only (parity). Some configurations allow the transmission of multicharacter groups but no block checks are made. There is no automatic retransmission of data following the detection of a data error. Currently available character mode protocols are Standard, Main Channel, and Reverse Channel.

Standard Communications is a term used to refer to point-to-point or single terminal communications. The terminal can be connected directly to a computer (hard-wired) or through a modem. In most block applications the terminal can use a simple "handshake" protocol with the ASCII DC1 character. This protocol can be used with Bell 103 or equivalent modems (full-duplex operation). There are two additional protocols available, Main Channel and Reverse Channel. These protocols are normally only used with Bell 202 or equivalent modems (half-duplex operation).

### Block Protocols

Block protocols transmit a block of characters at a time. Data checking is performed on an entire block of data. A separate block check character (BCC) is generated for each block. If a data error is detected, a retry of the data transmission is made automatically. The currently available block protocol is Multipoint.

Multipoint communications is the use of several terminals sharing a single communication line. The terminals can be directly connected to the computer or can be connected through modems. Multipoint communications require a special multipoint protocol. Additional information on multipoint operation is given later in this section.

Table 5-6. Protocol Characteristics

<b>Single Terminal (Character Mode Protocols)</b>	
Standard	Standard communication protocol is teletype compatible or can use the DC1 character to trigger multicharacter transfers.
Main Channel	Communication protocol uses special framing characters to control line turn-around.
Reverse Channel	Uses secondary channel signals to trigger line turn-around.
<b>Multiple Terminal (Block Mode Protocols)</b>	
Multipoint	Uses a polling protocol similar to IBM Bisync to serve multiple terminals on the same line.

The remainder of this section provides descriptions and samples of control and data transfer sequences for various protocols. Included are examples of typical single terminal and multiple terminal organizations together with sample communication programs. Detailed flowcharts of the various protocols are given in Appendix C.

## CHARACTER PROTOCOLS

The terminal can operate character-by-character as a completely interactive terminal or on a block of data at a time. Block transfers allow data to be composed and edited at the terminal allowing the user to verify and correct data before sending it to the computer.

### Operating at High Speeds

If the number of characters sent to the terminal in one sequence exceeds 80, the required terminal processing time may cause some of the characters to be lost. (This usually does not occur at data rates of 4800 baud or less.) The symptom of this problem is the appearance of the "■" (delete) or "\_" characters. (These characters do not appear if the terminal is in Graphics Text Mode.)

There are three ways of insuring that this problem will not arise:

- It is possible to use a call-and-answer procedure between the terminal and the computer. If the computer sends an ENQ (octal 5) character after sending 80 characters, the terminal will respond with the ACK (octal 6) after it has processed the characters. The computer can then send the next block of characters. This is the recommended technique. The ENQ/ACK handshake cannot be used when the terminal is in Compatibility Mode (refer to Section III, Compatibility Mode). Compatibility Mode automatically selects a larger data communications buffer in non-multipoint operation. (Refer to the description of alternate buffer size that follows.

- Delays can be inserted in the application or system software after each 80 character transfer from the computer to the terminal. Transmitting NULL characters (octal 0) is one way to accomplish this. Each NULL character has the effect of 4 millisecond delay when operating at 2400 baud, and 2 milliseconds at 4800 baud. As an aid in calculating needed time delays, a list of processing times for various terminal functions is provided in table 5-7. The times listed are typical and can vary greatly depending on such factors as the number of characters in the terminal memory or on the display, and the current operating mode.
- A larger data communications buffer can be selected using the P and Q switches on the Keyboard Interface PCA. The larger buffer must be selected by physically setting the switches. Programmatically setting the switches will not work. Opening either P or Q will allocate a 2048 byte communications buffer when the terminal is initialized (power on or a full reset). This will help to eliminate data overruns due to character bursts. The additional buffer space is taken from the available display memory space. This will result in a loss of display storage unless additional memory storage is added to the terminal. (With the 2048 byte buffer, the standard terminal can store 37 lines of 80 characters.) Refer to Section VII, Installation, for instructions for adding additional memory storage. Note that the P and Q switches are also used to select Compatibility Mode for graphics operation. (Refer to table 5-8.) Compatibility Mode is discussed in detail in Section III, Graphics Functions.

Table 5-7. Terminal Functions

TERMINAL FUNCTION	TYPICAL REQUIRED TIME (MILLISECONDS)
Text Character	0.7
Cursor Up/Down/Left/Right	1.4
Line Feed	1.4
Insert Char	4.5
Delete Char	7.0
Insert Char w/wrap	12.0
Delete Char w/wrap	19.0
Soft Reset (Tapes Stationary)	130
Hard Reset (No Tapes)	159
Forms Mode On	12.0
Forms Mode:	
Home	8.0
Tab	8.0
Back Tab	10.0
Erase to End-of-Line (40 characters)	10.0

## Character Mode

In Character Mode operation (BLOCK MODE key up), the terminal sends characters to the computer as they are typed. This mode of operation can be used for conversational exchanges with the computer.

## Example:

```

Computer:  Please type your company name

User types:  AJAX

Computer:  What file number would you like from
           the AJAX library?

User types:  12345

and so on   . . .

```

## Multicharacter Transfers

There are certain functions that always result in multicharacter (block) data transfers.

- device input/output and control operations, including tape transfers.
- special function keys
- status requests
- cursor sensing
- all transfers while in Block Mode

In order for the terminal to make a block transfer, it must first be enabled and then triggered by the computer. Transfers are enabled by the ENTER or special function keys while the terminal is in Block Mode (see figure 5-2). When a transfer is enabled from the keyboard, the terminal sends a DC2 character to the computer to indicate that a data block is ready for transmission. (This process can be modified by strap settings on the Keyboard Interface, refer to Section VII.) A transfer can also be enabled from the computer by an escape sequence requesting status (ESC ^), cursor sensing (ESC a), or device control (ESC & p . .) as shown in figure 5-3.

When the transfer is enabled the keyboard is locked out until the transfer is complete. Enabling sequences should not be entered from the keyboard or cartridge tapes because they will cause the keyboard to be locked until the computer responds with a DC1 character. (If the computer does not respond, a soft reset will cancel the transfer and re-enable the keyboard.)

Once a block transfer has been enabled, it must be triggered by the computer before the block of data is actually sent. The computer triggers the transfer by sending a DC1 character when it is ready to receive the data. The terminal also assumes that it has received the trigger when it is first powered up or fully reset, or when the REMOTE key is pressed (down).

The computer software must support the handshaking process used in multiple character transfers. The DC2 character must be recognized as a request to send data and

the DC1 character must then be sent to trigger the transfer after buffers have been allocated to receive the data. Additional software support may be needed depending on your need for terminal or device control. There are straps on the Keyboard Interface that can be used to modify the handshaking process. These are discussed later in this section.

NOTE

The computer should not be allowed to echo back information that has been transmitted as a block from the terminal.

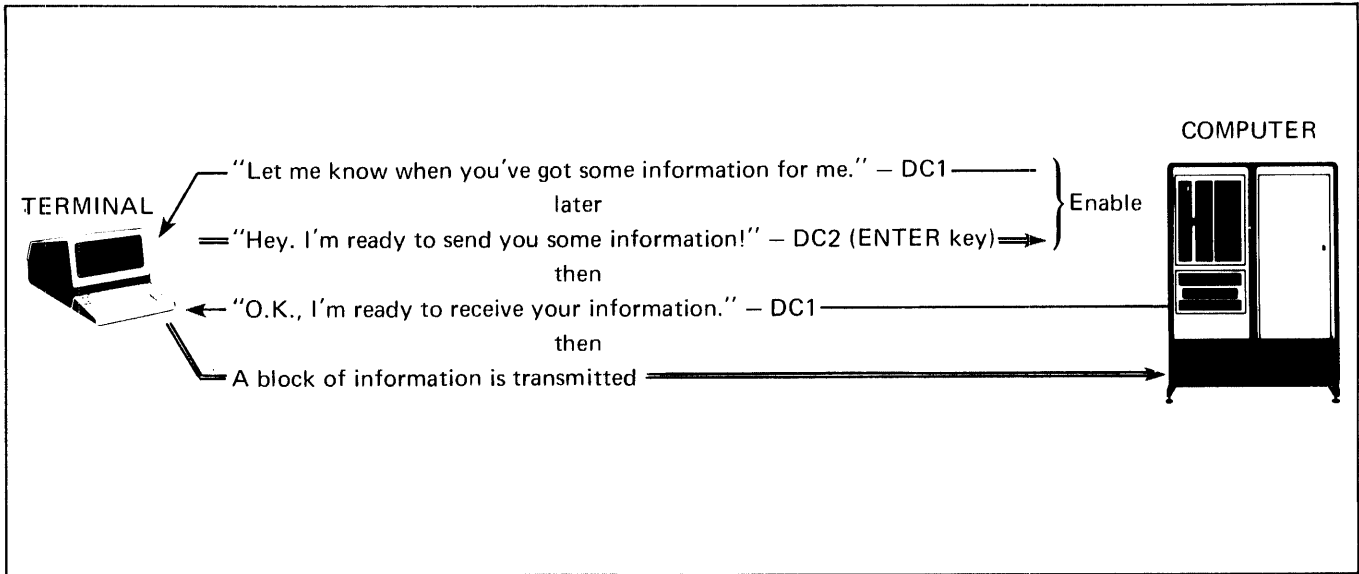


Figure 5-2. Block Transfer Enabled By The ENTER Key

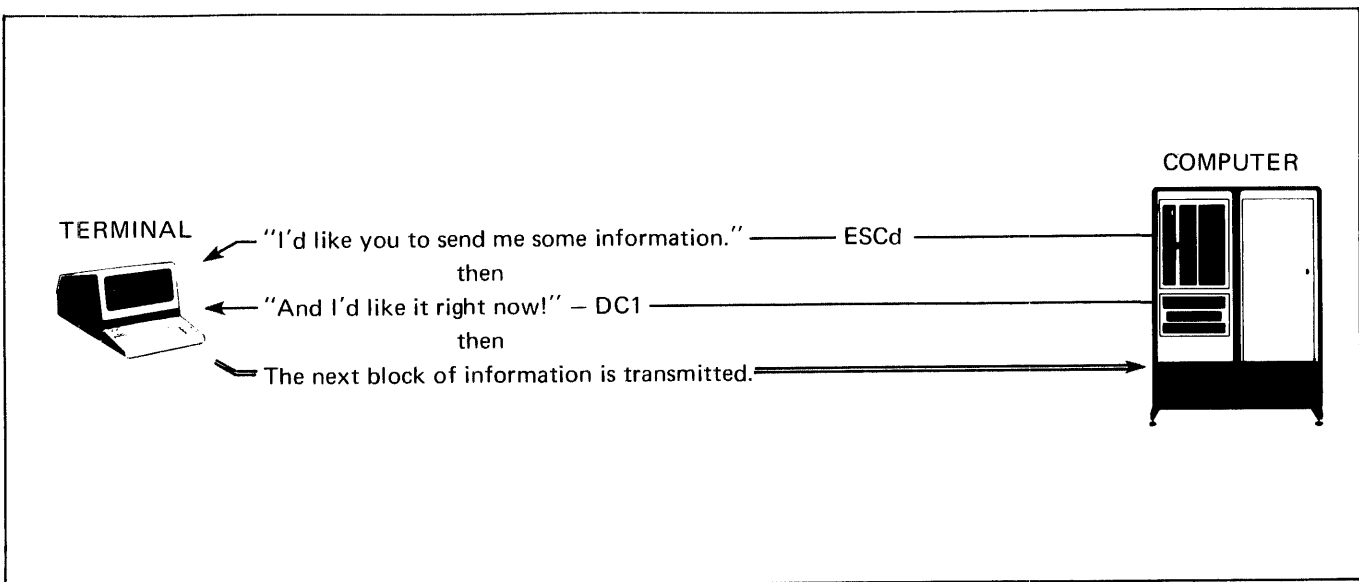


Figure 5-3. Block Transfer Enabled By The Computer

## Block Mode

When the terminal is in Block Mode (BLOCK MODE key down), characters are not transmitted as they are typed. Instead, the user can input data to the terminal, then edit and correct the data before sending it to the computer using the **ENTER** key. The data can be grouped into convenient blocks, either lines or pages (refer to the configuration procedures later in this section). Block Mode operation allows you to efficiently utilize computer and communication facilities.

The G and H switches on the Keyboard Interface PCA are used to control the terminal's response to block transfer requests (refer to table 5-8).

Switch G	Setting H	Block Operation
Closed	Closed	Data transfers used DC1/DC2 handshake. Other transfers are triggered by the receipt of a DC1 character.
Closed	Open	Data is sent when the <b>ENTER</b> key is pressed. Other block transfers are triggered by the receipt of a DC1 character.
Open	Closed	All block transfers require a DC1/DC2 handshake.
Open	Open	No DC1/DC2 handshake is required for any block transfer.

Note: In half duplex operation, a line turnaround is substituted for a DC1 character.

Note: In half duplex operation, a line turnaround is substituted for a DC1 character.

The size of the block of information transferred in BLOCK MODE, and the control characters used to separate fields and to terminate blocks differ somewhat, depending on the Line/Page Strapping of the terminal and whether or not the terminal is operating in FORMAT MODE. Figure 5-4 illustrates these differences.

In the example in figure 5-5, the user has an application in which order data is to be entered in the same format as a standard company form.

## Full Duplex Operation

In full duplex operation, the characters which are typed at the keyboard are transmitted to the computer and are not displayed unless they are returned by the computer. This setting is ignored when in Block Mode.

## Teletype Compatible Communications

In teletype compatible (full duplex, character mode) applications, the terminal can be quickly configured for use by following the instructions given in the Installation section. Note that if block data transfers are used the computer should be programmed to use the simple DC1/DC2 protocol described under Multicharacter Transfers.

## Half Duplex Operation (202 Modem Compatibility)

In half duplex operation, data is sent in only one direction at a time. In order to change the direction of data flow, a line turn around must occur. This means that the sender becomes the receiver and the receiver becomes the sender. Line turn arounds are controlled by half duplex line protocols. Both the computer and the terminal must use the same protocol otherwise malfunction and loss of data will result. The Main Channel and Reverse Channel protocols are examples of half duplex operation.

Initially the terminal is in the transmit state. While in this state the terminal will ignore data sent from the computer. The terminal will remain in the transmit state until one or more of the following occur:

- An ON to OFF transition on the SB (CCITT 122) line (Reverse Channel)
- An end of data character (ETX or EOT) is sent (Main Channel)

In the example in figure 5-5, the user has an application in which order data is to be entered in the same format as a standard company form.

- The user tries to send an end of data character from the keyboard (control-C, control-D)

The above conditions cause the terminal to switch to the receive state.

The terminal then receives the processed data until one of the following occurs:

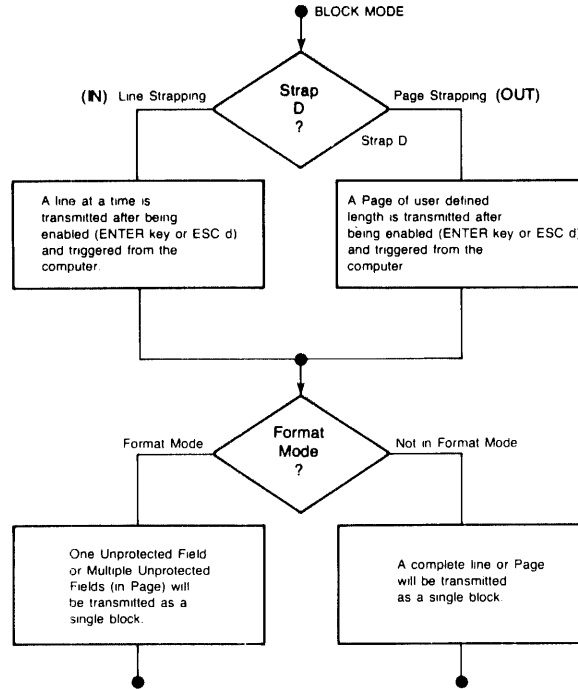
- An ON to OFF transition of the CF (CCITT 109) line (Reverse Channel)
- An end of data character (ETX or EOT) is received (Main Channel)

The terminal then requests the computer or modem for permission to transmit. The computer or modem responds with transitions of the CB (CCITT 106) and SB (CCITT 122) lines. (If the computer or modem does not respond within 2.6 seconds the terminal will return to the receive state.) If the computer is ready the terminal will begin to send any data present in its output buffer.

The terminal provides a range of half duplex line protocols, including Bell 202 modem compatible protocols. These protocols are selected by switch settings on the Keyboard Interface PCA. Table 5-8 contains a list of the communication switches that are used to select half-duplex protocols.

Half-duplex operation can be controlled either by RS232C signal lines or by control characters in the data being transferred or by a combination of characters and signals. The Main Channel protocol uses control characters while the Reverse Channel protocol uses control signal lines.





**STRAPPED FOR LINE**

**STRAPPED FOR PAGE**

**non-FORMAT MODE**

- data is transferred from the current cursor position to the end of the line or to a Record Separator (RS) control character, whichever occurs first.
- imbedded control characters are transmitted. If present, the RS character is sent.
- the Block is terminated by the transmission of a CR(LF), a Carriage Return and Line Feed if AUTO LF is depressed. (A local CR(LF) is executed to reposition the cursor; if no more information is present at or beyond the cursor the transmission consists of RS CR(LF).)

- data is transferred from the current cursor position to the end of the terminal's allocated memory or to the next RS, whichever occurs first. Thus the Block to be transferred could be several lines of information.
- imbedded control characters are transmitted. If present, the RS character is sent.
- if multiple lines are in the Block, they are separated by CR LF in the transfer. The Block is terminated by the transmission of an RS.

**FORMAT MODE**

- only information in Unprotected Fields is transmitted. If the cursor is not in an Unprotected Field it will be forwarded to the next one or RS CR(LF) will be transmitted if no such field exists. Data is transmitted from the cursor position to the end of the Field or an RS, whichever occurs first. Thus the Unprotected Field to be transferred could no be longer than one line in length.
- imbedded display control characters are not transmitted. If present, the RS character is sent.
- the Block is terminated by the transmission of a CR(LF) and the cursor is forwarded one character position.

- only information in Unprotected Fields is transmitted. If the cursor is not in an Unprotected Field it will be forwarded to the next one or RS will be transmitted if no such fields exist. Data found in Unprotected Fields is transmitted from the cursor until an RS or the end of memory is encountered.
- imbedded display control characters are not transmitted. If present the RS character is sent.
- a Unit Separator (US) control character (or RS character for multipoint) is transmitted between each Unprotected or Transmit Only field. The Block is terminated by the transmission of an RS.

Note: In Multipoint configuration the Group Separator character (GS) is used in place of RS.

Figure 5-4. Block Mode Operation

**STEP 1.** The user presses the Special Function key, which he has previously programmed in a remote computer routine to both automatically display the form shown and turn on FORMAT MODE. (REMOTE and BLOCK MODE are depressed.)

**STEP 2.** All areas of the display have been programmed to be protected except for the dark fields within the form itself. Thus, as data is typed at the keyboard only these dark areas can be written into. The cursor automatically will tab from one field to the next when a field boundary is encountered or by use of the **TAB** key. The user now inputs data from the keyboard.

ORDER #	COMPANY NAME	SHIPPING ADDRESS: STREET			
DATE	BILLING #	CITY	STATE	ZIP	
ITEM #	PRODUCT NAME	PRICE	QNTY	TOTAL	CODE

The complete form would look as follows:

ORDER #	COMPANY NAME	SHIPPING ADDRESS: STREET			
DATE	BILLING #	CITY	STATE	ZIP	
ITEM #	PRODUCT NAME	PRICE	QNTY	TOTAL	CODE

**STEP 3.** After filling out the form and correcting any noticed errors, the **ENTER** key is pressed once. The following sequence of events would then occur:

- Having received a DC1 from the computer, the terminal transmits a DC2.
- Computer software recognizes the DC2 and responds with a second DC1.

- The terminal receives the DC1 and transmits all data as one Block, fields separated by US's and the Block terminated by an RS.

**STEP 4.** The form full of data has been transmitted to the computer. The user could then Home the cursor, hit **ESC** to clear only the data from the form in FORMAT MODE, and enter a second set of data inputs — repeating the sequence and reusing the form.

Figure 5-5. Example of Format Mode with Page Strapping

Table 5-8. Keyboard Interface PCA Strapping Options for Point-to-Point

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
A	Function Key Transmission	The escape code sequence generated by the major function keys (such as, ROLL UP, ROLL DOWN, etc.) are executed locally, but not transmitted to the computer.	The escape code sequences generated by all keys are transmitted to the computer. If operating in half duplex, the function is also executed locally.
B	Space Overwrite (SPOW) Latch Enable	Spaces typed will overwrite existing characters.	When the SPOW latch is off, overwriting occurs as normal. When the SPOW latch is on, spaces cause the cursor to move forward but not overwrite any existing characters. The SPOW latch is turned on by a Carriage Return, and off by a Line Feed, Home or Tab.
C	Cursor End-of-Line Wraparound	At the end of each line, a local Carriage Return and Line Feed are generated; the cursor moves to the beginning of the next line.	A Carriage Return and Line Feed are not generated at the end of each line. The cursor remains in and overwrites column 80.
D	Line/Page	The terminal is set to transfer a line at a time in Block Mode.	Entire pages of information are transferred in Block Mode.
E	Paper Tape Mode	When the <b>READ</b> key is pressed with <b>AUTO LF</b> key latched down, each tape record begins with an LF and is terminated by a CR.	Each tape record is terminated by CR.
F	Fast Binary Read	The transmission rate is determined by the BAUD RATE switch on the keyboard.	When an FR (Fast Binary Read) is issued by the computer, the baud rate is switched automatically to 9600 baud (if the terminal is equipped with cartridge tape units).
G	Block Transfer Handshake	In Block Mode, all data transfers to the computer are sent upon receipt of a DC1 from the computer.	All Block Mode transfers (i.e., cursor sense, terminal and device status, device I/O responses, display memory, and function keys) are preceded by a DC2. The terminal sends the DC2 upon receipt of a DC1 from the computer. After the CPU receives the DC2 from the terminal, another DC1 is required to trigger transmission of data from the terminal.
H	Inhibit DC2	During Block Mode Handshake transfers, the terminal sends a DC2 in response to a DC1 prior to sending data. (See Block Transfer Handshake strapping above.)	A DC1 from the computer is not required to trigger data transfers to the computer. Also, the DC2 from the terminal is not sent during Block Mode Transfer handshakes. (See Block Transfer Handshake strapping above.) Additionally, when the <b>ENTER</b> key is pressed in Block Mode the cursor will be placed in the first column before transmission occurs if operating in Line/Field Mode (switch D closed) or Home'd if operating in Page Mode (switch D open.) Opening both switches G and H eliminate the terminal's use of the Handshake protocol entirely.
J	Auto Terminate	No effect.	When in BLOCK mode and the ENTER key is pressed, places a non-displaying terminator before the cursor position.
K	Clear Terminator	No effect	Clear terminator caused by Strapping Option J or CR.
L	Self Test Inhibit	No effect.	Self Test function is inhibited. Pressing TEST key or issuing CR z displays the NO TEST message. TAPE TEST and DATA COMM SELF TEST functions are not affected.
M	INSERT and DELETE CHAR with wrap (Reverse Sense)	No effect.	Reverses effect of <b>CNTRL</b> key on INSERT CHAR and DELETE CHAR keys (i.e., when key is pressed, line wrap around is in effect without having to press CNTRL key. When either key is pressed while pressing CNTRL, normal insert character and delete character functions are in effect.)

Table 5-8. Keyboard Interface PCA Strapping Options for Point-to-Point (Continued)

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
N	Escape Code Transfer to Printer	No effect.	Escape codes relating to the display (e.g., display enhancements, alternate character sets, format mode, fields, etc.) are sent to printer if it is selected as a destination device.
P,Q	Compatibility Mode	These switches set the terminal to be compatible with Tektronix control commands when initialized (power on or full reset).	
		P-closed, Q-closed P-closed, Q-open P-open, Q-closed P-open, Q-open	Normal operation Unscaled Compatibility Mode and 2048 byte data comm buffer. Scaled Compatibility Mode and 2048 byte data comm buffer. 2048 byte data comm buffer.
R	Circuit Assurance	The transition from receive state to transmit state occurs after both CB (106) (Clear to Send) and SB (119) (Secondary Receive Data) go on within 2.6 seconds. Otherwise, the terminal returns to the receive state.	The transition from receive state to transmit state occurs after CB (106) (Clear to Send) goes on.
S,T	Main Channel Protocol	Reverse Channel protocol (both switches closed).	<b>S-closed, T-open:</b> Main channel with STX/ETX as Start of Data and End of Data. <b>S-open, T-closed:</b> Main channel with EOT as End of Data. <b>S-open, T-open:</b> Main channel with ETX as End of Data.
U	CPU Break	The CPU can interrupt the terminal while it is in the transmit state. The CPU initiates an ON to OFF transition of the SB (119) (Secondary Receive Data) line. The terminal responds by turning off CA (105) (Request to Send) and going to the receive state.	The terminal ignores all transitions on the SB (122) (Secondary Receive Data) line from the modem in the transmit state.
V	Carrier Detect	When the terminal is in the receive state, an ON to OFF transition of CF (109) (Carrier Detect) line from the modem causes the terminal to go into the transmit state. Transitions of CF have no effect while the terminal is in the transmit state.	Transitions of CF (109) (Carrier Detect) line have no effect on the terminal.
W	Data Comm Self Test Enable	Enables DATA COMM SELF TEST from either the keyboard or escape sequence.	Disables DATA COMM SELF TEST. If self test is attempted (by either the keyboard or escape sequence), the test will be aborted and ERROR 0 will appear on the display.
X	Data Speed Select	Holds data speed signal low (CH (111) = 0).	Sets data speed signal high (CH (111) = 1).
Y	Transmit LED	The TRANSMIT light on the keyboard is turned on when CB (106) (Clear to Send) line from the modem is high. It is turned off when the CB (106) line goes low.	The TRANSMIT light on the keyboard is turned on when the CC (107) (Data Set Ready) line from the modem is high and the 13260B Extended Asynchronous Communications Interface PCA is used. It is turned off when the CC line goes low.
Z	Parity	The PARITY switch on the terminal keyboard is affected as follows:	
		<b>No Parity:</b> Send 8 bits and receive 8 bits. Force bit 8 to zero. Check for parity error. <b>Odd Parity:</b> Send 7 data bits + odd parity. Receive 7 data bits + odd parity. Check for parity error. <b>Even Parity:</b> Send 7 data bits + even parity. Receive 7 data bits + even parity. Check for parity error.	<b>No Parity:</b> Send 8 bits and receive 8 bits. Force bit 8 to one on send. No check for parity error. <b>Odd Parity:</b> Send 7 bits + odd parity. Receive 7 bits. No check for parity error. <b>Even Parity:</b> Send 7 data bits + even parity. Receive 7 data bits. No check for parity error.

**MAIN CHANNEL (CHARACTER CONTROL) PROTOCOL.** The Main Channel protocol is for use in half-duplex or Bell 202 modem equivalent networks where secondary channel signals are not available. The Main Channel protocol uses control characters to "frame" each data transmission. These framing characters indicate to the receiving station that a data transmission has begun or ended.

An ASCII STX (octal 002) character can be used to indicate the start of a data transmission. An ASCII ETX (octal 003) or EOT (octal 004) character is used to indicate the end of a data transmission. When these characters are received they are used to perform a line turn-around.

The following switch settings should be made on the Keyboard Interface PCA to operate using the Main Channel protocol:

SWITCH	SETTING	DESCRIPTION
R	Open	
S,T	Closed,Open Open,Closed Open,Open	<STX>data<ETX> data<EOT> data<ETX>

Note that at least one of the S or T switches must be open to select Main Channel protocol.

**Example:**

U,V,W,X,Y,Z All Open — Variations of the Main Channel Protocol are discussed under Other Protocols and in Appendix C.

The operation of the Main Channel protocol is shown in figure 5-6. Sample data transfers are shown in figure 5-7. Figures 5-7a and 5-7b illustrate the line turn-arounds that occur during a log-on sequence when in character mode. Figures 5-7c and 5-7d illustrate the transfers that occur during block mode operation.

**REVERSE CHANNEL (SIGNAL LINE CONTROL) PROTOCOL.** The Reverse Channel protocol is for use in half-duplex or Bell 202 modem equivalent networks where secondary channel signals are available. The Reverse Channel protocol uses changes on secondary channel lines SA (CCITT 120) and SB (CCITT 122) to control line turn-arounds.

The following settings should be made on the Keyboard Interface PCA to operate using the Reverse Channel protocol:

SWITCH	SETTING	DESCRIPTION
R	Closed	Monitor the CB line
S,T	Closed,Closed	Reverse Channel (no framing characters)
U	Closed	Watch for computer interrupts (SB>0)
V	Closed	Watch for Carrier (CF) transitions

**Example:**

W,X,Y,Z All Open — Variations of the Reverse Channel protocol are discussed under Other Protocols and in Appendix C.

The operation of the Reverse Channel protocol is shown in figure 5-8. Sample data transfers are shown in figure 5-9.

**OTHER PROTOCOLS.** In addition to the Main and Reverse Channel protocols you can select various features of both to configure a custom protocol to suit your own requirements. A flowchart of the overall Basic Communications function including the Half-Duplex settings is given in Appendix C. You can create a custom protocol using this flowchart and the switch descriptions in table 5-8. When more than one terminal must share a modem or hardwired communication line, the Multipoint protocol must be used. Refer to the description of Block Protocols.

**Monitor Mode**

Monitor Mode is an added feature available with the 13260A and 13260B interfaces. Refer to Multipoint Monitor Mode for a description.

**Configuration**

A procedure for configuring the terminal for point-to-point operation is given in figure 5-10.

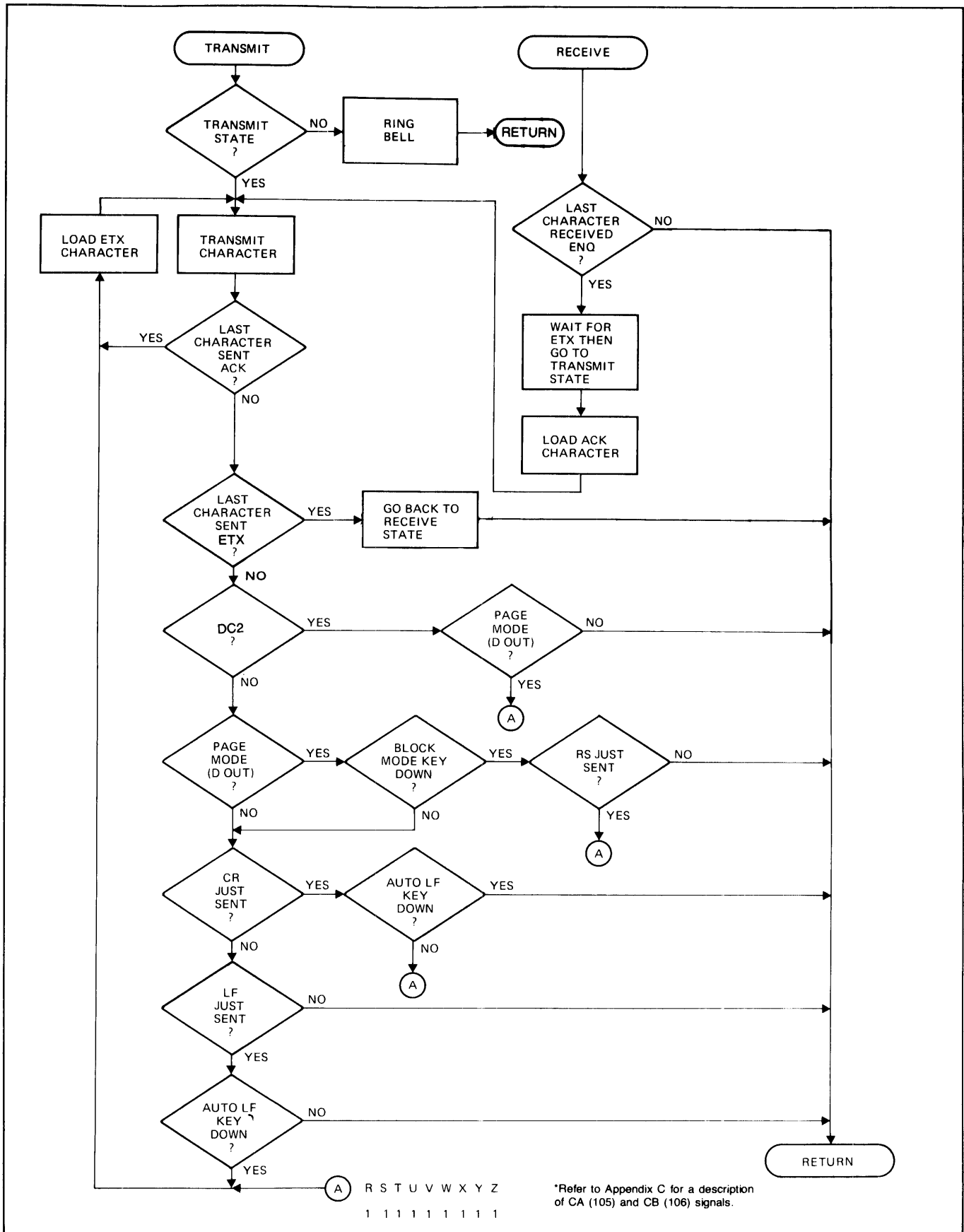


Figure 5-6. Main Channel Protocol

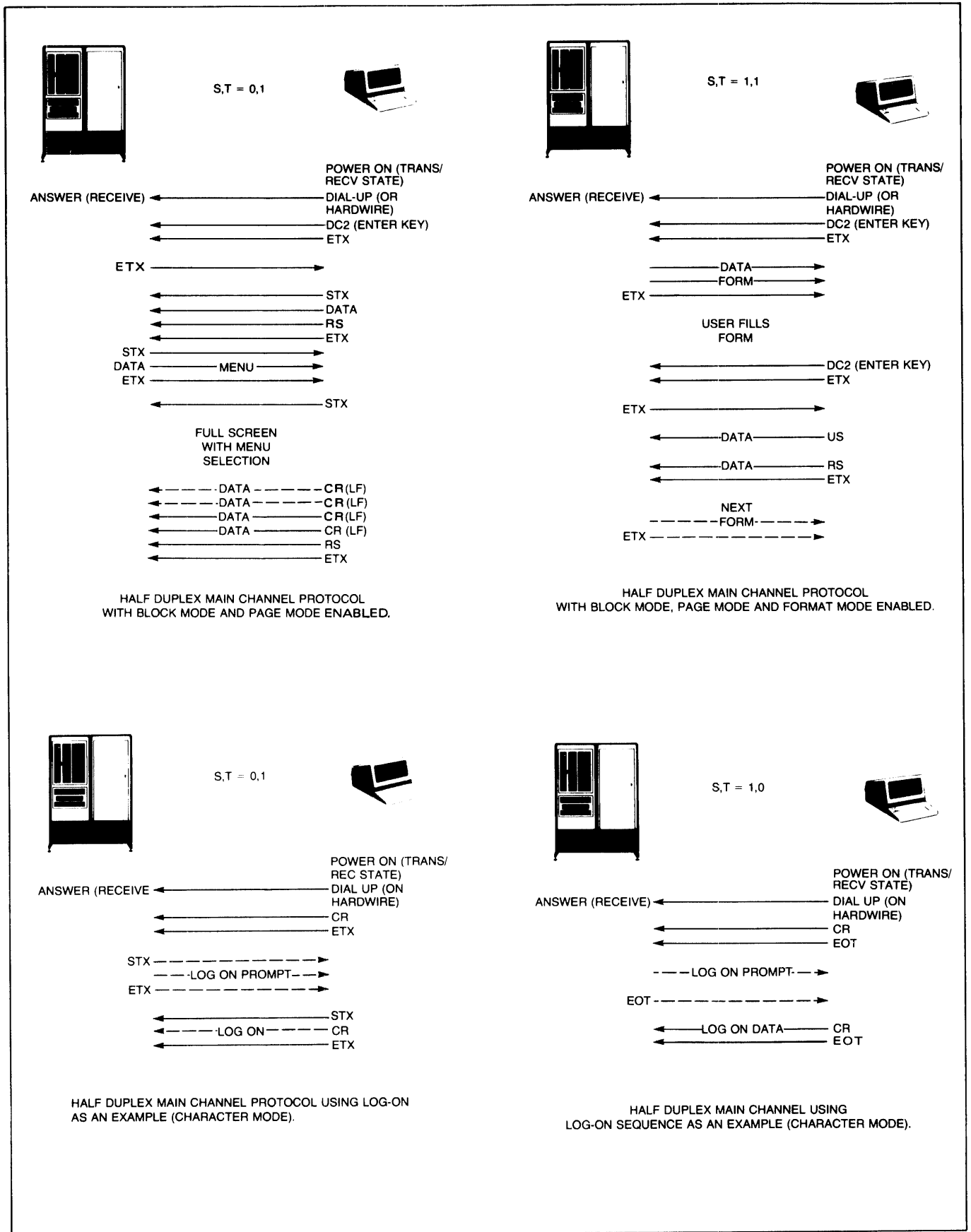


Figure 5-7. Sample Data Transfers Using Main Channel Protocol

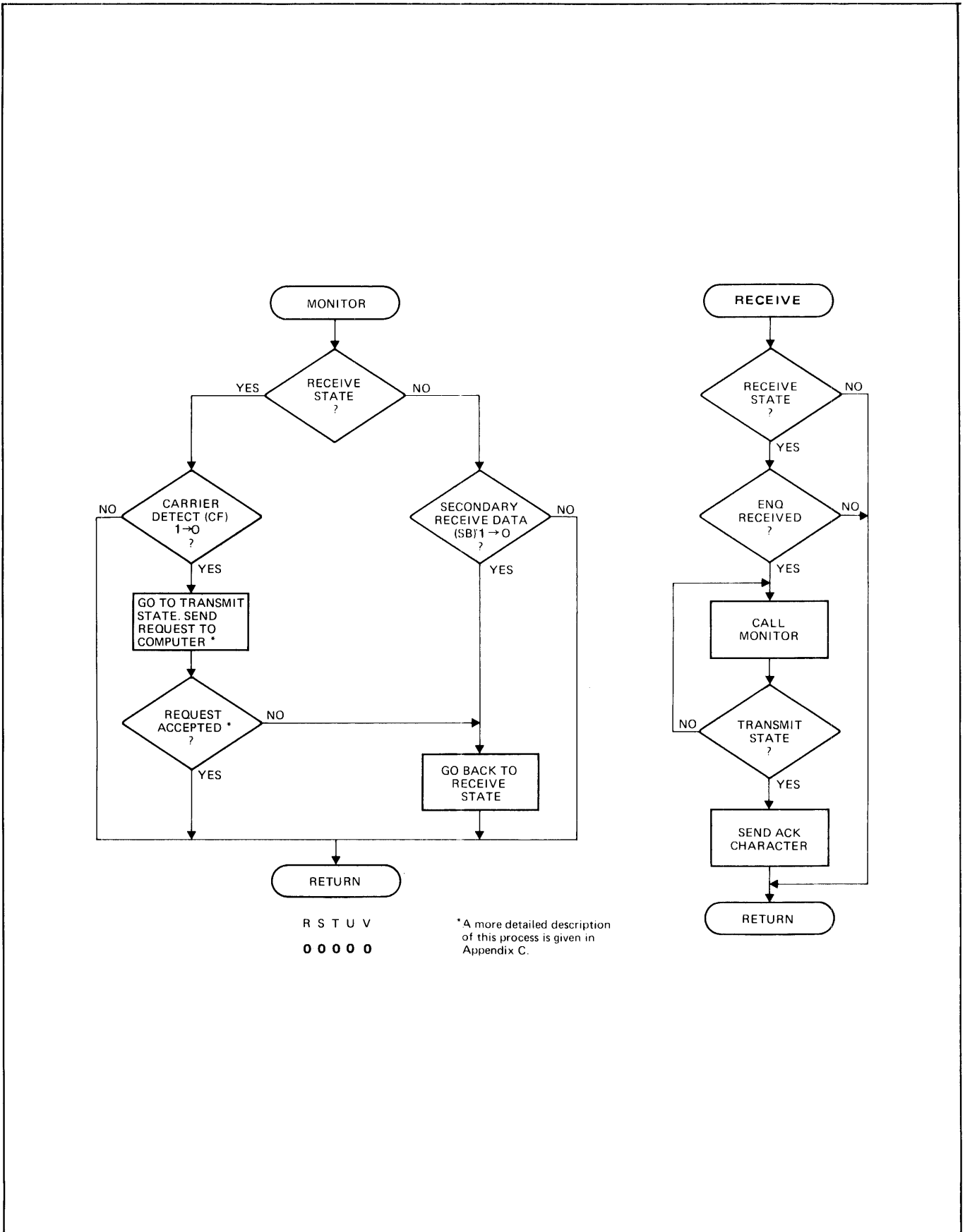
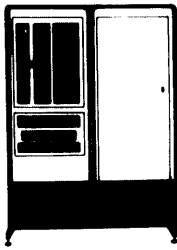


Figure 5-8. Reverse Channel Protocol



CHARACTER MODE HALF-DUPLEX REVERSE CHANNEL



POWER ON (TRANS/RECV STATE)

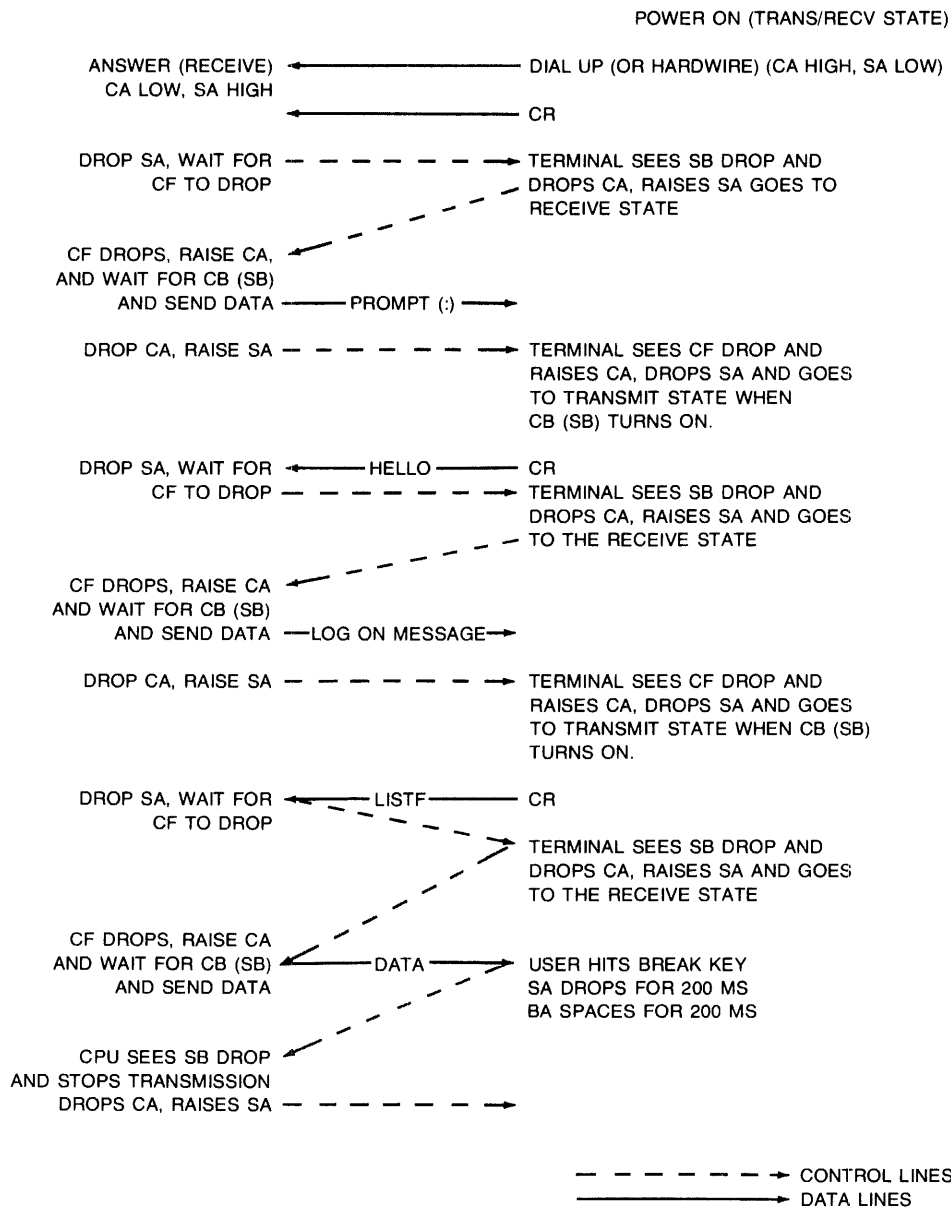


Figure 5-9. Sample Data Transfers Using Reverse Channel Protocol

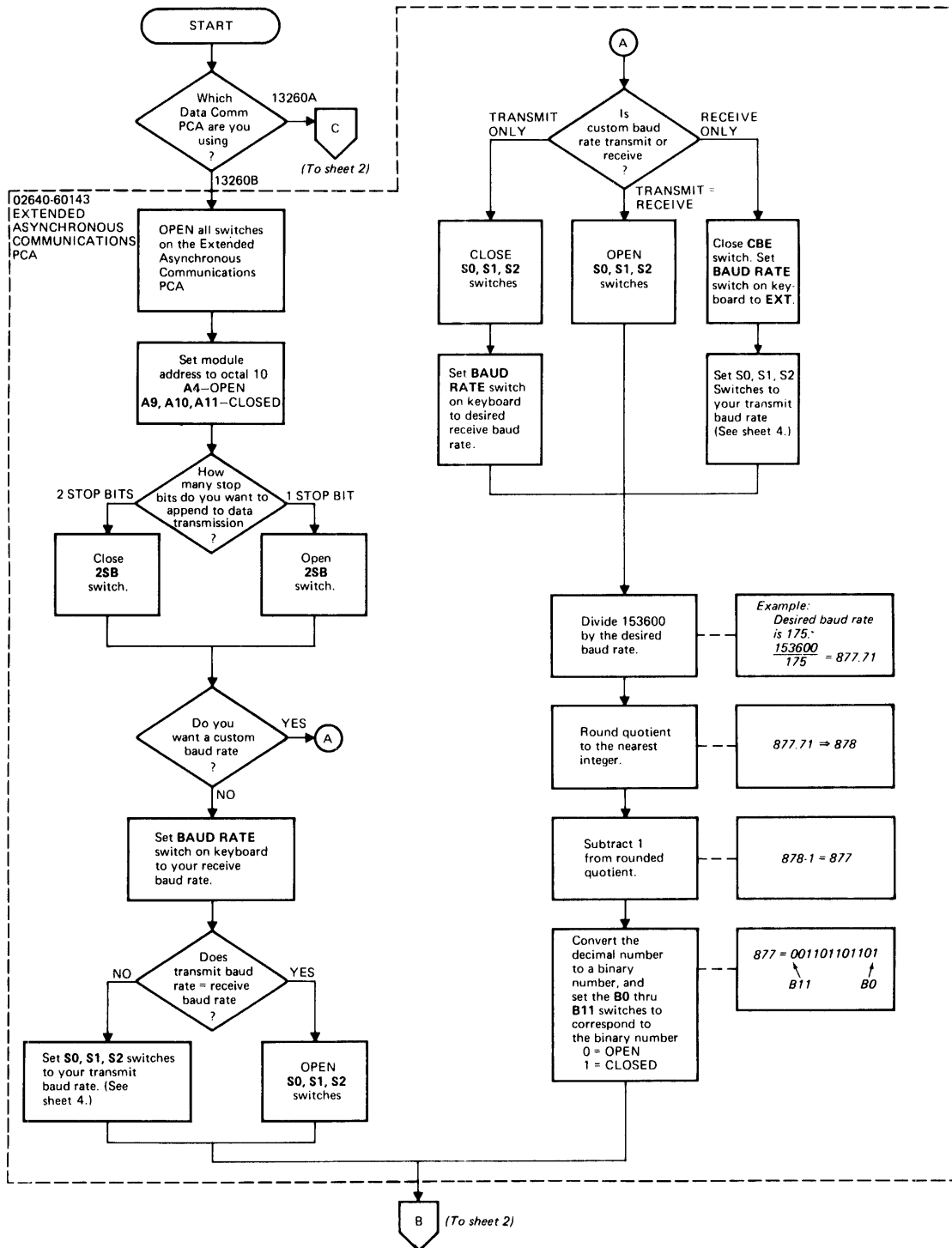


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 1 of 4)

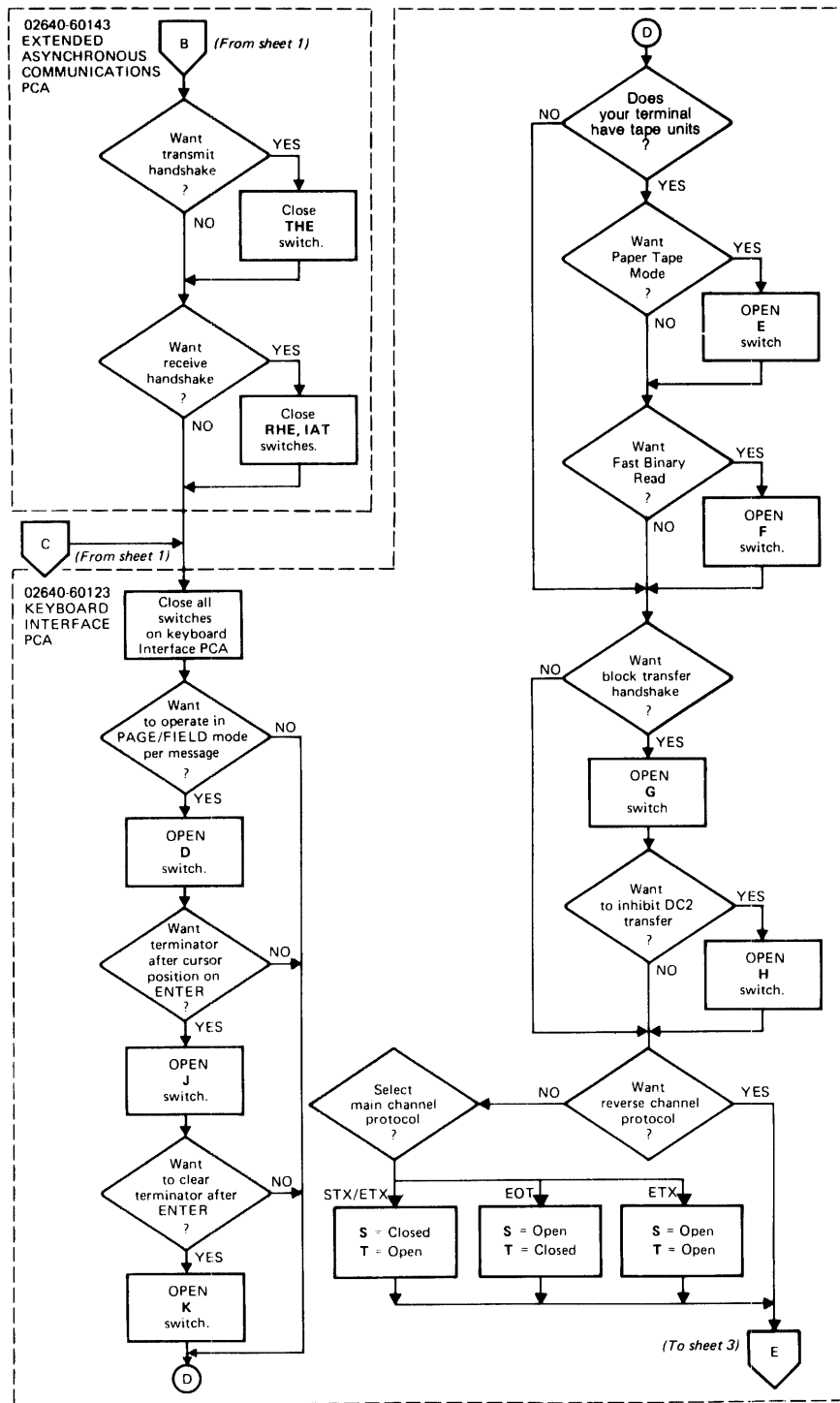


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 2 of 4)

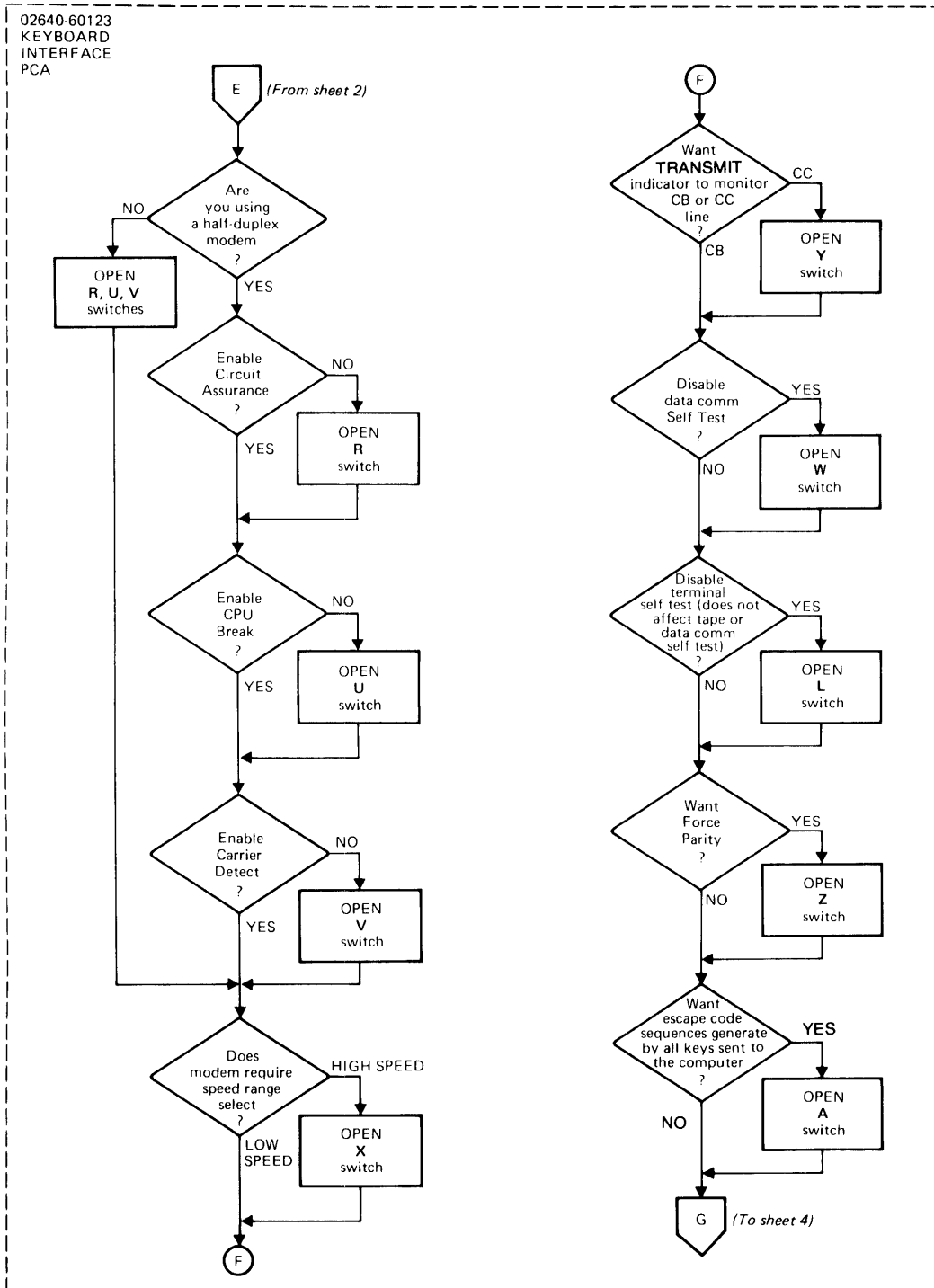


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 3 of 4)

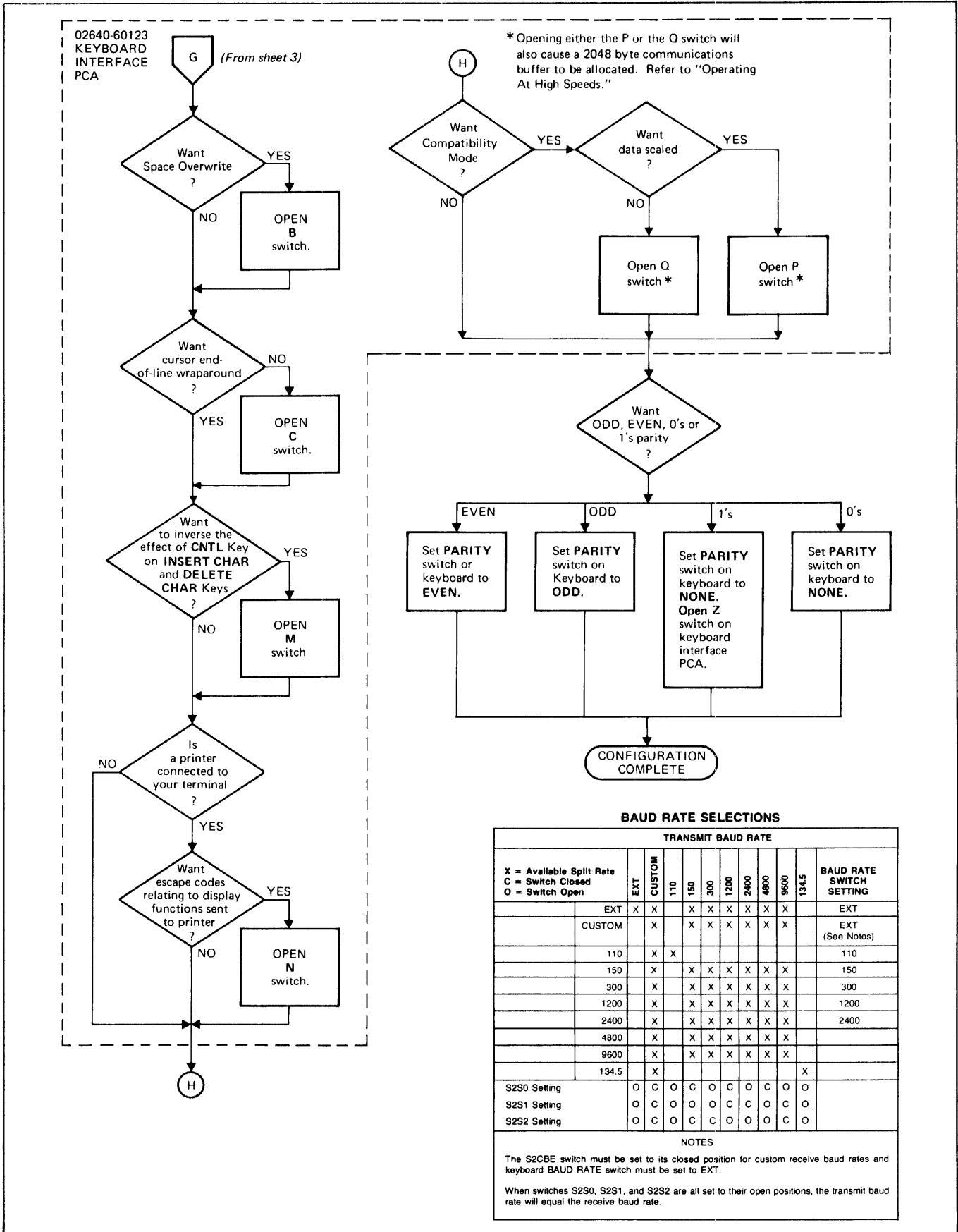


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 4 of 4)

## BLOCK PROTOCOLS WITH 13260C OR 13260D COMMUNICATIONS ACCESSORY

Block protocols transfer data in blocks. The blocks are made up of three parts:

- Block framing characters
- Text (1 to n characters, where n depends on terminal configuration).
- Block check character(s)

This data format is always present in block protocols. In addition, block protocols use special character sequences to control all data transfers.

A block check character is included at the end of each data block. If a data error is detected the protocol will normally automatically attempt a retransmission of the block.

### Character Mode Transfers

Character mode transfers are not permitted with block protocols. All data transfers are implemented using a block data structure.

### Multi Character Transfers

When the terminal makes a multi character response of fixed length to the computer (status etc.), the data sent will be in the form of a block with framing characters and one or two block check characters.

### Message Blocks

A message consists of one or more blocks of text data. The use of blocks enables the terminal to efficiently buffer data, respond to transmission errors and guarantee data integrity. Maximum block size is strap selectable and permits you to use the size best suited to computer requirements. (Refer to Buffer Size.)

### Block Operation

The block protocol is designed to operate using either synchronous or asynchronous communications. Data transmission is done in multiple character blocks. The block size used is limited by the terminal's communications buffer (refer to Configuration).

The input buffer size limits the size of the data block that can be sent to the terminal. For example, if the input buffer size is 500 bytes, sending a block of data larger than 500 bytes will result in a loss of data. If this happens an EOT character will be sent to the computer.

Two forms of text blocks are shown in figure 5-11. The first is a block received from a computer. Note that no ID characters are used since the terminal or terminals to receive the data have already been identified by a select sequence. The second block is one sent from a terminal. In multipoint configurations, since more than one terminal may have been polled, the first text block sent from each terminal must have the terminal ID included. The ID characters are not repeated (as in poll and select sequences) since they are included in the block check character.

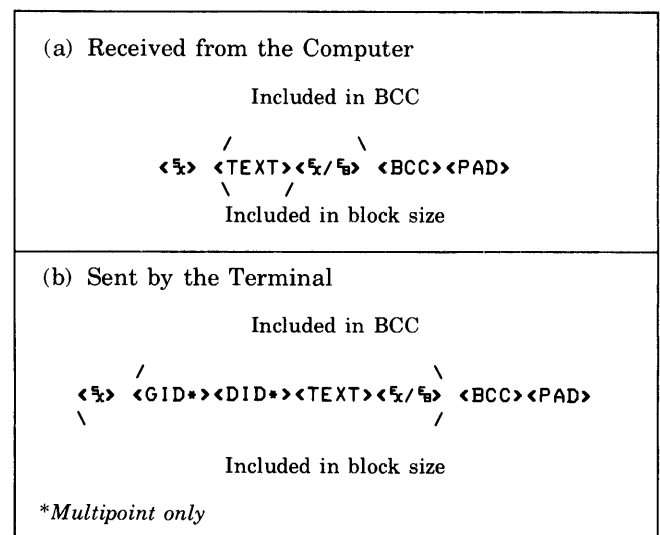


Figure 5-11. Examples of Block Transmission

**TEXT TERMINATION.** When the terminal is receiving text (Text-In mode) it will accept only ETB (octal 27), ETX (octal 3), or ENQ (octal 5) as a text block terminator. An ETB indicates the end of a block with one or more blocks to follow. An ETX indicates the end of the current block and the end of the text transfer. An ENQ character indicates that the current block has been aborted. The terminal will respond to the ENQ with a NAK to request the retransmission of the aborted text block. When the terminal is sending data (Text-Out mode), it will terminate text blocks with either an ETB or an ETX character.

All characters sent or received between the STX character and the terminating character must not be more than 40 milliseconds apart for asynchronous operation (13260C). Synchronous operation requires SYN characters to be sent as fill characters if no text characters are ready for transmission.

The terminal may send a STX ENQ as a Temporary Text Delay (TTD) notification. This indicates that there is more text to come but that it is not ready to be transmitted (i.e., it is still being read from a cartridge tape). A TTD should be answered with a NAK to request the transmission of the text block, or an EOT to reset the terminal to control mode.

**DATA CHECKING.** There are two types of data checking used with the multipoint protocol. The first is a check of each character as it is received and is called a vertical redundancy check (VRC). This check is only used for ASCII coded characters. The second is a check of an entire block of data and is called a block check. Two types of block check are available. The first is a longitudinal redundancy check (LRC). You can also choose a more complex block check called a cyclic redundancy check (CRC).

**Character Checking.** The vertical redundancy check is also known as a parity check. When an ASCII character is transmitted by the computer or the terminal, the high order (eighth bit of each character is set to a "1" or a "0" to make the number of "1" bits in the character either even (EVEN parity) or odd (ODD parity). The parity must be the same for both the computer and the terminal. For example, if even parity is used the high order bit of each character would be set to cause the number of "1" bits in the character to be even.

Character checking is not done when EBCDIC code is used or when operating in transparency mode. The parities available are listed in table 5-9.

Table 5-9. Parities Available with ASCII Data

ODD PARITY	Input characters are checked for odd parity. Output characters are supplied with odd parity.
EVEN PARITY	Input characters are checked for even parity. Output characters are supplied with even parity.
NONE PARITY	Input characters are checked for a "0" parity bit. Output characters are supplied with a "0" bit for parity.

**Block Checking.** Each block includes a block check character (BCC). The BCC character(s) is in addition to the parity bit set for each character transmitted. This BCC can be a one (LRC) or a two (CRC16) character check sum. The type of BCC and parity desired can be set to match almost any communications requirements.

**LRC.** The LRC check character is a 7 bit check sum obtained by exclusive "OR"ing the low order 7 bits of each character included in the text block. A parity bit (VRC) is then added to this character when it is transmitted. For EBCDIC, all 8 bits are "OR'ed" together and no parity bit is added.

**CRC16.** The CRC16 check is a 16 bit (two character) check sum calculated using the following formula:

$$X^{16} + X^{15} + X^2 = 1$$

This is compatible with the CRC16 check sum used by IBM.

**STRAP SELECTABLE OPTIONS.** The following options are strap selectable on the interfaces used by the block protocols:

- Code Selection (ASCII/EBCDIC)
- Block Check Character (LRC/CRC16)
- Data Comm Buffer Size
- Space Compression
- Transparent Transmission
- Synch Characters

(For backward compatibility strapping, refer to Tables C-4 and C-5 in Appendix C.)

**Code Selection (ASCII/EBCDIC).** The terminal can be set to use either ASCII or EBCDIC data codes. All data and most control characters translate directly from one code to the other (map to the same graphic). A list of the characters and their codes is given in Appendix C. Control characters that do not translate directly between the two codes are:

ASCII	
graphic	octal
ACK0    0	020 060
ACK1    1	020 061
WACK    ;	020 073
RVI     <	020 074
EBCDIC	
graphic	octal
ACK0    (no graphic equivalent)	020 160
ACK1	020 141
WACK    ,	020 153
RVI     @	020 174
EBCDIC characters that have no equivalent ASCII character are converted to a "?" character.	

All terminals on the same communication line must use the same code type.

The J07 switch is used to select the code to be used to represent data. The codes available are as follows:

<b>J07</b>	
0 (closed)	= ASCII
1 (open)	= EBCDIC

**Block Check Character (BCC).** The Block Check Character is used to verify the accuracy of transmitted data. Switch J06 allows you to select the type of test used. The terminal will then automatically perform the proper test and generate the same type of check character sent to the computer. The types of check character available are as follows:

**J06**

0 (closed)	=	Longitudinal redundancy check (LRC)
1 (open)	=	Cyclic Redundancy Check (CRC 16)

**EXTENDED TEXT FEATURE.** The Extended Text Feature is selected by setting the J05 switch on the multipoint interface to the 1 (open) position.

**J05**

1 (open)	=	Extended features
0 (closed)	=	No effect

The Extended Text feature can be used to generate and delete three special characters used with an IBM 3270 terminal. After the computer has selected the terminal to receive data, the first text block will have the following form:

3 leading characters		
/		\
⌘	⌘	⌘
	1 WCC TEXT	

Note that the characters follow STX and precede the text block. Since these characters are not used by the terminal, they would normally be accepted as a part of the text block. Selecting the Extended Text feature will cause the terminal to discard these three characters before processing the text.

When the first block of text is sent to the computer in response to a POLL sequence, the computer expects to see the following:

⌘	GID DID AID CCA CCA TEXT	⌘
	\ /	
	3 leading characters	

The leading characters that are sent by the terminal are as follows:

**AID** — attention I.D. This character will normally be an apostrophe (') 47 octal ('047'). If you use the PA or PF functions refer to their descriptions elsewhere in this section.

**CCA** — Current Cursor Address. This is a two character address and will always be SP,SP ('040 040'). This is the cursor home position (0,0).

Note that if you have configured the terminal for text mode compatibility and are not operating with such a system, the first three characters in the first text block

received by the terminal will be ignored. Also, the three leading characters (AID, CCA, CCA) will be embedded in the transmitted text block.

**Buffer Size.** You must set the amount of terminal memory allocated for use as input and output communication buffers. When the terminal is inputting data it uses this space for a single input buffer. When the terminal is outputting data the buffer space is divided into two or more output buffers. The basic terminal configuration uses a 500 byte input buffer or two 250 byte output buffers.

When the terminal is selected, any data waiting in the output buffers is lost. The output buffers then become an input buffer to hold data sent from the computer until the terminal can process the characters.

The terminal will respond to select sequences with a WACK when there are fewer than 250 bytes available in the input buffer. The terminal will respond with an ACK as soon as 250 bytes of buffer space becomes available. Note that if too large a block is sent to the terminal following the ACK it may result in a buffer overflow and an EOT will be returned.

It is often desirable to increase the size of the communication buffers to optimize use of the computer. The size of the terminal's input buffer can be set on the communication interface. The input buffer size can range from 500 to 4000 bytes.

If there is no RAM memory installed in the address range 48K-52K, buffer space is allocated from the terminal's display memory (see figure 7-10). This means that the larger the buffer size, the smaller the amount available to display memory.

Input buffer size is set by switches J17 and J16 as follows:

J17	J16	Input Buffer
0	0	500 bytes
0	1	1000 bytes
1	0	2000 bytes
1	1	4000 bytes

where 1 = open  
0 = closed

Output buffer size can range from 250 to 2000 bytes. Output buffer sizes are limited to a maximum of one half the input buffer size. Output buffer sizes are set with switches T and U on the Keyboard Interface as follows:

T	U	
0	0	2 buffers, each one half the size of the input buffer (2000 max)
1	0	250 bytes
0	1	500 bytes
1	1	1000 bytes

where 1 = open  
0 = closed



## Data Communications

Note that if the output buffer is inadvertently set larger than one half of the input buffer size the terminal will default to the 0,0 setting of the T and U switches.

Between 4 and 10 additional header and framing characters will be added to the output buffers depending on other configuration operations selected. Note that if the output buffer is inadvertently set larger than one half of the input buffer size the terminal will default to the 0,0 setting of the T and U switches.

**Space Compression.** The terminal can be configured to compress multiple space characters within a text block into a single space.

This can reduce the time needed to transmit a given block of data.

Example:

### Initial Text

AJAX Corp.  
110 N. Sea Road  
New York, NY 11011

**Uncompressed (59 bytes)** Δ = space

ΔΔAJAXΔΔ CorpΔΔΔΔΔΔ 110ΔN.Δ SeaΔ RoadΔΔΔ  
ΔΔNewΔYork,ΔNYΔΔΔΔΔ11011

**Compressed (45 bytes)** Δ = space

ΔAJAXΔ CorpΔ110Δ N.ΔSeaΔ RoadΔ NewΔ York,Δ NYΔ  
11011

Space Compression is selected by opening the S switch on the Keyboard Interface PCA.

<b>S</b>	
0 (closed)	= No effect
1 (open)	= Space Compression

**Synch Characters.** In asynchronous configurations Opening Switch V on the Keyboard Interface PCA causes SYN characters to be inserted at the beginning of each transmission and at 1 second intervals until the end of the

transmission. This allows the use of a single generalized data communication driver for both synchronous and asynchronous operation.

**TRANSPARENCY MODE (BINARY OPERATION).** Transparency mode allows you to send and receive 8 bit binary data. This allows the sending of data bit patterns that might otherwise be interpreted as control characters.

This mode is controlled with the following character sequences:

DLE STX Starts transparency.

DLE ETX Ends transparency.  
or  
DLE ETB

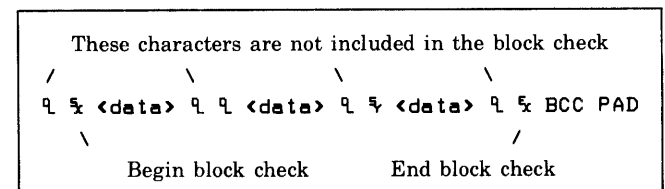
DLE DLE Allows one DLE character to be sent. Note that this will vary with the parity used.

DLE SYN Allows one SYN character to be sent (for synchronous operation). Not included in text or BCC.

DLE ENQ Aborts current transmission. A BCC character is not expected.

Once in transparency mode, in order to send control characters and have them interpreted as control characters rather than binary data, the control character must be preceded with a single DLE character. Single DLE characters are seen as the beginning of control sequences rather than data. The first DLE character of the above sequences is never included in the block check.

### Example:



The terminal will always accept transparent data. Escape sequences can be used to cause the terminal to send transparent data. The Z strap can also be used to cause the terminal to send transparent data at all times.

Note that whenever control character sequences are used in transparent mode they must have proper parity or they will not be interpreted as control characters.

## Block Protocol Control Sequences

Block protocols require specific control sequences to acknowledge text block transfers, terminate text transfers or to inform the sender or receiver of status changes. These sequences consist of one or more data link control characters. A list of these control characters are given in table 5-10. A summary of the uses of these characters is given in table 5-11.

Figure 5-12 illustrates the operation of the various control characters used in the block protocol.

**KEYBOARD INTERFACE STRAPS.** The Keyboard Interface straps permit you to select various communication features. A list of the selectable features is given in table 5-12.

**KEYBOARD DATA COMM SWITCHES.** The keyboard data comm switches are shown in figure 5-13.

Table 5-10. Block Protocol Control Characters

CONTROL CHARACTER	ASCII CODE (OCTAL)	DESCRIPTION
Data link control characters. These characters are used to frame messages and acknowledgements for both transmitted and received text blocks. They are also used to control all communications in an orderly fashion.		
DLE	020	Data Link Escape. This is the first character in two byte control characters. It is used to indicate that the second character is to be interpreted as a control rather than a data character. The DLE character has no meaning when used alone.
ACK0 (DLE 0)	020 060	Acknowledge 0. These control characters are sent by the terminal after being selected to tell the computer that the terminal is ready to accept a text block. They are also sent by the receiving station (computer or terminal) after even text blocks (2, 4, etc.) to tell the sending station (terminal or computer) that the block was received properly (see ACK 1). The alternating ACK0/ACK1 sequence is initialized to ACK0 following a select sequence or to ACK1 after a poll sequence.
ACK1 (DLE 1)	020 061	Acknowledge 1. These control characters are sent by the receiving station (computer or terminal) after odd text blocks (1, 3, 5, etc.) to tell the sending station (terminal or computer) that the block was received properly (see ACK 0).
WACK (DLE ;)	020 073	Wait Before Transmit. These characters are sent by the receiving station to indicate that the last block was properly received but that the receiving station requests that the sender wait before sending the next block. The sending station should then send an ENQ. The receiving station will then return an ACK0/1 if it is ready to receive data or a WACK in order to continue waiting.
NAK	025	Negative Acknowledge. This character is returned in response to a text block to indicate that the block was rejected because of a bad block check or because of improper framing characters. When received by the terminal after it has sent a text block, the terminal will retransmit the block.
ENQ	005	Enquiry. This character is always used as to terminate a POLL or SELECT sequence. It is also used by the sending station to request a retransmission of the acknowledgement for the previous text block. When used as a block terminator, ENQ indicates that the computer has aborted the block (forward abort or TTD). The terminal will respond with a NAK to acknowledge the abort command.
STX	002	Start of Text. This character must be the first character in every text block. It tells the receiving station to begin accumulating a block check character. The STX character is not included in the block check.

Table 5-10. Block Protocol Control Characters (Continued)

CONTROL CHARACTER	ASCII CODE (OCTAL)	DESCRIPTION
ETB	027	End of Transmission Block. This character is used to tell the receiving station to stop accumulating a block check character and that the next character transmitted will be the block check character. When used the ETB character must always follow the last character in the text block. The ETB character is included in the block check character accumulation. (See the ETX character.)
ETX	003	End of Text. This character must be the last character of the last (or only) text block in a message. It tells the receiving station to stop accumulating a block check character. The ETX character is included in the block check character. (See the ETB character.)
EOT	004	End of Transmission. When this character is sent or received by the terminal, it causes the terminal to switch to Control Mode. It is sent by the terminal when it detects a data overflow condition while receiving text (buffer full), after sending the last text block of a message to the computer, or in response to a POLL sequence when it has no data to send. An EOT is sent by the computer following the last text block in a message to indicate that the computer has no more data to send or when the computer wants to abort the communication sequence.
RVI (DLE <)	020 074	Reverse Interrupt. This character is sent by the computer to acknowledge that the last text block was properly received (see ACK0 and ACK1) and at the same time to request that the terminal stop sending as soon as possible. When this character is received by the terminal, the terminal will immediately send an EOT to the computer. The terminal sends the RVI sequence when in Text-In mode and the <b>STOP</b> key is held down. This indicates that the terminal properly received the last text block but requests the computer to stop sending text as soon as possible.
TTD (STX ENQ)	002 005	Temporary Text Delay. This character is sent to inform the receiving station that the sender is temporarily out of text but that there is more to follow. The receiver must respond with a NAK for the sender to continue. This sequence will continue until the sender has more data to send. This sequence might be used in transferring data to and from the CTU's.
Transmission control characters. These characters are used to initialize, synchronize, and terminate data without affecting data integrity.		
SYN	026	Synchronous Idle. This character is used only in synchronous communications to establish and maintain character timing between sending and receiving stations. At the beginning of each transmission a minimum of three SYN characters are required. During transmission two pair of SYN characters are inserted at one second intervals.
PAD	377	PAD. This character is used to ensure that the last character of every transmission has time to be properly received before the receiving station begins transmitting. All transmissions must be terminated with a trailing PAD. In addition a trailing PAD must always be used after an EOT when it is used in a POLL or SELECT sequence. (Note that accuracy of the PAD character cannot be guaranteed.) If the trailing PAD character is not used, the communications interface will wait 40 msec before continuing to allow all data to be properly received. This may significantly slow communications.
DLE EOT	020 004	Disconnect. When this sequence is received by the terminal instead of a normal response or text block, the terminal will attempt to disconnect the modem attached to the communication line. This sequence is only used on switched lines.

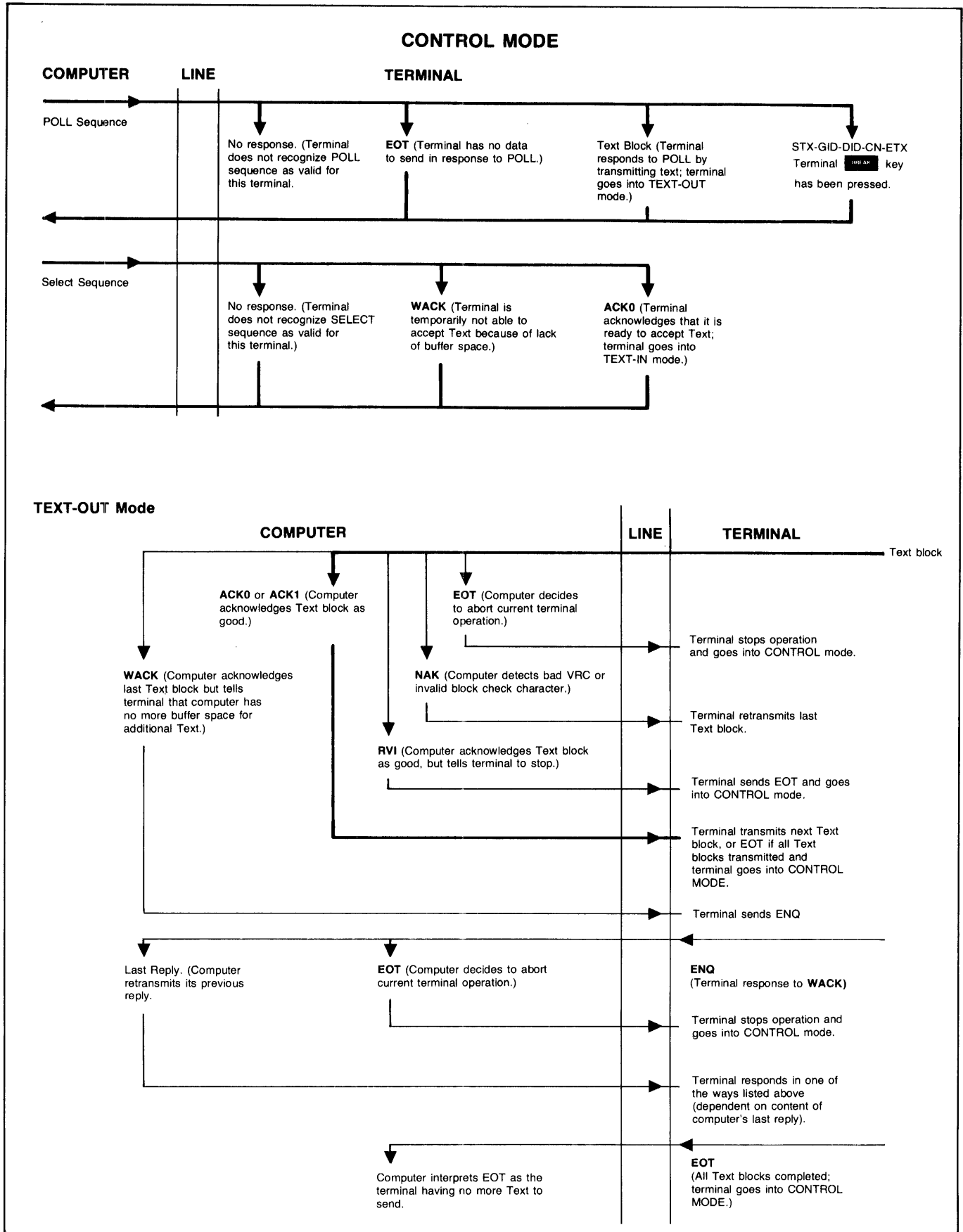


Figure 5-12. Operation of Block Protocol Control Characters (Sheet 1 of 2)

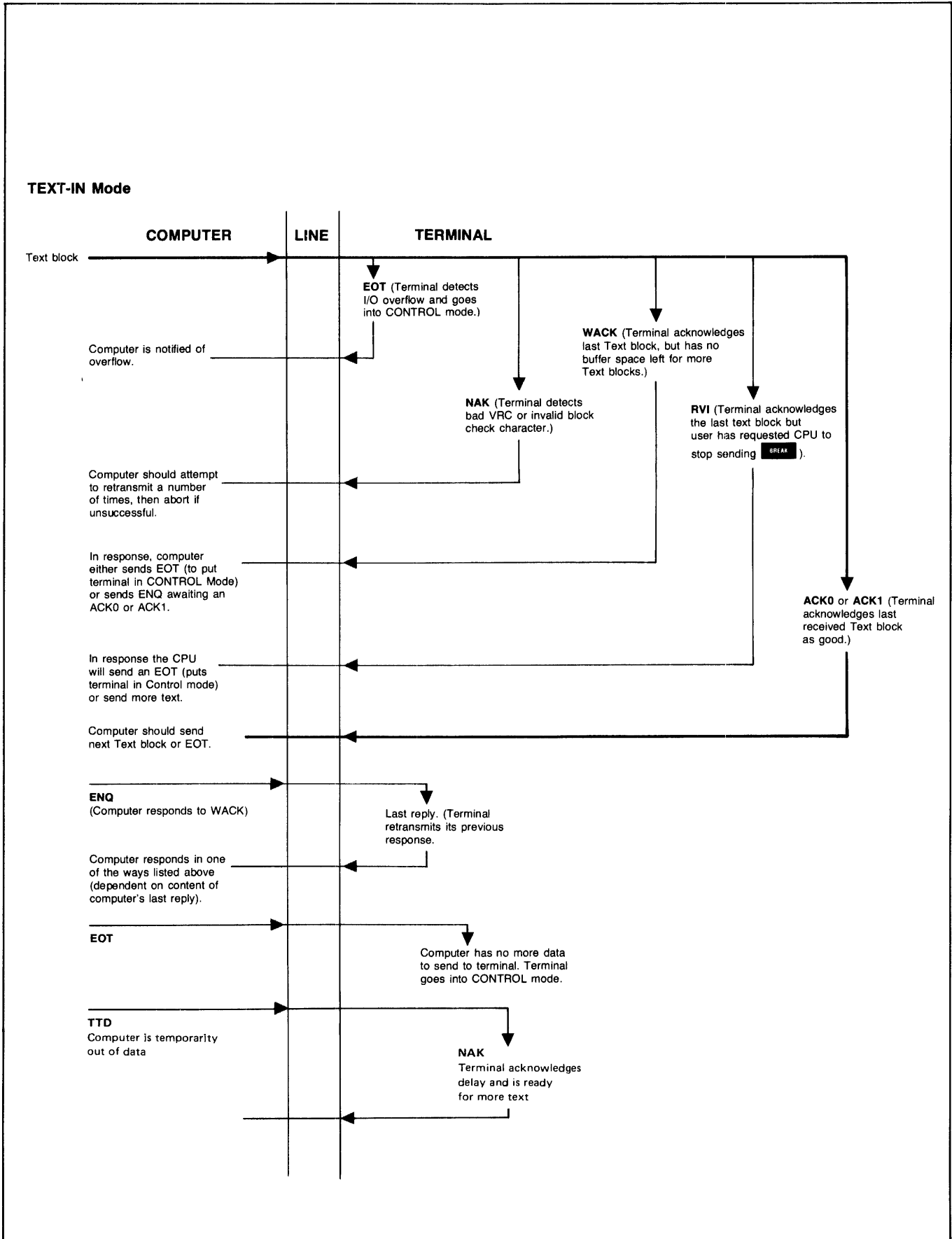


Figure 5-12. Operation of Block Protocol Control Characters (Sheet 2 of 2)

Table 5-11. Summary of Block Protocol Control Characters

	CONTROL		TEXT-IN		TEXT-OUT	
	POLL RESPONSE	SELECT RESPONSE	RECEIVED	TRANSMITTED	RECEIVED	TRANSMITTED
STX-"TEXT"-ETB-ETX	Positive response to POLL.		Sent by CPU as a response to an ACK received from terminal.			Sent by terminal as a response to an ACK received from CPU.
"EOT"	Negative response to POLL. Terminal has no TEXT to xmit.		CPU has no more TEXT to xmit to terminal.	Terminal has detected data overflow.	CPU has decided to abort terminal xmission.	Term has no more TEXT to send to CPU or has just received an "RVI".
"ENQ"			CPU requests terminal send last TEXT acknowledgement.			Term requests CPU retransmit last acknowledgement to TEXT.
"RVI"				Terminal acknowledges last text block and requests the CPU to stop sending ( <b>BREAK</b> ).	CPU acknowledges last TEXT block & requests term send "EOT".	
"ACK0/ACK1"		Terminal tells CPU that it is ready to accept TEXT (ACK0).		Terminal tells CPU that last TEXT block was received OK.	CPU tells term that last TEXT that term sent was OK.	
"WACK"		Term is temporarily busy (term has no available buffers). Cannot accept TEXT.		Term acknowledges last TEXT block received.OK but now term has no more buffers & cannot accept more TEXT.	CPU acknowledges last TEXT block sent by term but tell term to wait because CPU does not have any more buffs.	
"NAK"				Term detected error in last TEXT block CPU sent. Invalid VRC/BCC or frame chars.	CPU detected error in last TEXT block term sent. Invalid VRC/BCC or frame chars.	
STX-GID-DID-CN-ETX	<b>BREAK</b> has been pressed. Any data that is waiting to be sent to the CPU is lost.					
STX-ENQ ("TTD")			CPU is temporarily out of data. The terminal must respond with a NAK.			Terminal is temporarily out of data.

Table 5-12. Keyboard Interface Straps for Block Operation Using 13260C or 13260D Communications Accessory

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
A	Function Key Transmission	The escape code sequence generated by the major function keys (such as, ROLL UP, ROLL DOWN, etc.) are executed locally, but not transmitted to the computer.	(Same as switch closed.)
B	Space Overwrite (SPOW) Latch Enable	Spaces typed will overwrite existing characters.	When the SPOW latch is off, overwriting occurs as normal. When the SPOW latch is on, spaces cause the cursor to forward but not overwrite any existing characters. The SPOW latch is turned on by a Carriage Return, and turned off by a Line Feed, Home, or Tab.
C	Cursor End-of-Line Wrap Around	At the end of each line, a local Carriage Return and Line Feed are generated; the cursor moves to the beginning of the next line.	A Carriage Return and Line Feed are not generated at the end of each line. The cursor remains in and overwrites column 80.
D	Line/Page	The terminal is set to transfer a line at a time from display memory, an unprotected field in format mode, or a record from the tape cartridge.	Transfers the entire contents of display memory (a "page"), all unprotected fields in format mode, or a file from the tape cartridge.
E	Paper Tape Mode	When the READ key is pressed with the AUTO LF down, each tape record begins with an LF if the AUTO LF key is down and is ended with a CR.	Each tape record is terminated by CR(LF).
F	(Not Used)		
G	Block Transfer Handshake	No effect.	No effect.
H	Inhibit DC2	No effect.	No effect.
J	Auto Terminate	No effect.	When the ENTER key is pressed a non-displaying terminator is placed after cursor position.
K	Clear Terminator	No effect.	Clear terminator caused by strapping option J above.
L	Self Test Inhibit	No effect.	Self Test function is inhibited. Pressing TEST key or issuing ESC z has no effect. TAPE TEST and DATA COMM SELF TEST functions are not affected.
M	Reverse Sense of INSERT and DELETE CHAR with Wrap.	No effect.	Reverses control function of INSERT CHAR and DELETE CHAR keys (i.e., when key is pressed, line wrap around is in effect without having to press CNTL key. When either key is pressed while pressing CNTL, normal insert character and delete character functions are in effect.)
N	Escape Code Transfer To Printer	No effect.	Escape codes relating to the display (e.g., display enhancements, alternate character sets, format mode, fields, etc.) are sent to printer if it is selected as a destination device.
P,Q	Compatibility Mode	These switches set the terminal to be compatible with Tektronix control commands when initialized (power on or full reset). Refer to Section III for additional information on Compatibility Mode.  P-closed, Q-closed    Normal operation P-closed, Q-open    Unscaled Compatibility Mode P-open, Q-closed    Scaled Compatibility Mode P-open, Q-open    Normal operation	
R <sup>1</sup>	Data Set Ready	No effect.	Provides an internal Data Set Ready (CC) signal to the terminal. (Used in applications with the HP 30037A Asynchronous Repeater, and the Group Poll feature.)

Table 5-12. Keyboard Interface Straps for Block Operation Using 13260C or 13260D Communications Accessory (Continued)

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
S <sup>1</sup>	Space Compression	Space characters are sent normally.	Space characters are compressed.
T,U	Output Block Size	<b>T</b> <b>U</b>	<b>BLOCK SIZE (BYTES)</b>
		C      C O      C C      O O      O	1/2 Data Comm Buffer (refer to switches J16, J17 on multipoint PCA). 250 max 500 max 1000 max
C = closed, O = open			
V <sup>1</sup>	Synch Characters	Asynchronous operation without SYN characters.	SYN characters are inserted during Asynchronous operation.
W	Data Comm Self Test	Enables DATA COMM SELF TEST from either the keyboard or escape sequence.	Disables DATA COMM SELF TEST. If self test is attempted (by either the keyboard or escape sequence), the test will be aborted and ERROR 0 will appear on the display.
X	Data Speed Select	Holds data speed signal low (CH = off).	Sets data speed signal high (CH = on).
Y	Transmit Indicator	Lights TRANSMIT indicator on keyboard when terminal is communicating with the computer.	Lights TRANSMIT indicator on keyboard when Data Set Ready (CC) is on, and it goes out when CC goes off.
Z <sup>1</sup>	Transparency	No effect.	Causes all data sent from the terminal to be transparent.

<sup>1</sup> For backward compatibility strapping, refer to tables C-4 and C-5 in Appendix C.

**Parity Switch.** This switch is used only with ASCII code and is ignored when EBCDIC code is used. The settings are as follows:

ODD = Odd parity required.  
EVEN = Even parity required.  
NONE = "0"s required.

**Speed Switches.** Block protocols can be used at speeds from 300 to 9600 baud. Speed selections outside this range are ignored. Keyboard speed settings can only be made for asynchronous operation. Synchronous communications speeds are selected on the Synchronous Communications Interface or are supplied by the modem.

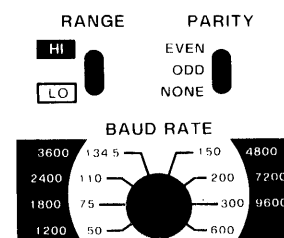


Figure 5-13. Block Keyboard Data Communication Switches



## Multipoint Communications

The terminal is capable of operating in a polled multipoint environment. This means that one or more terminals can share the same communication line. The terminal can be used in networks using asynchronous or synchronous communications. Operation is similar to IBM Bisynchronous communications. Multipoint operation requires the following:

- All communications follow a strict protocol.
- Each terminal must have an address that is unique within its communication line.
- Data is transmitted in blocks.
- All data transfers are initiated by the computer.
- All terminals on the same communication line must use the same code (ASCII/EBCDIC) and parity.

**MULTIPOINT PROTOCOL.** The terminal uses a multipoint protocol that is similar to IBM Bisync. The protocol is made up of sequences of one or two control characters. Table 5-10 contains a list of the control characters used along with a short description.

**BREAK Key Operations.** The **BREAK** key allows the user to tell the CPU or application program that he wants to abort the current operation. (Long text transfers from the CPU can be stopped by holding down **BREAK**.) When the terminal is in Text-In mode and the **BREAK** key is held down, an RVI (DLE <) is sent to the CPU instead of an ACK0 or ACK1 after the current text block is received. The CPU software must then respond to the RVI in an appropriate manner.

If the terminal is in the Text-Out or Control mode and the **BREAK** key is pressed, the terminal will clear all data in the data comm output buffers (the data is lost) and then it will send **5GID DID 5x** in response to the next poll from the CPU. The CPU software must then respond in an appropriate manner.

**PA and PF Key Functions.** Multipoint operation allows you to enter an escape sequence to select operation comparable to the CLEAR, PA, and PF keys on the IBM 3270 terminal. The escape sequence can be entered from the keyboard or a tape unit or datacomm. The PA and PF functions allow you to send a single character to the computer or preface the data with a special character. Then depending on how the computer is programmed, it can use this character to branch to various data handling routines.

The escape sequence is as follows:

**Esc & g ### {F or A}**

where: ### is the octal code of the character to be transmitted. It can be made up of 0 to 3 octal digits. This character must have an octal code in the range 040 to 176. Note that the DELETE character (octal 177) cannot be used. If no character is defined, the default value returned will be an octal 047 (27 hex) if Extended Text Mode is selected.

The softkeys ( **f1** - **f8** ) can be loaded with the escape sequences. Refer to the description of "soft keys" elsewhere in this manual. Note that the soft keys are cleared by a full reset.

**PA Operation.** If the last character of the escape sequence is an A, it will cause the single character indicated by the octal code to be sent to the computer the next time the terminal is polled. This is done by creating a new text block in the output buffer.

**Example:** **Esc & g 122A** (Note: R = 122 octal)

This would cause the following text block to be sent to the computer:

**<5><GID><DID><R><5><BCC><PAD>**

**PF Operation.** If the last character in the escape sequence is an F, it will cause the defined character together with the data currently displayed on the terminal screen to be sent to the computer the next time the terminal is polled.

**Example:** **Esc & g 120F** (Note: P = 120 octal)

This would cause the following text block to be sent to the computer.

**<5><GID><DID><P><screen data><5><BCC><PAD>**

Note that if the screen data exceeded the terminal block size the transmission would use the normal multiblock format.

When in Extended Text mode the PF escape sequence will cause the character coded in the sequence to be sent as the AID character. (Refer to Extended Text Feature.)

**Example:** **Esc & G 120F** (Note: P = 120 Octal)

This would result in the following text block:

**<5><GID><DID><P><CCA><CCA><screen text><5><BCC><PAD>**

**Typical Applications of the PA and PF Functions.** Use of the PA and PF functions allow the computer to use a general poll to find out more than just which terminals have data ready to send. If the PA and PF functions are used the character returned from each terminal can be used to determine whether the terminal has data, no data, a large amount of data, or high priority data. For example the terminal's programmable function keys (f1—f8) can be programmed with the following sequences:

F1 = **Esc & g 110F** (H = high priority, text follows)  
 F2 = **Esc & g 114A** (L = one line of text ready)  
 F3 = **Esc & g 116A** (N = no data ready)  
 F4 = **Esc & g 120A** (P = full page of text is ready)

The computer can then respond by polling the individual terminals in a logical order after allocating the necessary resources required for each transfer.

**TERMINAL ADDRESSES.** Each terminal on a communications line must have an address that is unique on that line. (The same address can be used on a different line). An address is made up of a one character group ID

and a one character device ID. This address is set on the data comm interface during installation. The characters that can be used are @, A through Z, and SPACE. This allows for 28 groups of up to 28 terminals each.

Table 5-13. Terminal Address Characters

GROUP OR DEVICE NUMBER	COLUMN 1 USED FOR: *DEVICE ID *GROUP ID FOR POLL *ID RETURN ADDRESS			COLUMN 2 USED FOR: *GROUP ID FOR SELECT		
	ASCII I/O CHAR	ASCII HEX	ASCII OCTAL	ASCII I/O CHAR	ASCII HEX	ASCII OCTAL
0	@	40	100	'	60	140
1	A	41	101	a	61	141
2	B	42	102	b	62	142
3	C	43	103	c	63	143
4	D	44	104	d	64	144
5	E	45	105	e	65	145
6	F	46	106	f	66	146
7	G	47	107	g	67	147
8	H	48	110	h	68	150
9	I	49	111	i	69	151
10	J	4A	112	j	6A	152
11	K	4B	113	k	6B	153
12	L	4C	114	l	6C	154
13	M	4D	115	m	6D	155
14	N	4E	116	n	6E	156
15	O	4F	117	o	6F	157
16	P	50	120	p	70	160
17	Q	51	121	q	71	161
18	R	52	122	r	72	162
19	S	53	123	s	73	163
20	T	54	124	t	74	164
21	U	55	125	u	75	165
22	V	56	126	v	76	166
23	W	57	127	w	77	167
24	X	58	130	x	78	170
25	Y	59	131	y	79	171
26	Z	5A	132	z	7A	172
27	SP	20	040	-	20	055

////// Shaded values only are used by IBM 3270 configurations. The SP character only is allowed in Group Polls and the - character only is allowed in Group Selects.

**EXAMPLES:**  
 POLL DEVICE 6 in GROUP 20  
 GROUP ADDR TT  
 DEVICE ADDR FF  
 SELECT DEVICE 6 in GROUP 20  
 GROUP ADDR tt  
 DEVICE ADDR FF

The terminal ID characters are listed in table 5-13. The characters in column 1 are used for group and device IDs in polling sequences and for device IDs in select sequences. Characters in column 2 are used for group IDs in select sequences. The lower case group IDs let the terminal tell a poll sequence from a select sequence. Figure 5-14 gives an example of terminal address assignments.

The first 'group' shown in figure 5-14 contains three terminals. Two of the terminals have the same group ID character. Terminals with the same group ID can be controlled by group function commands sent from the computer. Group functions allow you to address all terminals having the same group ID simultaneously. In this way a single command can be used to send a message to up to 28 terminals. Similarly, all of the terminals in a group can be requested to send data to the computer. The terminals send data according to their position in the group, the terminal closest to the communication line being first.

Note that all terminals in the same group must be connected to the same modem.

Additional information on terminal addresses is given under Polling and Selection. Procedures for installing multipoint networks are given in Section VII, *Installation*.

**Terminal I.D. Number.** Each terminal in a multipoint network must be assigned a unique identification number. The identification number is made up of a Group I.D. number (GID) and a Device I.D. number (DID). The terminal I.D. number is set by switches on the data communications interface printed circuit assemblies (02640-60106 or 02640-60107).

**Device I.D. Number.** The Device I.D. number may be 0 to 27 and is set with switches J14 through J10. Each bit corresponds to a power of 2. If the number is set to a number greater than 27, the terminal will set the number to 27. For example, Device 6 would be set with the following switch configuration:

```

Device ID = 6
J14 (closed)
J13 (closed)
J12 (open)   = 22 = 4
J11 (open)   = 21 = 2
J10 (closed)

```

6

**Group I.D. Number.** The Group I.D. number may be 0 to 27 and is set with switches J04 through J00. Each bit corresponds to a power of 2. In order to use group functions, all terminals in a group must be "daisy chained" together (connected to the same modem if a modem is used).

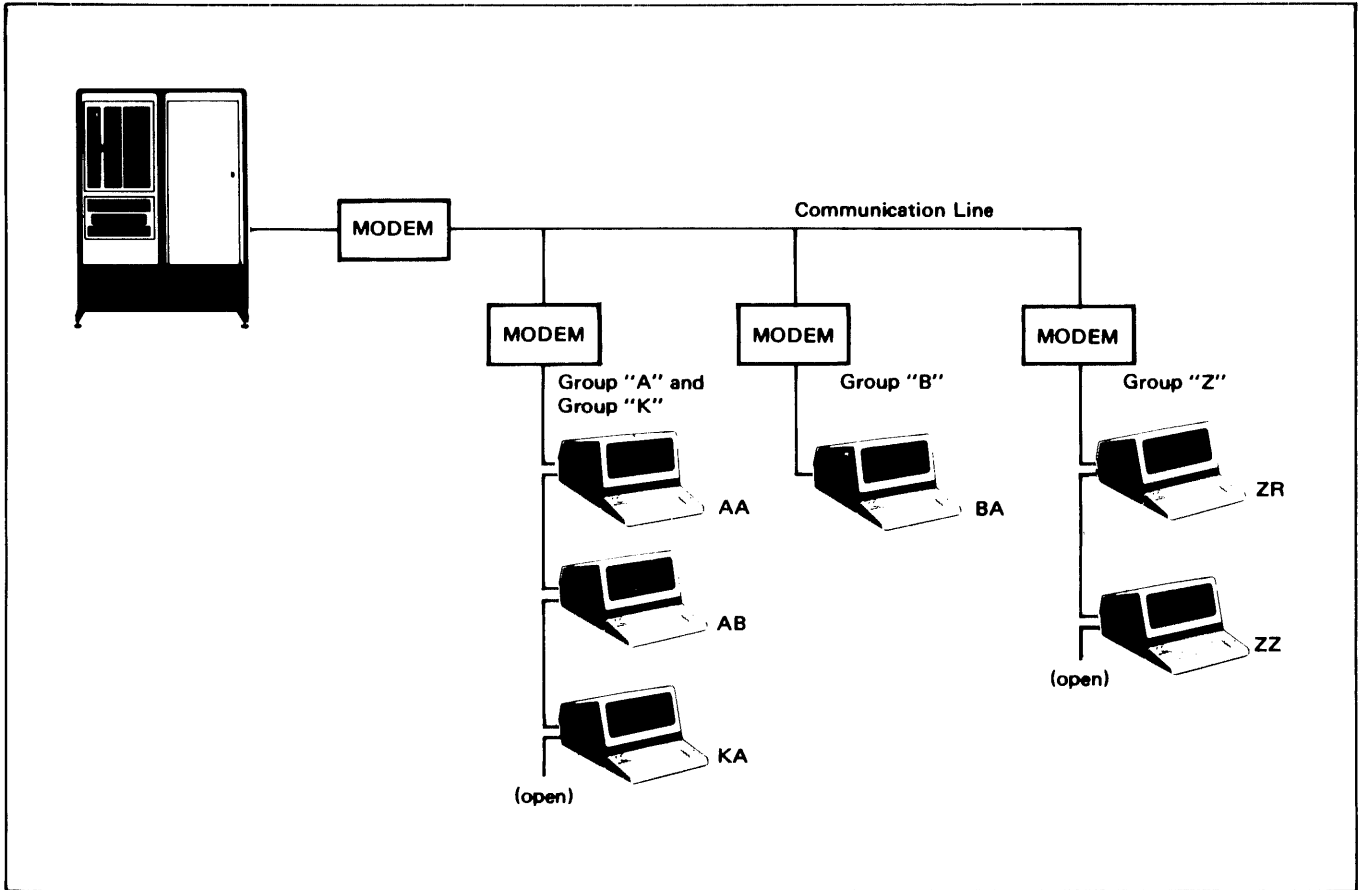


Figure 5-14. Terminal Addressing

If the I.D. is set to a number greater than 27, the terminal will set the I.D. to 27. For example, Group 20 would be set with the following switch configuration:

Group ID = 20  
 J04 (open) = 2<sup>4</sup> = 16  
 J03 (closed)  
 J02 (open) = 2<sup>2</sup> = 4  
 J01 (closed)  
 J00 (closed)

---

20

**INITIATION OF A DATA TRANSFER.** All data transfers are initiated by the computer in one of two ways, Polling or Selection. In both cases, device addresses are used to call a specific terminal or group of terminals.

**CONTROL SEQUENCES.**

**Polling.** The computer requests terminals with data ready for transmission to begin sending by "polling" the terminals. The terminals can then respond in order according to their position on the communication line. Those at the far end of the communication line being held off until all terminals ahead of them on the string have completed their data transfers.

For example, a poll of terminal D in group A would consist of the following character sequences:

Asynchronous

<E><PAD>†<GROUP ID><GROUP ID><DEV ID><DEV ID><E><PAD>

Synchronous

<E\*><E><PAD>†<E\*><GROUP ID><GROUP ID><DEV ID><DEV ID><E><PAD>

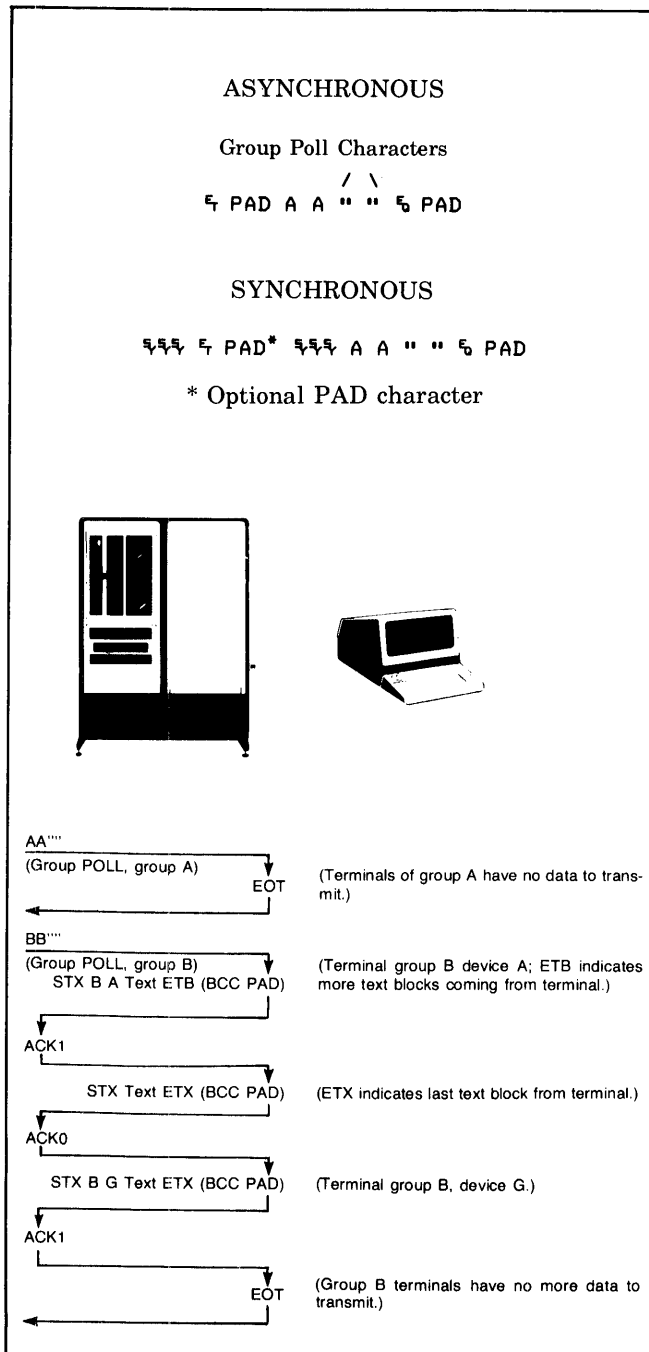
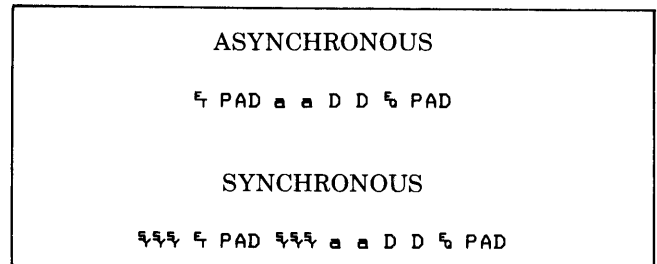
\*3 or more SYN characters.

†These PAD characters must have the correct ASCII parity.

**Group Polling.** In order to reduce the time and programming required to poll each terminal on a communication line you can perform a group poll. This will allow all of the terminals in a group (terminals having the same group ID) with data ready to send, to respond to a single poll sequence. When the last terminal in the group with data to transfer is through sending it will send an EOT to indicate that the group has finished.

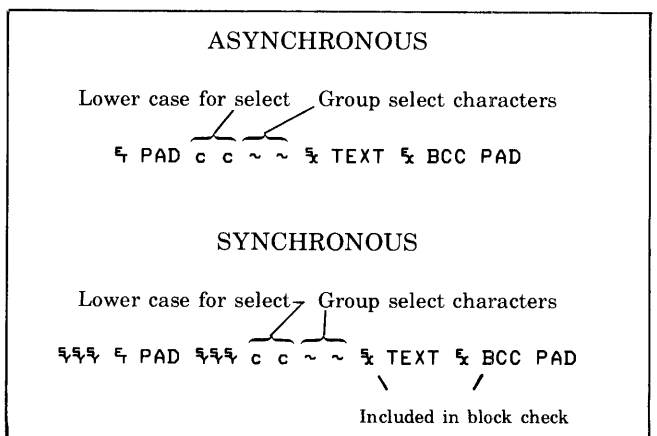
The group poll sequence is similar to the normal poll sequence. The " character (042 octal) is used in place of the device ID characters. For example, to poll all of the terminals in group A you can use the following sequence:

**Selection.** "Selection" occurs when the computer directs a specific terminal or group of terminals to accept a data transmission. The character sequences used in selection are the same as those used in polling. The only difference is that the lower case ID characters are used. This is to tell the terminals that a selection is being sent instead of a poll. For example, to address the same device as in the polling example, the sequence would be as follows:

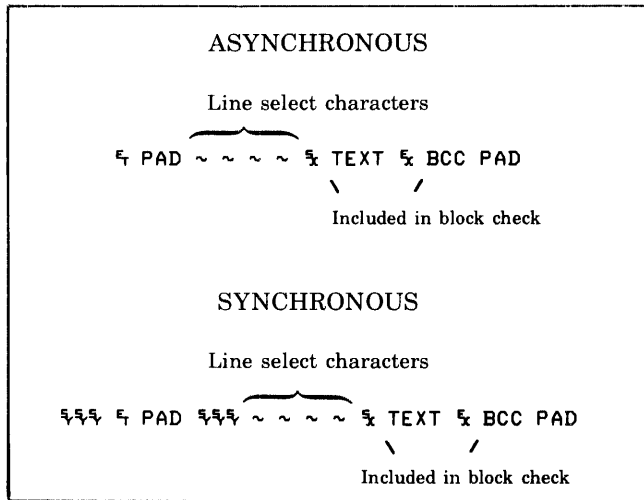


Note that both the group ID and device ID characters are transmitted twice to eliminate line errors during Poll and Select sequences. (These transmissions do not use Block Check characters.) The two group ID characters must be the same and the two device ID characters must be the same for a terminal to accept a poll or select sequence. Then, if the group and device IDs are the same as the terminal's, the terminal will respond with an ACK0. After receiving the first block of data the terminal will respond with an ACK1.

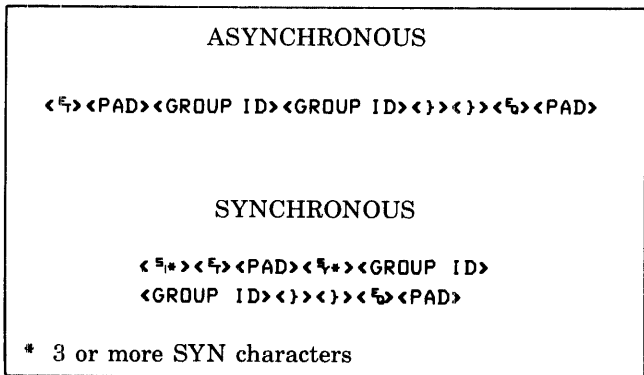
**Group Select.** A "group select" sequence can be used to send data to all of the terminals in a group. The terminals will not send any response to group select. (Since there is no response there is no guarantee that the terminals will receive the text.) The text transmission is appended directly to the end of the group select sequence. The group select is the same as a device select sequence except that the device ID character is replaced with a tilde (~) (octal 176). For example, to send data to all of the terminals in group C the following sequences would be used:



**Line Select.** A "line select" allows you to select all of the terminals on a communication line. This is also known as "Broadcast" mode. Both the group and device id characters are replaced with tildes (~).



**CONFIGURATION STATUS — WHO ARE YOU (WRU).** The Who Are You (WRU) control sequence is a status request from the computer to a terminal group. It is similar to a group poll except that the terminals respond with status information instead of the normal text data. All terminals in the group that are turned on will send in their status. The status sequence is shown below. The right brace character (175 octal) is used in place of the device id. This tells the terminal that a status request is being made.



Three bytes of status information are returned for each responding terminal. Figure 5-15 shows a typical status request and responses from a terminal group.

The status bytes contain terminal hardware and firmware configuration information. The content of each of the status bytes is explained in figure 5-16.

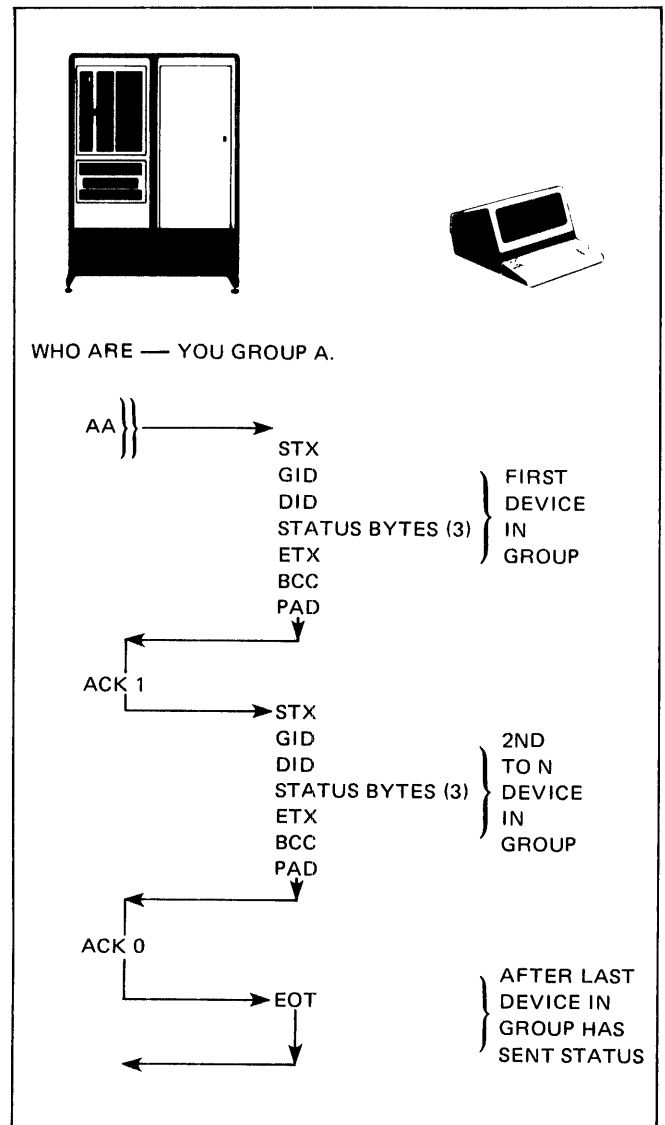


Figure 5-15. Typical Configuration Status Request and Response Sequence

### Configuration Procedure

After you have determined the required multipoint settings for your application follow the flowchart given in figure 5-17.

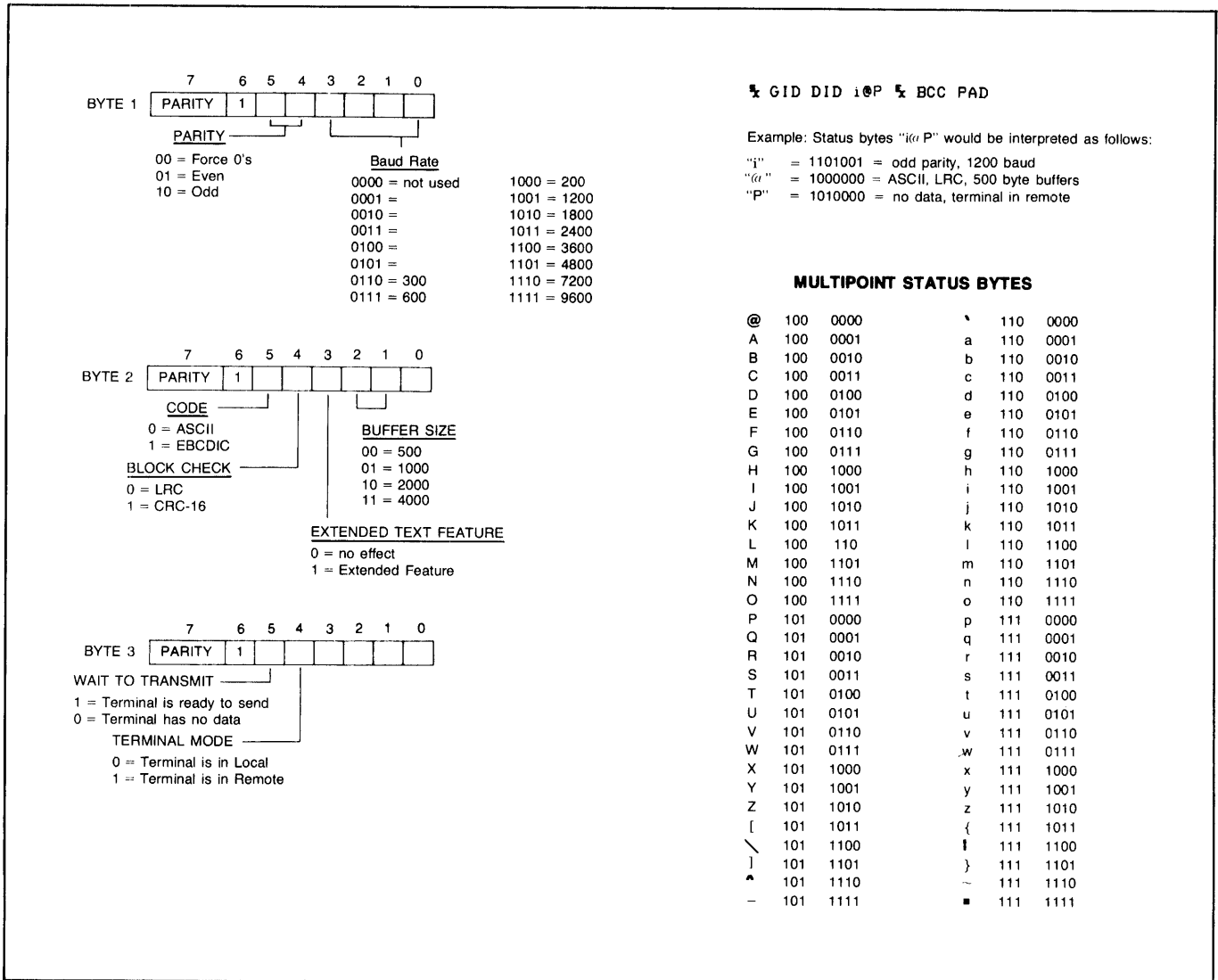


Figure 5-16. Configuration Status Byte Contents

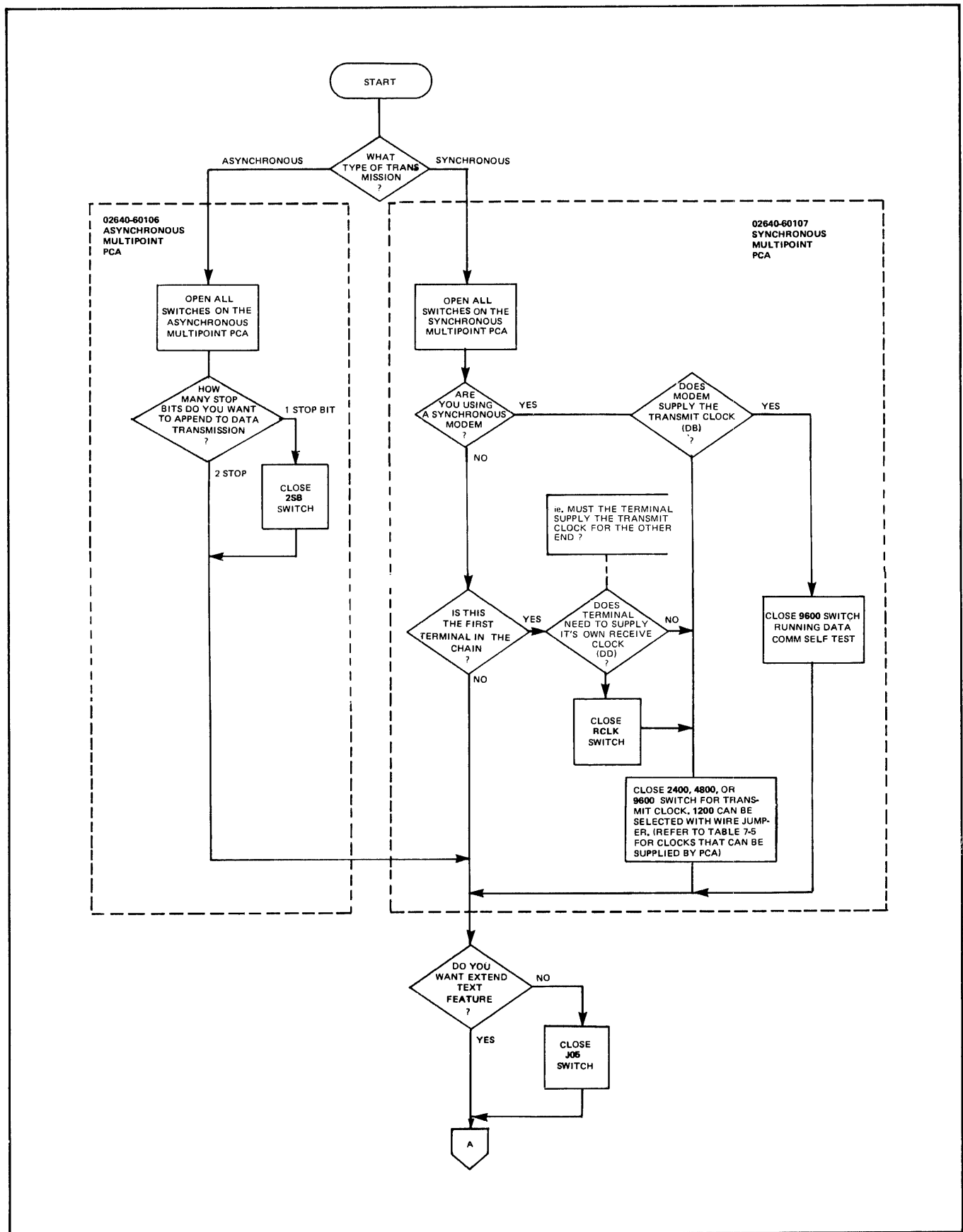


Figure 5-17. Multipoint Data Communications Configuration (Sheet 1 of 4)

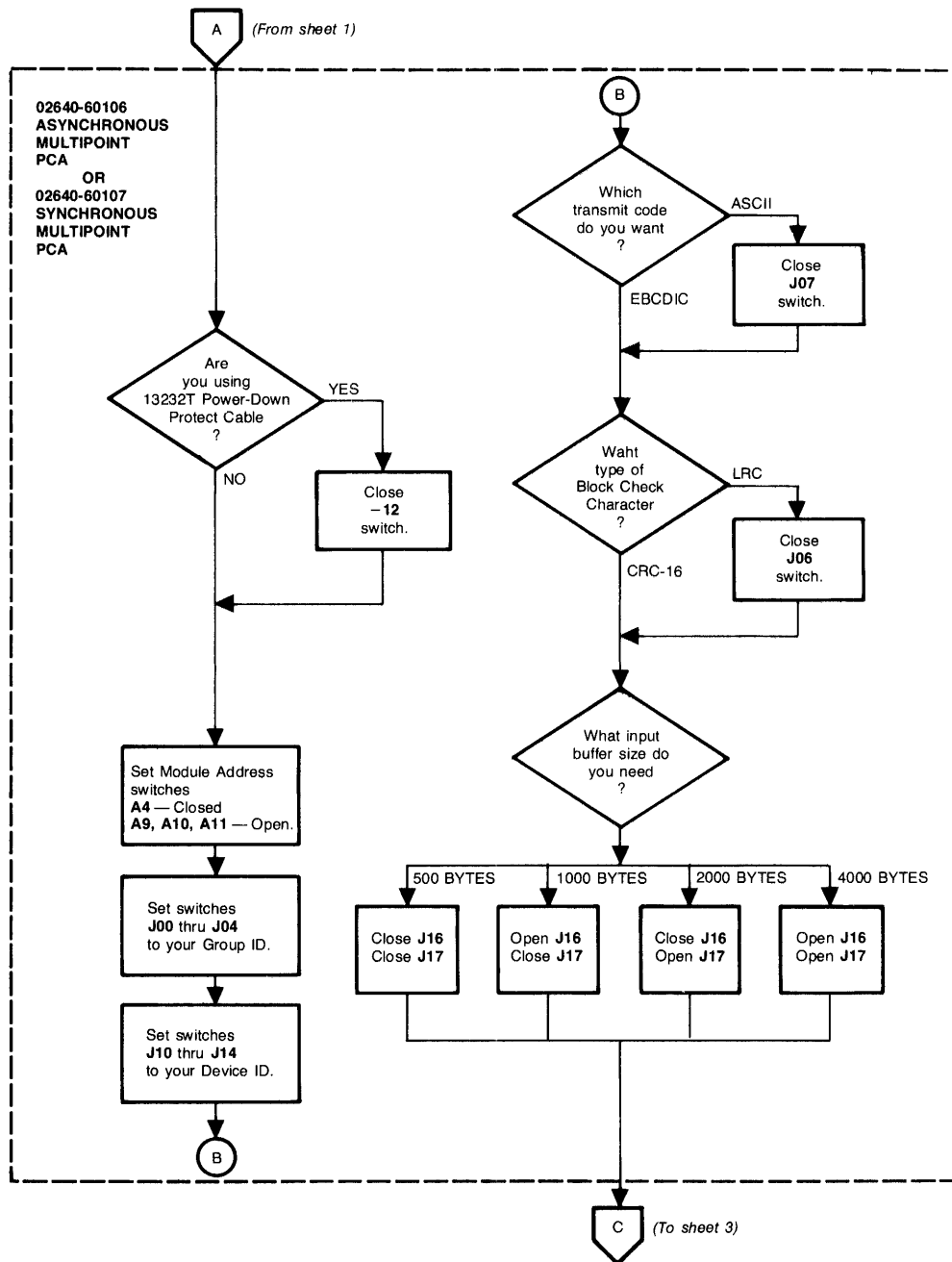


Figure 5-17. Multipoint Data Communications Configuration (Sheet 2 of 4)



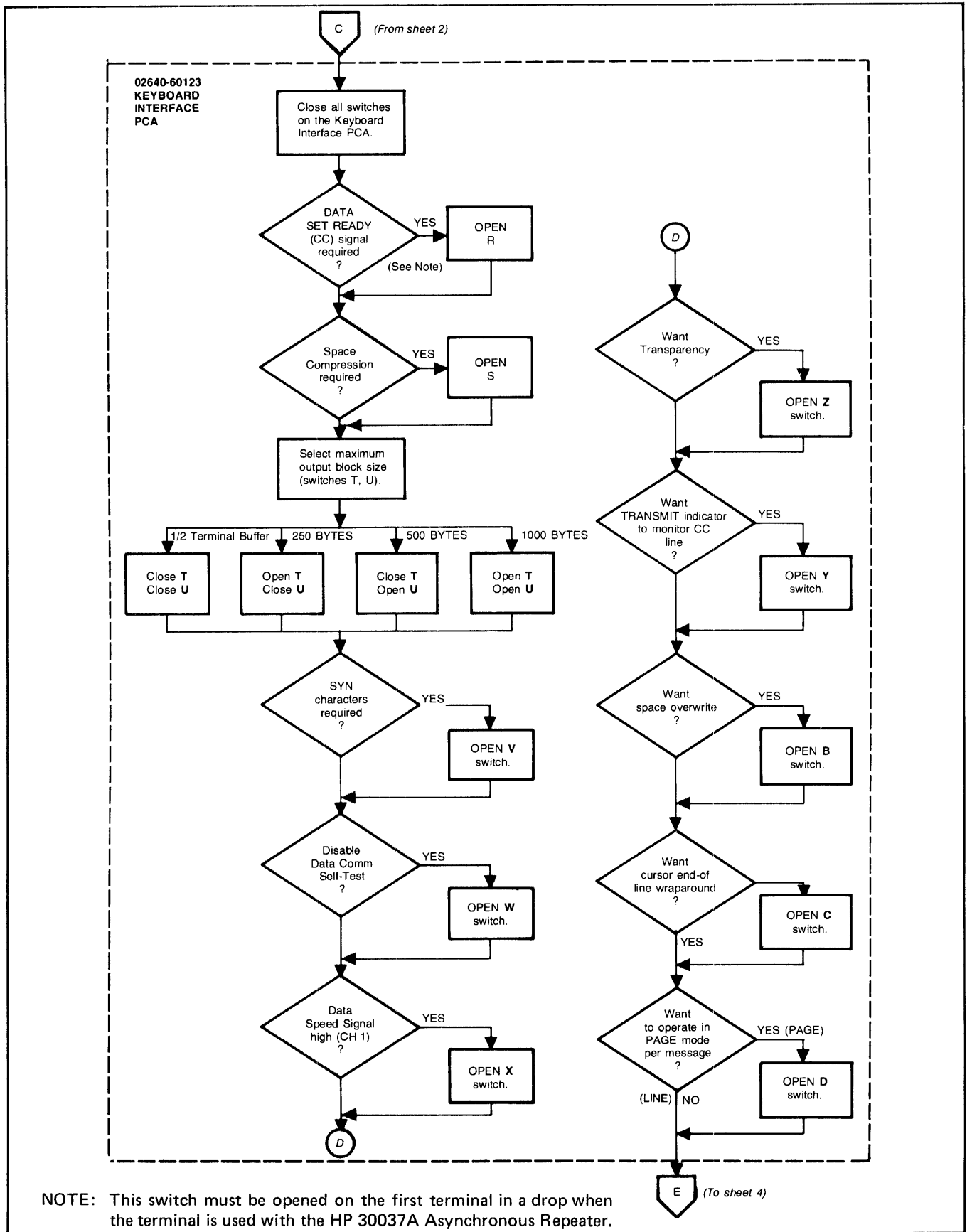


Figure 5-17. Multipoint Data Communications Configuration (Sheet 3 of 4)

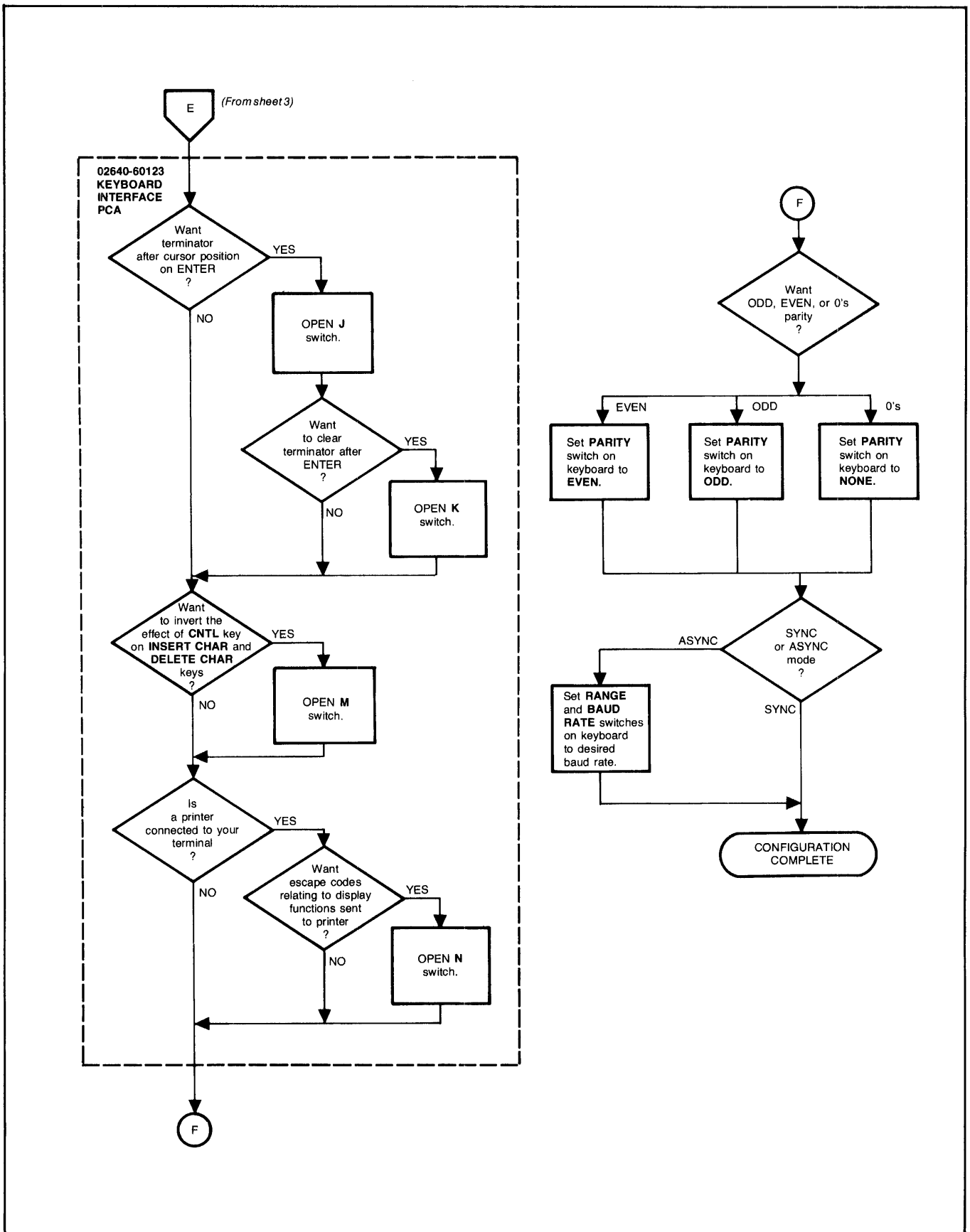


Figure 5-17. Multipoint Data Communications Configuration (Sheet 4 of 4)

## Monitor Mode

Monitor mode is an added multipoint feature available as an option to the 13260C and 13260D interfaces. It allows a terminal to monitor the data transfers between the computer or driver terminal (refer to Driver Mode) and other multipoint terminals on the same communication line. This is a useful technique when developing communications programs or testing multipoint networks.

The monitor must be placed in the line between the computer and the other terminals in order to monitor both sides of the communication exchanges. Figure 5-18 shows a sample communication line using a terminal in monitor mode. (Note that the monitor cannot detect data sent from terminal AA to the computer.)

Note that the monitor will not respond to poll or select sequences addressed to it while in Monitor Mode.

Once the Monitor option has been installed (refer to Section VII, *Installation*, for procedures), Monitor mode is selected by the following:

**Step 1.** **RESET TERMINAL**, **RESET TERMINAL** ( **RE MOTI** key down).

**Step 2.** **CNTRL** **DISPLAY FUNCTIONS** (The **DISPLAY FUNCTIONS** indicator should begin blinking).

Pressing the **DISPLAY FUNCTIONS** key again will turn off the indicator and return the terminal to normal operation.

While in Monitor mode data communications between the computer and "downstream" terminals will be displayed on the monitor. Data from terminals will be framed in left and right arrows (<data>) and will include control and block check characters (see figure 5-19). All untranslatable EBCDIC characters will be displayed as "?" characters.

In group poll operations the last three characters of the poll sequence (second **"**, **ENQ,PAD**) may be distorted due to the response of polled terminals (see figure 5-20). Once a terminal detects the first double quote character (**"**), it begins its response with a transition on the Request to Send Line (**CA**). This causes the monitor to begin watching for data from the terminal instead of the computer. This distortion occurs only within the monitor and does not affect the operation of either the computer or the other terminals.

If the monitor terminal is configured with a communications buffer that is smaller than either the computer or responding terminals, a data overflow may occur. A cancel character (octal 030) will be displayed at the point where the data overflow occurred (see figure 5-21). To prevent data overflow, make sure that the buffer used in the monitor is at least as large as the largest buffer used by any responding terminal. (Refer to the Installation section for buffer configuration information.)

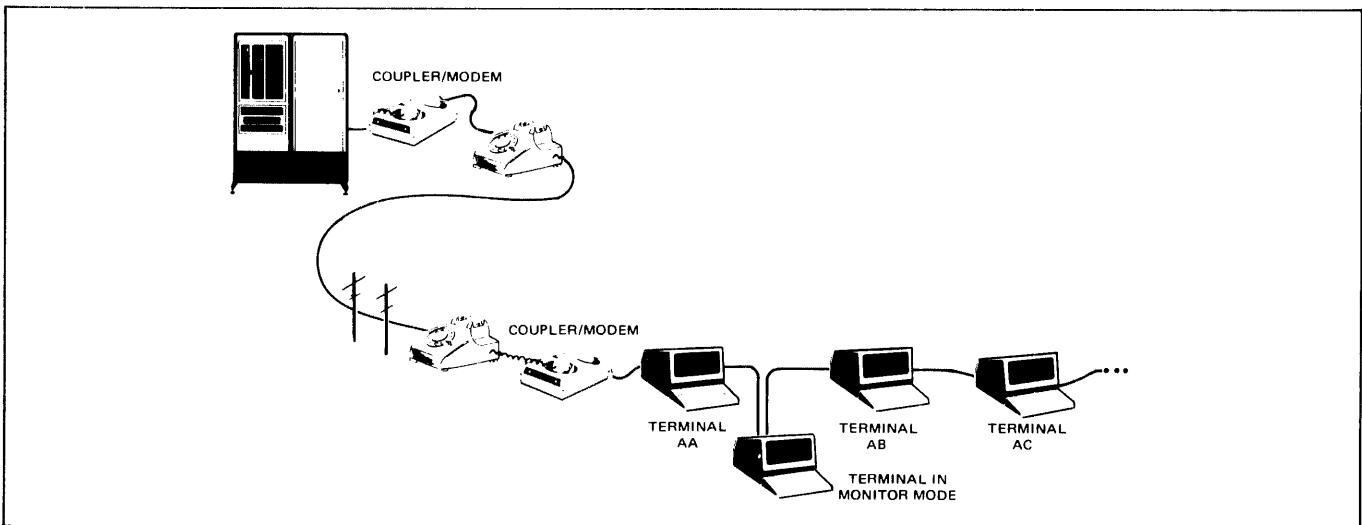


Figure 5-18. Communication Line Using a Monitor



## Driver Mode

Driver Mode is an additional multipoint feature available with the Monitor Mode option to the 13260C and 13260D interfaces. It allows you to use a terminal to control a multipoint communication line. This technique is very useful in developing communication drivers or testing networks without the need of a computer or modem.

Figure 5-22 shows two typical networks using the driver mode option. The network in figure 5-22a is the simplest case. A more useful network is shown in figure 5-22b. Here a multiterminal network is being driven while a terminal in monitor mode is used to display or record all communication transactions. All data from the downstream terminals as well as the driver terminal is displayed.

Once the Monitor/Driver Mode option has been installed (refer to Section VII, *Installation*, for installation procedures), Driver mode is selected as follows:

**Step 1.** RESET TERMINAL , RESET TERMINAL , <ENTER> , CLEAR DISPLAY

**Step 2.** Type DVR-<GID DID><gid DID>

where: GID DID = the group and device IDs to be used in poll sequences.

gid DID = the group and device IDs to be used in select sequences.

### Examples:

DVR-ABaB (uses terminal B in group A for both poll and select)

DVR-A" aB (polls all terminals in group A but selects only terminal B)

**Step 3.** ENTER

**Step 4.** CTRL DISPLAY FUNCTIONS (The DISPLAY FUNCTIONS indicator should blink.)

The Driver will begin sending out the polling sequence at 4 to 5 second intervals using the poll ID characters loaded with the ENTER key. You can also type in text to be sent to the terminal identified for select operations. Block transfers are triggered by the ENTER key.

**Example:** This is a block of text to be sent to a terminal.

This line would be sent to and displayed on the destination terminal just as it was typed. Note that if a monitor terminal were in the network between the driver terminal and the destination terminal it would display all of the framing characters as well as the block check character. Figure 5-23 shows the way this transfer would appear.

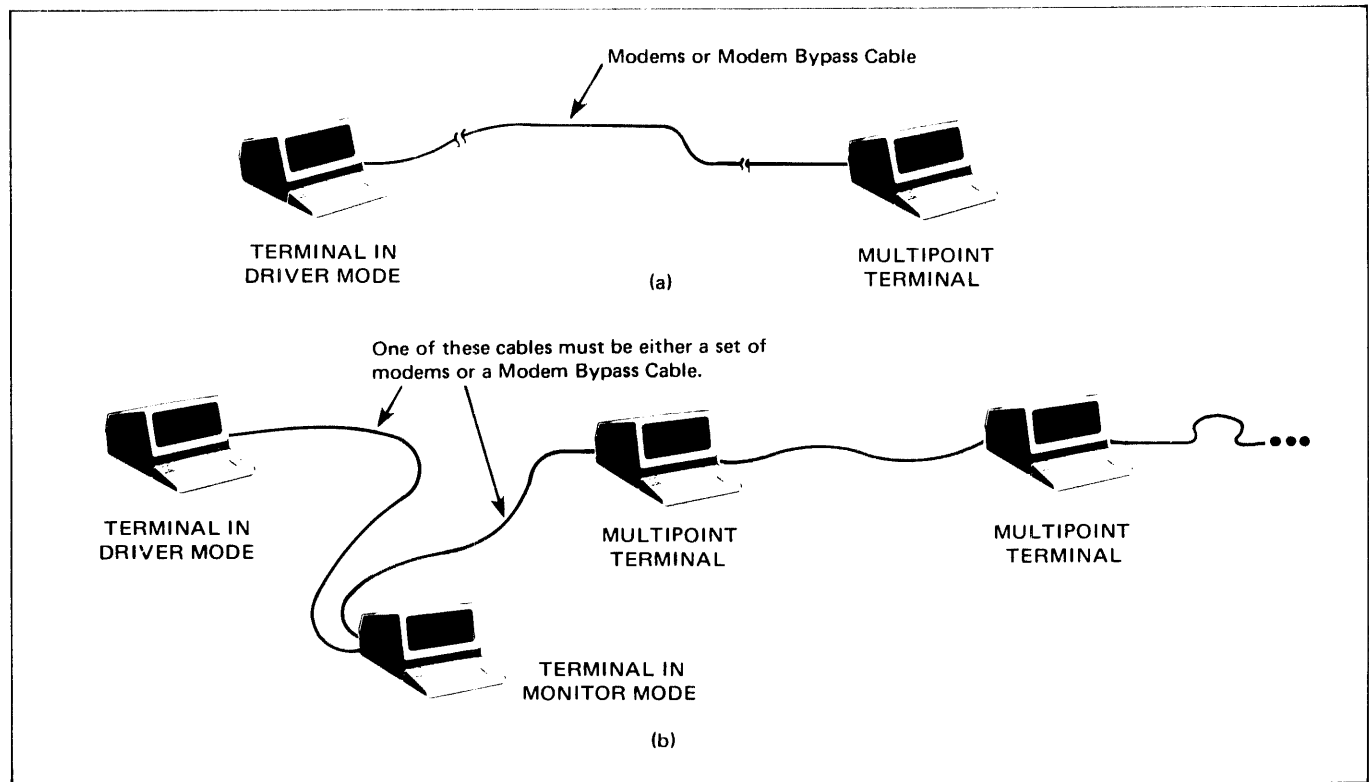


Figure 5-22. Driver Mode Configurations

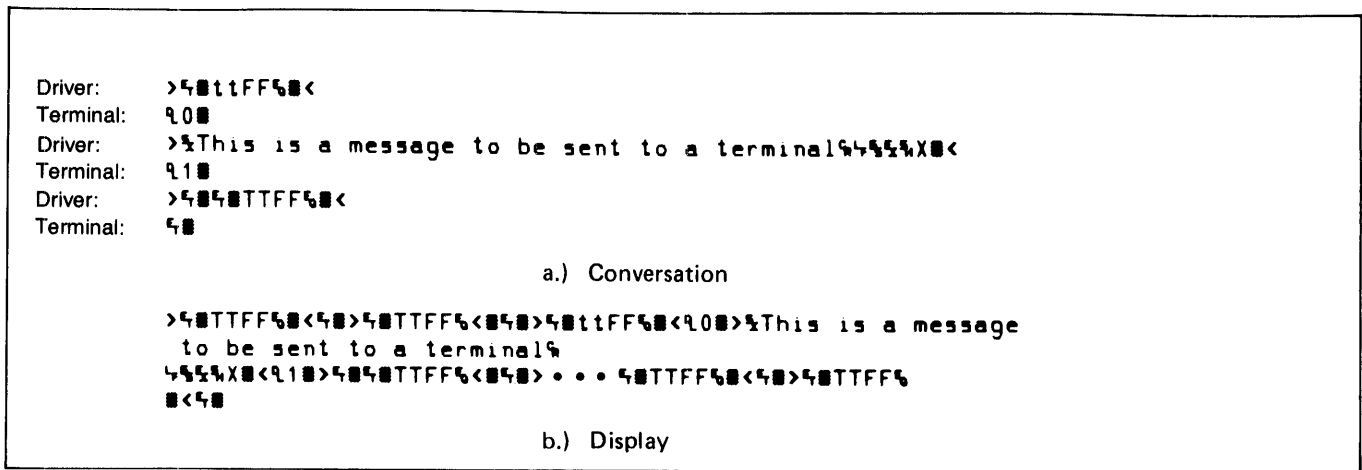


Figure 5-23. Sample Select Sequence Using Driver Mode

Normally all 128 ASCII characters are displayed on the screen of the driver terminal. You can press the DISPLAY FUNCTIONS key (indicator goes out) and still remain in driver mode. This will prevent control characters from being displayed (see figure 5-24).

Data can be transferred from a multipoint terminal to the driver terminal by entering the data and pressing the ENTER key. The terminal will then respond to a poll sequence by sending the data the same as it would in normal multipoint operation (see figure 5-25).

A full reset returns the driver terminal to normal operation.

All multipoint group functions except broadcast can be used in driver mode. Note that you can poll an entire group but can only address one terminal with a select sequence.

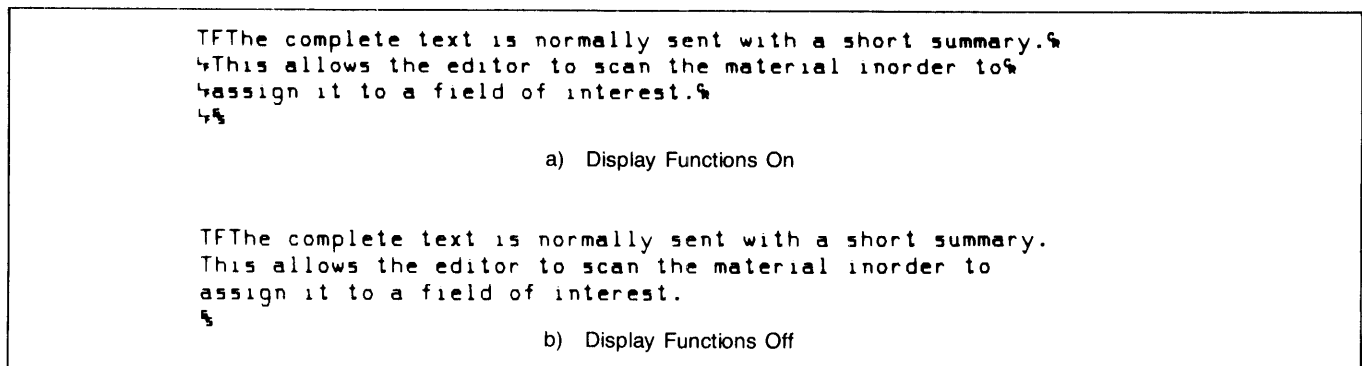


Figure 5-24. Control Character Display On Driver Terminal

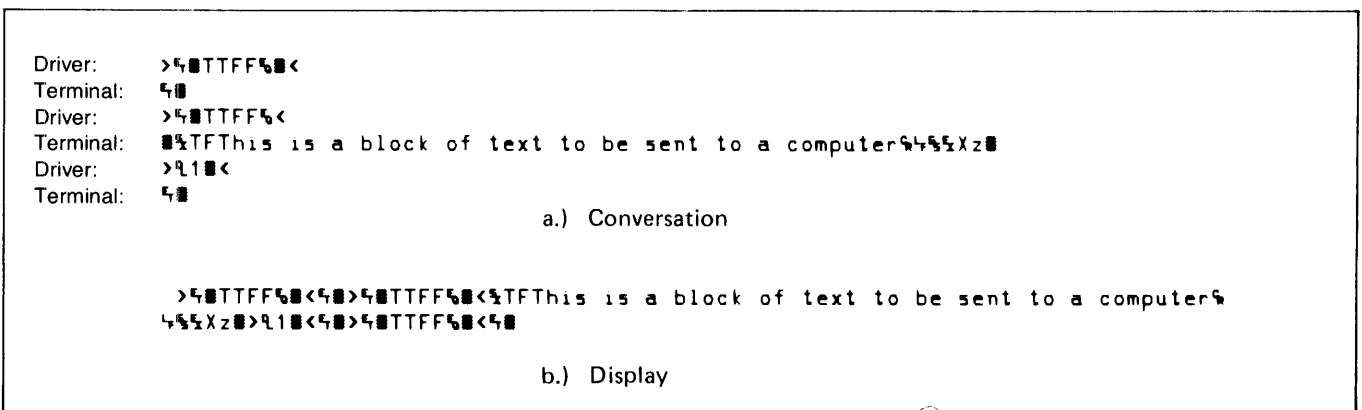


Figure 5-25. Terminal Input

## INTRODUCTION

This section contains information on how to obtain and interpret terminal status information. In addition to normal terminal status, you can also obtain status information on graphics operations and input/output devices used with the terminal.

Status requests are made by sending an escape code sequence to the terminal to select the desired status information. All status requests are treated as block transfers. (Refer to Multicharacter Transfers in Section V). The examples that follow use the DC1 character to trigger the status transfer (Basic Communication Protocol). Only one status request at a time can be enabled. The last status block requested will be returned when the DC1 is received.

## INTERPRETING STATUS

In response to status requests the terminal returns an escape code sequence followed by one or more bytes. The status bytes are followed by a terminator. The terminator received may be a CR(LF), RS or GS depending on the

communications protocol and terminal configuration (refer to Section V). The examples that follow use the CR character as a terminator.

The status information is normally contained in the lower four bits of each status byte. The upper five bits of the bytes are set so that the byte will have the value of an ASCII character. Each byte can be interpreted as one of 32 characters as shown in table 6-1.

Some graphics status requests return numeric data such as x and y coordinate values. The terminal returns actual numeric values using ASCII characters for these requests. Refer to the detailed descriptions of the individual graphics status requests for additional information.

## TERMINAL STATUS


Terminal status is made up of 14 status bytes (bytes 0-13) containing information such as display memory size, switch settings, keyboard interface configuration, and terminal errors. These fourteen status bytes are displayed below the terminal Self-Test pattern when the  key is pressed. (Refer to Section VII for a discussion of Self-Test.) There are two terminal status requests, primary and secondary. Each returns a set of 7 status bytes. The terminal status bytes are shown on pages 6-3 and 6-5.

Table 6-1. ASCII Status Characters

ASCII CHARACTER	BINARY	ASCII CHARACTER	BINARY
SPACE	0010 0000	0	0011 0000
!	0010 0001	1	0011 0001
"	0010 0010	2	0011 0010
#	0010 0011	3	0011 0011
\$	0010 0100	4	0011 0100
%	0010 0101	5	0011 0101
&	0010 0110	6	0011 0110
'	0010 0111	7	0011 0111
(	0010 1000	8	0011 1000
)	0010 1001	9	0011 1001
*	0010 1010	:	0011 1010
+	0010 1011	;	0011 1011
,	0010 1100	<	0011 1100
-	0010 1101	=	0011 1101
.	0010 1110	>	0011 1110
/	0010 1111	?	0011 1111

## Primary Terminal Status

The first block of terminal status (bytes 0-6) is requested by sending the following escape sequence:

Primary Terminal Status Request:



The terminal will respond with an ESC \ and 7 status bytes followed by a terminator. A typical primary terminal status request and response is shown in figure 6-1. The example is for a configuration requiring the DC1 character to trigger block transfers.

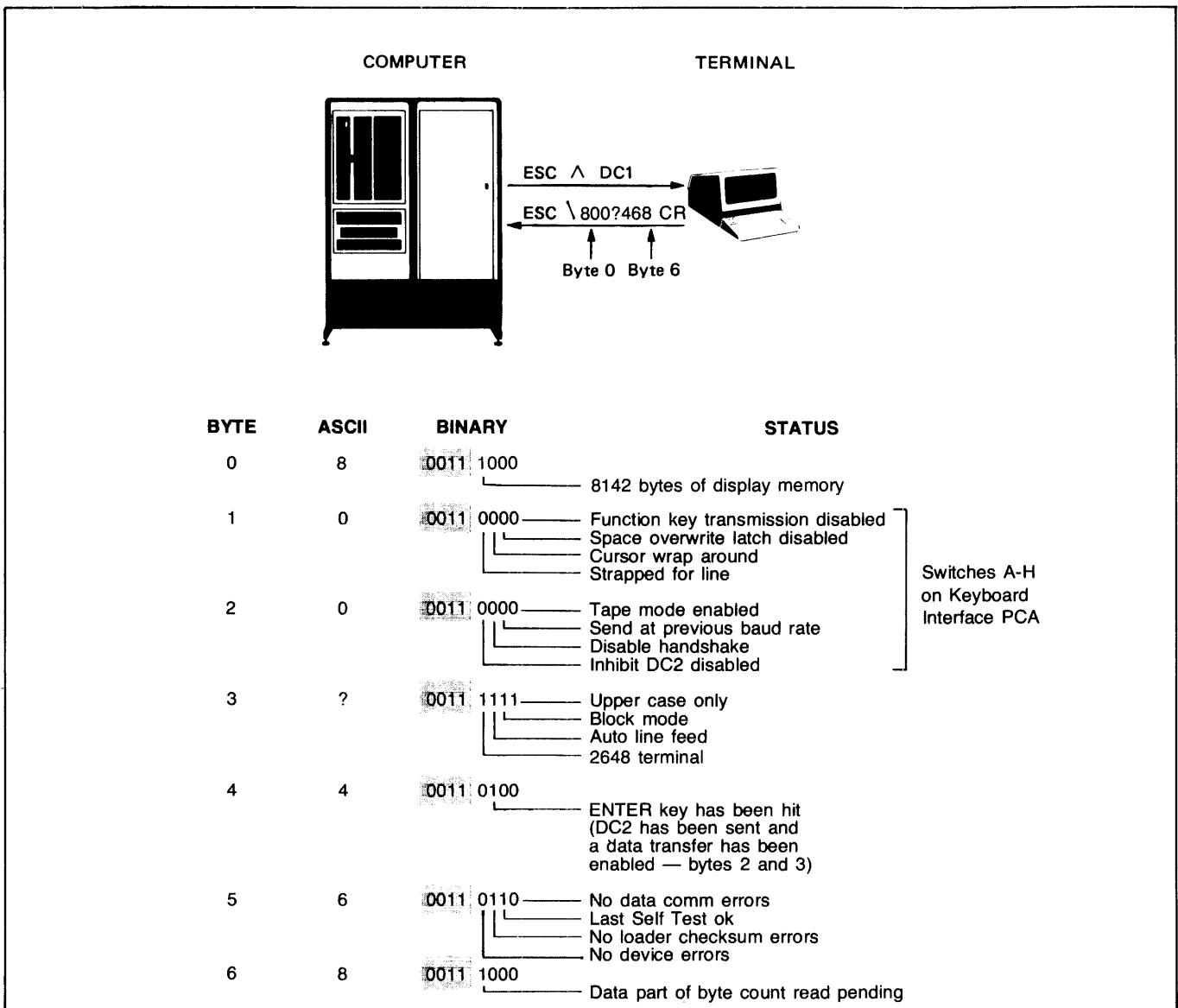
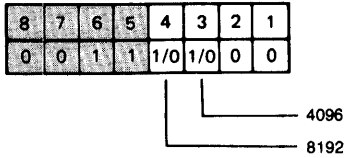


Figure 6-1. Primary Terminal Status Example



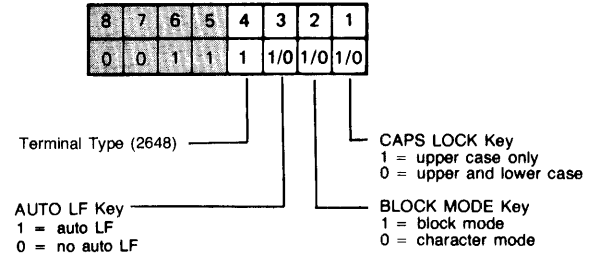
## PRIMARY TERMINAL STATUS

### BYTE 0 DISPLAY MEMORY SIZE

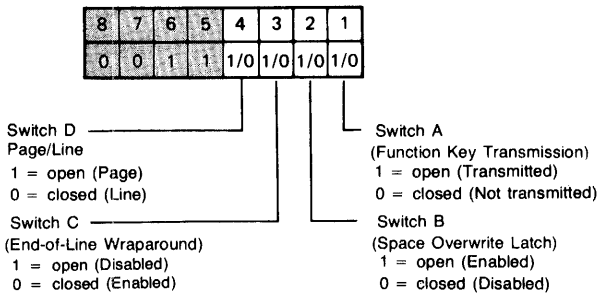


The amount of display memory (blocks of 1K) available in the terminal is returned. The amount can range from 4096 to 12,288 bytes.

### BYTE 3 LATCHING KEYS

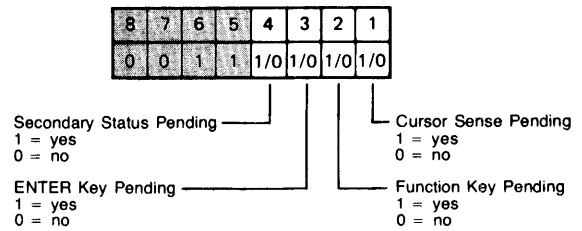


### BYTE 1 KEYBOARD INTERFACE SWITCHES (A-D)

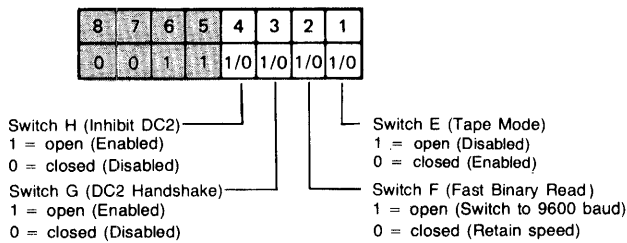


Refer to Section V for a detailed description of Keyboard Interface switches.

### BYTE 4 TRANSFER PENDING FLAGS

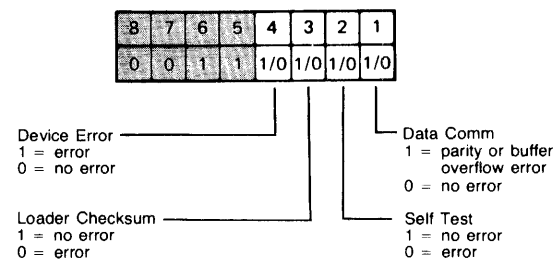


### BYTE 2 KEYBOARD INTERFACE SWITCHES (E-H)

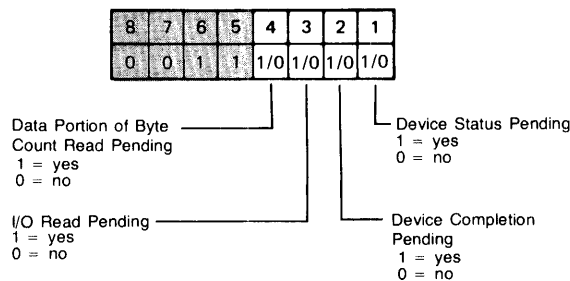


Refer to Section V for a detailed description of Keyboard Interface switches.

### BYTE 5 ERROR FLAGS



### BYTE 6 DEVICE TRANSFER PENDING FLAGS



## Secondary Terminal Status

The second block of terminal status (bytes 7-13) is requested by sending the following escape sequence:

Secondary Terminal Status Request: ESC ~

The terminal will respond with an ESC | and 7 status bytes followed by a terminator. A typical secondary terminal status request and response are shown in figure 6-2.

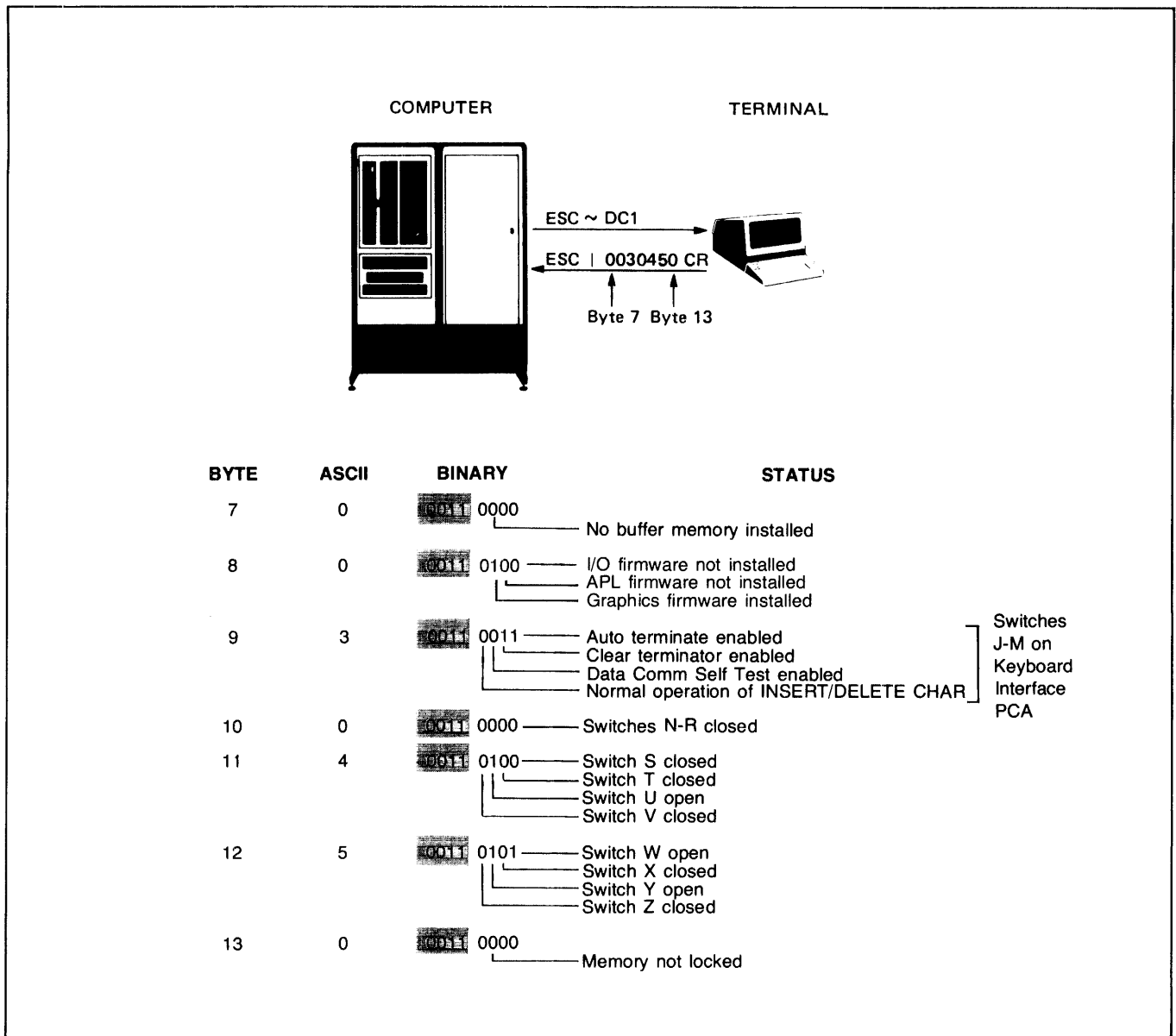


Figure 6-2. Secondary Terminal Status Example

## SECONDARY STATUS BYTES

### BYTE 7 BUFFER MEMORY

8	7	6	5	4	3	2	1
0	0	1	1	0	1/0	0	U

1 = 4096 bytes  
0 = none

Memory installed in addition to display memory that is available for use as data buffers.

### BYTE 8 TERMINAL FIRMWARE CONFIGURATION

8	7	6	5	4	3	2	1
0	0	1	1	0	1	0	1/0

1 = Graphics firmware installed  
0 = No Graphics firmware

1 = I/O firmware installed  
0 = not installed

1 = APL Firmware  
0 = No APL Firmware

The device support firmware is required before tape units or printers can be used with the terminal.

### BYTE 9 KEYBOARD INTERFACE SWITCHES (J-M)

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

Switch M (Alternate Operation — INSERT and DELETE CHARACTER Keys)  
1 = open (Invert wrap sense)  
0 = closed (normal)

Switch L (Self Test Inhibit)  
1 = open (Inhibit test)  
0 = closed (Allow test)

Switch J (Auto Terminate)  
1 = open (Enabled)  
0 = closed (Disabled)

Switch K (Clear Terminator)  
1 = open (Enabled)  
0 = closed (Disabled)

Refer to Section V for a detailed description of Keyboard Interface switches.

### BYTE 10 KEYBOARD INTERFACE KEYS (N-R)

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

Switch R (varies with communications protocol)  
1 = open  
0 = closed

Switch Q Compatibility Mode (Unscaled)  
1 = Enabled  
0 = Disabled

Switch N Printer (Escape Code Transfer)  
1 = open (Send ESC code)  
0 = closed (Do not send code)

Switch P Compatibility Mode (Scaled)  
1 = Enabled  
0 = Disabled

Note that if either or both of the P or Q switches is enabled an extended data comm buffer is selected. Refer to Section V for detailed descriptions of these switches.

### BYTE 11 KEYBOARD INTERFACE KEYS (S-V)

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

Switch V  
1 = open  
0 = closed

Switch U  
1 = open  
0 = closed

Switch S  
1 = open  
0 = closed

Switch T  
1 = open  
0 = closed

The use switches S to V varies depending on the communication protocol used. Refer to Section V for detailed descriptions of their functions.

### BYTE 12 KEYBOARD INTERFACE SWITCHES (W-Z)

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

Switch Z (Parity)  
1 = (Force Parity)  
0 = (Do not Force Parity)

Switch Y (Transmit light)  
1 = open (On when CC high)  
0 = closed (On when CB high)

Switch W (Data Comm Test)  
1 = open (Inhibit)  
0 = closed (Allow)

Switch X (Speed Select)  
1 = open (CH = ON)  
0 = closed (CH = OFF)

The use switches W to Z varies depending on the communication protocol used. Refer to Section V for detailed descriptions of their functions.

### BYTE 13 MEMORY LOCK/BI-LINGUAL MODE

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	0	0

1 = APL Mode  
0 = ASCII Mode

1 = locked  
0 = unlocked

## GRAPHICS STATUS

In addition to normal terminal status you can request graphics status information. All graphics status requests are initiated by sending an `␣ * s` followed by a single parameter (1 through 12 followed by a `^`). The single parameter selects the particular status block desired. If an invalid parameter is used, the terminal will simply return its I.D. (see Device I.D. Request, parameter = 1).

Graphics Status Request:

```
␣ * s <parameter> ^
```

where: `␣ * s` is the graphics status escape sequence.

`<parameter>` is 1-12 and selects one of twelve blocks of graphics status information.

The graphics status blocks that can be requested are listed in table 6-2 together with the format of the terminal's response. Detailed descriptions of each of the status requests are contained in the following paragraphs.

The terminal will respond with one or more bytes of status information followed by a block terminator. All status information is returned in ASCII format, separated by commas. Coordinates are returned in a fixed format, consisting of a sign and five digits. Leading zeros are used as

required to provide a fixed number of digits (i.e. +00100, -01234). This allows you to use simple input statements without the need to mask or shift bits.

If the DC1 handshake protocol is enabled (keyboard straps G and H, refer to Section V), the status block is not actually sent until receipt of a DC1 character. If the DC1 character is used, only one status request can be enabled while the terminal is waiting for a DC1. When the DC1 is received, the last status block requested will be sent.

The keyboard straps determine the terminating characters sent following the status block (`␣`, `␣-L`, `␣`, or `␣`). Graphics status requests turn on an echo suppress mode in the terminal. This prevents information echoed back from the computer from being displayed on the screen. Once a status block has been sent, characters received by the terminal will not be displayed until one of the following control characters is received: `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, `␣`, or `␣`. With the exception of `␣` and `␣` the terminating control code itself will be executed.

The terminal expects the status information to be echoed and uses the terminating control character to turn off the suppress echo mode. If the computer does not echo the status back, a suitable control character must be returned to the terminal to turn off the echo suppress mode.

The graphics status blocks that can be requested are:

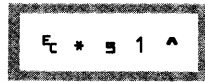
Table 6-2. Graphics Status Requests

Parameter	Request	Response
1	Read device I.D.	2648A
2	Read current pen position	<X>, <Y>, <PEN>
3	Read graphics cursor position	<X>, <Y>
4	Read graphics cursor position with wait	<X>, <Y>, <KEY>
5	Read display size	<LLX>, <LLY>, <URX>, <URY>, <MMX>, <MMY>
6	Read device capabilities	<b1>, <b2>, . . . <b15>, <b16>
7	Read graphics text status	<X size>, <Y size>, <origin>, <angle>, <slant>
8	Read zoom status	<size>, <ON/OFF>
9	Read relocatable origin	<X>, <Y>
10	Read Reset status	<RESET>, <b1>. . . <b6>, <b7>
11	Read area shading capability	1, 8, 8
12	Read dynamics capability	1, 1

### Read Device I.D. (Parameter=1)

When you request a device I.D. the terminal responds with its Hewlett-Packard model number, 2648A.

Device I.D.  
Request:

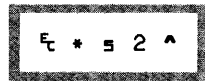


The terminal responds: 2648A <terminator>

### Read Current Pen Position (Parameter=2)

The pen position and status are returned as a string of ASCII characters.

Pen Position  
Request:



The terminal responds: <X>,<Y>,<Pen>,<terminator>

where: <X> = X coordinate  
 <Y> = Y coordinate  
 <Pen> = Pen state, 0=pen up, 1=pen down

For example, assume that the pen is at 360, 80, the pen is up, and the terminal is set for the DC1 handshake, with CR as the terminator:

The computer sends: Esc \* S 2 ^ <terminator> DC1

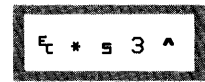
The terminal responds: +00360,+00080,0^

↙ X coordinate      ↖ Pen state  
 ↘ Y coordinate

### Read Graphics Cursor Position (Parameter=3)

The graphics cursor position is returned as a string of ASCII characters.

Read Graphics  
Cursor Request:



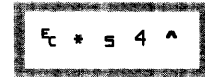
The terminal responds: <X>,<Y> <terminator>

where: <X> = X coordinate  
 <Y> = Y coordinate

### Read Cursor Position with Wait (Parameter=4)

This request allows the user to position the cursor, then strike a key to return the position. The ASCII decimal code for the key struck is also returned (not the actual character). The code is returned as three digits. For example, striking an uppercase A would return 065. Only ASCII character keys will generate a response (i.e. ROLL UP, ROLL DOWN, etc. are ignored). The graphics cursor is turned on, if not already on. If an escape sequence is received by the terminal after it has received the READ CURSOR with WAIT command and before a key is struck, the READ CURSOR command will be aborted. The new sequence will be executed instead.

Read Graphics Cursor  
with Wait Request:



The terminal responds: <X>,<Y>,<key code>  
 <terminator>

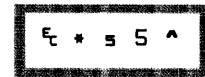
where: <X> = X coordinate  
 <Y> = Y coordinate  
 <key code> = Decimal value of key struck

The position bytes are ordered as in the read pen request.

### Read Display Size (Parameter=5)

This request returns the number of displayable units in the X and Y axes. It also returns the number of units per millimeter in the display. This request allows you to scale data for use on graphic devices with varying display areas.

Read Display  
Size Request:



The terminal responds: <LLX>,<LLY>,<URX>,<URY>,  
 <MMX>,<MMY><terminator>

where: <LLX>,<URX> = Lower left and upper right x coordinates  
 <LLY>,<URY> = Lower left and upper right y coordinates  
 <MMX>,<MMY> = number of units per millimeter in the x and y axes, (five digits and a decimal point)

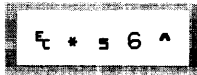
The terminal will always return a fixed response. The lower left corner has coordinates of 0,0. The upper right corner has coordinates of 719,359. There are approximately 3 units per millimeter in each axes.

Terminal response: +00000,+00000,+00719,+00359,  
 00003.,00003.<terminator>

## Read Device Capabilities (Parameter=6)

The device capabilities request returns a list of graphic and plotting features available in the terminal. This allows you to use one program for a variety of graphic devices. Not all of the features listed are available in the terminal. The absence of a feature is indicated by a 0. If a feature is present, it may be necessary to send an additional request to determine the exact capabilities present. Where multiple response values are possible the terminal's standard response is shaded.

Device Capability Request:



The terminal responds:

<b1>, <b2>, <b3>, <b4>, <b5>, <b6>, <b7>, <b8>, <b9>, <b10>, <b11>, <b12>, <b13>, <b14>, <b15>, <b16><terminator>

where:

<b1> - Clear Display  
 0 = no clear  
 1 = paper advance  
 2 = clear (total erase)  
 3 = partial clear by area

<b2> - Number of Pens (1)

<b3>, <b4> - Not Used (0,0)

<b5> - Area Shading  
 0 = no  
 1 = yes (see Read Area Shading Capability)

<b6>, <b7> - Not Used (0,0)

<b8> - Dynamic Modification  
 0 = no  
 1 = yes (see Read Modification Capability)

<b9> - Graphics Character Size  
 0 = fixed  
 1 = integer multiples of the basic cell size  
 2 = any size

<b10> - Graphics Character Angles  
 0 = fixed  
 1 = multiples of 90°  
 2 = multiples of 45°  
 3 = any angle

<b11> - Graphics Character Slant  
 0 = fixed  
 1 = 45°  
 2 = any angle

<b12> - Dot-Dash Line Patterns  
 0 = none  
 1 = predefined only  
 2 = user defined and predefined

<b13>-<b16> - Not Used (0,0,0,0)

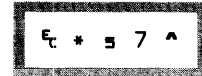
The terminal will always respond:

3,1,0,0,1,0,0,1,1,1,1,2,0,0,0,0<terminator>

## Read Graphics Text Status (Parameter=7)

The terminal returns the current text size, orientation, slant, and type of justification. Refer to Section III, Graphics Functions for a description of graphics text characteristics.

Read Graphics Text Request:



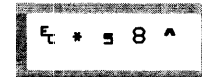
The terminal returns: <x size>, <y size>, <origin>, <angle>, <slant><terminator>

where: <x size> - X dimension of the character cell (three digits)  
 <y size> - Y dimension of the character cell (three digits)  
 <origin> - Relative position of text to cursor (see text origin command)  
 <angle> - Text angle 0, 90, 180, or 270 (five digits and a decimal point)  
 <slant> - 00000. or 00045. degrees

## Read Zoom Status (Parameter=8)

This request returns the terminal's zoom setting.

Read Zoom Status Request:



The terminal responds:

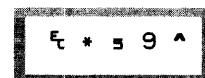
<zoom size>, <zoom on/off><terminator>

where: <zoom size> - zoom setting, 1-16 (three digits and a decimal point)  
 <zoom on/off> - 0 for Off, 1 for On

## Read Relocatable Origin (Parameter=9)

The position of the relocatable origin is returned as x and y coordinates.

Read Relocatable Origin Request:



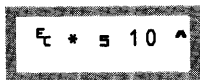
The terminal responds:

<X coordinate>, <Y coordinate><terminator>

## Read Reset Status (Parameter=10)

You can determine whether or not the terminal has executed a full reset (or Power On) since the last time reset status was checked. This will tell you whether or not you need to reestablish terminal settings or images before resuming terminal functions. An additional seven bytes are returned but are not used.

Read Reset  
Status Request:



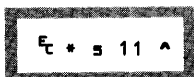
The terminal responds: <reset>, <b1>, <b2>, <b3>, <b4>, <b5>, <b6>, <b7><terminator>

where: <reset status> = 0 No full reset since last check  
or  
1 Terminal has been reset  
<b1>-<b7> = 0 (not used)

## Read Area Shading Capability (Parameter=11)

The area shading capability of the terminal can be read. These are fixed for the terminal.

Read Area  
Shading Request:



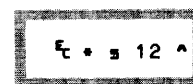
The terminal will always respond: 1,8,8 <terminator>

The "1" indicates that the area shaded must be rectangular. The first "8" indicates that the shading pattern is 8 units wide. The second "8" indicates that the shading pattern is 8 units high.

## Read Graphic Modification Capabilities (Parameter=12)

You can read the terminal's dynamic graphics capabilities. This is the ability of the terminal to change selected portions of the display. These are fixed for the terminal.

Read Graphic Modification  
Capabilities Request:



The terminal will always respond: 1,1 <terminator>

These two bytes indicate that the terminal has selective erase and compliment capabilities.

## Any Other Parameter

Any other parameter which has not been assigned causes the terminal I.D. to be returned. This is to prevent an invalid status request from tying up the requesting computer while waiting for a response.

2648A <terminator>

## DEVICE STATUS

The status of a tape unit or printer can be obtained by a device status request. This request would typically be made following an input/output operation or as a result of testing bytes 5 and 6 of the terminal status. The device status bytes are shown on the following page.

Device status is requested by sending the following escape sequence:

Device Status Request:

```
ESC & p <device code>
```

where: <device> is 1, 2, or 4 and  
 1 = left tape  
 2 = right tape  
 4 = printer  
 5 = HP-IB

The terminal will return an ESC \ p <device code> and 3 bytes of device status followed by a terminator. A typical device status request and response are shown in figure 6-3. A status request from device 3 (display) will be ignored. A status request from device 5 (HP-IB) will return three zeros.

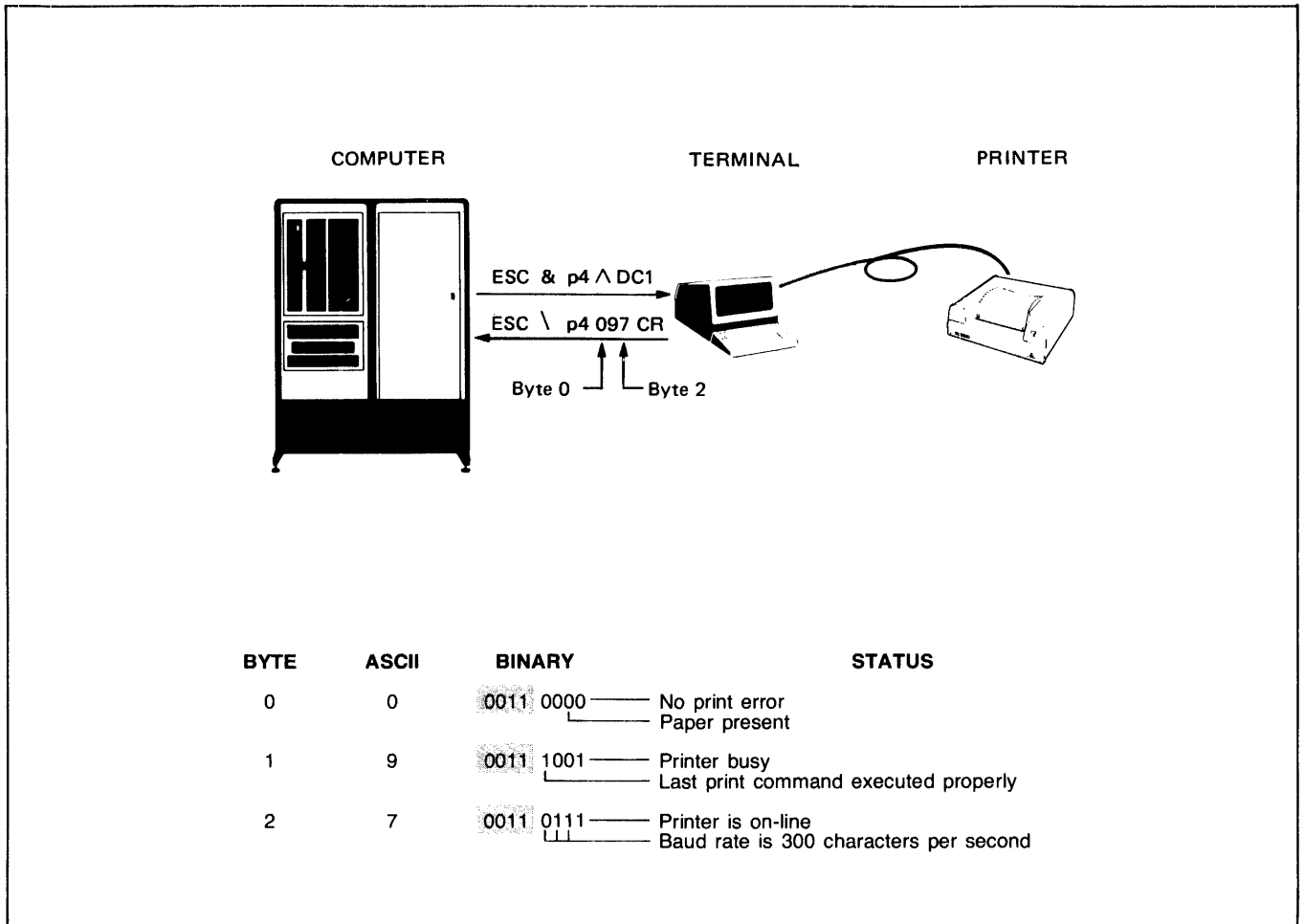


Figure 6-3. Device Status Example



**TAPE UNITS**

**BYTE 0**



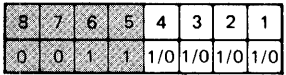
End of File  
1 = at end of file (tape positioned after the file mark)  
0 = not at end of file

Load Point  
1 = at load point  
0 = not at load point

Write Error (Write/Backspace/Read Mode only)  
1 = error  
0 = no error

End of Tape  
1 = at end of tape  
0 = not at end of tape

**BYTE 1**



Command Execution  
1 = last command performed  
0 = last command aborted

Write Protect  
1 = protected  
0 = not protected

Tape Busy  
1 = busy\*  
0 = not busy

Read Error  
1 = error during last read  
0 = no error

\*A "busy" indication is returned when the terminal is:

- conditioning the tape
- rewinding the tape
- finding a file (keyboard or cartridge tape initiated)
- skipping lines (keyboard or cartridge tape initiated)
- no tape present

Since the terminal cannot process a status request while performing a normal read or write operation, these functions will not result in a "busy" indication.

**BYTE 2**



Soft Error (read/write error-recovered)  
1 = yes  
0 = no

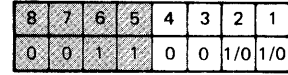
Hard Error (10 read/write failures)  
1 = yes  
0 = no

Tape Inserted  
1 = yes  
0 = no

End of Valid Data  
1 = yes  
0 = no

**PRINTERS**

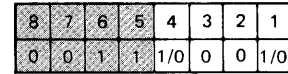
**BYTE 0**



Paper Out (varies with printer)  
1 = yes  
0 = no

Print Error (varies with printer)  
1 = yes  
0 = no

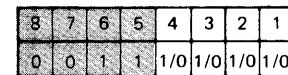
**BYTE 1**



Command Execution  
1 = last command performed  
0 = last command aborted

Printer Busy  
1 = yes  
0 = no

**BYTE 2**



Printer Baud Rate

Printer Connected  
1 = yes  
0 = no

rate	bit		
	4	3	2
external	0	0	0
110	0	0	1
150	0	1	0
300	0	1	1
1200	1	0	0
2400	1	0	1
4800	1	1	0
9600	1	1	1

## INTRODUCTION

This section contains installation instructions for the terminal. Also included are instructions for selecting optional ac operating voltages (115 or 230V), selecting optional operating functions, and installing terminal add-on accessories.

**WARNING**

*Hazardous voltages are present inside equipment. The procedures contained in this section shall be performed only by qualified service personnel.*

**VORSICHT**

*Innerhalb des Geräts bestehen gefährliche Spannungen. Die in diesem Abschnitt enthaltenen Arbeiten dürfen nur durch Betriebsfachpersonal durchgeführt werden.*

**ATTENTION**

*Des tensions dangereuses sont présentes à l'intérieur du matériel. Les opérations décrites dans cette section ne devront être effectuées que par un personnel qualifié.*

**AVVISO**

*Pericolo: Alta tensione presente in questa apparecchiatura. Le procedure contenute in questa sezione debbono essere effettuate soltanto da qualificato personale di servizio.*

**ADVERTENCIA**

*Hay voltaje peligroso en el interior de este equipo. Los procedimientos expuestos en esta sección sólo deberá llevarlos a cabo el personal de servicio calificado.*

**高圧危険**

内部装置に危険な高電圧がきています。この章にある処置や手続に関しては、専門のサービスマンによるのみ行なって下さい。

## OPENING THE TERMINAL

To gain access to the terminal internal components, open the terminal as follows (also see figure 7-1):

- a. Set mainframe rear panel ~ LINE switch to OFF and disconnect power cord from ~ LINE connector.

### NOTE

Mainframe top cover is unlocked by inserting access key supplied with terminal in each of the keyways located on right and left sides of top cover. Inserting keys into keyways unlock top cover. No key rotation is required.

- b. From front of terminal, insert access key into right keyway and unlock right side of terminal by slightly raising right side of top cover. (figure 7-1, A and B).
- c. While maintaining upward pressure to keep right side of terminal unlocked, insert access key into left keyway and raise top cover until both right and left sides of terminal are unlocked. (figure 7-1, C).

- d. Using both hands, carefully swing top cover up until it latches into the half open position. (figure 7-1, D).

### NOTE

The half open position provides adequate room for performing most service routines. However, if extensive repairs are to be made or if components contained in the top cover are to be serviced, fully open mainframe in accordance with step e.

### CAUTION

Mainframe top hinges are open hinge type. When fully opening terminals do not allow top hinges to slip off hinge pins.

- e. Firmly grasp top cover in one hand and release safety latch (see figure 7-2) by pressing it inboard with other hand. Then, using both hands, swing top cover up and over to a full open position (resting on its top).

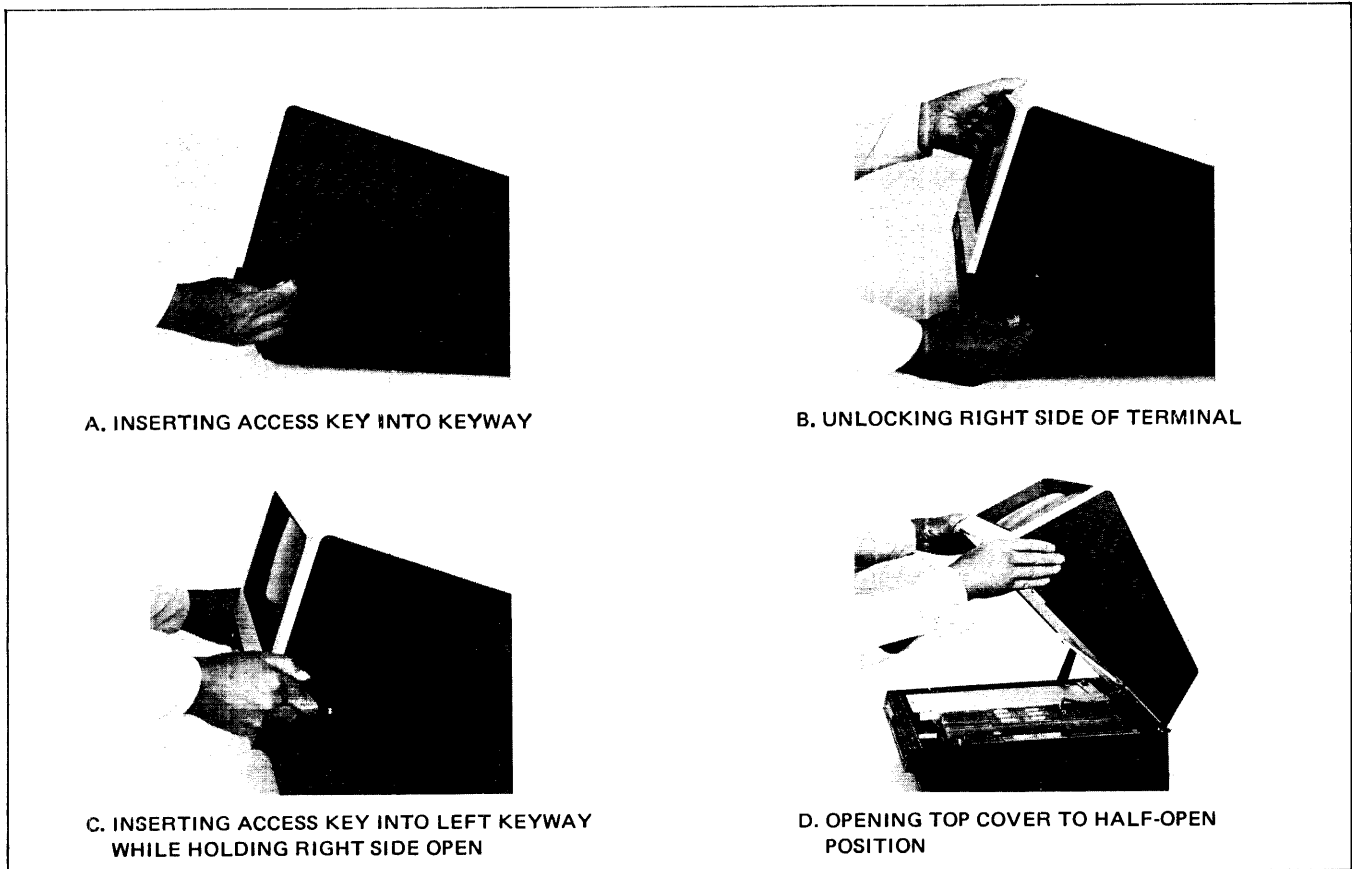


Figure 7-1. Opening the Terminal

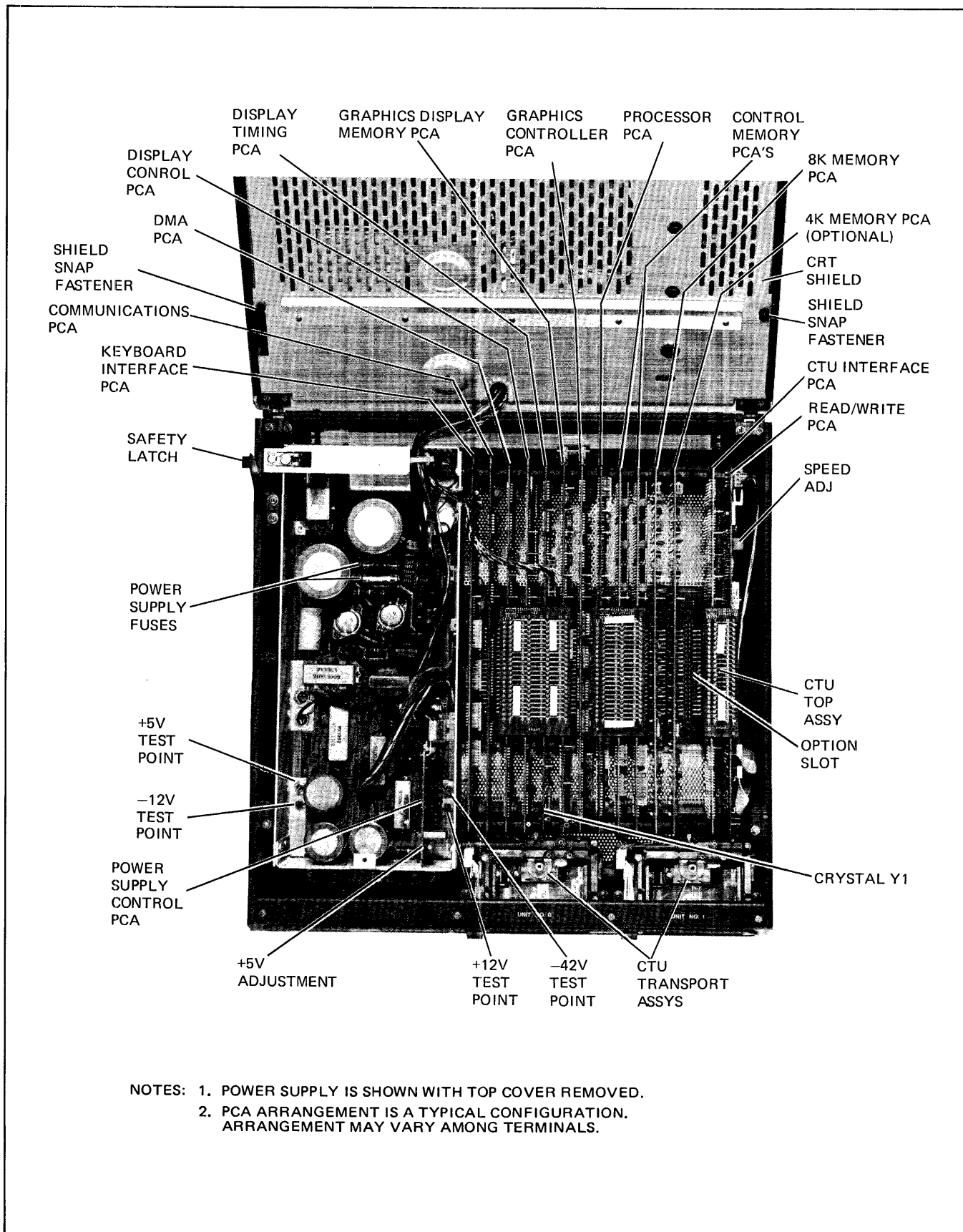


Figure 7-2. Mainframe Bottom Part Locations

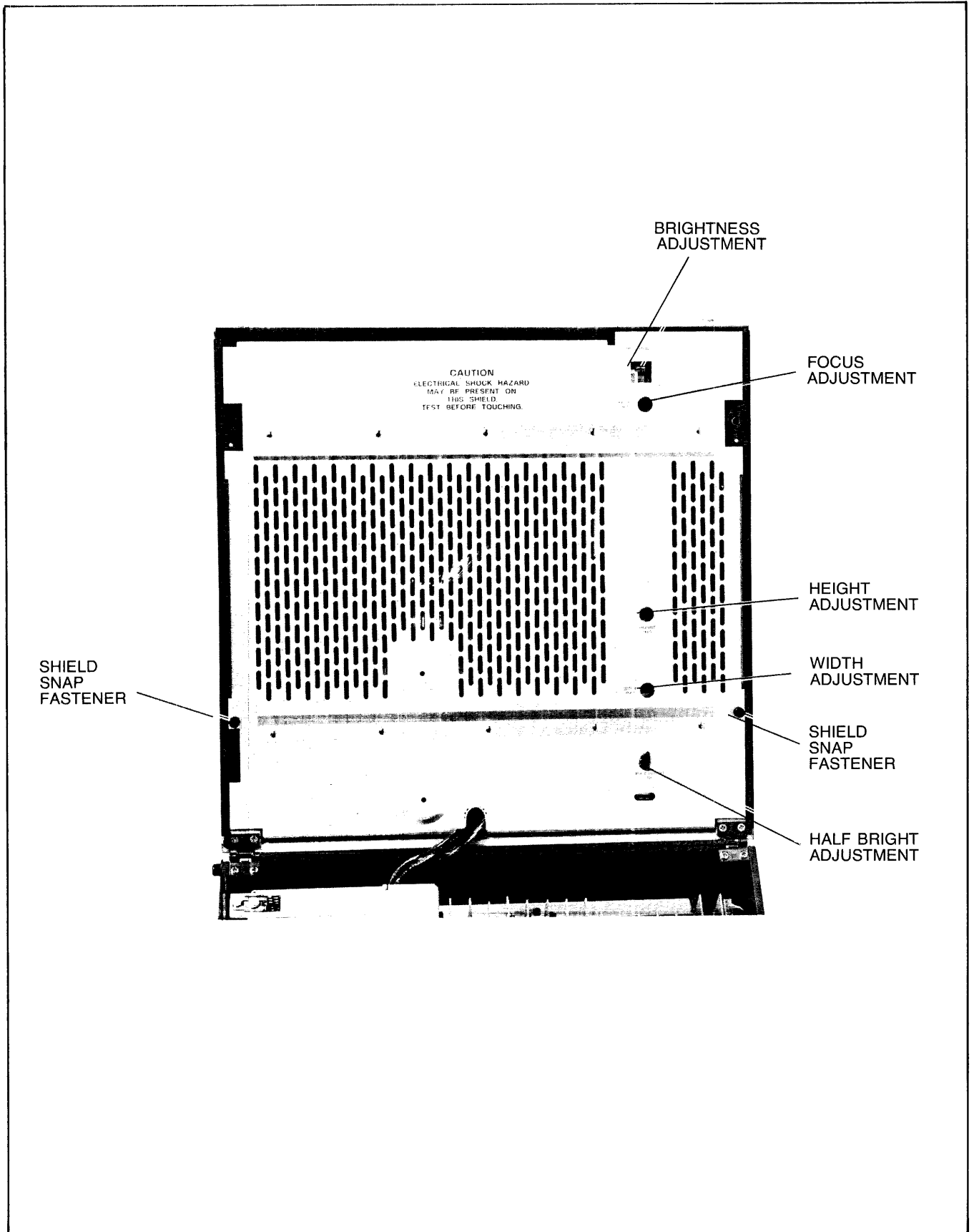


Figure 7-3. Mainframe Top Part Locations

## GROUNDING REQUIREMENTS

To protect operating personnel, the National Electrical Manufacturers' Association (NEMA) recommends that the terminal's frame be grounded. The terminal is equipped with a three-conductor power cable which, when connected to an appropriate power receptacle, grounds the frame of the terminal. To preserve this protection feature, do not operate the terminal from an ac power outlet with no ground connection.

## SELECTING LINE VOLTAGE

The terminal can be operated from either 115 or 230V, 60 Hz line voltage (230V, 50 Hz optional). When shipped from the factory, the line voltage for which the terminal is configured is stamped on the mainframe rear panel identification label. If it is necessary to change the operating line voltage, ensure that power cord is disconnected and proceed as follows:

1. ( ) Open terminal to its half open position in accordance with "Opening the Terminal" paragraph.
2. ( ) Remove power supply cover by removing the screw at the front of the cover and pulling the cover up and out of the mainframe.
3. ( ) Select the operating voltage by inserting the proper fuses into the appropriate locations shown in figure 7-4. For 115 volts, use a 0.5A, SB, 250V fuse and a 4A, SB, 250V fuse. For 230 volts, use a 0.20A, SB, 250V fuse and a 2A, SB, 250V fuse.
4. ( ) If changing from 60 Hz to 50 Hz operation or vice versa, ensure that crystal Y1 on the Display Timing PCA (figure 7-2) is changed. For 60 Hz operation, use a 21.06 MHz crystal (part no. 0410-0647) and for 50 Hz operation, use a 17.55 MHz crystal (part no. 0410-0646).
5. ( ) Check and, if necessary, adjust power supply in accordance with "Power Supply Adjustment".
6. ( ) Replace power supply cover, and secure in place with the screw.
7. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with other hand. Then, using both hands, carefully lower top cover to its closed position.
8. ( ) Perform terminal self-test (refer to "Self-Test").

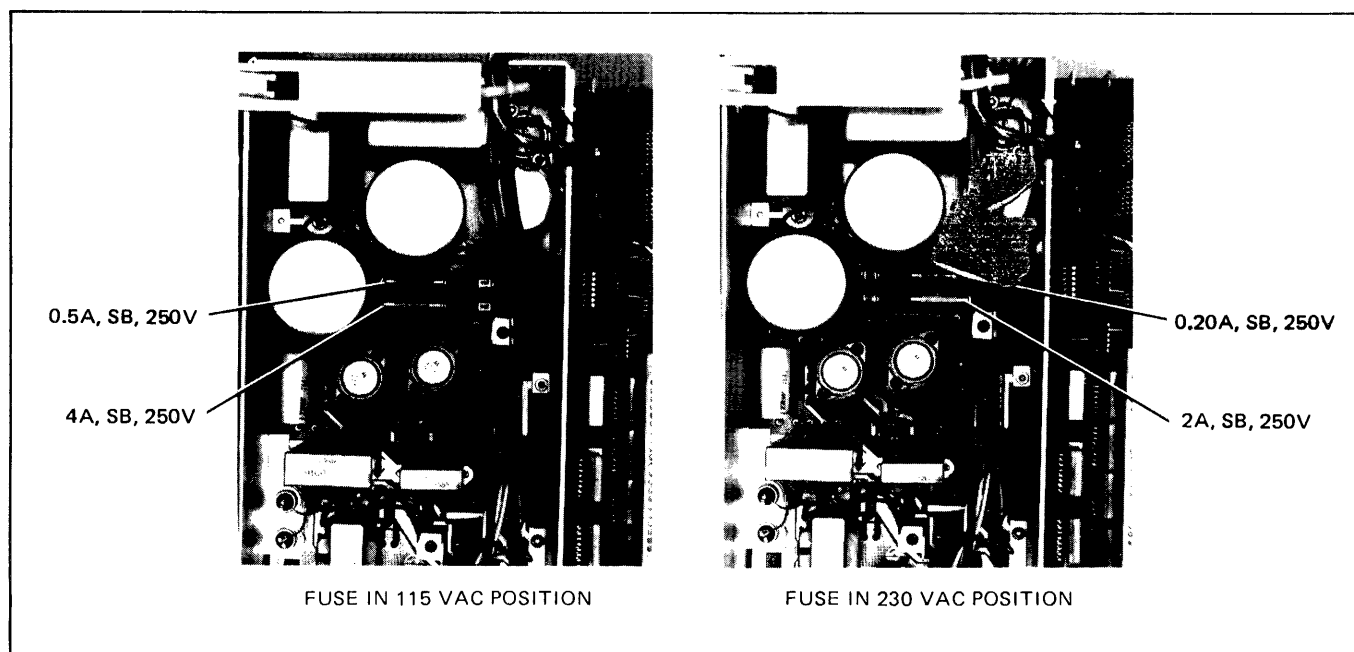


Figure 7-4. Fuse Positions for 115 VAC and 230 VAC Line Voltage

## ACCESSORY INSTALLATION PROCEDURES

Instructions for installing add-on accessories to the standard model terminal are contained in the following paragraphs. Refer to figures 1-1 and 7-2 for typical PCA configurations.

### NOTE

After installing any accessory, always use the terminal self-test feature (refer to "Self-Test") to ensure proper operation.

## HP 13231A Display Enhancements

These instructions apply to the HP 13231A-201 and HP 13231A-203 accessories as well as the HP 13231A accessory. The HP 13231A accessory consists of a Display Enhancement PCA, part no. 02640-60024; a four-wide Top Plane Connector Assembly, part no. 02640-60022; a five-wide Top Plane Connector Assembly, part no. 02640-60016; a Line Drawing Set ROM IC, part no. 1816-0641; and a Connector Removal Tool, part no. 02640-00029. The HP 13231A-201 and -203 accessories consist of the same five items with the applicable ROM IC's mounted on the Display Enhancement PCA. Install any of these accessories as follows:

The alternate character sets are configured with jumpers located on the upper right corner of the Display Enhancement PCA. There are six jumpers, two for each of the three possible alternate character sets. Jumpers 1 and 2 are for alternate character set 1 (referred to as set A in the User's Manual), jumpers 3 and 4 are for alternate character set 2 (set B in the User's Manual), jumpers 5 and 6 are for alternate character set 3 (set C in the User's Manual).

The first jumper for each set (jumpers 1, 3, and 5) indicates whether the set is composed of 128 (jumper in) or 64 (jumper out) characters. The second jumper for each set (jumpers 2, 4, and 6) indicates whether the character set data is in alphanumeric (jumper in) or microvector (jumper out) format. A detailed description of data formats for alternate character sets is given in the application note: *2640 Series Character Set Generation* (part number 13245-90001).

When using the three standard alternate character sets (Math Set, Line Set and Large Character Set) the jumpers would normally be configured as follows:

Math Set (placed in the first socket of set 1)

Jumper 1 = Out, since only 64 characters are used.

Jumper 2 = In, since character data is in alphanumeric format.

Line Set (placed in the first socket of set 2)

Jumper 3 = Out, since only 64 characters are used.

Jumper 4 = Out, since character data is in microvector form.

Large Character Set (placed in the first select of set 3)

Jumper 5 = Out, since only 64 characters are used.

Jumper 6 = Out, since character data is in microvector form.

Note that the Math Set has been shown as alternate character set 1 (A in the User's Manual), the Line Set as alternate 2 (B in the User's Manual), and the Large Character Set as alternate 3 (C in the User's Manual). They could have been configured as any combination of the three possible alternate sets. There is no requirement that the sets be configured in any order.

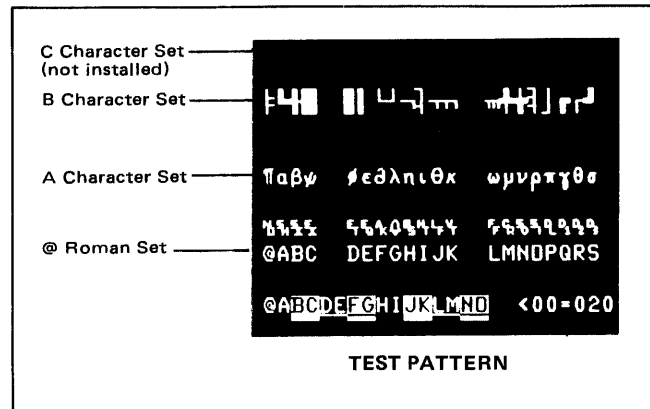


Figure 7-5. Sample Character Set Configuration

### NOTE

Do not confuse the 128/64 character jumpers for *alternate character sets* with the 128 character jumper for the *standard character set*.

## EFFECT OF IMPROPER JUMPER PLACEMENT.

*128 Characters Strapped for 64.* When a 128 character set is used and is jumpered for 64 characters, only the first 64 characters in the set will be accessed. This will cause the "q" character for example to access the same display character as the "Q" character.

*64 Characters Strapped for 128.* Any attempt to access one of the lower case 64 characters ("a", "q", etc.) will result in a blank being displayed.

Table 7-1. Display Enhancement PCA Jumper Protocol

ALTERNATE SET	128/64 (JUMPER IN/JUMPER OUT) CHARACTERS	ALPHANUMERIC/MICROVECTOR (JUMPER IN/JUMPER OUT) CHARACTER DATA
A	JUMPER 1	JUMPER 2
B	JUMPER 3	JUMPER 4
C	JUMPER 5	JUMPER 6

*Alphanumeric Data Strapped as Microvector.* Alphanumeric data strapped as microvector will normally result in characters that are skewed or fuzzy.

*Microvector Data Strapped as Alphanumeric.* Microvector data strapped as alphanumeric will display blanks for the microvector characters.

**INSTALLATION PROCEDURE**

- ( ) Using figure 7-6 and table 7-1 as a guide, check that Display Enhancement PCA jumpers are arranged correctly for the ROM character set configuration. If there are no alternate character set ROM's installed (HP 13231A), all jumpers should be in the jumper socket.

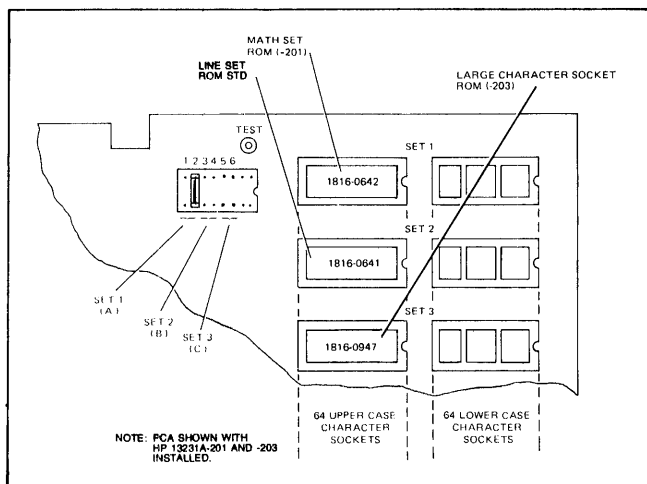


Figure 7-6. Display Enhancement PCA Jumper and ROM Socket Locations

- ( ) Open terminal to its half open position (refer to "Opening the Terminal").
- ( ) Insert connector removal tool under Top Plane Assembly as shown in figure 7-7.
- ( ) Remove Top Plane Assembly by pressing down on connector removal tool handle. Retain Top Plane Assembly for possible future use.

- ( ) Rearrange the PCA's in the Backplane Assembly so that the Display Enhancement PCA is adjacent to the Display Memory Access (DMA), Display Control, Display Timing, and Graphics Display PCA's.

**NOTE**

PCA arrangement can be in any configuration with the following exceptions. The Keyboard Interface and data communications PCA's should be installed in one of the first three Backplane Assembly connectors closest to the power supply.

The Display Enhancement, DMA, Display Control, Display Timing, and Graphics Display PCA's must always be installed as a group in adjacent connectors. No Backplane Assembly connectors can be left vacant between any PCA's. In addition, the Processor PCA must be installed adjacent to the display PCA's described previously.

- ( ) Install five-wide Top Plane Connector Assembly, part no. 02640-60016 on Display Enhancement, DMA, Display Control, Display Timing, and Graphics Display PCA connectors.

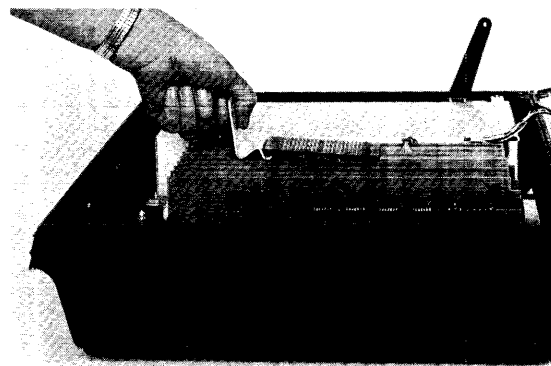


Figure 7-7. Top Plane Assembly Removal



Installation

- 7. ( ) Check and, if necessary, adjust power supply (refer to "Power Supply Adjustment").
- 8. ( ) Depress TEST key and observe last line of test pattern for correct display enhancements. If enhancements are correct skip to step 11. If adjustment is necessary, perform step 10.
- 9. ( ) Perform brightness, half bright, focus, and field adjustments in accordance with the *Service Manual*, part no. 02648-90003.

- 10. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with other hand. Then, using both hands, carefully lower top cover to its closed position.

**HP 13232 Cable Assemblies**

The HP 13232 cable assemblies provide interface connections between the terminal and modems, printers, and computers. Table 7-2 below lists the particulars of each cable.

Table 7-2. 13232 Cable Assemblies

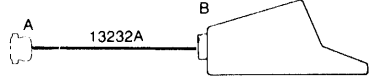
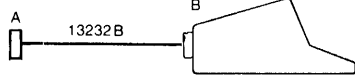
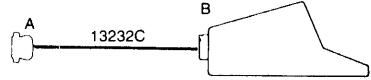
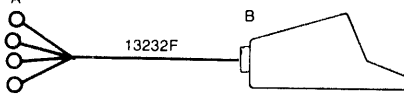
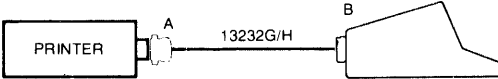
CABLE	FUNCTION	CONNECTORS			LENGTH	HOOKUP
		A	B	C		
13232A	Connects data communications interface PCA to modem 103/202. (Cable part no. 02640-60043.)	RS232 (male)	Hood	—	4.57 Metres 15 feet	
13232B	Connects 12531/12880 teleprinter interface PCA to terminal. (Cable part no. 02640-60058.)	Hood	Hood	—	15.25 Metres 50 feet	
13232C	Connects data communications interface PCA to RS232 connector. (Cable part no. 02640-60059.)	RS232C (female)	Hood	—	1.52 Metres 5 feet	
13232F	Provides current loop connections for 13260B data communications interface. (Cable part no. 02640-60097.)	4 terminal lugs	Hood	—	1.52 Metres 5 feet	
13232G	Connects 13250A Serial Printer Interface to RS232 compatible printers. (Cable part no. 02640-60098.)	RS232C (male)	Hood	—	4.57 Metres 15 feet	
13232H	Same as 13232G. (Cable part no. 02640-60099.)	RS232C (female)	Hood	—	4.57 Metres 15 feet	

Table 7-2. 13232 Cable Assemblies (Continued)

CABLE	FUNCTION	CONNECTORS			LENGTH	HOOKUP
		A	B	C		
13232J	Connects 13238A Duplex Register PCA to 9871A Printer. (Cable part no. 02640-60116.)	9871A printer (female)	Hood	—	1.83 Metres 6 feet	
13232K	Connects 13254A Video Interface PCA to Tektronix 4632/7 Video Copier. (Cable part no. 02640-60120.)	RS232 (male)	Hood	—	4.57 Metres 15 feet	
13232L	Connects 13254A Video Interface PCA to Conrac Monitor. (Cable part no. 02640-60121.)	BNC	Hood	—	7.61 Metres 2.5 feet	
13232N	Connects data communications interface PCA to modem. (Cable part no. 02640-60131.)	RS232C (male)	Hood	—	4.57 Metres 15 feet	
13232P	Connects 13260C or 13260D data communication interface PCA to modem in multipoint configurations. (Cable part no. 02640-60132.)	RS232C (male)	Hood	Multipoint (female)	4.57 Metres 15 feet (each leg)	
13232Q	Connects 13260C or 13260D data communications interface PCA to other terminals in downstream multipoint configuration. (Cable part no. 02640-60133.)	Multipoint (male)	Hood	Multipoint (female)	4.57 Metres 15 feet (each leg)	
13232R	Provides 100-foot extension to 13232P, Q, T, multipoint cables. (Cable part no. 02640-60134.)	Multipoint (male)	Multipoint (female)	Multipoint (female)	30.5 Metres 100 feet	
13232S	Connects 13238A Duplex Register PCA to 9866A/B Printer. (Cable part no. 02640-60135.)	9866 printer (male)	Hood	—	1.83 Metres 6 feet	
13232T	Provides power-down protection for a terminal in multipoint configuration. (Cable part no. 02640-60151.)	Multipoint (male)	Hood	Multipoint (female)	4.57 Metres 15 feet (each leg)	
13232U	Provides direct connection to a computer by replacing the modem connections. (Cable part no. 5060-2403.)	RS232C (female)	RS232C (female)	—	1.52 Metres 5 feet	

### HP 13234A (4K) Terminal Memory Module

Install the HP 13234A (+4K) memory accessory as follows:

1. ( ) Open terminal to its half open position (refer to "Opening the Terminal").
2. ( ) Locate and ensure that the Control Memory PCA's, part no. 02640-60192, are jumpered properly. (See figures 7-2 and 7-8.)
3. ( ) Using figure 7-9 as a guide, locate memory jumpers on 4K Memory PCA.
4. ( ) Using figures 7-10 and 7-11 as guides, arrange PCA starting address jumpers to select appropriate memory starting address for the size memory being configured. For example: to add 4K of display memory remove all but the 8K jumper from the 4K Memory PCA you are installing. To add a 4K block of data comm buffer remove all but the 4K and 8K jumpers from the 4K Memory PCA.
5. ( ) Install memory PCA's in any vacant Backplane Assembly connectors ensuring that no connectors are left vacant between any PCA's other than the two cartridge tape unit PCA's (if installed).
6. ( ) Check and, if necessary, adjust power supply (refer to "Power Supply Adjustment").

7. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with other hand. Then, using both hands, carefully lower top cover to its closed position.

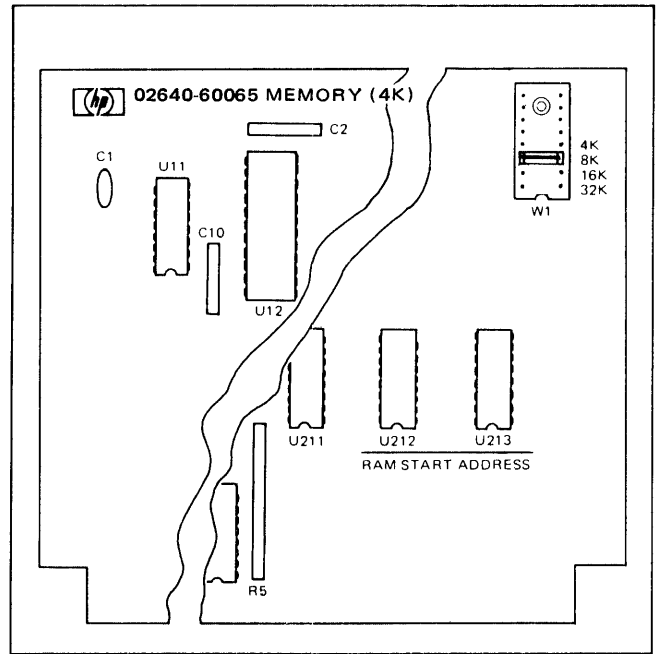


Figure 7-9. 4K Memory PCA Jumper Socket Location

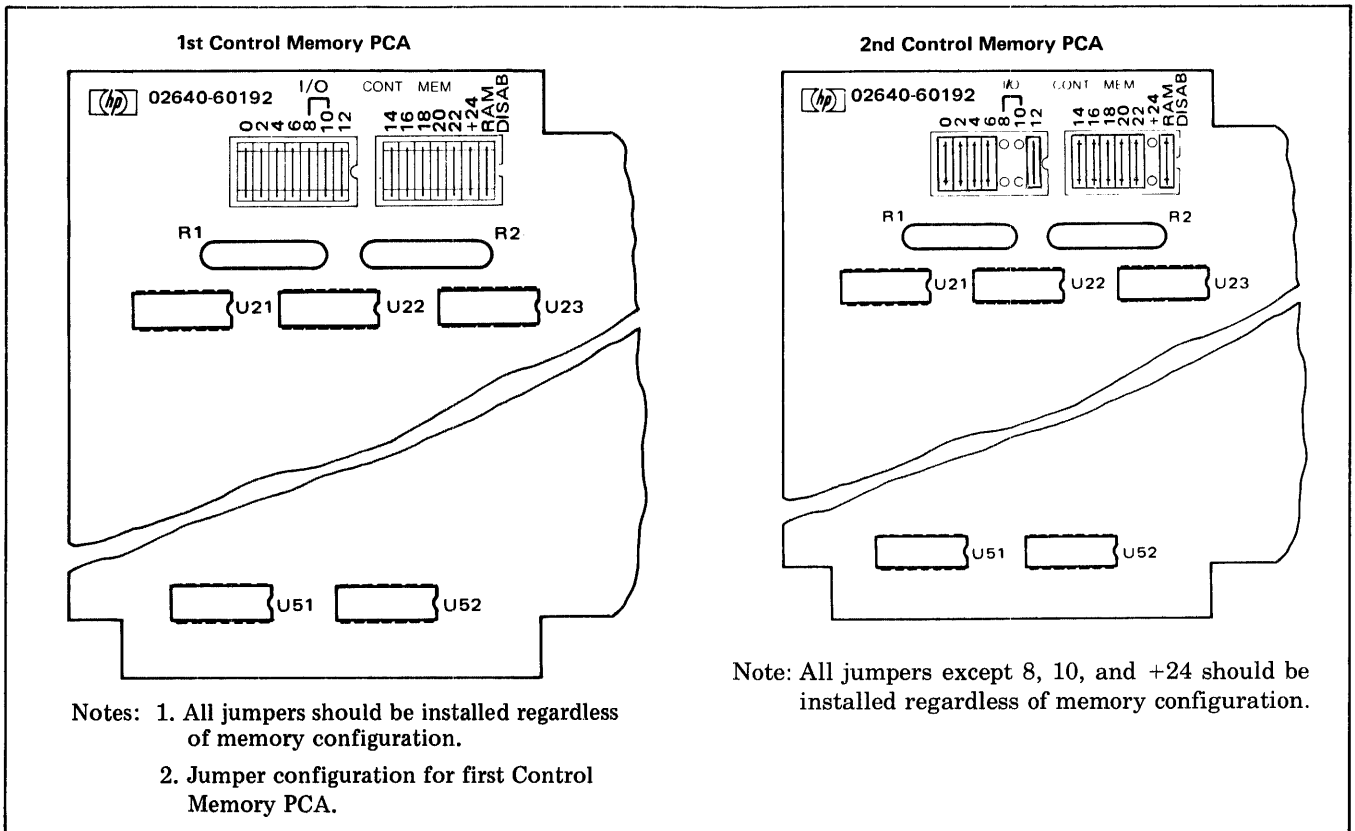


Figure 7-8. Control Memory PCA Jumper Socket Location

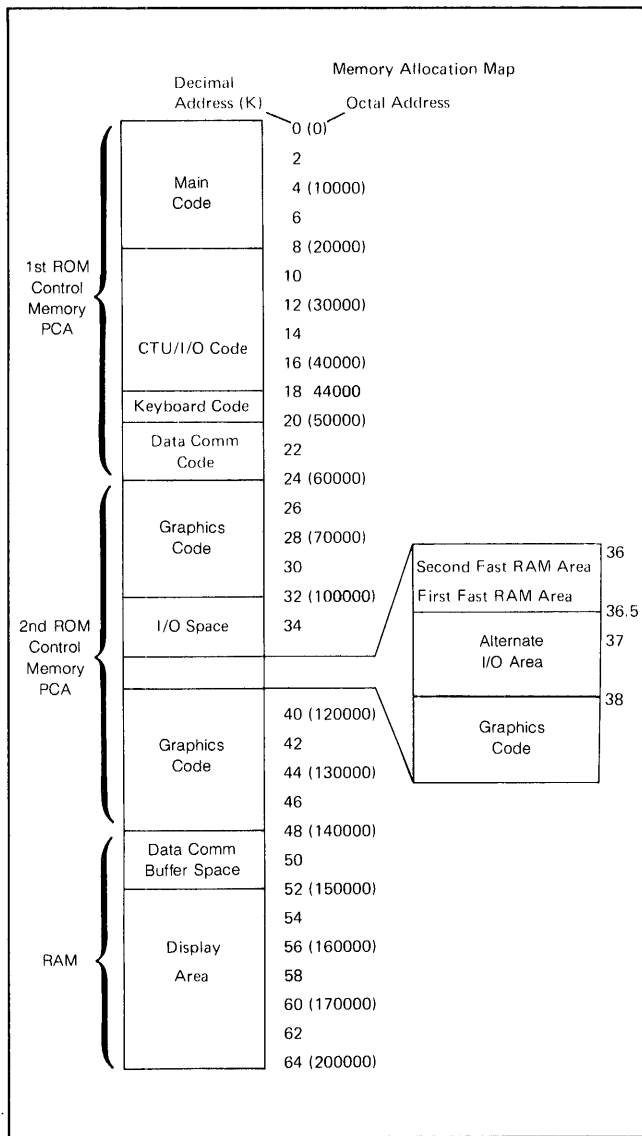
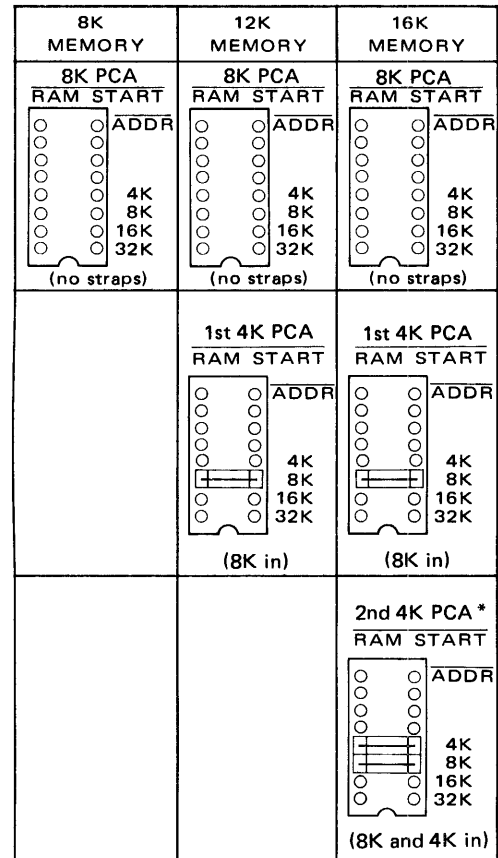


Figure 7-10. Typical Memory Map



\*Used for data comm buffer only.

Figure 7-11. Configuring 4K Memory PCA Jumpers

## HP 13236B Cartridge Tape Unit

The HP 13236B Cartridge Tape Unit is provided to upgrade the terminals to include mass storage capabilities. This accessory consists of two cartridge tape (CTU) Transport Assemblies, two Tape Cartridges, a CTU Interface PCA, a Read/Write PCA, a CTU Top Plane Assembly, a Motor Cable Assembly, a front bezel and required attaching hardware. Installation procedures are contained in the HP 13236A/B Cartridge Tape Unit Accessory Manual, part no. 13236-90004.

## HP 13238A Terminal Duplex Register

To install the HP 13238A accessory, perform all the following steps except steps 4 and 5.

1. ( ) Open terminal to its half open position (refer to "Opening the Terminal").
2. ( ) Configure jumpers in Terminal Duplex Register PCA jumper sockets as shown in figure 7-12.
3. ( ) Install Terminal Duplex Register PCA in first vacant Backplane Assembly connector adjacent to existing PCA's.

### NOTE

To ensure proper terminal operation, all PCA's must be installed in adjacent Backplane Assembly connectors. There should never be vacant connectors between PCA's except for the two CTU PCA's in option 007 (Read/Write PCA and CTU Interface PCA) which can be separated from the others.

4. ( ) Open mainframe rear door by twisting two lock extrusions.

5. ( ) Holding Terminal Duplex Register PCA firmly in place, carefully connect hood connector of the cable assembly, supplied with the printer subsystem, to PCA connector P2.

### NOTE

The hood connector and PCA connector P2 are identically keyed to prevent inadvertent erroneous connections. Connecting the two together requires minimal hand pressure. If excessive resistance is encountered, an incorrect connection is being attempted.

For printer interfacing information refer to the *HP 9866A/B Printer Operator's Manual*, part no. 09866-90901, or the *HP 13349A Printer Subsystem Operating Manual*, part no. 13349-90901.

6. ( ) Check and, if necessary, adjust power supply (refer to "Power Supply Adjustment").
7. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with other hand. Then, using both hands, carefully lower top cover to its closed position.

## HP 13245A Character Set Generation Kit

The Character Set Generation Kit Accessory requires 1 backplane slot and consists of a PROM Character PCA, part no. 02640-60053 and a Connector Assembly, part no. 02640-60070. Install the HP 13245A accessory as follows:

1. ( ) Open terminal to its half open position (refer to "Opening the Terminal").

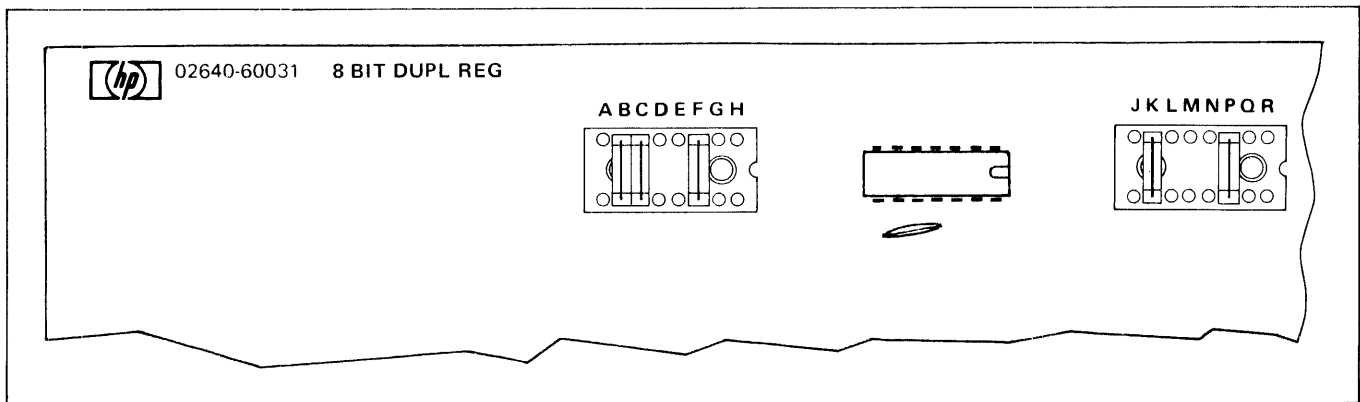


Figure 7-12. Terminal Duplex Register PCA Jumper Configuration

2. ( ) Rearrange PCA's in the Backplane Assembly so that an unused connector is available for the PROM Character PCA adjacent to either the Display Control PCA or Display Enhancement PCA depending on the character set(s) to be replaced. If the base character set is to be replaced, vacate a connector adjacent to the Display Control PCA. If an alternate character set(s) is to be replaced, vacate a connector adjacent to the Display Enhancement PCA.
5. ( ) Attach Connector Assembly, part no. 02640-60070 between the two interface connectors (P2) on the PROM Character PCA and Display Control PCA or Display Enhancement PCA.
6. ( ) Check and, if necessary, adjust power supply (refer to "Power Supply Adjustment").
7. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with other hand. Then, using both hands, carefully lower top cover to its closed position.

#### NOTE

PCA arrangement can be in any configuration with the following exceptions. The Keyboard Interface and data communications PCA's should always be installed in one of the first three Backplane Assembly connectors closest to the power supply. The Display Enhancement, DMA, Display Control, Graphics Display, and Display Timing PCA's must always be installed as a group in adjacent connectors to accommodate the Top Plane Connector Assembly. The CTU Interface and Read/Write PCA's (option 007) must always be installed in adjacent connectors. No Backplane Assembly connectors can be left vacant between any PCA's except for the two CTU PCA's which can be separated from the others.

3. ( ) Install PROM Character PCA in vacated Backplane Assembly connector.

#### NOTE

The base or alternate character set ROM(s) to be replaced by the user generated PROM set(s) must be removed from the applicable PCA in accordance with the instructions contained in the *Character Set Generation Kit Application Note*, part no. 13245-90001.

4. ( ) When connected to the Display Enhancement PCA, the PROM Character PCA character sets 1 and 2 replace the Display Enhancement PCA character sets 1 and 2 respectively. If an alternate set(s) is to be replaced, first determine if the user generated PROM set(s) is alphanumeric or microvector. Then, using table 7-1 and figure 7-6 as a guide, correctly arrange Display Enhancement PCA jumpers 2 and 4 for the PROM character set type(s). (Jumpers 1 and 3 can either be removed or left installed.)

## HP 13250B Serial Printer Interface

The HP 13250B provides an RS232C interface to serial printers for alphanumeric output. You can configure the 13250B to be compatible with many RS232 serial printers requiring handshake or full-character protocol. For details on configuring and installing the interface, refer to the HP 13250 accessory manual, part no. 13250-90004.

## HP 13254A Video Interface

The 13254A consists of a Video Interface PCA, part no. 02640-60019, and a Sweep Extender Cable, part no. 02640-60122. This accessory is used to link the terminal to a compatible video monitor or hard copy device. The accessory requires one option slot. Detailed installation and operating information is contained in the *13254A Accessory Manual*, part no. 13254-90001.

## HP 13260A,B,C,D Data Communications Accessories

The HP 13260A,B,C,D Data Communications Accessories provide various types of data communications from teletypewriter compatible data communications to asynchronous or synchronous multipoint polling. (Refer to Section V for details of these accessories.) Only one data communications interface may be installed in the terminal at any time. Each accessory consists of the items listed in table 7-3.

### CAUTION

MOS integrated circuits can be damaged by electrostatic discharge. Use the following precautions:

Table 7-3. Contents of 13260 Data Communications Accessories for the 2648A

ITEM	PART NUMBERS									
	13260A-003	13260B-003	13260C	13260D						
Interface Printed Circuit Assembly	02640-60086	02640-60143	02640-60106	02640-60107						
ROM IC's	1818-0547 <sup>1</sup>	1818-0547 <sup>1</sup>	1818-0584 (std only) <sup>2</sup> 1818-0583 (opt 001 only) <sup>2</sup> 1818-0585 <sup>2</sup>	1818-0584 (std only) <sup>2</sup> 1818-0583 (opt 001 only) <sup>2</sup> 1818-0585 <sup>2</sup>						
Keyboard Overlay			7120-6925	7120-6925						
Baudrate Label	7120-6388	7120-6388	7120-6386	7120-6386						
Cable Assembly		02640-60083	02640-60083	02640-60083						
Switch Cover			4040-1356	4040-1356						
<p><sup>1</sup> Earlier units used 1818-0411. If this older part number must be replaced, refer to the Service Manual for additional compatibility information.</p> <p><sup>2</sup> Earlier units use one of the following sets of IC's:</p> <table style="width: 100%; border: none;"> <tr> <td style="width: 50%;">1818-0214 (std only)</td> <td style="width: 50%;">1818-0434 (std only)</td> </tr> <tr> <td>1818-0261 (opt. 001 only)</td> <td>1818-0433 (opt. 001 only)</td> </tr> <tr> <td>1818-0288</td> <td>1818-0435</td> </tr> </table> <p>If replacement is necessary, replace with the same part number (i.e., do not use 1818-0584 to replace 1818-0214 or 1818-0434. IC's from different sets cannot be mixed. Also, if it is necessary to replace an entire set, it is recommended that the same set be used. If replacement is made with a different set, it may be necessary to reconfigure the terminal. (Refer to Appendix C for backdating information.)</p>					1818-0214 (std only)	1818-0434 (std only)	1818-0261 (opt. 001 only)	1818-0433 (opt. 001 only)	1818-0288	1818-0435
1818-0214 (std only)	1818-0434 (std only)									
1818-0261 (opt. 001 only)	1818-0433 (opt. 001 only)									
1818-0288	1818-0435									

DO NOT wear clothing subject to static charge buildup, such as wool or synthetic materials.

DO NOT handle MOS circuits in carpeted areas.

DO NOT remove the circuit from its conductive foam pad until you are ready to install it.

AVOID touching the circuit leads. Handle by the plastic package only.

ENSURE that the circuit, work surface (table, desk, etc.) and PCA are all at the same ground potential. This can be done by touching the foam pad to the PCA and then touch the foam pad, circuit, and PCA to the work surface.

1. ( ) Perform complete terminal SELF TEST (refer to "Self-Test") to verify proper terminal operation before installing the accessory.
2. ( ) Turn off ~ LINE switch at rear of terminal and disconnect power cord.
3. ( ) Open terminal as described in "Opening the Terminal."
4. ( ) Locate the first Control Memory PCA, part no. 02640-60192 (see figure 7-2). All jumpers are installed on the first Control Memory PCA.
5. ( ) Insert connector removal tool under Top Plane Assembly as shown in figure 7-7, and remove Top Plane Assembly by pressing down on connector removal tool handle.

6. ( ) Remove the first Control Memory PCA.
7. ( ) Remove IC's, if present, from DATA COMM sockets 20 and 22 of Control Memory PCA (see figure 7-13).
8. ( ) Carefully unpack and inventory 13260 accessory parts per table 7-3 above.
9. ( ) Carefully insert ROM(s) contained in accessory package into DATA COMM socket(s) so that ROM pin 1 is at upper right corner of socket (see figure 7-13). Also, be sure that all jumpers are installed as shown in figure 7-8, "1st Control Memory PCA."
10. ( ) Reinstall Control Memory PCA into backplane assembly connector from which it was removed.
11. ( ) Reinstall Top Plane Assembly on Processor and Control Memory PCA's top connectors.
12. ( ) Configure the 13260B,C, or D Interface PCA's for your particular application by setting the switches on the PCA. (Refer to "Selecting Optional Operating Functions", page 7-17.)
13. ( ) Install the Interface PCA in the first vacant backplane assembly connector adjacent to existing PCA's. For 13260C/D installation, connect ground cable assembly (part no. 02640-60083) between power supply chassis ground and PCA ground connector lug.

**NOTE**

To ensure proper terminal operation, all PCA's must be installed in adjacent Backplane Assembly connectors. There

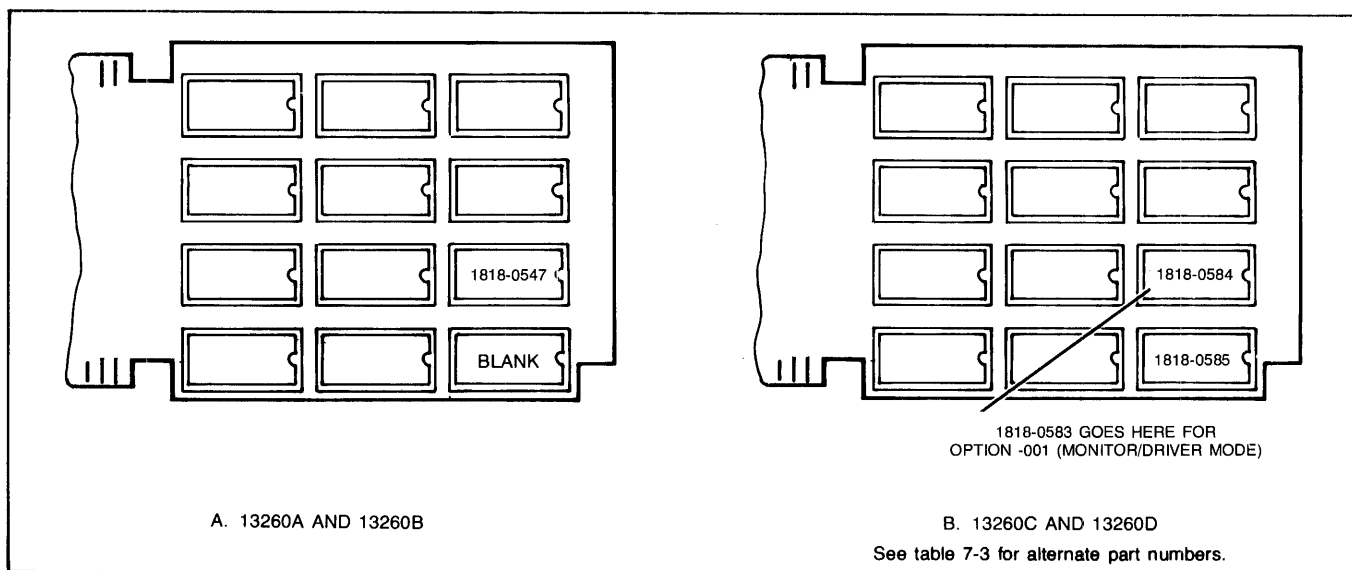


Figure 7-13. Control Memory PCA Data Comm Firmware IC Locations

should never be vacant connectors between PCA's except for the two CTU PCA's (Read/Write PCA and CTU Interface PCA) in option 007 which can be separated from the others.

14. ( ) Open mainframe rear door by twisting the two lock extrusions.
15. ( ) Holding the communications interface PCA firmly in place, carefully connect Test Connector Assembly, part no. 02645-60002, (supplied with the terminal) to PCA, and perform the DATA COMM SELF-TEST to verify proper operation of the PCA (refer to "Data Communications Self-Test"). After self test is performed, remove the Test Connector Assembly.
16. ( ) Holding the communications interface PCA firmly in place, carefully connect hood connector of a 13232C, F,N,P,Q, or T Cable Assembly to PCA. For cabling information, refer to "HP 13232 Cable Assemblies" and "Data Communications Cabling."

## NOTE

The hood connector and PCA connector P2 are identically keyed to prevent inadvertent erroneous connections. Connecting the two together requires minimal hand pressure. If excessive resistance is encountered, an incorrect connection is being attempted.

17. ( ) Install baudrate switch overlay and keyboard overlay as shown in figure 7-14.

18. ( ) Firmly grasp top cover in one hand, and release safety catch by pressing it inboard with your other hand. Then, using both hands, carefully lower top cover to its closed position.

### HP 13261A-003 Device Support Firmware

The 13261A-003 Device Support Firmware provides the firmware required to operate peripheral devices from the terminal (such as printers, hard copy unit, etc.). This firmware is not required if option 007 (the cartridge tape units) is installed.

The accessory consists of five ROM IC's (part nos. 1818-0406, 1818-0407, 1818-0408, 1818-0409, and 1818-0746) mounted on a conductive foam pad. The pad prevents the ROM IC's from being damaged by electrostatic discharge.

### CAUTION

MOS integrated circuits can be damaged by electrostatic discharge. Use the following precautions:

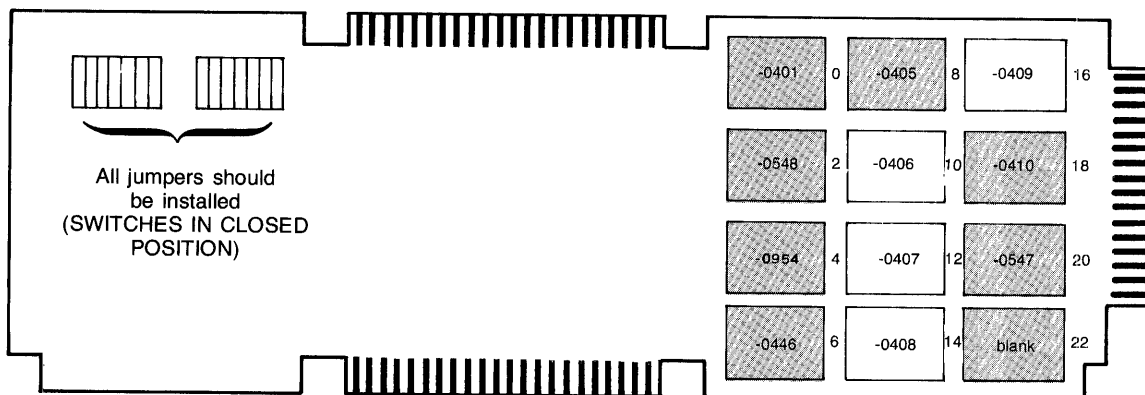
DO NOT wear clothing subject to static charge buildup, such as wool or synthetic materials.

DO NOT handle MOS circuits in carpeted areas.

DO NOT remove the circuit from its conductive foam pad until you are ready to install it.



**1st Control Memory PCA (02640-60192)**



NOTE: UNSHADED ROM LOCATIONS INDICATE DEVICE SUPPORT FIRMWARE ROMS.

**2nd Control Memory PCA (02640-60192)**

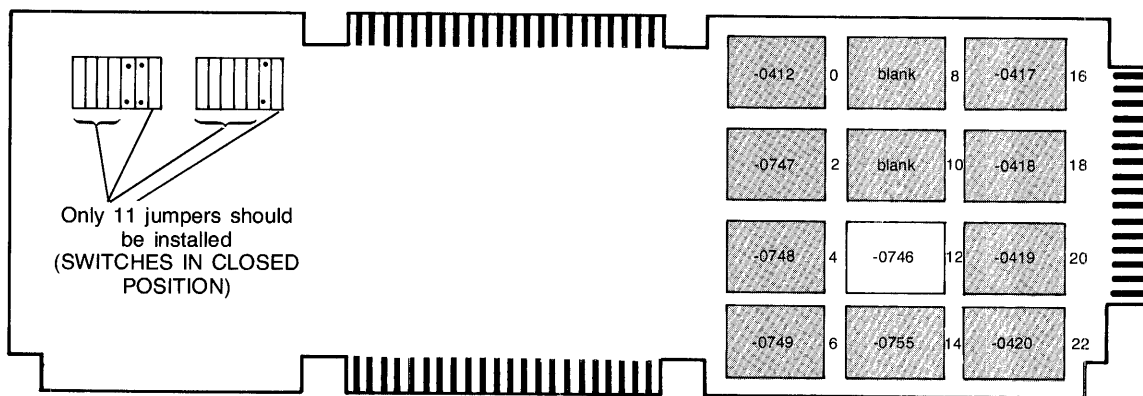


Figure 7-15. 13261A-003 Device Support Firmware ROM IC Locations on Control Memory PCA's

## HP 13296A Shared Peripheral Interface

This accessory provides the means for communicating between your terminal and various external devices using the Hewlett-Packard Interface Bus (HP-IB) as prescribed in the IEEE Standard Document 488-1975.

The HP 13296A accessory includes the following items:

1. An HP-IB Interface PCA (part no. 02640-60128).
2. An HP-IB Interface Adapter (part no. 02640-60215).
3. A 2-meter HP-IB interconnecting cable (part no. 8120-1834).
4. Option -048 includes a ROM IC (part no. 1818-0746) which adds raster dump capability to the existing device support firmware in earlier version terminals. Later version terminals include this ROM as part of the device support firmware; therefore, option -048 is not required.

To configure and install this accessory, proceed as follows:

1. ( ) Using table 7-3A as a guide, set the switches on the interface PCA to the appropriate positions.
2. ( ) Open the terminal to its half open position (refer to "Opening the Terminal").

Table 7-3A. HP-IB Interface Switch Settings

SWITCH(ES)	SETTING(S)								
A4,A11,A10,A9	These four switches specify the PCA module address and must be set as follows: <table style="margin-left: auto; margin-right: auto;"> <tr> <td>A4</td> <td>A11</td> <td>A10</td> <td>A9</td> </tr> <tr> <td>Closed</td> <td>Open</td> <td>Closed</td> <td>Closed</td> </tr> </table>	A4	A11	A10	A9	Closed	Open	Closed	Closed
A4	A11	A10	A9						
Closed	Open	Closed	Closed						
PL6 through PL0	These seven switches are reserved for future use. PL6 should be set to the closed position and switches PL5 through PL0 should be set to the open position.								
ATN, ATN2	These switches are reserved for future use and should be set to the open position.								
FC, TA, and LA	These three switches are reserved for future use and should be set to the closed position.								
B4 through B0	These five switches specify the HP-IB address of your terminal. Set these switches as follows: <table style="margin-left: auto; margin-right: auto;"> <tr> <td>B1</td> <td>B0, B2 thru B4</td> </tr> <tr> <td>Closed</td> <td>Open</td> </tr> </table>	B1	B0, B2 thru B4	Closed	Open				
B1	B0, B2 thru B4								
Closed	Open								
SC	Set to open (system controller).								

3. ( ) If your interface accessory includes option -048, install ROM IC (part no. 1818-0746) in the appropriate Control Memory PCA. Follow the HP 13261A-003 Device Support Firmware installation procedures on page 7-15. Figure 7-15 shows the proper location for the ROM IC.)
4. ( ) Install the HP-IB interface PCA in a vacant Backplane Assembly option slot. (To allow room for mounting the HP-IB Adapter in step 5, you may need to create a slot position by moving some of the PCA's in the backplane one slot position to the right.)
5. ( ) Holding the HP-IB Interface PCA firmly in place, carefully connect the hood connector of the HP-IB Interface Adapter to PCA connector P2.

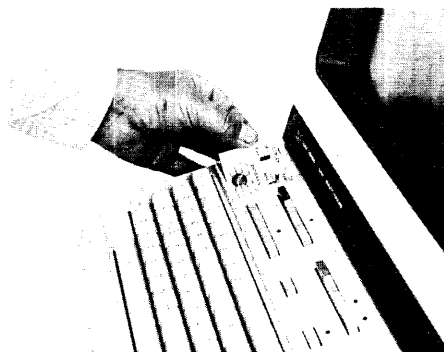
### NOTE

The hood connector and PCA connector P2 are identically keyed to prevent inadvertent erroneous connections. Connecting the two together requires minimal hand pressure. If excessive resistance is encountered, an incorrect connection is being attempted.

6. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with your other hand. Then, using both hands, carefully lower top cover to its closed position.
7. ( ) Connect one end of the 2-meter HP-IB cable to the connector on the interface adapter. Connect the other end of the cable to the peripheral device. (Refer to Section 8, "HP-IB Configurations" for further information on peripheral connections.)



A. REMOVING A KEYBOARD OVERLAY



B. INSTALLING BAUDRATE OVERLAY



C. INSTALLING KEYBOARD OVERLAY

AVOID touching the circuit leads. Handle by the plastic package only.

ENSURE that the circuit, work surface (table, desk, etc.) and PCA are all at the same ground potential. This can be done by touching the foam pad to the PCA and then touch the foam pad, circuit, and PCA to the work surface.

1. ( ) Perform complete terminal SELF-TEST (refer to "Self-Test") to verify proper terminal operation before installing the accessory.
2. ( ) Turn off AC POWER switch at rear of terminal and disconnect power cord.
3. ( ) Open terminal (refer to "Opening the Terminal").
4. ( ) Locate the Control Memory PCA's, part no. 02640-60192 (see figure 7-2).
5. ( ) Insert connector removal tool under Top Plane Assembly as shown in figure 7-7.
6. ( ) Remove both Control Memory PCA's.
7. ( ) Carefully unpack and inventory 13261A accessory parts.
8. ( ) Carefully insert the appropriate ROM contained in accessory package into the *empty* ROM sockets 10 through 16 on the Control Memory PCA (see figure 7-15). Be sure that ROM pin 1 is at upper right corner of each socket. Be sure that all jumpers are installed (see figure 7-15).
9. ( ) Carefully insert the appropriate ROM contained in accessory package into empty ROM socket 14 on the other Control Memory PCA (see figure 7-15). Be sure that ROM pin 1 is at upper right corner of each socket. Be sure that the appropriate jumpers are installed (see figure 7-15).
10. ( ) Reinstall the Control Memory PCA's into backplane assembly connectors from which they were removed.
11. ( ) Reinstall Top Plane Assembly on Processor and Control Memory PCA's top connectors.
12. ( ) Perform terminal SELF-TEST to verify proper terminal operations after installing accessory.

Figure 7-14. Installing Keyboard Overlays

## Power Supply Adjustment

After installing or removing accessories, you should adjust the +5 volt output of the terminal's power supply. Only this voltage need be adjusted because the +5 volts provides reference for the other supply voltages. The adjustment requires a 20,000 ohms/volt voltmeter.

To adjust the +5V, proceed as follows:

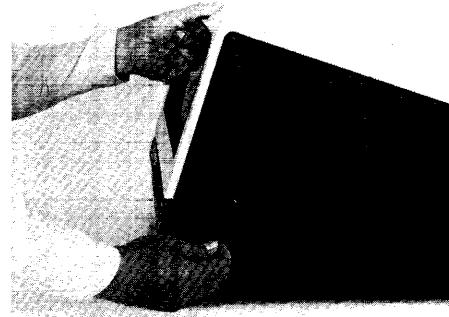
1. ( ) Open the terminal, and remove power supply cover.
2. ( ) Turn on ac power to terminal, and ensure that neither cartridge tape transport motor is running (if installed).
3. ( ) Check the voltages at the following points with the multimeter. (See figure 7-2.)

TEST POINT	VOLTAGE TOLERANCE
+5V diode	+4.85V to +5.25V
-42V diode	-40V to -46V
+12V diode	+11.8V to +12.6V
-12V diode	-11.8V to -12.6V

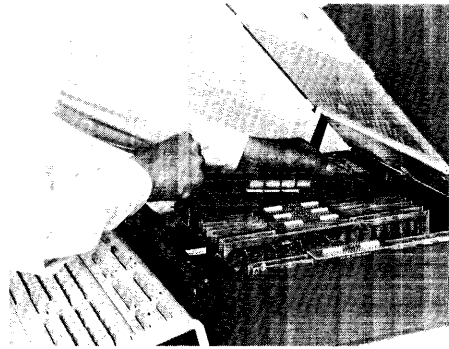
4. ( ) Adjust +5 volt potentiometer until all voltages are within tolerance.
5. ( ) When all voltages are within tolerance, turn off power, disconnect multimeter, and replace power supply cover.

## SELECTING OPTIONAL OPERATING FUNCTIONS

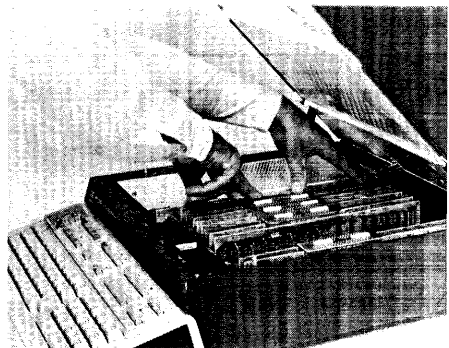
The terminal is equipped with jumper and switch selectable options that can be used to alter some of its operating functions (see figure 7-16). These options and their effects on terminal operation are discussed in tables 7-4 through 7-8. To select an operating function, proceed as follows:



1. ( ) Open the terminal to its half open position (refer to "Opening the Terminal").



2. ( ) Locate the particular PCA, and remove the cable hood connector from the PCA. Then, remove the PCA from the Backplane Assembly connector.
3. ( ) Using figure 7-17, 7-18, 7-19, or 7-20 (as applicable) and table 7-4, 7-5, 7-6, 7-7 or 7-8 (as applicable), select the desired operating functions, and set the switches to the appropriate positions. (More information on configuration is given in Section V.)



4. ( ) Reinstall the PCA into the vacated Backplane Assembly connector.



5. ( ) Firmly grasp mainframe top cover in one hand and release safety latch by pressing it inboard with your other hand. Then, using both hands, carefully lower top cover to its closed position.

6. ( ) Perform SELF-TEST (refer to "Self-Test").

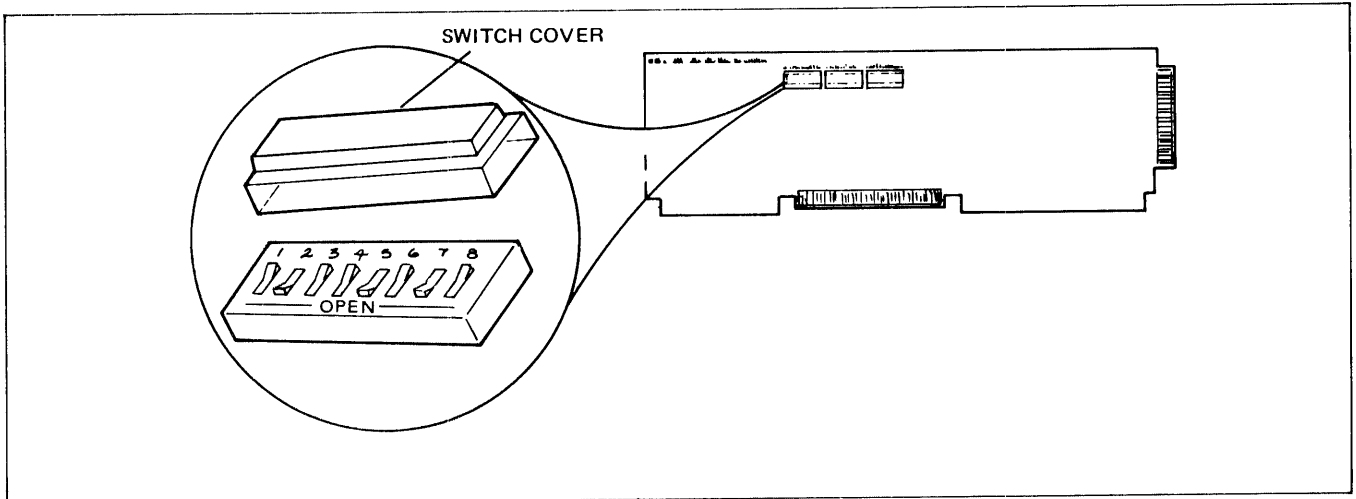


Figure 7-16. Typical Strapping Option Switch Assembly

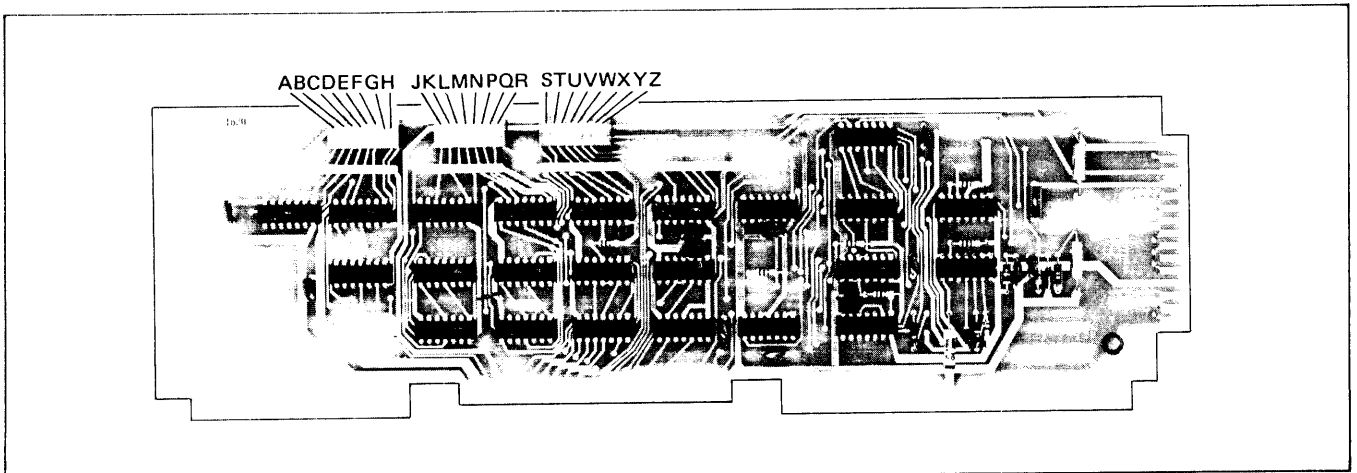


Figure 7-17. Keyboard Interface PCA Strapping Options

Table 7-4. Keyboard Interface PCA Strapping Options for Point-to-Point

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
A	Function Key Transmission	The escape code sequence generated by the major function keys (such as, ROLL UP, ROLL DOWN, etc.) are executed locally, but not transmitted to the computer.	The escape code sequences generated by all keys are transmitted to the computer. If operating in half duplex, the function is also executed locally.
B	Space Overwrite (SPOW) Latch Enable	Spaces typed will overwrite existing characters.	When the SPOW latch is off, overwriting occurs as normal. When the SPOW latch is on, spaces cause the cursor to move forward but not overwrite any existing characters. The SPOW latch is turned on by a Carriage Return, and off by a Line Feed, Home or Tab.
C	Cursor End-of-Line Wraparound	At the end of each line, a local Carriage Return and Line Feed are generated; the cursor moves to the beginning of the next line.	A Carriage Return and Line Feed are not generated at the end of each line. The cursor remains in and overwrites column 80.
D	Line/Page	The terminal is set to transfer a line at a time in Block Mode.	Entire pages of information are transferred in Block Mode.
E	Paper Tape Mode	When the <b>READ</b> key is pressed with <b>AUTO FEED</b> key latched down, each tape record begins with an <b>LF</b> and is terminated by a <b>CR</b> .	Each tape record is terminated by <b>CR</b> .
F	Fast Binary Read	The transmission rate is determined by the BAUD RATE switch on the keyboard.	When an <b>FR</b> (Fast Binary Read) is issued by the computer, the baud rate is switched automatically to 9600 baud (if the terminal is equipped with cartridge tape units).
G	Block Transfer Handshake	In Block Mode, all data transfers to the computer are sent upon receipt of a DC1 from the computer.	All Block Mode transfers (i.e., cursor sense, terminal and device status, device I/O responses, display memory, and function keys) are preceded by a DC2. The terminal sends the DC2 upon receipt of a DC1 from the computer. After the CPU receives the DC2 from the terminal, another DC1 is required to trigger transmission of data from the terminal.
H	Inhibit DC2	During Block Mode Handshake transfers, the terminal sends a DC2 in response to a DC1 prior to sending data. (See Block Transfer Handshake strapping above.)	A DC1 from the computer is not required to trigger data transfers to the computer. Also, the DC2 from the terminal is not sent during Block Mode Transfer handshakes. (See Block Transfer Handshake strapping above.) Additionally, when the <b>ENTER</b> key is pressed in Block Mode the cursor will be placed in the first column before transmission occurs if operating in Line/Field Mode (switch D closed) or Home'd if operating in Page Mode (switch D open.) Opening both switches G and H eliminate the terminal's use of the Handshake protocol entirely.
J	Auto Terminate	No effect.	When in BLOCK mode and the ENTER key is pressed, places a non-displaying terminator before the cursor position.
K	Clear Terminator	No effect	Clear terminator caused by Strapping Option J or <b>CR</b> .
L	Self Test Inhibit	No effect.	Self Test function is inhibited. Pressing TEST key or issuing <b>CR</b> z displays the NO TEST message. TAPE TEST and DATA COMM SELF TEST functions are not affected.
M	INSERT and DELETE CHAR with wrap (Reverse Sense)	No effect.	Reverses effect of <b>CTRL</b> key on INSERT CHAR and DELETE CHAR keys (i.e., when key is pressed, line wrap around is in effect without having to press CNTL key. When either key is pressed while pressing CNTL, normal insert character and delete character functions are in effect.)

Table 7-4. Keyboard Interface PCA Strapping Options for Point-to-Point (Continued)

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
N	Escape Code Transfer to Printer	No effect.	Escape codes relating to the display (e.g., display enhancements, alternate character sets, format mode, fields, etc.) are sent to printer if it is selected as a destination device.
P,Q	Compatibility Mode	These switches set the terminal to be compatible with Tektronix control commands when initialized (power on or full reset).	
		P-closed, Q-closed	Normal operation
		P-closed, Q-open	Unscaled Compatibility Mode and 2048 byte data comm buffer.
		P-open, Q-closed	Scaled Compatibility Mode and 2048 byte data comm buffer.
		P-open, Q-open	2048 byte data comm buffer.
R	Circuit Assurance	The transition from receive state to transmit state occurs after both CB (106) (Clear to Send) and SB (119) (Secondary Receive Data) go on within 2.6 seconds. Otherwise, the terminal returns to the receive state.	The transition from receive state to transmit state occurs after CB (106) (Clear to Send) goes on.
S,T	Main Channel Protocol	Reverse Channel protocol (both switches closed).	<b>S-closed, T-open:</b> Main channel with STX/ETX as Start of Data and End of Data. <b>S-open, T-closed:</b> Main channel with EOT as End of Data. <b>S-open, T-open:</b> Main channel with ETX as End of Data.
U	CPU Break	The CPU can interrupt the terminal while it is in the transmit state. The CPU initiates an ON to OFF transition of the SB (119) (Secondary Receive Data) line. The terminal responds by turning off CA (105) (Request to Send) and going to the receive state.	The terminal ignores all transitions on the SB (122) (Secondary Receive Data) line from the modem in the transmit state.
V	Carrier Detect	When the terminal is in the receive state, an ON to OFF transition of CF (109) (Carrier Detect) line from the modem causes the terminal to go into the transmit state. Transitions of CF have no effect while the terminal is in the transmit state.	Transitions of CF (109) (Carrier Detect) line have no effect on the terminal.
W	Data Comm Self Test Enable	Enables DATA COMM SELF TEST from either the keyboard or escape sequence.	Disables DATA COMM SELF TEST. If self test is attempted (by either the keyboard or escape sequence), the test will be aborted and ERROR 0 will appear on the display.
X	Data Speed Select	Holds data speed signal low (CH (111) = 0).	Sets data speed signal high (CH (111) = 1).
Y	Transmit LED	The TRANSMIT light on the keyboard is turned on when CB (106) (Clear to Send) line from the modem is high. It is turned off when the CB (106) line goes low.	The TRANSMIT light on the keyboard is turned on when the CC (107) (Data Set Ready) line from the modem is high and the 13260B Extended Asynchronous Communications Interface PCA is used. It is turned off when the CC line goes low.
Z	Parity	The PARITY switch on the terminal keyboard is affected as follows:	
		<p><b>No Parity:</b> Send 8 bits and receive 8 bits. Force bit 8 to zero. Check for parity error.</p> <p><b>Odd Parity:</b> Send 7 data bits + odd parity. Receive 7 data bits + odd parity. Check for parity error.</p> <p><b>Even Parity:</b> Send 7 data bits + even parity. Receive 7 data bits + even parity. Check for parity error.</p>	<p><b>No Parity:</b> Send 8 bits and receive 8 bits. Force bit 8 to one on send. No check for parity error.</p> <p><b>Odd Parity:</b> Send 7 bits + odd parity. Receive 7 bits. No check for parity error.</p> <p><b>Even Parity:</b> Send 7 data bits + even parity. Receive 7 data bits. No check for parity error.</p>

Table 7-5. Keyboard Interface Straps for Block Operation Using 13260C or 13260D Communications Accessory

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
A	Function Key Transmission	The escape code sequence generated by the major function keys (such as, ROLL UP, ROLL DOWN, etc.) are executed locally, but not transmitted to the computer.	(Same as switch closed.)
B	Space Overwrite (SPOW) Latch Enable	Spaces typed will overwrite existing characters.	When the SPOW latch is off, overwriting occurs as normal. When the SPOW latch is on, spaces cause the cursor to forward but not overwrite any existing characters. The SPOW latch is turned on by a Carriage Return, and turned off by a Line Feed, Home, or Tab.
C	Cursor End-of-Line Wrap Around	At the end of each line, a local Carriage Return and Line Feed are generated; the cursor moves to the beginning of the next line.	A Carriage Return and Line Feed are not generated at the end of each line. The cursor remains in and overwrites column 80.
D	Line/Page	The terminal is set to transfer a line at a time from display memory, an unprotected field in format mode, or a record from the tape cartridge.	Transfers the entire contents of display memory (a "page"), all unprotected fields in format mode, or a file from the tape cartridge.
E	Paper Tape Mode	When the READ key is pressed with the AUTO LF down, each tape record begins with an LF if the AUTO LF key is down and is ended with a CR.	Each tape record is terminated by CR(LF).
F	(Not Used)		
G	Block Transfer Handshake	No effect.	No effect.
H	Inhibit DC2	No effect.	No effect.
J	Auto Terminate	No effect.	When the ENTER key is pressed a non-displaying terminator is placed after cursor position.
K	Clear Terminator	No effect.	Clear terminator caused by strapping option J above.
L	Self Test Inhibit	No effect.	Self Test function is inhibited. Pressing TEST key or issuing ESC z has no effect. TAPE TEST and DATA COMM SELF TEST functions are not affected.
M	Reverse Sense of INSERT and DELETE CHAR with Wrap.	No effect.	Reverses control function of INSERT CHAR and DELETE CHAR keys (i.e., when key is pressed, line wrap around is in effect without having to press CNTL key. When either key is pressed while pressing CNTL, normal insert character and delete character functions are in effect.)
N	Escape Code Transfer To Printer	No effect.	Escape codes relating to the display (e.g., display enhancements, alternate character sets, format mode, fields, etc.) are sent to printer if it is selected as a destination device.
P,Q	Compatibility Mode	These switches set the terminal to be compatible with Tektronix control commands when initialized (power on or full reset). Refer to Section III for additional information on Compatibility Mode.  P-closed, Q-closed    Normal operation P-closed, Q-open    Unscaled Compatibility Mode P-open, Q-closed    Scaled Compatibility Mode P-open, Q-open    Normal operation	
R <sup>1</sup>	Data Set Ready	No effect.	Provides an internal Data Set Ready (CC) signal to the terminal. (Used in applications with the HP 30037A Asynchronous Repeater, and the Group Poll feature.)



Table 7-5. Keyboard Interface Straps for Block Operation Using 13260C or 13260D Communications Accessory (Continued)

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
S <sup>1</sup>	Space Compression	Space characters are sent normally.	Space characters are compressed.
T,U	Output Block Size	T      U	<b>BLOCK SIZE (BYTES)</b>
		C      C	1/2 Data Comm Buffer (refer to switches J16, J17 on multipoint PCA). 250 max 500 max 1000 max
		O      C C      O O      O	
		C = closed, O = open	
V <sup>1</sup>	Synch Characters	Asynchronous operation without SYN characters.	SYN characters are inserted during Asynchronous operation.
W	Data Comm Self Test	Enables DATA COMM SELF TEST from either the keyboard or escape sequence.	Disables DATA COMM SELF TEST. If self test is attempted (by either the keyboard or escape sequence), the test will be aborted and ERROR 0 will appear on the display.
X	Data Speed Select	Holds data speed signal low (CH = off).	Sets data speed signal high (CH = on).
Y	Transmit Indicator	Lights TRANSMIT indicator on keyboard when terminal is communicating with the computer.	Lights TRANSMIT indicator on keyboard when Data Set Ready (CC) is on, and it goes out when CC goes off.
Z <sup>1</sup>	Transparency	No effect.	Causes all data sent from the terminal to be transparent.

<sup>1</sup> For backward compatibility strapping, refer to tables C-4 and C-5 in Appendix C.

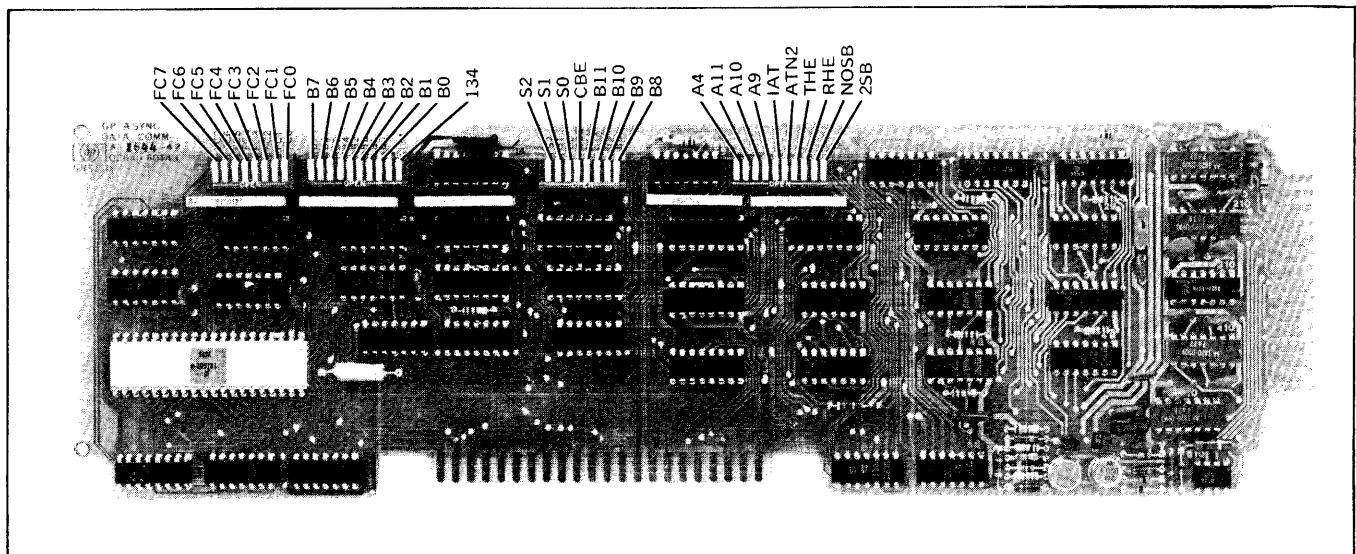


Figure 7-18. Extended Asynchronous Communications PCA Strapping Options

Table 7-6. Extended Asynchronous Communications Interface Strapping Options

STRAP	STRAPPING OPTION	DESCRIPTION																																				
FC0 thru FC7	(Not Used)	(This switch should always be open.)																																				
B0 thru B7	Custom Baud Rate Select	The switches are set to the binary equivalent of a number determined by the formula: $\text{INT} \left( \frac{153600}{\text{baud rate}} \right) - 1$ (See example in figure 5-10.)																																				
134	134.5 Baud	(This switch should always be open.)																																				
S0 thru S2	Transmit Baud Rate	<table border="1"> <thead> <tr> <th colspan="3">SWITCH SETTING</th> <th>TRANSMIT BAUD RATE</th> </tr> <tr> <th>S0</th> <th>S1</th> <th>S2</th> <th></th> </tr> </thead> <tbody> <tr> <td>O</td> <td>O</td> <td>O</td> <td>Transmit baud rate = receive baud rate.</td> </tr> <tr> <td>O</td> <td>C</td> <td>C</td> <td>110</td> </tr> <tr> <td>C</td> <td>O</td> <td>C</td> <td>150</td> </tr> <tr> <td>O</td> <td>O</td> <td>C</td> <td>300</td> </tr> <tr> <td>C</td> <td>C</td> <td>O</td> <td>1200</td> </tr> <tr> <td>O</td> <td>C</td> <td>O</td> <td>2400</td> </tr> <tr> <td>C</td> <td>C</td> <td>C</td> <td>Custom</td> </tr> </tbody> </table>	SWITCH SETTING			TRANSMIT BAUD RATE	S0	S1	S2		O	O	O	Transmit baud rate = receive baud rate.	O	C	C	110	C	O	C	150	O	O	C	300	C	C	O	1200	O	C	O	2400	C	C	C	Custom
		SWITCH SETTING			TRANSMIT BAUD RATE																																	
S0	S1	S2																																				
O	O	O	Transmit baud rate = receive baud rate.																																			
O	C	C	110																																			
C	O	C	150																																			
O	O	C	300																																			
C	C	O	1200																																			
O	C	O	2400																																			
C	C	C	Custom																																			
O = open, C = closed																																						
CBE	Custom Baud	<b>Closed:</b> Enables custom receive baud rates. (The keyboard BAUD RATE switch must be set to EXT.) <b>Open:</b> Receive baud rate is set by keyboard BAUD RATE switch.																																				
B8 thru B11	Custom Baud Rate Select	The switches are set to the binary equivalent of a number determined by the formula: $\text{INT} \left( \frac{153600}{\text{baud rate}} \right) - 1$																																				
A4,A9 thru A11	Module Address	Provides PCA address so that firmware can address the PCA. These switches should always be set to 10, (A4 open, A9 thru A11 closed).																																				
IAT	Inhibit Attention	(This switch must be closed when receive handshake is used.)																																				
ATN2	Enable Attention Two	(This switch should always be open.)																																				
THE	Transmit Handshake Enable	<b>Closed:</b> Permits the associated external device (a or computer) to signal a "busy" condition on CB (Clear to Send) or SCF (Secondary Carrier) control lines and temporarily stop data transmission from the terminal. <b>Open:</b> Transmit Handshake disabled.																																				
RHE	Receive Handshake Enable	<b>Closed:</b> Permits the terminal to signal a "busy" condition on the CD (Data Terminal Ready) control line and temporarily stop data transmission from the associated external device (a computer). <b>Open:</b> Receive Handshake Disabled.																																				
NOSB	SCF Inhibit	<b>Closed:</b> Inhibits RS232 SCF (Secondary Carrier) control line. <b>Open:</b> Enables RS232 SCF (Secondary Carrier) control lines.																																				
2SB	Stop Bit Select	Selects the number of stop bits to be appended to the data bits during transmission. <b>Closed:</b> Selects 2 stop bits. <b>Open:</b> Selects 1 stop bit. NOTE: Selecting 110 baud automatically appends 2 stop bits.																																				

Table 7-7. Asynchronous Multipoint Communications Interface Strapping Options

STRAP	STRAPPING OPTION	DESCRIPTION		
J10 thru J14	Device ID	Selects device ID code (0-27) which identifies one terminal from another on a particular communication line. For example: to set an ID code of 6, set switches J14 through J10 to 00110 respectively. (See "Device I.D. Number" and "Configuration Procedure" in Section V.) 0 = closed, 1 = open		
J15 <sup>1</sup>	(not used)	(should be set to OPEN.)		
J16, J17	Input Buffer Size	<b>J17</b>	<b>J16</b>	<b>BUFFER SIZE</b>
		C	C	500 bytes
		C	O	1000 bytes
		O	C	2000 bytes
		O	O	4000 bytes
C = closed, O = open				
J00 thru J04	Group ID	Selects group ID code (0-27) which identifies the communications line that the terminal is on. For example: to set an ID code of 20, set switches J04 thru J00 to 10100 respectively. (See "Device I.D. Number" and "Configuration Procedure" in Section V.) 0 = closed, 1 = open		
J05 <sup>1</sup>	Extended Text Mode	<b>Open:</b> Enable Extended Text features. See "Extended Text Mode", Section V. <b>Closed:</b> Disable Extended Text features.		
J06	BCC (Block Check Character)	Determines which type of parity check will be used for an entire block of data in Block Mode. <b>Closed = 0:</b> LRC (longitudinal redundancy check) <b>Open = 1:</b> CRC — 16 (cyclic redundancy check)		
J07	Code Select	Selects data character and control character code format. <b>Open = 1:</b> EBCDIC <b>Closed = 0:</b> ASCII		
INT	Firmware Interrupt	This switch should always be open.		
PL0 thru PL6	Poll Bits	These switches should always be open.		
A4, A9 thru A11	Module Address	Provides PCA address so that the firmware can address the PCA. These switches should always be set to 7 (A4 closed, A9 thru A11 open).		
-12	13232T Accessory Power	<b>Closed:</b> Provides -12 volts for operation of relays in the 13232T Power Protect Multipoint Cable. <b>Open:</b> No power supplied.		
2SB	Stop Bit Select	Selects the number of stop bits to be appended to the data bits during transmission. <b>Open:</b> Selects 2 stop bits. <b>Closed:</b> Selects 1 stop bit (normal position)		

<sup>1</sup> For backward compatibility strapping, refer to tables C-4 and C-5 in Appendix C.

Table 7-8. Synchronous Multipoint Communications Interface Strapping Options

STRAP	STRAPPING OPTION	DESCRIPTION		
J10 thru J14	Device ID	Selects device ID code (0-27) which identifies one terminal from another on a particular communication line. For example: to set an ID code of 6, set switches J14 thru J10 to 00110 respectively. (See "Device I.D. Number" and "Configuration Procedure" in Section V.) 0 = closed, 1 = open		
J15 <sup>1</sup>	(not used)	(should be set to OPEN.)		
J16, J17	Input Buffer Size	<b>J17</b>	<b>J16</b>	<b>BUFFER SIZE</b>
		C	C	500 bytes
		C	O	1000 bytes
		O	C	2000 bytes
		O	O	4000 bytes
C = closed, O = open				
J00 thru J04	Group ID	Selects group ID code (0-27) which identifies the communications line that the terminal is on. For example: to set an ID code of 20, set switches J04 thru J00 to 10100 respectively. (See "Device I.D. Number" and "Configuration Procedure" in Section V.) 0 = closed, 1 = open		
J05 <sup>1</sup>	Extended Text Mode	<b>Open:</b> Enable Extended Text features. See "Extended Text Mode", Section V. <b>Closed:</b> Disable Extended Text features.		
J06	BCC (Block Check Character)	Determines which type of block check will be used for an entire block of data. <b>Closed = 0:</b> LRC (longitudinal redundancy check) <b>Open = 1:</b> CRC — 16 (cyclic redundancy check)		
J07	Code Select	Selects data character and control character code format. <b>Open = 1:</b> EBCDIC <b>Closed = 0:</b> ASCII		
-12	13232T Accessory Power	<b>Closed:</b> Provides -12 volts for operation of relays in the 13232T Power Protect Multipoint Cable. <b>Open:</b> No power supplied.		
A4, A9 thru A11	Module Address	Provides PCA address so that the firmware can address the PCA. These switches should always be set to 7 (A4 closed, A9 thru A11 open).		
RCLK	Receive Data Clock	When the terminal is directly connected to a computer (no modem) by using the 13232U Modem Bypass Cable, the PCA can provide the receive data clock (DD) by closing this switch. (This applies only to the first terminal in the multipoint chain.)  Normally, this switch is open. One of the transmit data clock switches (see below) must be selected for this function.		
2400 4800 9600	Transmit Data Clock	Usually, the modem or computer provides both the receive (DD) and transmit (DB) data clocks for timing the data transfers. If the modem requires a terminal-supplied transmit clock (DA), select the appropriate rate for that modem.  If using the 13232U Modem Bypass Cable, select the desired rate.  9600 must be closed for the DATA COMM SELF TEST.		
<div style="border: 1px solid black; padding: 5px; display: inline-block; margin: 10px auto; width: 100px;"> <b>CAUTION</b> </div> <p>Close only one switch, otherwise damage to the PCA may result.</p>				

<sup>1</sup> For backward compatibility strapping, refer to tables C-4 and C-5 in Appendix C.

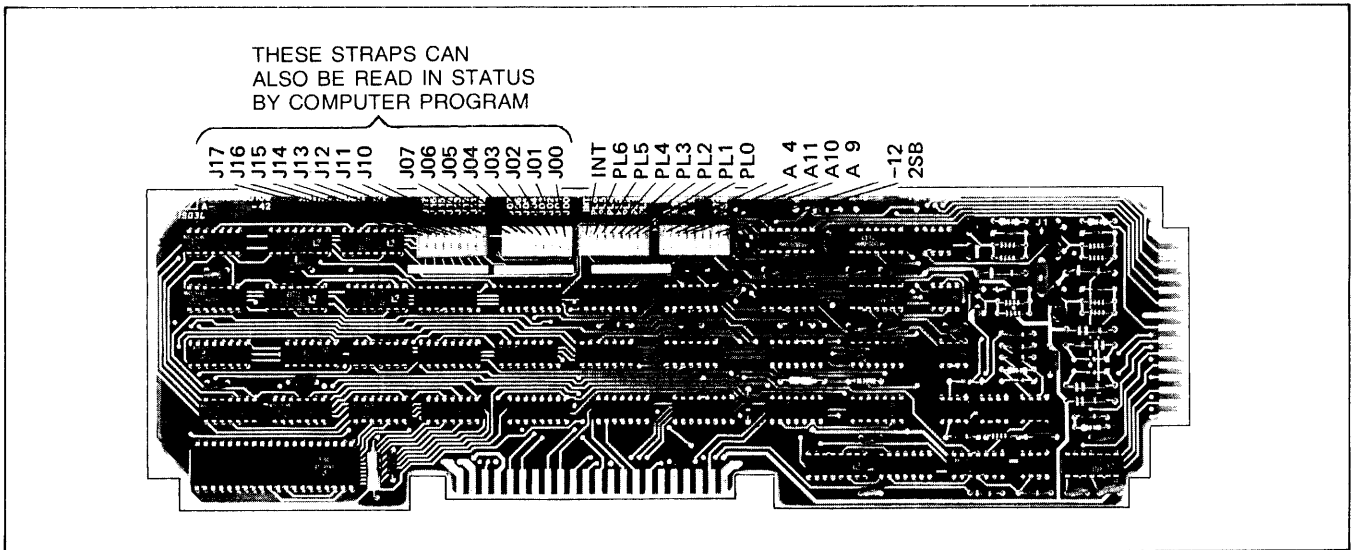


Figure 7-19. Asynchronous Multipoint Communications PCA Strapping Options

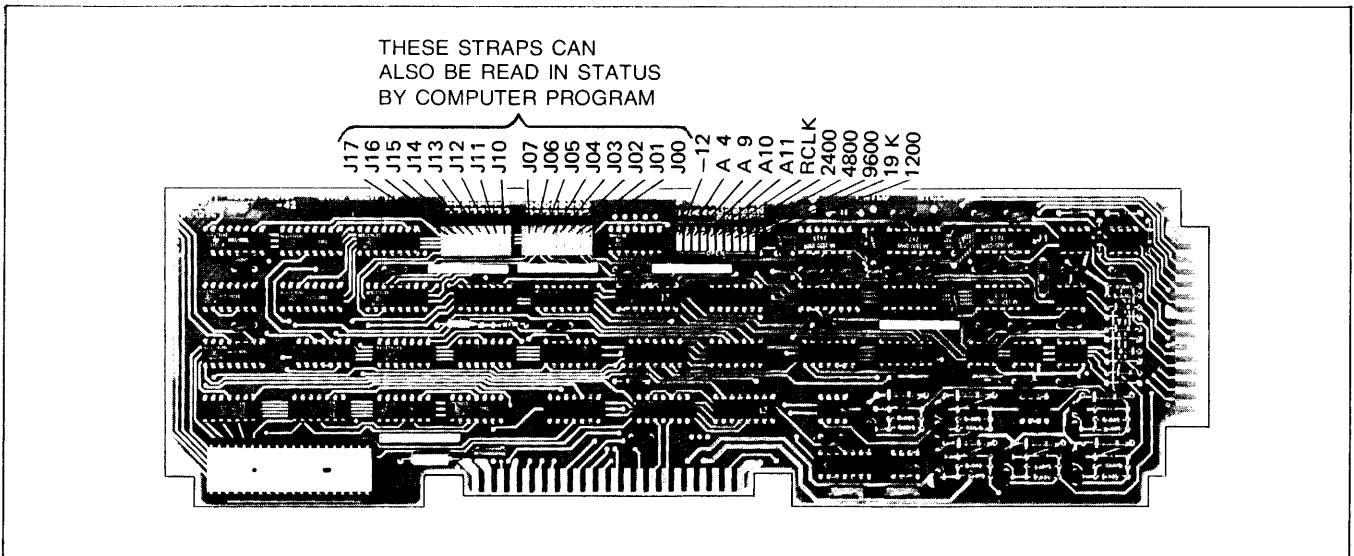


Figure 7-20. Synchronous Multipoint Communications PCA Strapping Options

## DATA COMMUNICATIONS CABLING

### Interface Signals

The RS232 signals available on each of the communication interfaces are listed in table 7-9. This information can be used to verify interface capability or to fabricate special interface cables. Refer to "Fabricating Your Own Data Communications Cable" for additional cabling information.

### Logic Levels

Table 7-10 gives the logic levels of signals used by the 13260 series data communications accessories.

### Cable Types

There are five cable types that are available for use in multipoint networks. These cables are described in table 7-11.

Table 7-9. EIA RS232C and CCITT V24 Interface Data and Control Signals

CONNECTOR					CIRCUIT		DESCRIPTION	MODEM		GND	DATA	CON- TROL	TIMING
R S 2 3 2	P2 13260				R S 2 3 2	C C I V T 2 T 4		TO	FROM				
	A	B	C	D									
—	A	A	A	A	AA	—	Protective Ground			X			
7	H	H	H	H	AB	102	Signal Ground/Common Return			X			
2	B	B	B	B	BA	103	Transmitted Data	X			X		
3	C	C	C	C	BB	104	Received Data		X		X		
4	D	D	D	D	CA	105	Request to Send	X				X	
5	E	E	E	E	CB	106	Clear to Send		X			X	
6	F	F	F	F	CC	107	Data Set Ready		X			X	
20	P	P	P	P	CD	108.2	Data Terminal Ready	X				X	
22	—	—	14	14	CE	125	Ring Indicator		X			X	
8	J	J	J	J	CF	109	Received Line Signal Detector		X			X	
—	—	—	—	—	CG	110	Signal Quality Detector	X				X	
23	—	R	R	R	CH	111	Data Rate Selector (DTE Source)	X				X	
—	—	—	—	—	CI	112	Data Rate Selector (DCE Source)		X			X	
24	—	—	S	S	DA	113	Transmitter Timing (DTE Source)	X					X
15	—	—	—	12	DB	114	Transmitter Timing (DCE Source)		X				X
17	—	—	—	13	DD	115	Receiver Timing		X				X
—	—	—	—	—	SBA	118	Secondary Transmitted Data	X			X		
—	—	—	—	—	SBB	119	Secondary Received Data		X		X		
19	M	M	M	M	SCA	120	Secondary Request to Send	X				X	
—	—	—	—	—	SCB	121	Secondary Clear to Send		X			X	
12	N	N	N	N	SCF	122	Secondary Received Line Detector		X			X	

Table 7-10. Data Communications Signal Levels

<b>DATA:</b>		
Name	Space	Mark
Logic	0	1
Voltage	> +3V but < +25V	< -3V but > -25V
<b>CONTROL:</b>		
	ON (true)	OFF (false)
<b>CLOCK SIGNALS:</b>		
	0 = ground	1 = +5V

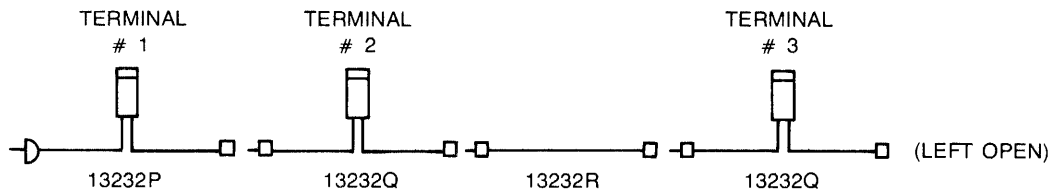
Table 7-11. Multipoint Data Communication Cables

ACCESSORY NO.	DESCRIPTION	SYMBOL
13232N	Male RS232C Modem to terminal cable	
13232P	RS232 Modem to terminal plus multipoint	
13232Q	Multipoint terminal to terminal	
13232R	Multipoint extender	
13232T	Power Down protect (same symbol as Q)	
13232U	Modem Bypass	

where:

- = RS232 (Modem) connector (female)
- = Terminal hood connector
- = Multipoint connector (female)
- = Male connector i.e., or

For example, to connect 3 terminals you could use the following configuration:



### Point-To-Point Communications Cabling

Figures 7-21 through 7-23 show the cable connections and signals used by the 13260A/B/C/D and 13250A/B accessories.

### Multipoint Communications Cabling

Figures 7-24 and 7-25 show the cable connections and signals used by the 13260C/D accessories in the multipoint configuration.

### Power Down Protect Cabling

Figure 7-26 and 7-27 show the cable configuration and effects of signal switching during power-down.

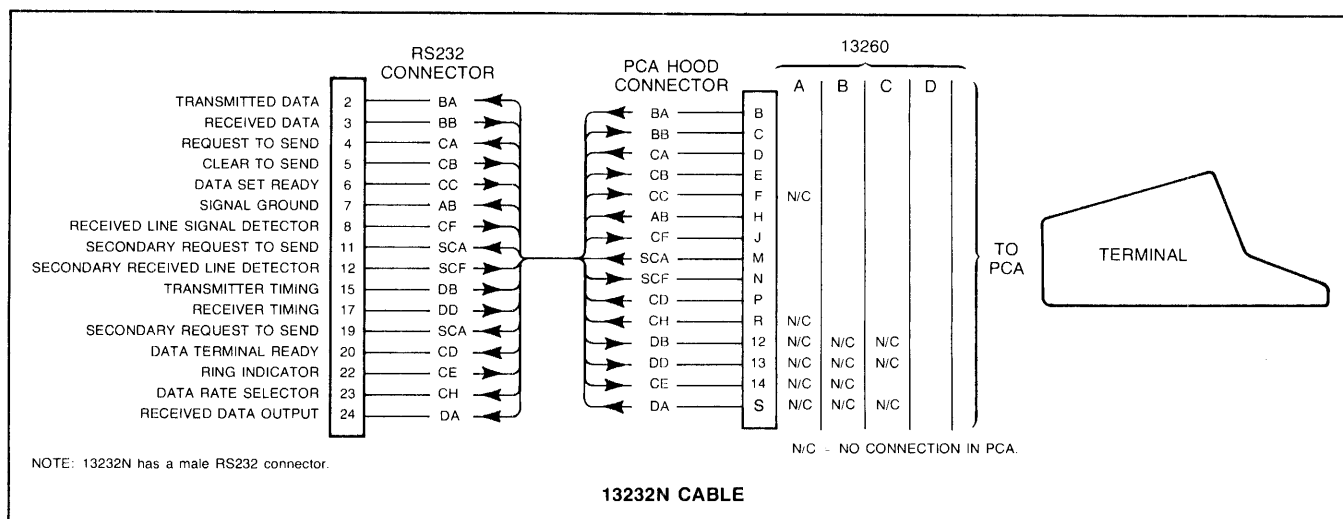


Figure 7-21. Point-to-Point Communications Cabling

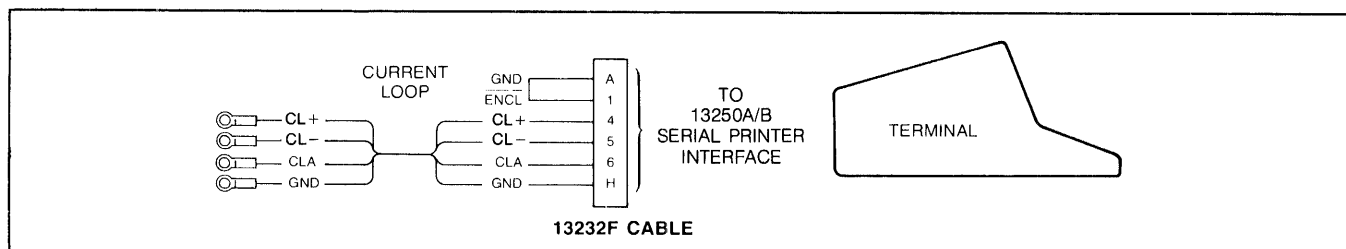


Figure 7-22. Current Loop Cabling

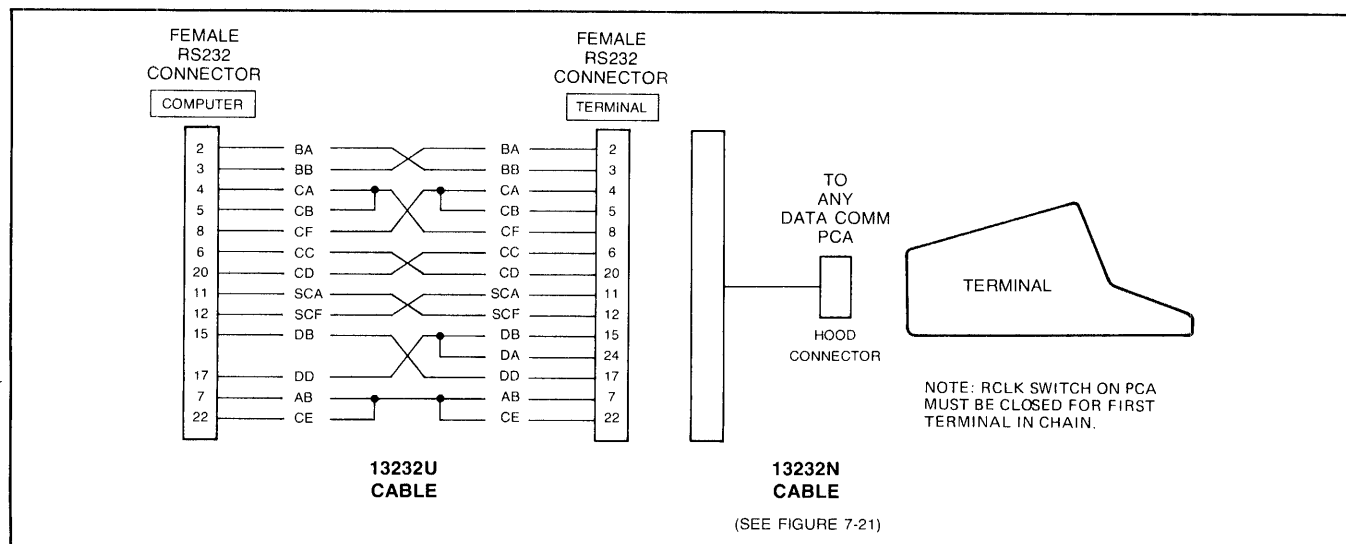


Figure 7-23. Modem By-Pass Cabling



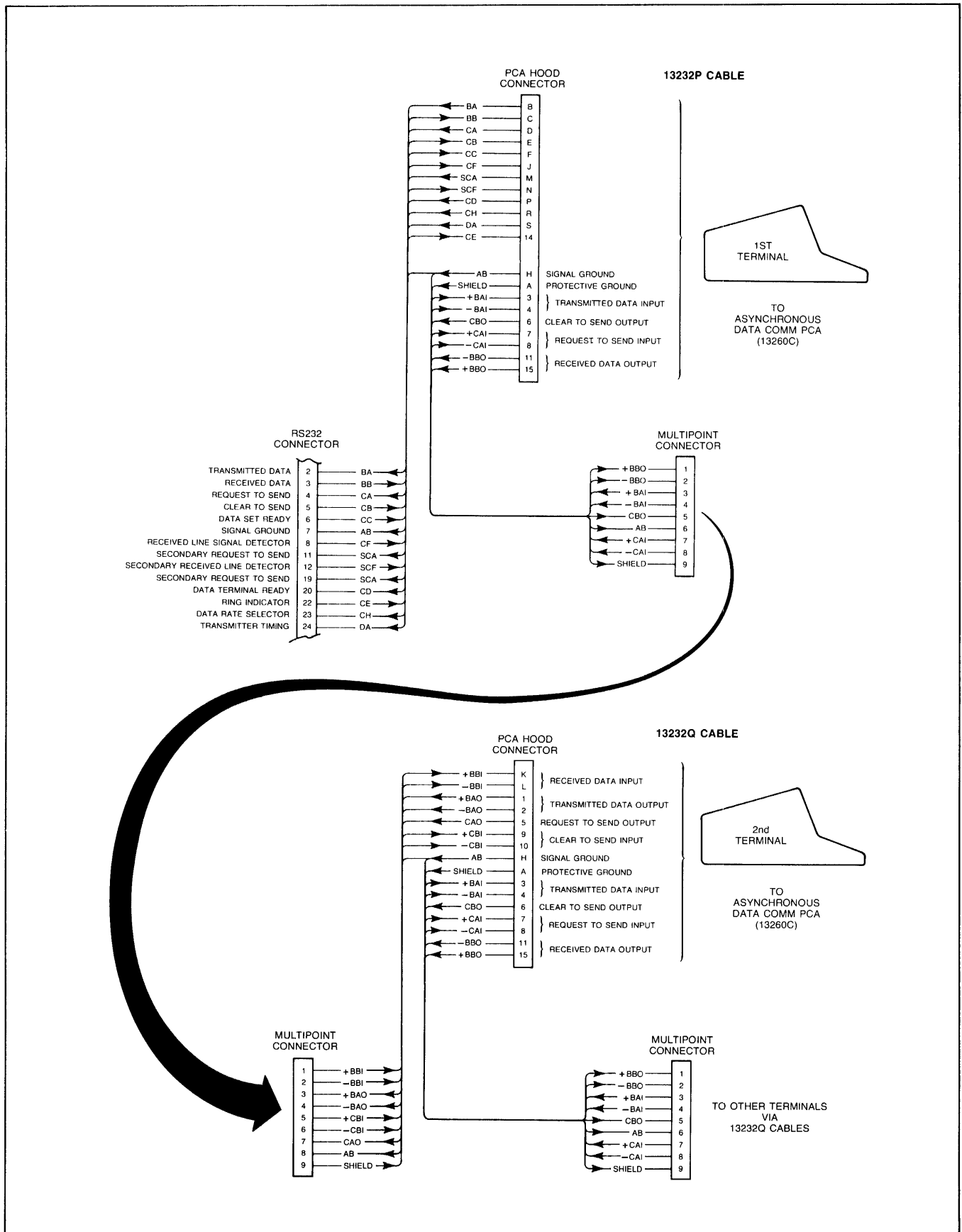


Figure 7-24. Asynchronous Multipoint Cabling

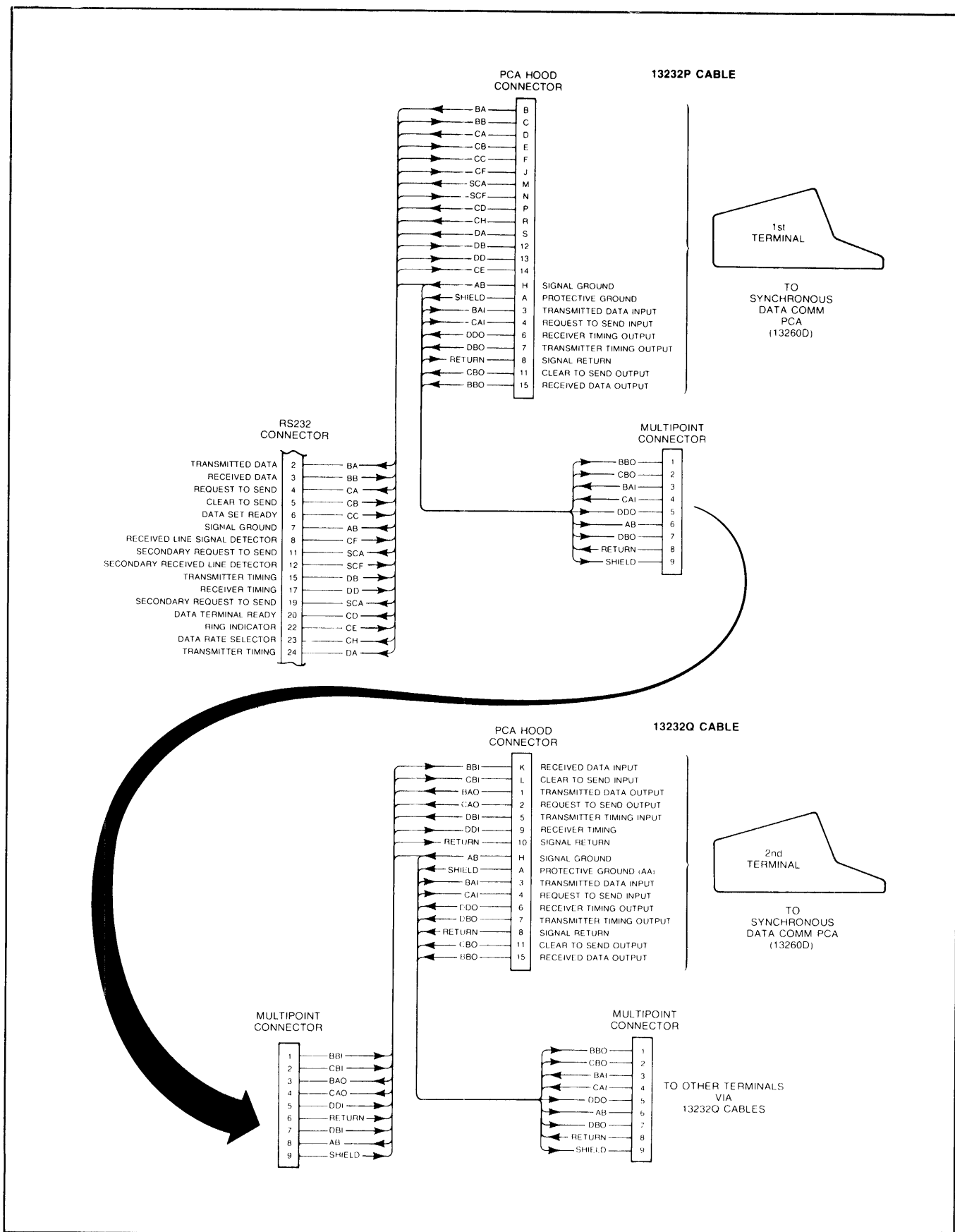


Figure 7-25. Synchronous Multipoint Cabling

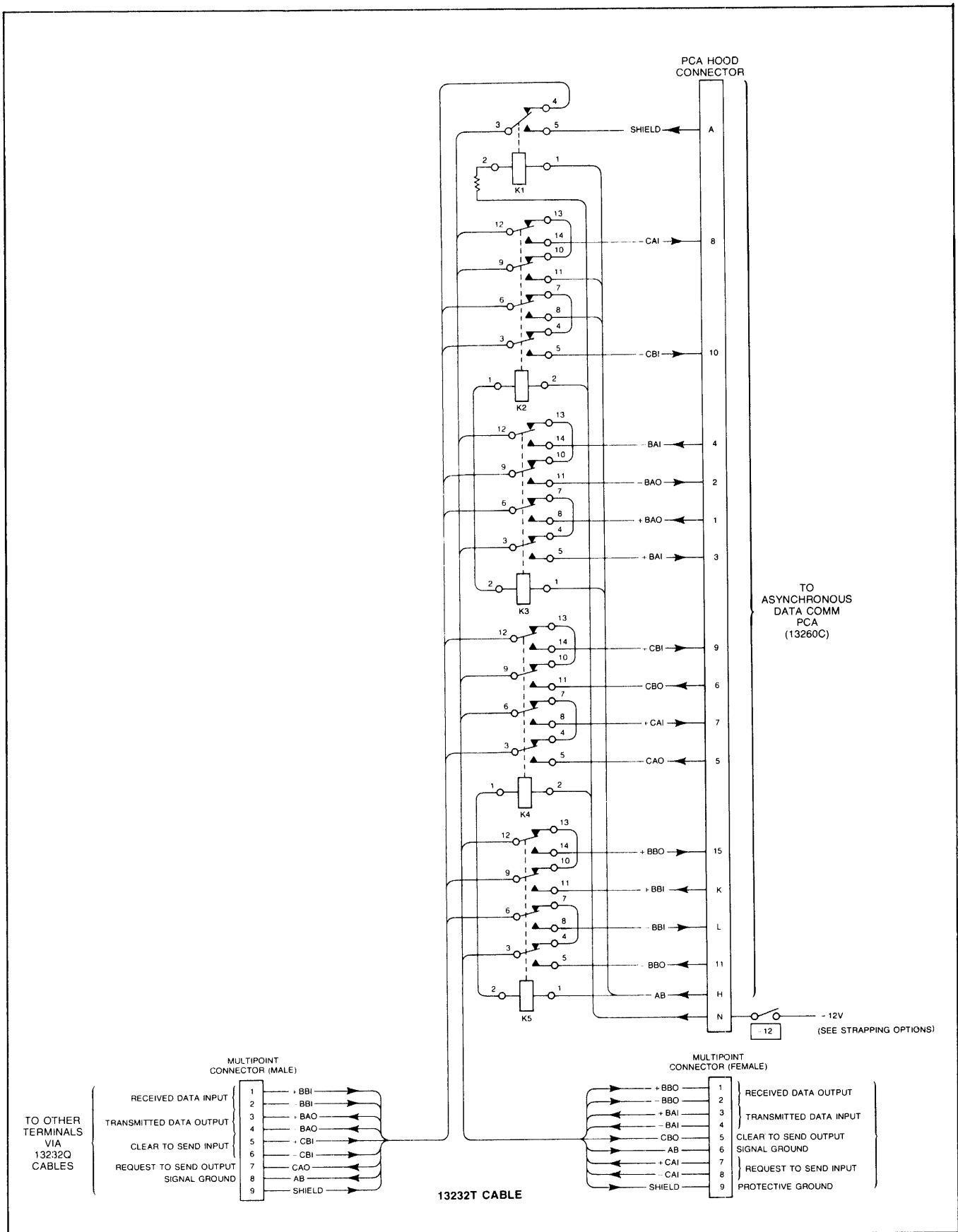


Figure 7-26. Power-Down-Protect Cabling for Asynchronous Multipoint Configuration

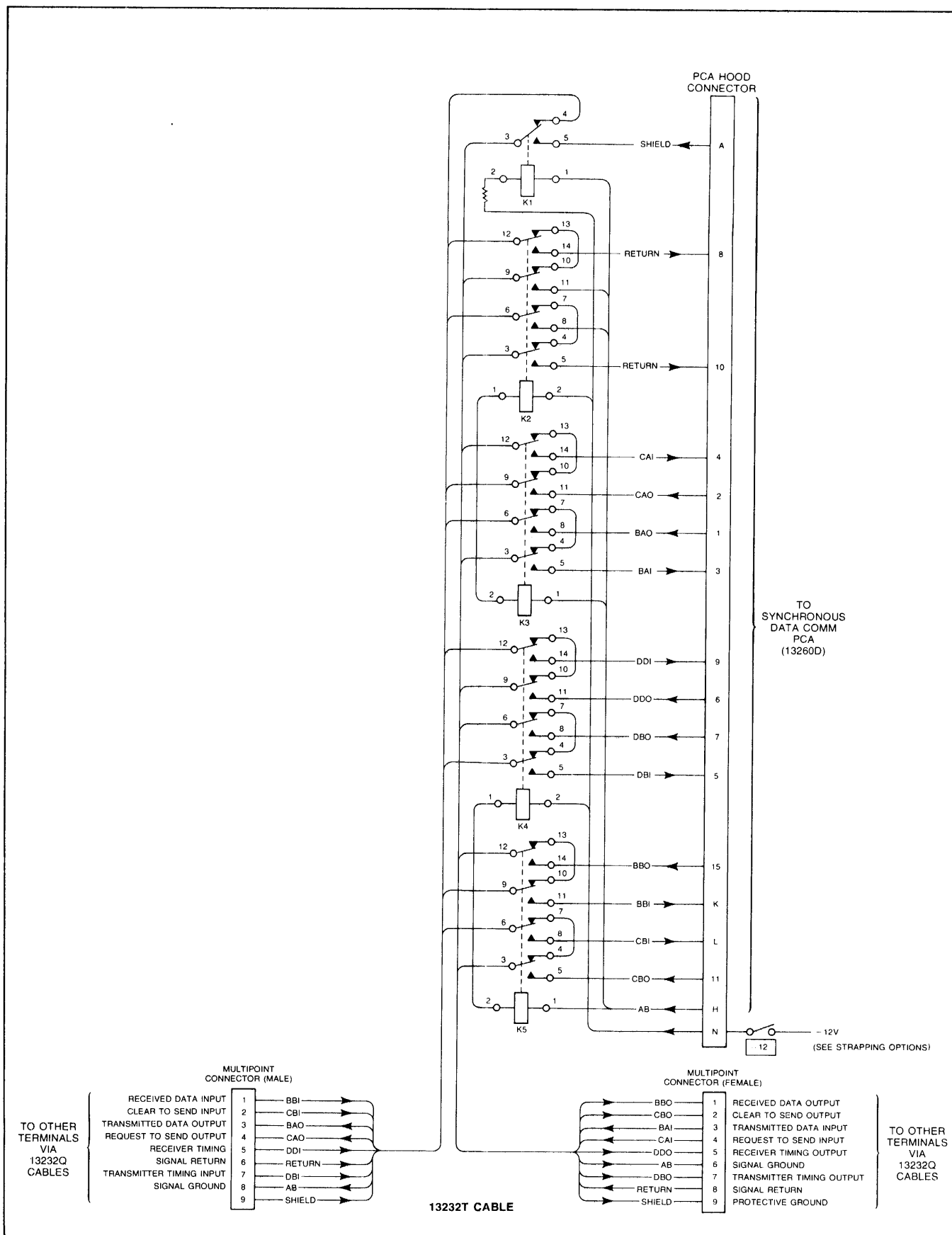


Figure 7-27. Power-Down-Protect Cabling for Synchronous Multipoint Configuration

## Fabricating Your Own Data Communications Cable

PCA hood connectors, RS232C connectors, multipoint connectors, and cables are available should you need to fabricate your own data communications cable. Part numbers of the items are given in table 7-12.

Figures 7-28 through 7-30 show the details of assembling each type of connector. Table 7-13 lists the interface signals on each of the data communications PCA's. Also, the illustrations of the HP cables (figures 7-21 through 7-27) may be used as a guide.

There are maximum length limitations on each type of cable. The following may be used as a guide for length considerations.

### Maximum Distances:

Modem/Computer to first terminal: 50 feet (RS232-C standard)

Modem/Computer to terminal: 1000 feet (current loop on 13260B)

Terminal to terminal —

Note: Maximum total distance 16,000 feet.

Asynchronous (13260C) @ 300 to 9600 bits per second: up to 2000 feet between terminals with up to 32 terminals per line.

Synchronous (13260D) (2000 feet maximum between terminals):

Terminals/ Line	Bits/ Sec.		
	2400	4800	9600
4	2000 ft	2000 ft	2000 ft
8	2000 ft	2000 ft	1200 ft
16	2000 ft	1200 ft	480 ft
32	1200 ft	480 ft	120 ft

Table 7-12. Parts for Fabricating Your Custom Data Communications Cable

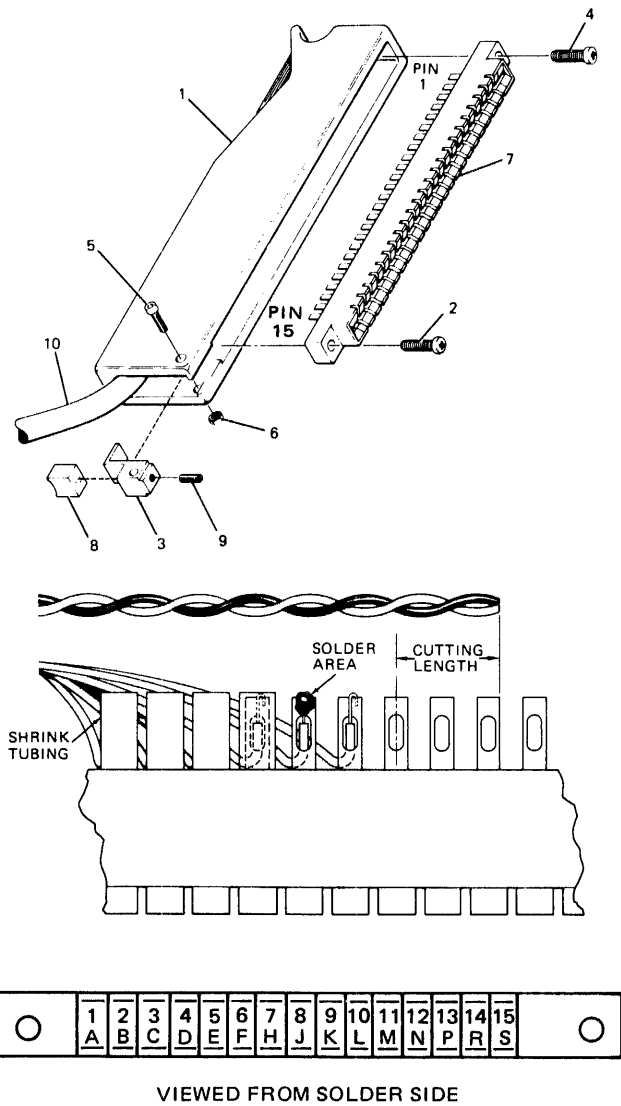
ITEM	HP PART NO.	ALTERNATE SOURCE	DESCRIPTION
RS232 Connector	5061-2405		(See figure 7-29.)
PCA Hood Connector	5061-1340		(See figure 7-28.)
Multipoint Connector	5061-2401		(See figure 7-30.)
PCA Hood to RS232 Connector Cable	8120-1903 or 8120-1930		26 AWG (or greater) Low Voltage Computer Cable.
Multipoint Cable	8120-2305	Brand Rex POSS4P22	22 AWG, 4 twisted pairs, overall shield, 75 ohm differential mode characteristic impedance.

Note: All connectors include contacts.

Table 7-13. 13260 Series Data Communications PCA Signal Names

P2 PIN	SIGNAL NAMES			
	13260A	13260B	13260C	13260D
1	(no connection)	ENCL (see note)	+BAO	BAO
2	(no connection)	INI	-BAO	CAO
3	(no connection)	CL+ 12	+BAI	BAI
4	(no connection)	CL+ (see note)	-BAI	CAI
5	(no connection)	CL- (see note)	CAO	DBI
6	(no connection)	CLA (see note)	CBO	DDO
7	(no connection)	CLP	+CAI	DBO
8	(no connection)	INO	-CAI	RET-D
9	(no connection)	PON	+CBI	DDI
10	(no connection)	ISB	-CBI	RET-U
11	(no connection)	XECL	-BBO	CBO
12	(no connection)	TTY IN	(no connection)	DB
13	(no connection)	+5V	(no connection)	DB
14	(no connection)	CE	CE	CE
15	TEST	TEST	+BBO	BBO
A	AB	GND	AA	AA
B	BA	BA	BA	BA
C	BB	BB	BB	BB
D	CA	CA	CA	CA
E	CB	CB	CB	CB
F	(no connection)	CC	CC	CC
H	AB	AB (see note)	AB (GND)	AB (GND)
J	CF	CF	CF	CF
K	X8OUT	X8OUT	+BBI	BBI
L	X16OUT	X16OUT	-BBI	CBI
M	SCA	SCA	SCA	SCA
N	SCF	SCF	SCF	SCF
P	CD	CD	CD	CD
R	(no connection)	CH	CH	CH
S	X16IN	X16IN	(no connection)	DA

NOTE: Used in current loop mode.

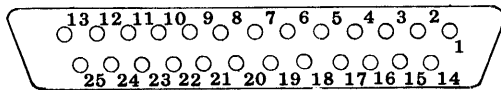
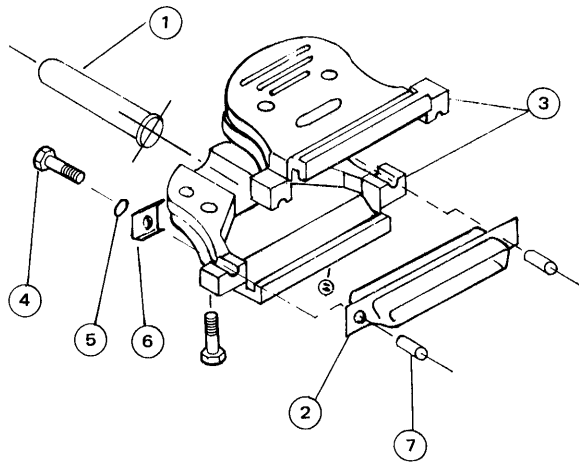


VIEWED FROM SOLDER SIDE

Assembly Procedure:

1. ( ) Insert approximately 10 inches of cable (item 10) into the connector hood (item 1).
2. ( ) Strip the outer jacket of the cable back 5 inches.
3. ( ) Remove approximately 1/4-inch of insulation from each signal wire.
4. ( ) Starting at the end of the 30-pin connector (item 7) nearest pins S and 15, solder the signal wires to the appropriate pins on the connector, and insulate each pin with tubing as shown at left.
5. ( ) Install the 30-pin connector in the connector hood using the two self-tapping screws (items 2 and 4).
6. ( ) Install the cable clamp (items 3 and 8), and tighten it in place with the screw and nut (items 5 and 6).
7. ( ) Tighten the cable clamp on the cable with the setscrew (item 9).

Figure 7-28. Assembling the PCA Hood Connector



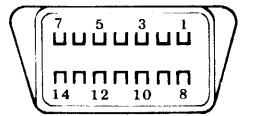
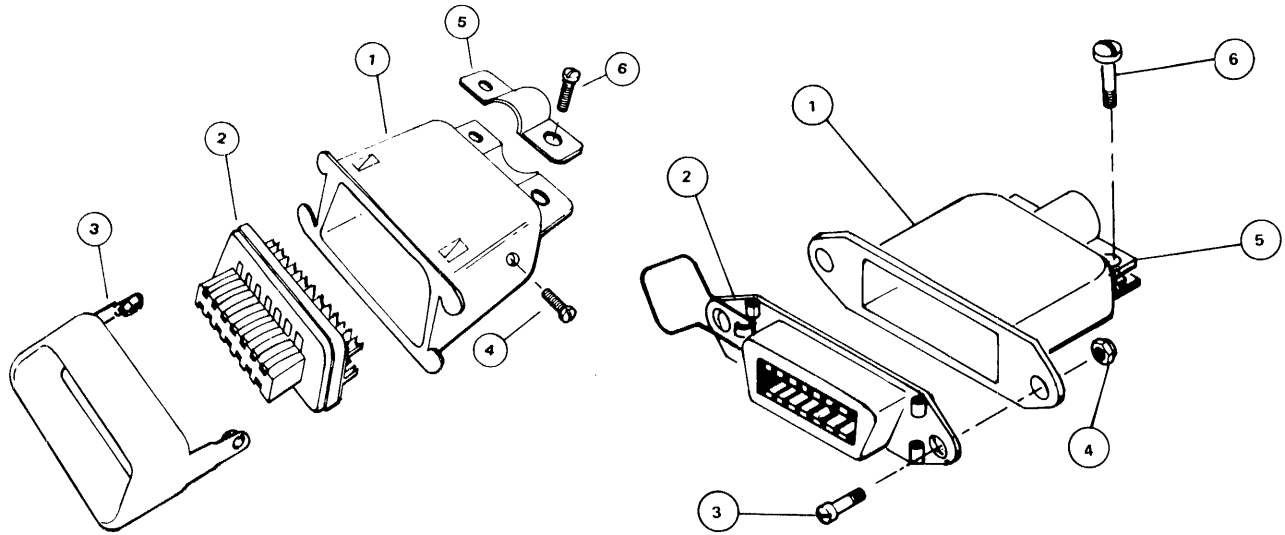
VIEWED FROM SOLDER SIDE

Assembly Procedure:

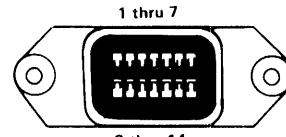
- |   |  |
|---|--|
| <ol style="list-style-type: none"> <li>1. ( ) Slide rubber bushing (item 1) over end of cable, leaving about 6 inches of cable end exposed for wire stripping, etc.</li> <li>2. ( ) Strip back the cable jacket 1-inch.</li> <li>3. ( ) Clip the unused conductor wires to the edge of the cable jacket.</li> <li>4. ( ) Remove 1/4-inch of insulation from the ends of the conductor wires to be used.</li> <li>5. ( ) Solder the conductor wires onto the contacts of the contact assembly (item 2). (Select either the male or female contact assembly provided for your particular application.)</li> </ol> | <ol style="list-style-type: none"> <li>6. ( ) Slide the rubber bushing to the end of the cable such that the rubber bushing flange is flush with the stripped end of the cable jacket.</li> <li>7. ( ) Assemble the two halves of the connector (item 3) onto the contact assembly (item 2). (Use the screws and nuts provided.)</li> <li>8. ( ) Mount the two screws, threaded spacers, and other hardware (items 4 thru 7) onto the contact assembly.</li> </ol> |
|---|--|

Figure 7-29. Assembling the RS232C Connector



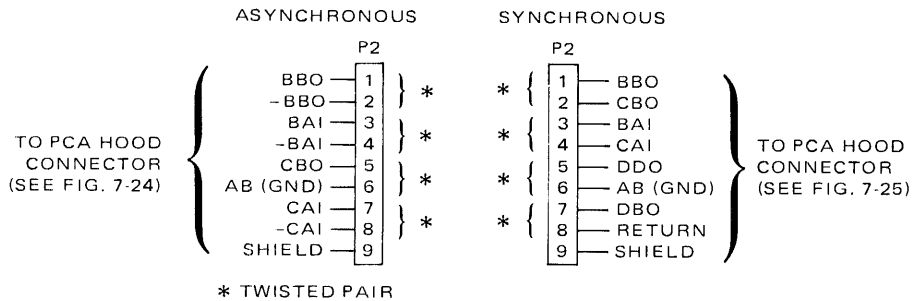


MULTIPOINT CONNECTOR P2  
(VIEWED FROM SOLDER SIDE)



FEMALE  
MULTIPOINT CONNECTOR  
(VIEWED FROM SOLDER SIDE)

MULTIPOINT CONNECTOR CABLING



Assembly Procedure:

1. ( ) Insert cable through the outer housing (item 1).
2. ( ) Strip back the cable jacket 1-inch.
3. ( ) Clip any unused conductor wires to the edge of the cable jacket.
4. ( ) Remove 1/4-inch of insulation from the ends of the conductor wires to be used.
5. ( ) Solder the conductor wires onto the contacts of the contact assembly (item 2).
6. ( ) Assemble the multipoint connector by sliding the inner housing (item 3) over the contact assembly (item 2). Slide the outer housing over items 2 and 3 until the screw holes are aligned. Secure the entire assembly with the two screws (item 4).
7. ( ) Mount the cable clamp (item 5), and secure with the two screws (item 6).

Figure 7-30. Assembling the Multipoint Connector

## SELF-TEST

The terminal tests itself. Should you suspect a malfunction while operating the terminal, you can perform the SELF-TEST function to checkout the terminal. Also, after installing any accessory, the terminal's self-test function should be performed to insure that the terminal is functioning properly. There are three types of self-test, each testing a specific function of the terminal. (Testing the HP-IB, if installed, is contained in Section VIII, "HP-IB Configurations.")

### Basic Self-Test

Pressing **TAPE TEST**, checks out the terminal, except for the cartridge tape units (if installed) and the data communications. The following is performed when the **TAPE TEST** key is pressed (also see the flowchart in figure 7-32):

#### NOTE

The test pattern cannot be recorded because of imbedded Record Separators (RS).

- The light-emitting diodes (indicators) on the keyboard are turned on briefly as an indication that the power supply and microprocessor board are functioning.
- A checksum test is done on the read-only memory (ROM). This verifies that the firmware is working properly. An error here causes a ROM ERROR message to be displayed. (See flowchart, figure 7-32.)
- A checkerboard test is performed on the random access memory. An error here causes a RAM ERROR message to be displayed. (See flowchart, figure 7-32.)

- A graphics test is performed. The test checks both the vector generating function as well as the graphics memory. This is done by drawing a series of horizontal and vertical lines. In addition, a series of graphics characters is drawn at different character sizes and zoom settings. The final checkerboard pattern is panned. If the graphics memory test fails, a message indicating the failing graphics memory component is displayed (see the flowchart in figure 7-32).
- The bell is beeped indicating success up to this point.
- The entire character set contained in the terminal is displayed.
- A line of characters, @ABCDEFGHIJKLMNO, is displayed. If the Display Enhancement option is installed, then Underline, Half-Bright, and Blinking will be displayed with Inverse Video in all of the possible Display Enhancement combinations by this line of characters.
- The 14 bytes of status information are displayed. (See Section VI "Status" for an explanation of the status bytes.)

Generally, if the terminal beeps and the display shows a pattern similar to those shown in figure 7-31 then the terminal is functioning properly (only those character sets actually present in the terminal will be displayed in the test pattern and consequently the actual test pattern displayed will be dependent on which features are present in each terminal).

**RESET TERMINAL** must be pressed to resume operation if any error occurred. However, the station's operation will not be reliable if the Self-Test failed.

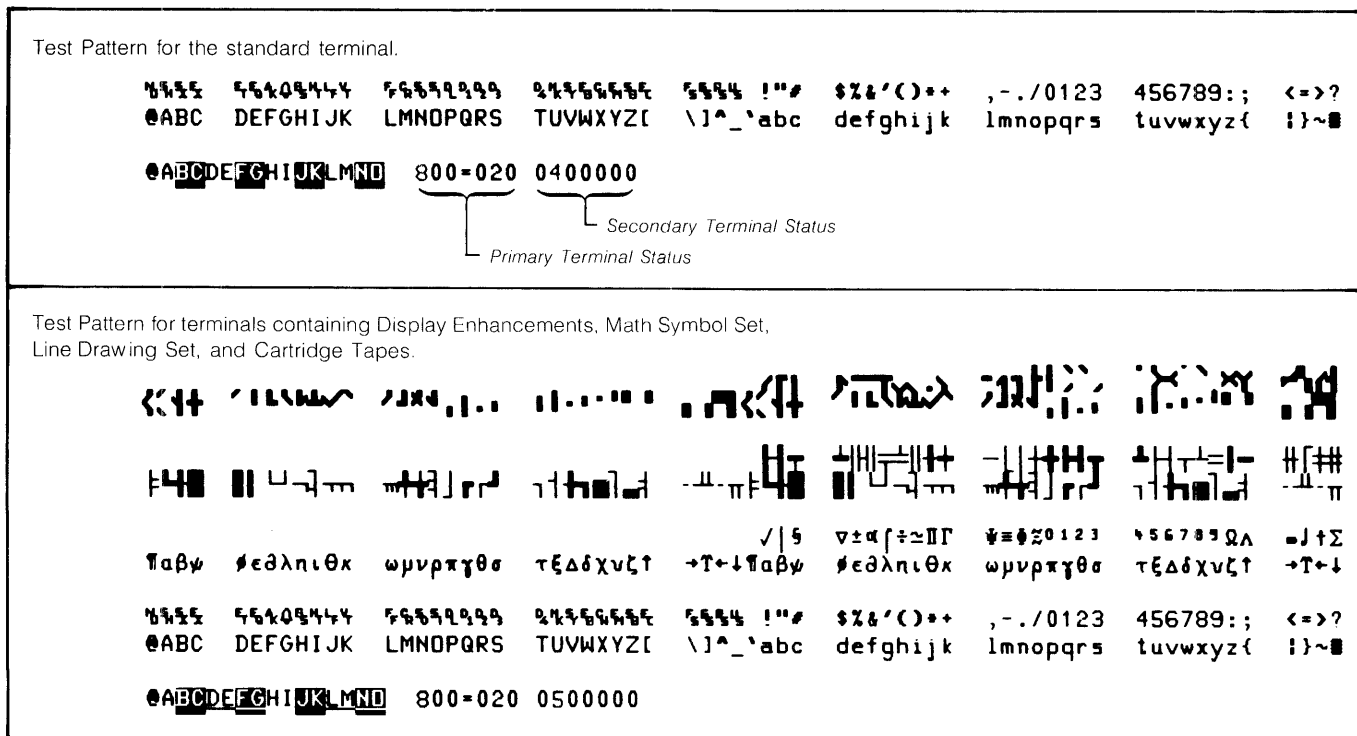


Figure 7-31. Basic Self-Test Patterns

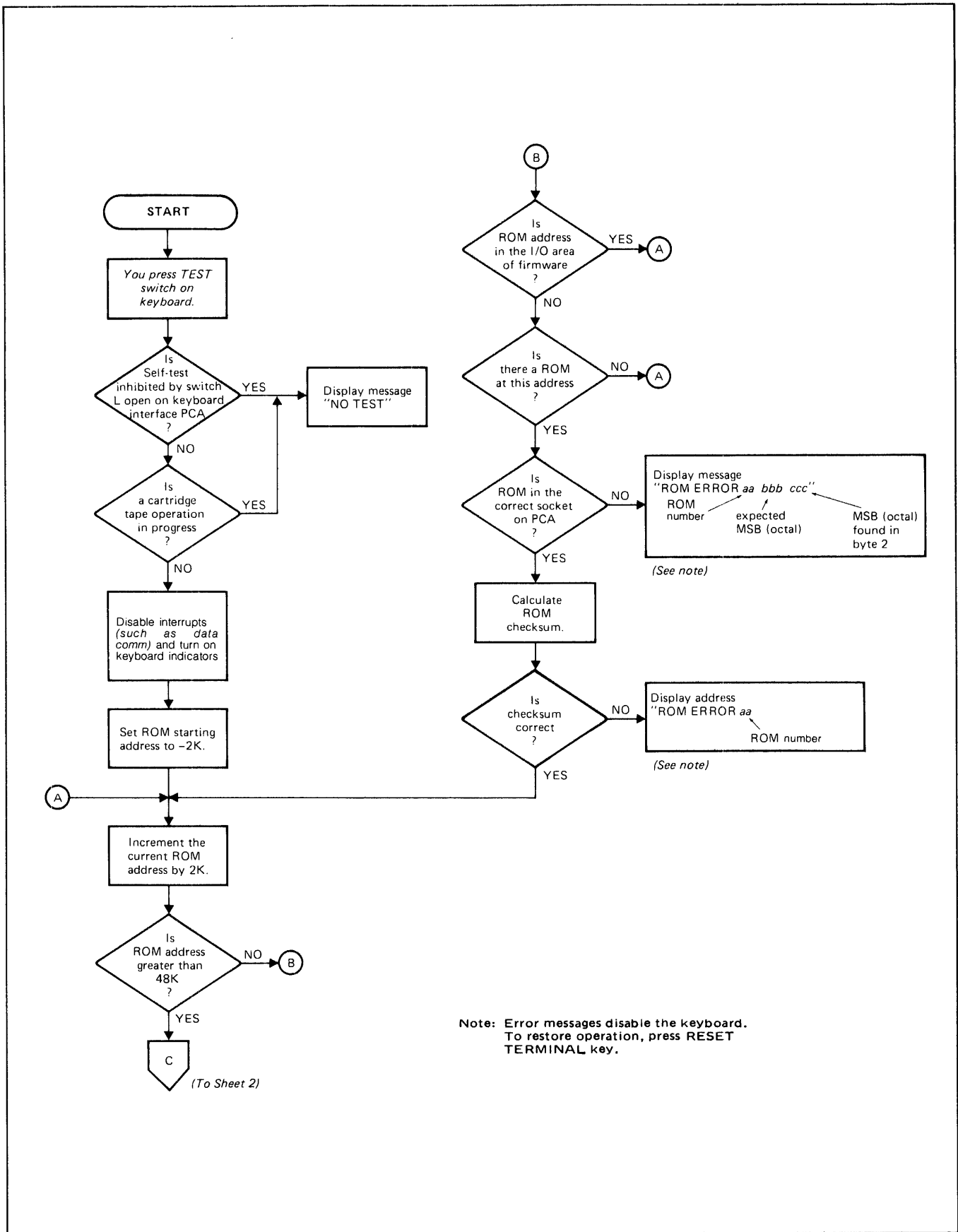


Figure 7-32. Basic Terminal Self-Test Flowchart (Sheet 1 of 3)

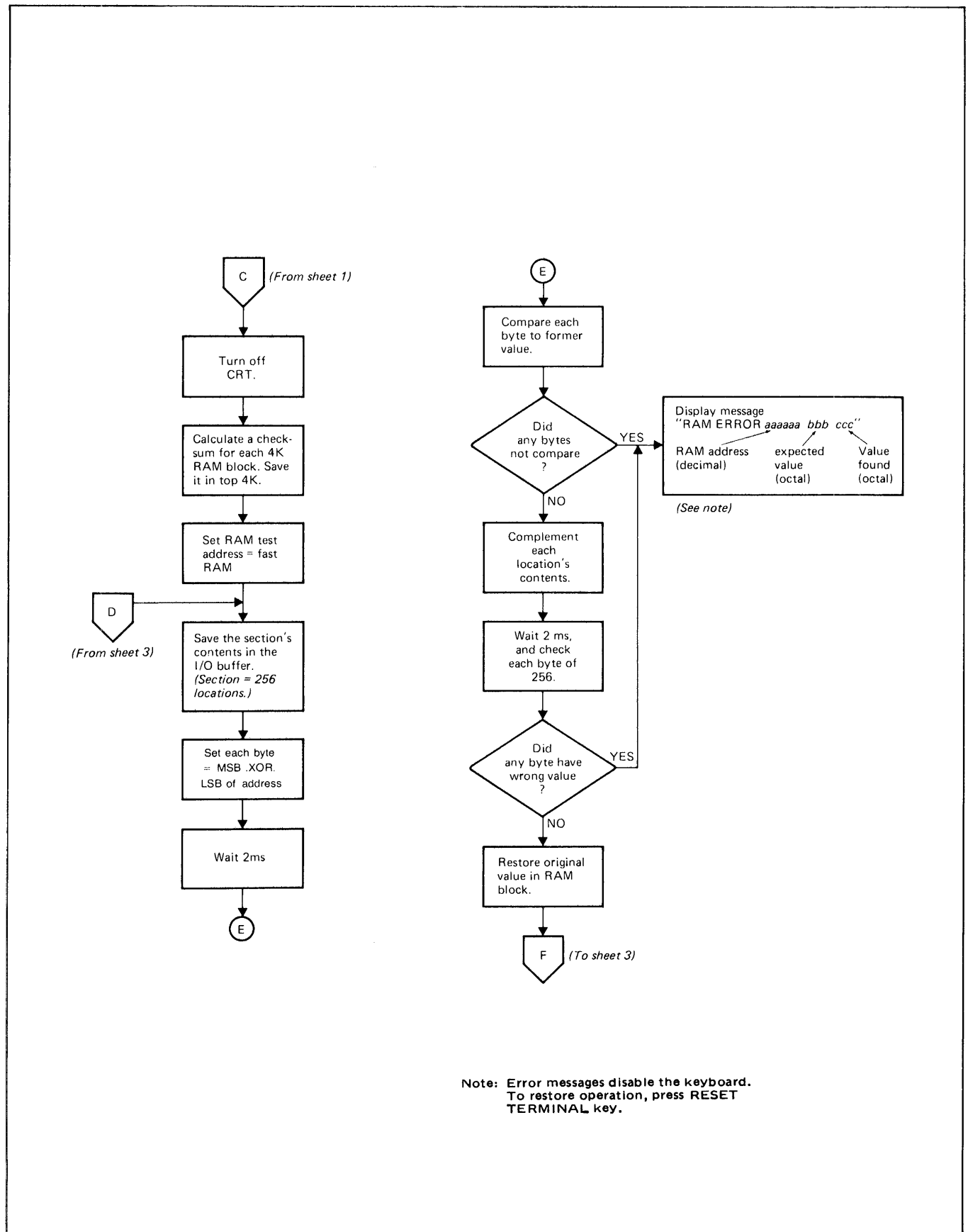


Figure 7-32. Basic Terminal Self-Test Flowchart (Sheet 2 of 3)

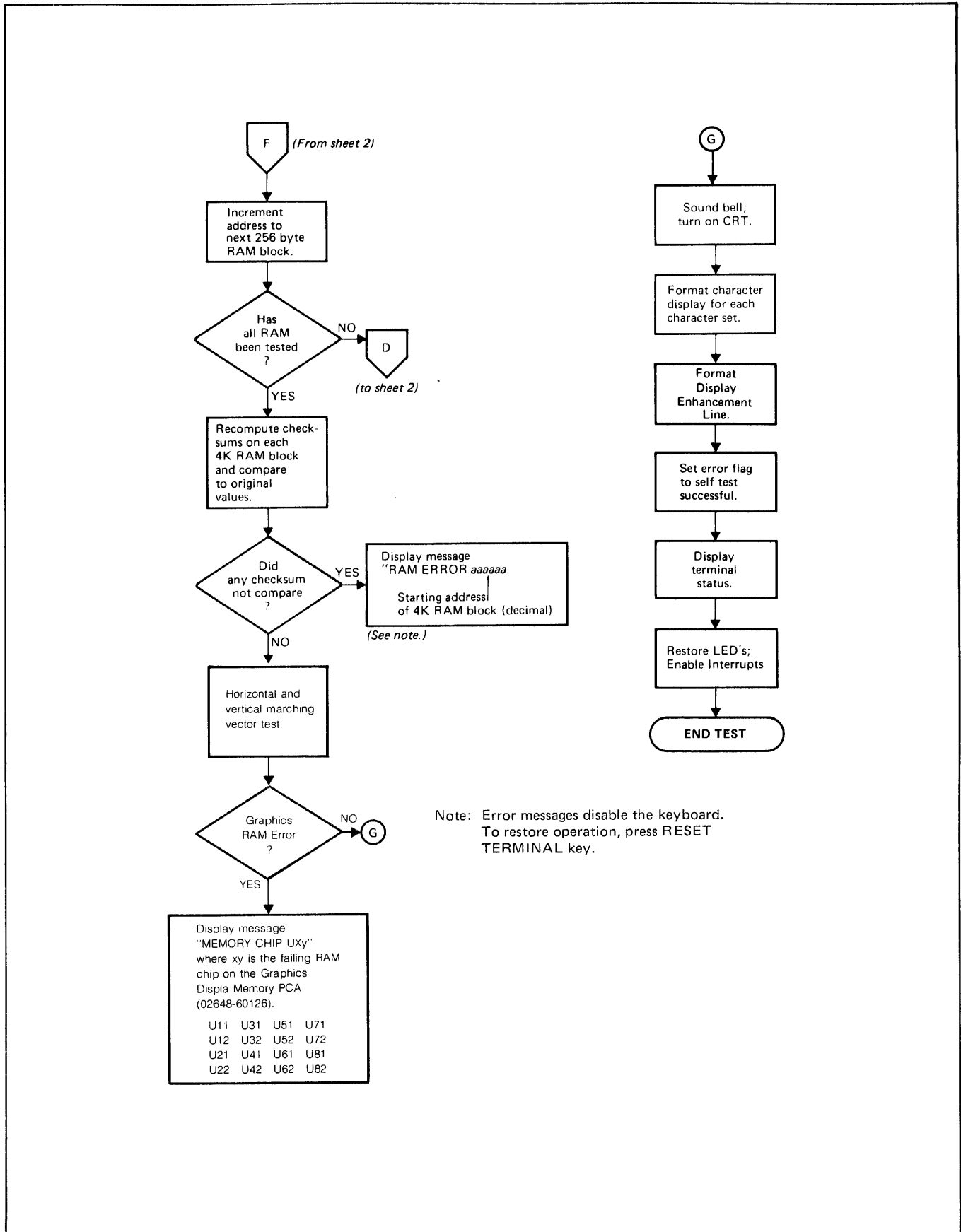


Figure 7-32. Basic Terminal Self-Test Flowchart (Sheet 3 of 3)

## Cartridge Tape Unit Self-Test

### CAUTION

The following self-test is performed with two unprotected tape cartridges. Make sure that any data on these tapes need not be saved.

**FROM THE KEYBOARD.** The following is performed when the **(GREEN)** key is pressed then the **TAPE TEST** key is pressed:

- A test is performed on the left tape unit:
  - A worst case data pattern (“%Z” repeated 128 times to form a 256 character record) is recorded on the tape cartridge.
  - The tape is backspaced over the record to the beginning of the test pattern.
  - The test pattern is read and verified.
  - A file mark is recorded.
- Two basic self-tests are performed as described previously.
- A test is performed on the right tape unit (same as the left tape unit).
- Another basic self-test is performed.

If a fault is detected during the tape transport test, the eject button will be lit on the transport being tested, the test will not proceed any further, and one of the error messages shown below will be displayed.

NO TAPE, RUNOFF, DATA PROTECTED, FAIL  
WRITE FAIL, STALL, or END OF TAPE

These messages are explained in the *User's Manual*.

If a hardware failure has occurred during the self-test, the reliability of the terminal cannot be assured. If any error occurred, press **RESET TERMINAL** to restore normal operation. Try replacing the tape cartridge and running the self-test again to make sure that the error is a hardware malfunction. Servicing procedures are contained in the *Service Manual*.

You may verify that the tapes you record may be used by other terminals as follows:

- Perform the tape transport test.
- Rewind the tapes.
- Exchange tape between the left and right transports.
- Read each tape, and check that a line of “%Z” appears on the screen. If this does not happen, a hardware malfunction may exist in one of the transports.

**FROM COMPUTER.** The tape transports may be tested from your program by coding:

ESC & p 1u 7C (for the left tape transport)  
ESC & p 2u 7C (for the right tape transport)

After the test is performed, the terminal will respond with an “S” CR(LF) if the test was successful or an “F” CR(LF) if the test failed. The status of the tested tape unit may be interrogated to determine the reason for the failure. (See Section VI, “Status”.)

## Data Communications Self-Test

This self-test checks the data communications PCA (13260A, B, C or D) and the associated network cabling. Test connectors are used with the self-test function to provide signal loop-back while the internal diagnostic is being run. A description of the test connectors is provided in table 7-14. To run the self-test, follow the instructions in table 7-15 or 7-16, whichever is applicable. Flowcharts of the self-test are contained in figures 7-33 and 7-34.

Table 7-14. Data Communications Self-Test Connectors


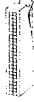

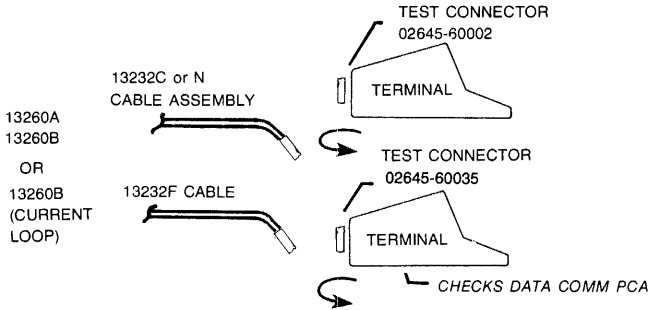
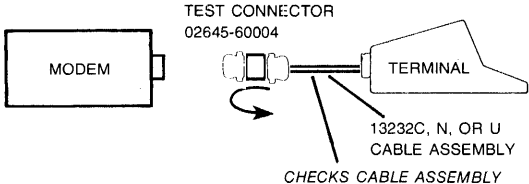
SELF-TEST CONNECTORS	HP PART NO.	USED FOR
	02645-60002	Checks RS232 circuits on 13260A, B, C, D accessory PCAs. This connector is supplied with the terminal. (Does not check multipoint circuits on PCA; use 02645-60004 test connector below.)
	02645-60035	Checks current loop circuits on 13260B accessory PCA. This connector is supplied with the 13232F Current Loop Cable.
	02645-60004	Provides loop-back of RS232 signals at RS232 connector end of cable. Used during self-test of multipoint configurations. This connector is supplied with the 13232P Multipoint/Modem Cable.

Table 7-15. Point-to-Point Data Communications Self-Test Procedure

<p><b>STEP 1.</b></p> <ol style="list-style-type: none"> <li>Ensure power is off, and disconnect cable data communications PCA.</li> <li>Connect PCA Test Connector, part no. 02645-60002, to data communications PCA.</li> <li>Turn on power; set the terminal to REMOTE, FULL DUPLEX, NONE (parity), and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>Refer to data comm self-test flowcharts for diagnosing possible error messages.</li> <li>If operating in current loop, turn power off and use test connector part no. 02645-60003 to connect to the 13260B Data Communications PCA. Turn on power, and type characters on the keyboard. The characters should be echoed back (two characters displayed if the terminal is set for Half Duplex). This verifies proper operation of current loop send and receive circuits.</li> </ol>	
<p><b>STEP 2.</b></p> <ol style="list-style-type: none"> <li>Turn off power, and connect 13232C or N Cable Assembly to 13260A, B, C, or D data communications PCA. (If operating in current loop, connect 13232F cable to 13260B data communications PCA.)</li> <li>Connect RS232 Test Connector, part no. 02645-60004, to RS232 connector on 13232C or N cable.</li> <li>Turn on power, set the terminal to REMOTE, and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>Refer to data comm self-test flowcharts for diagnosing possible error messages.</li> </ol>	
<p><b>NOTE:</b> For terminals without cartridge tapes, use ESC x to perform data comm self-test.</p>	

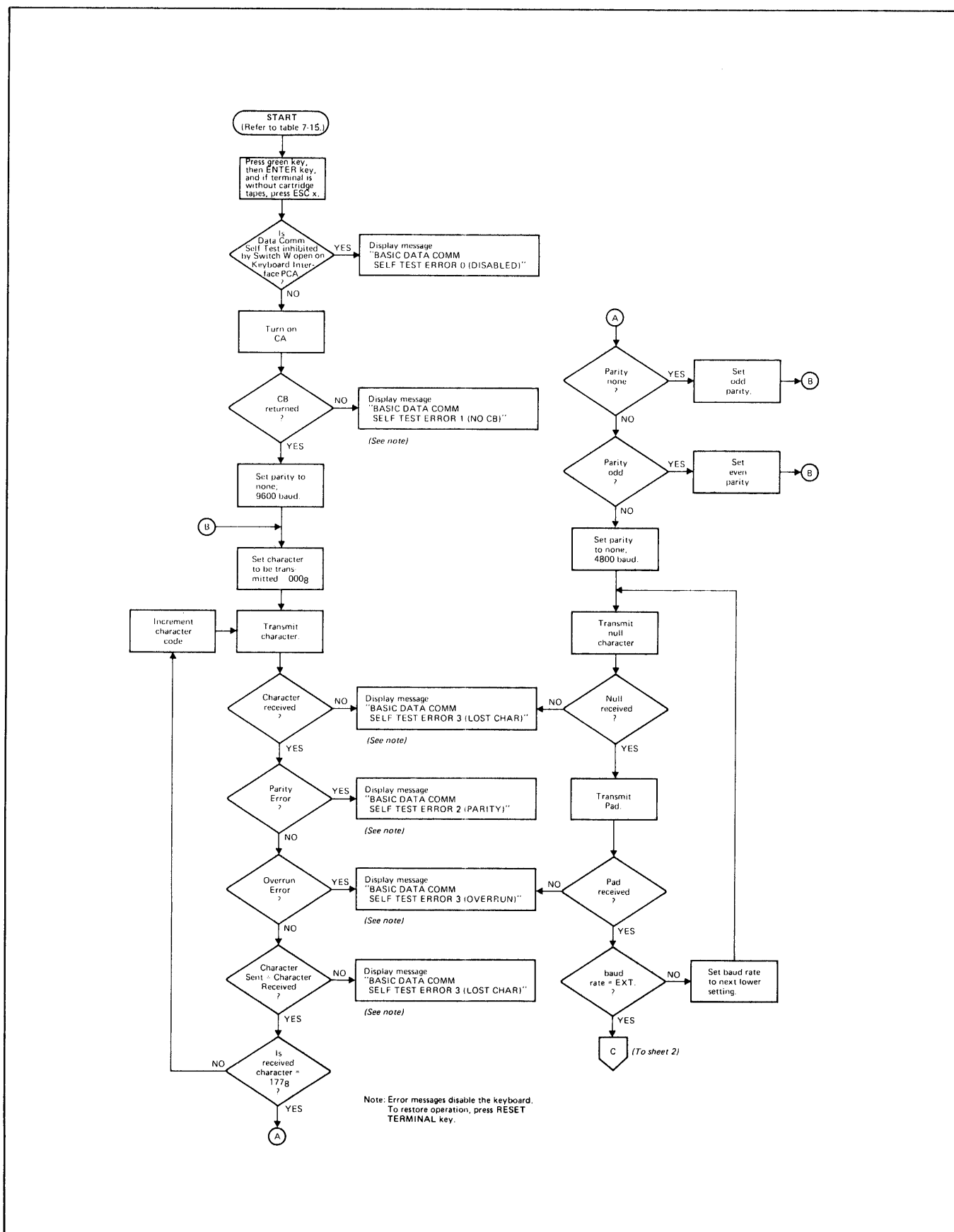
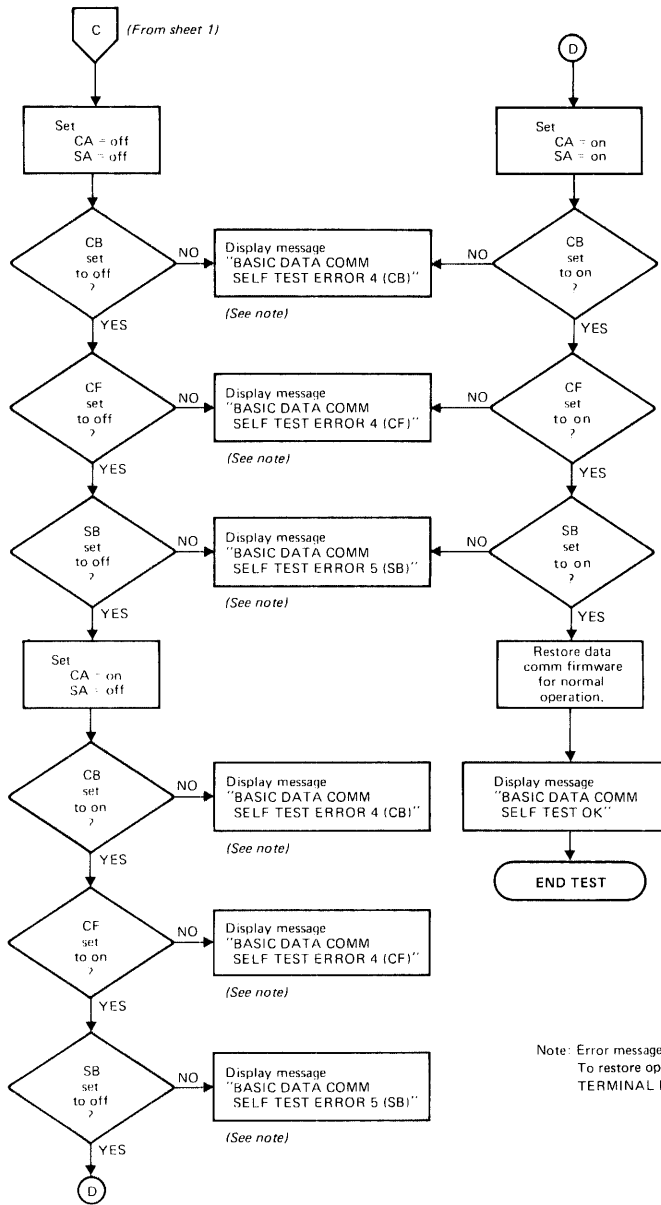


Figure 7-33. Basic Data Comm Self-Test Flowchart (Sheet 1 of 2)



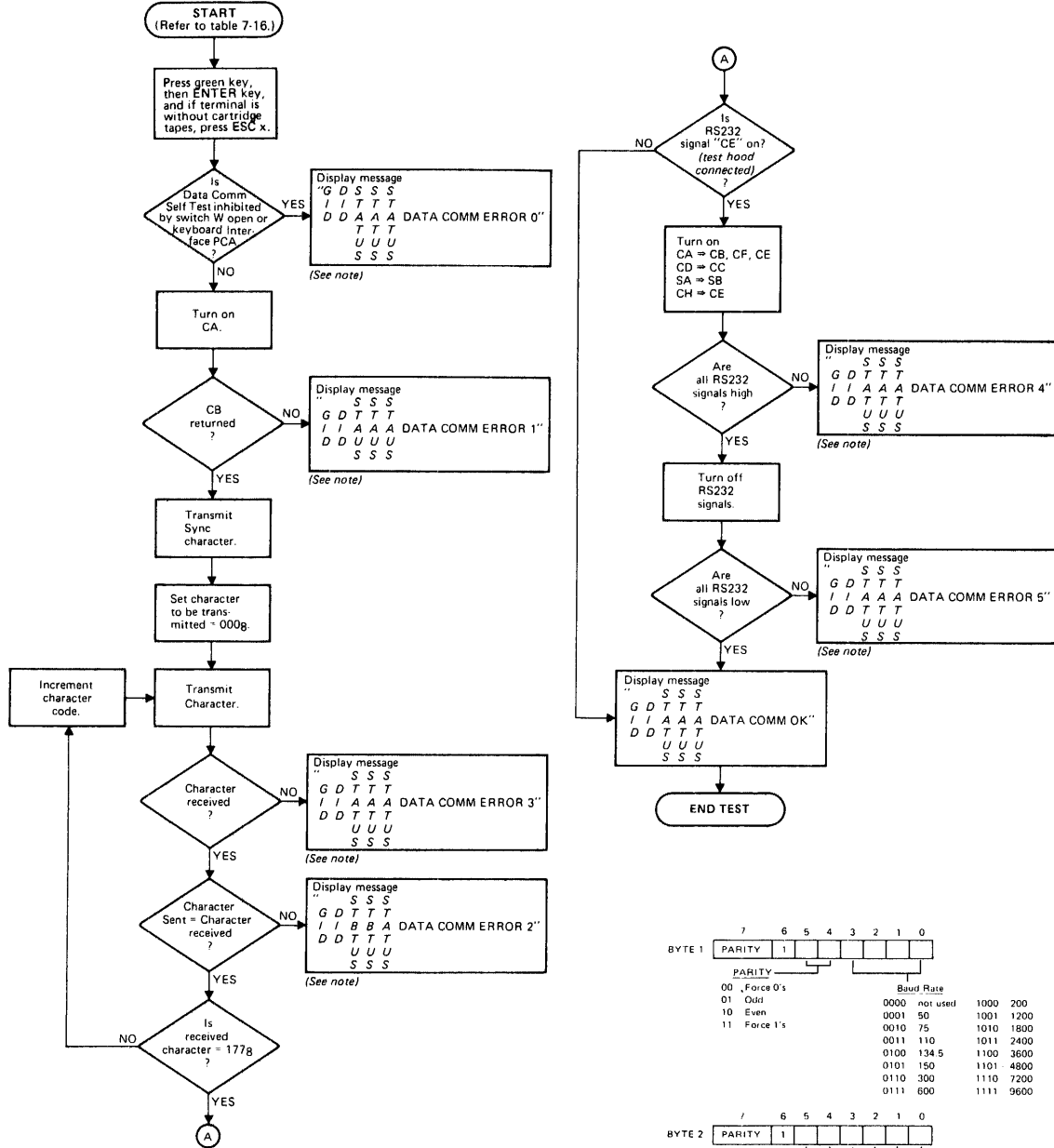


Note: Error messages disable the keyboard.  
To restore operation, press RESET  
TERMINAL key.

Figure 7-33. Basic Data Comm Self-Test Flow chart (Sheet 2 of 2)

Table 7-16. Multipoint Data Communications Self-Test Procedure

<p><b>STEP 1.</b></p> <ol style="list-style-type: none"> <li>Ensure power is off, and disconnect cable from 13260C or D communications PCA.</li> <li>Connect PCA Test Connector, part no. 02645-60002, to 13260C or D data communications PCA.</li> <li>Turn on power; set the terminal to REMOTE, FULL DUPLEX, NONE (parity), and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>Refer to multipoint data comm self-test flowchart for diagnosing possible error messages.</li> </ol>	<p style="text-align: center;">DATA COMM TESTING (MULTIPOINT)</p>
<p><b>STEP 2.</b></p> <ol style="list-style-type: none"> <li>Turn off power, and reconnect cable to 13260C or D data communications PCA.</li> <li>Connect RS232 Test Connector, part no. 02645-60004, to RS232 connector on 13232P cable.</li> <li>Turn on power, and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>Refer to multipoint data comm self-test flowchart for diagnosing possible error messages.</li> </ol>	
<p><b>STEP 3.</b></p> <ol style="list-style-type: none"> <li>Turn off power, and connect 13232P cable to modem.</li> <li>Switch modem to loop-back mode (if possible).</li> <li>Turn on power, and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>If self-test did not pass, the modem may be malfunctioning. Refer to multipoint data comm self-test flowchart for diagnosing possible error messages.</li> </ol>	
<p><b>STEP 4.</b></p> <ol style="list-style-type: none"> <li>Switch modem back to normal operation.</li> <li>Switch far-end modem to loop-back mode (if possible).</li> <li>Turn on power, and press <b>(GREEN)</b>, <b>ENTER</b> (see note).</li> <li>If self-test did not pass, the modem may be malfunctioning. Refer to multipoint data comm self-test flowchart for diagnosing possible error messages.</li> </ol>	
<p>NOTE: For terminals without cartridge tapes, use ESC x to perform data comm self-test.</p>	



**Note 1.** Error messages disable the keyboard. To restore operation, press RESET TERMINAL key.

**Note 2.** Display Message Legend.

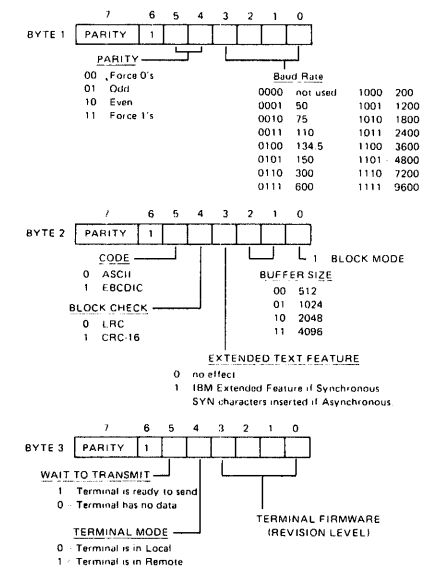
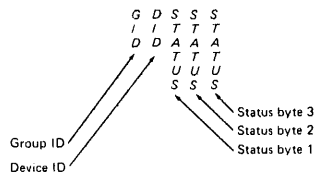


Figure 7-34. Multipoint Data Comm Self-Test Flowchart

## INTRODUCTION

The HP 13296A Shared Peripheral Interface includes the following items.

1. An HP-IB Interface PCA (part no. 02640-60128).
2. A standard 2 meter HP-IB interconnecting cable (figure 8-1). This cable has a double-sided male/female connector on both ends to allow for stacked interconnection of multiple cables.
3. An HP-IB Interface Adapter (part no. 02640-60215). See figure 8-2. This item serves two purposes. First, it provides a means of connecting an HP-IB cable to the interface PCA. Second, it provides you with a means of increasing the maximum combined amount of interconnecting cable permitted in your HP-IB configuration.

HP-IB interconnecting cables are available in three lengths:

- HP 10631A HP-IB Cable, 1 meter (3.3 feet), part no. 8120-1833
- HP 10631B HP-IB Cable, 2 meters (6.6 feet), part no. 8120-1834
- HP 10631C HP-IB Cable, 4 meters (13.2 feet), part no. 8120-1835
- HP 10631D HP-IB Cable, ½ meter (1.5 feet), part no. 8120-2237

Each of these cables has a double-sided male/female connector on both ends so that multiple cables can conveniently be stacked for parallel connection. To order any of the above three cables, contact your nearby HP Sales and Service Office.

The installation procedure for the HP 13296A Shared Peripheral Interface accessory is presented in Section VII, *Installation*. The various commands for interacting with other devices in the HP 13296A Shared Peripheral configuration are described in Section IV, *Device Control*.

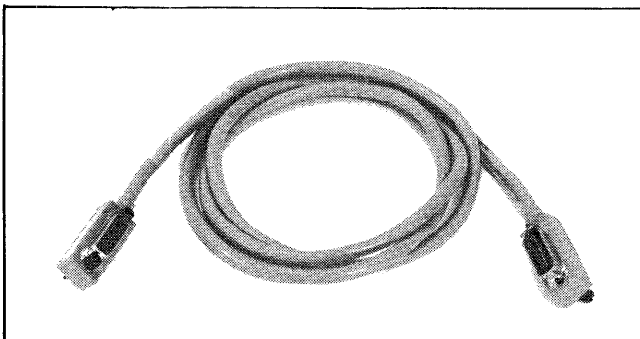


Figure 8-1. HP-IB Interconnecting Cable

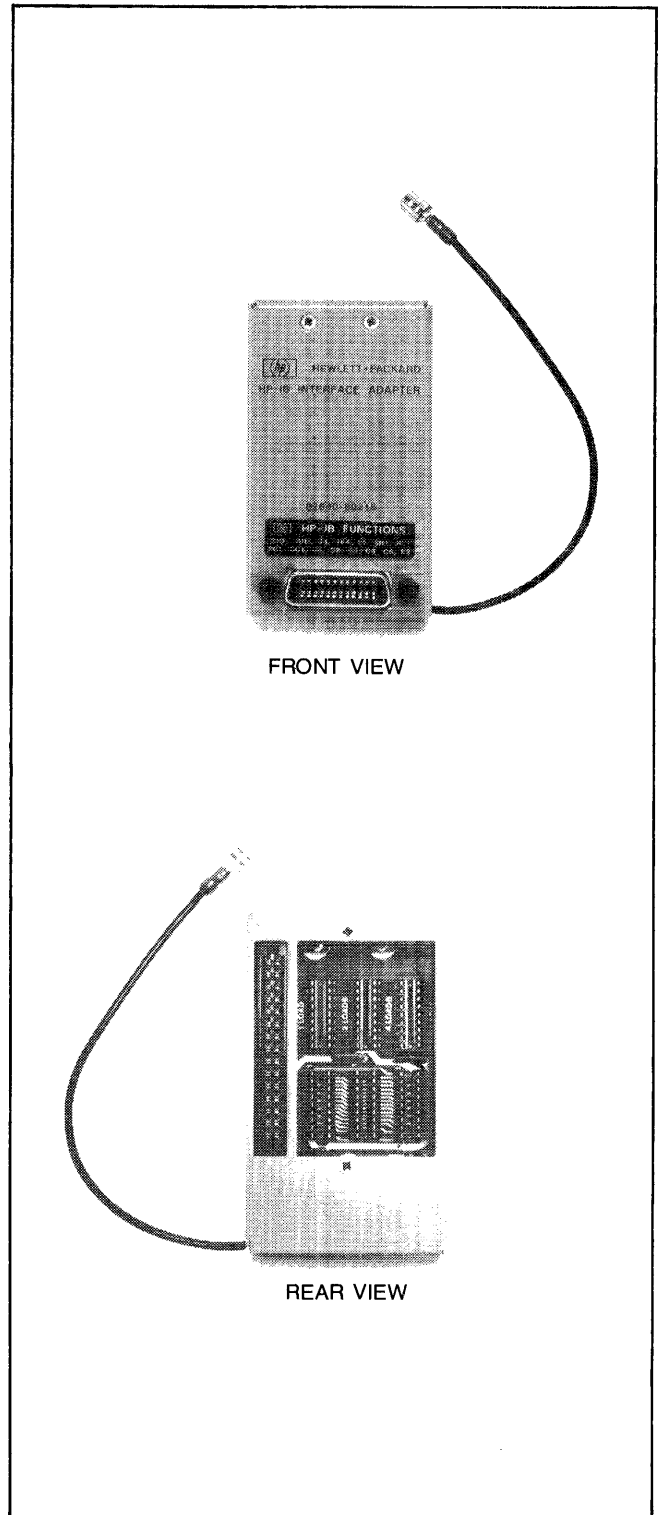


Figure 8-2. HP-IB Interface Adapter

## CABLING CONSIDERATIONS

Your HP-IB configuration is limited to a maximum combined amount of interconnecting cable that averages two meters of cable per device unless a device specifies that an average of 1 meter is required. The HP-IB Interface Adapter allows you to select 0-7 additional device loads (the terminal itself always counts as one device load). These simulated device loads make it appear as though the selected number of devices have actually been connected to the HP-IB configuration, thereby permitting you to use more meters of interconnecting cable. Additional cable availability obtained in this manner may be used anywhere within your HP-IB configuration.

The number of additional device loads is selected by moving the three IC chips (located inside the HP-IB Interface Adapter) back and forth between the upper and lower sockets. The upper sockets are inactive and the lower sockets are active. The lower left socket has the value "1", the lower middle socket has the value "2", and the lower right socket has the value "4". The number of additional device loads being simulated by the adapter is equal to the combined value of all filled sockets in the lower row.

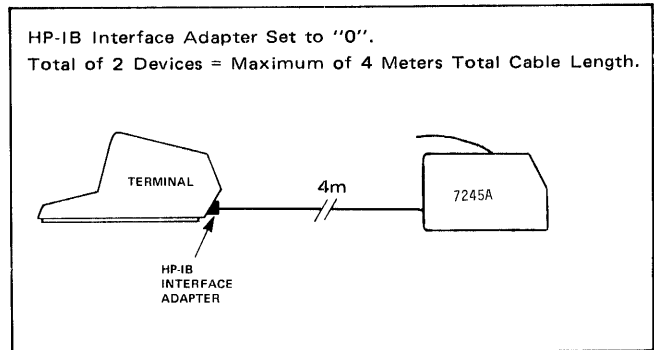
### CAUTION

The three IC chips are not interchangeable. The IC chip for selecting one additional device load must always be in either the upper or lower "1 LOAD" socket, the IC chip for selecting two additional device loads must always be in either the upper or lower "2 LOADS" socket, and the IC chip for selecting four additional device loads must always be in either the upper or lower "4 LOADS" socket. The part numbers of the three IC chips are as follows:

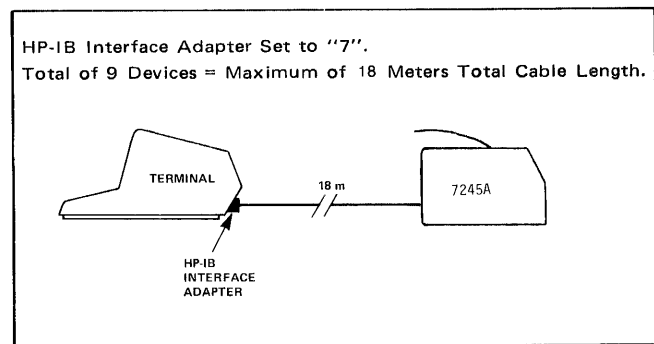
- 1820-0408 (1 LOAD)
- 1820-0410 (2 LOADS)
- 1820-0409 (4 LOADS)

The IC chips must be installed in the sockets with the notched edge at the top. The ground cable from the adapter should be connected to the terminal. The ground cable connects to the terminal at the ground plug adjacent to the power cord plug at the rear of the terminal.

Figure 8-3 illustrates several possible configurations. An IC chip extractor tool is included with the HP-IB Interface Adapter. For example, if you are connecting your terminal to an HP 7245A Plotter/Printer and the HP-IB Interface Adapter is set to "0", you would be limited to a total cable length of 4 meters as illustrated below.



By setting the HP-IB Interface Adapter to "7" instead, you could have up to 18 meters of cable between the two devices as illustrated below.



As you can see from the above examples, should you need extra cable length at some point within your configuration you can obtain it by simulating extra device "loads" with the HP-IB Interface Adapter. *Your configuration is, however, always limited to a combined total of 20 meters of interconnecting cable or a total of 15 devices (actual plus simulated).*

If you have a known combined length of HP-IB interconnecting cable and you want to know how many dummy device loads you must simulate in order to make use of all the available cable, use the following formula:

$$\# \text{ of dummy loads} = (\text{total cable length}/2) - \# \text{ of actual devices}$$

If this formula yields a fractional result, round the result up to the next higher integer value.

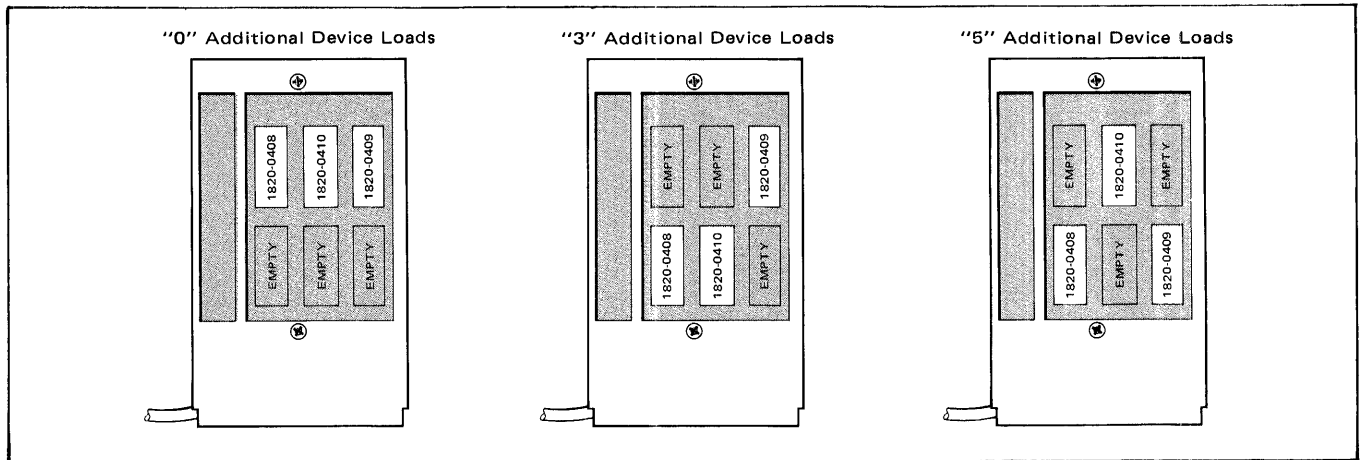


Figure 8-3. Selecting Additional Device Loads

### CABLING LIMITATIONS

Certain types of devices which can be included in an HP-IB configuration require that the maximum combined amount of interconnecting cable average one meter per device (instead of two, as described above). Such devices will have a label at their HP-IB connector which calls your attention to this limitation. When one of these devices is included in your HP 13296A configuration, each additional device load provided by the HP-IB Interface Adapter allows you one extra meter of cable in your configuration (not two). In this case, the formula for determining the required number of dummy device loads is as follows:

$$\# \text{ of dummy loads} = \text{total cable length} - \# \text{ of actual devices}$$

### OPERATIONAL CONSIDERATIONS

The terminal in any configuration is designated as the System Controller. This is accomplished by setting switch SC on the terminal's HP-IB interface PCA to the "open" position. Also, the terminal's HP-IB address is set to "29". (Refer to table 7-3A.)

### DEVICE CONSIDERATIONS

When using the HP 13296A interface, your overall configuration may include up to ten devices. The HP 13296A supports the following six peripheral devices:

- HP 2631G Printer (raster dump compatible)
- HP 2631A-046 Printer
- HP 9871A-001 Printer
- HP 7245A-001 Plotter Printer (raster dump compatible)
- HP 9872A Plotter
- HP 7225A Plotter

Note the HP 2631A/G and 9871A Printers are devices which limit the combined amount of interconnecting cable to an average of one meter per device (refer to "Cabling Limitations", above.)

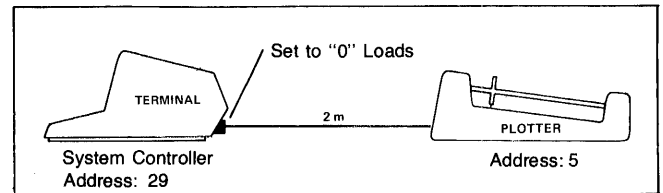


Figure 8-4. Terminal-to-Plotter Configuration

Also, note that the plotters must be driven by a program running on the remote computer system. The plotter commands are sent from the program to the plotter via the HP-IB. (Refer to the "HP-IB Operations" in Section IV, Device Control, for further information.)

### SAMPLE CONFIGURATIONS

Figure 8-4 illustrates an HP 13296A Shared Peripheral Interface configuration containing only two devices: an HP 2648A Graphics Terminal and an HP 9872A Plotter.

The terminal is designated as the System Controller with an HP-IB device address of 29. To configure the terminal in that manner, switches B4 through B0 and SC on the HP-IB Interface PCA (part no. 02640-60128) are set as follows:

LSB	B0:	Open	
	B1:	Closed	These switch settings specify the terminal's HP-IB device address as 29.
	B2:	Open	
	B3:	Open	
MSB	B4:	Open	
	SC:	Open	This switch setting indicates that the terminal is the System Controller.

Since the plotter is located adjacent to the terminal in this sample configuration, the standard 2 meter HP-IB interconnecting cable is adequate and the HP-IB Interface Adapter of the terminal is set to "0" additional device loads.

The plotter is assigned HP-IB device address 5 by setting the address switches A1 through A5 on the back panel as follows:

```

LSB  A1:  1 (on)
      A2:  0 (off)
      A3:  1 (on)
      A4:  0 (off)
MSB  A5:  0 (off)

```

Once the configuration is established as described above, you may drive the plotter under program control from a remote computer system. Be sure that the program sets the LISTENER/TALKER addresses to "5" as specified in "HP-IB Operations" in Section IV, *Device Control*.

Figure 8-5 illustrates an HP 13296A Shared Peripheral Interface configuration containing three devices: one HP 2648A Graphics Terminal, one HP 7245A Plotter/Printer, and one HP 2631A/G Printer.

The terminal is assigned HP-IB device address 29 and is designated as the System Controller. To configure the terminal in this manner, switches B4 through B0 and SC on the HP-IB Interface PCA is set as follows:

```

LSB  B0:  Open
      B1:  Closed  These switch settings specify the
      B2:  Open    terminal's HP-IB device address as
      B3:  Open    29.
MSB  B4:  Open
      SC:  Open    This switch setting indicates that
                   the terminal is the System Controller.

```

The HP 7245A Plotter/Printer is assigned HP-IB device address 5 by setting the address switches A1 through A5 on the back panel as follows:

```

LSB  A1:  1
      A2:  0
      A3:  1
      A4:  0
MSB  A5:  0

```

Note: Address 5 is assigned to the Plotter function of the HP 7245A; address 6 is automatically assigned to the Printer function of the HP 7245A. (The printer function is always automatically assigned one address greater than the plotter address on the HP 7245A.)

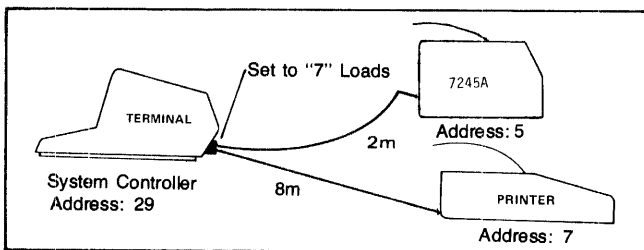


Figure 8-5. Plotter/Line Printer Configuration

The HP 2631A/G Printer is assigned HP-IB device address 7 by setting the address switches 1 through 5 on the back panel as follows:

```

LSB  5:  1 (closed)
      4:  1 (closed)
      3:  1 (closed)
      2:  0 (open)
MSB  1:  0 (open)

```

Note that switches 6 and 7 on the back panel of the HP 2631A/G should both be set to the open position.

The HP 2631A/G Printer is one of those devices which limits the overall amount of HP-IB interconnecting cable to an average of one meter per device load. With three actual devices plus seven additional device loads simulated by the HP-IB Interface Adapter of the System Controller terminal, we can therefore have up to 10 meters of HP-IB interconnecting cable in our sample configuration.

Once the configuration is established as described above, you may specify the plotting function of the HP 7245A Plotter/Printer by setting the LISTEN and TALK addresses to "5" in your program. To specify the printer function of the HP 7245A, set the LISTEN and TALK addresses to "6".

You may copy the content of the terminal's graphics display memory to the HP 7245A printer function by setting the LISTEN and TALK addresses to "6", then copying the graphics memory to the printer as described under "Graphics Transfer Operation (Raster Dump) in Section IV, *Device Control*.

Note that if the HP 2631 Printer is the G version, you may do the same with that device by specifying LISTEN and TALK address 7 in the above commands.

You may also copy the content of the terminal's alphanumeric display memory to either of the printers by specifying the appropriate printer function address (6 or 7), assigning "Source" and "Destination" devices, then specifying "Copy All" as described under "Copying to End of Medium" in Section IV, *Device Control*.

Figures 8-6 illustrates larger HP 13296A Shared Peripheral Interface configuration. The commands for interacting between your HP 2648A Graphics Terminal and the other devices in the configuration are as described in the previous two examples.

## TESTING THE HP-IB

Should the printer connected to the HP-IB not respond to commands from the terminal, the following procedure may be used to help locate the problem.

**Step 1.** Disconnect the interface cable(s) from the terminal.

**Step 2.** Enter at least six lines of text on the display, and home the cursor.

**Step 3.** Copy the text on the display to the printer. Select the devices by pressing:

**(GOLD)**     **f3**     **INSERT**  
**CHAR**

Start the copy operation by pressing:

**(GREEN)**     **f1**

The cursor should move down the display like it would during a normal copy operation. This would indicate that the printer is malfunctioning, or that the printer's HP-IB address does not agree with the TALKER/LISTENER addresses in the terminal's HP-IB driver. (Refer to Section VIII for setting the printer address and to Section IV for setting the TALKER/LISTENER addresses. The *User's Manual* gives instructions for setting the TALKER/LISTENER addresses from the keyboard.)

If the cursor moves down one line only, then the HP 13296A interface PCA may be faulty. To isolate the problem to the PCA, proceed as follows:

**Step 4.** Turn power off, and remove the interface PCA.

**Step 5.** Turn power on, and perform steps 1, 2, and 3 above.

After 10 to 20 seconds, an "HP-IB DOWN" message should appear on the display. If it does appear, then the interface PCA may be faulty. If it does not appear, then the ROM, part no. 1818-0746, on the Control Memory PCA may be faulty.

If "NO DEVICE DRIVER" message appears, then either the ROM is not installed on the Control Memory PCA, or the ROM is not installed correctly.

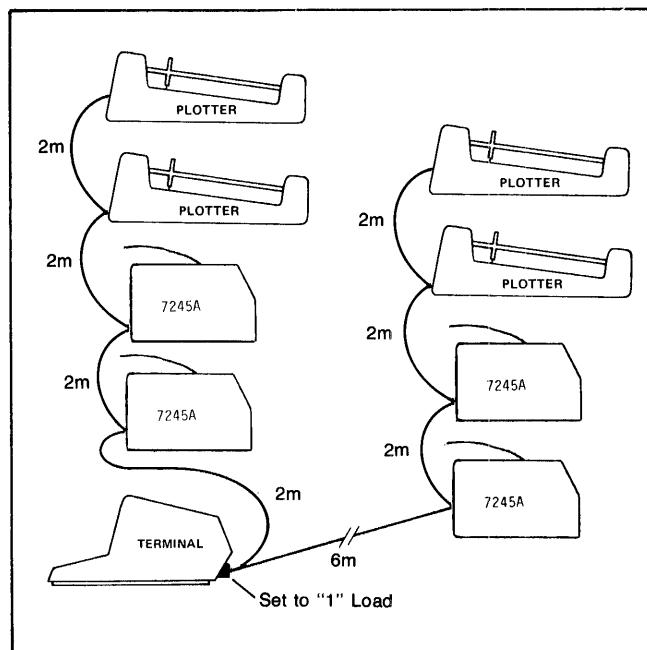


Figure 8-6. Maximum Cable Length Configuration (20 m)



## INTRODUCTION

This appendix contains sample applications using the terminal's unique features. They can be used as a guide in developing your own applications.

## SAMPLE HP3000 BASIC PROGRAM TO CONTROL THE HP 9872A PLOTTER

```
DIGITIZE
 10 REM SAMPLE PROGRAM TO DEMONSTRATE CONTROL OF 9872 FROM HP3000
 20 REM USING THE 2648 RASTER DUMP ROM
 30 REM A DIGITIZE OPERATION IS INITIATED, AND THE PLOTTER
 40 REM STATUS AND DIGITIZED POINT ARE READ BACK
 50 REM
 60 REM
 70 DIM S$(51),D$(20)
 80 REM S$ IS USED TO ACCEPT THE 'S', 'F', OR 'U' STATUS FROM THE
 90 REM TERMINAL AFTER EACH DEVICE CONTROL ESCAPE SEQUENCE.
100 REM IT IS NOT TESTED IN THIS PROGRAM
110 REM
111 REM SET LISTEN AND TALK ADDRESSES TO 5 FOR PLOTTER
112 REM
113 PRINT '27+"&p5p5u2C"
114 ENTER 255,M,S$
115 PRINT '27+"&p5p5u1C"
116 ENTER 255,M,S$
117 REM
120 PRINT "DIGITIZE POINT FROM 9872A PLOTTER BY READING STATUS UNTIL"
130 PRINT "THE 'ENTER' KEY IS HIT"
140 PRINT
150 REM
160 REM SEND DIGITIZE POINT COMMAND TO HP-IB
170 PRINT '27+"&-5dWDP;"
180 ENTER 255,M,S$
190 REM
200 REM SEND OUTPUT STATUS COMMAND TO HP-IB
210 PRINT '27+"&p5dWDS;"
220 ENTER 255,M,S$
230 REM
240 REM INPUT THE STATUS FROM THE HP-IB
250 PRINT '27+"&p5sR";
260 ENTER 255,M,S
270 REM
280 REM CHECK TO SEE IF 'ENTER' BUTTON HAS BEEN HIT
290 REM
300 IF S>=64 THEN S=S-64
310 IF S>=32 THEN S=S-32
320 IF S>=16 THEN S=S-16
330 IF S>=8 THEN S=S-8
340 IF S<4 THEN 210
350 REM
360 REM ENTER KEY HAS BEEN HIT
370 REM READ, THEN PRINT THE X,Y COORDINATES RETURNED BY THE PLOTTER
380 REM
390 REM SEND 'OUTPUT DIGITIZED POINT' COMMAND
400 PRINT '27+"&p5dWDD;"
410 ENTER 255,M,S$
420 REM
430 REM READ THE DIGITIZED POINT
440 PRINT '27+"&p5sR"
450 ENTER 255,M,D$
460 REM
470 PRINT "DIGITIZED INPUT POINT:";D$
480 GOTO 170
```

### MULTIPOINT EXAMPLE

Suppose we have a group of terminals set up in a synchronous multipoint configuration. We wish to determine status of the left cartridge of the terminal with group ID D and device ID A.

We wish to send the printer status query escape sequence to terminal DA. In multipoint, this requires selecting that terminal first. Send the select sequence

```
S S S E P S S S S           E P
Y Y Y O A Y Y Y Y           N A
N N N T D N N N N   d d A A Q D
                    Lower Upper
                    Case  Case
```

Note the sync characters present before and after the EOT PAD (there must be at least 3 in both places). Turn the line around to receive (multipoint is half duplex) and the terminal will send an ACK0 to indicate it is ready to receive:

```
S S S S D
Y Y Y Y L 0
N N N N E
```

Turn around the line again and give the terminal the cartridge status request embedded in the appropriate protocol characters:

```
S S S S E           E B P
Y Y Y T S & p       T C A
N N N X C           X C D

      I/O Control
      escape
      sequence
      ^
      |
      | 1
      |
      | STA-
      | TUS
      |
      | 1 or 2 chars
      | you have
      | calculated
      | depending
      | on type of
      | BCC chosen
      |
      | LEFT
      | Car-
      |tridge
```

The terminal returns an ACK1 after the next turnaround

```
S S S S D
Y Y Y Y L 1
N N N N E
```

if the data was OK and a NAK

```
S S S S N
Y Y Y Y A
N N N N K
```

if not. In the latter case retransmit the escape sequence above; either a parity error occurred or the BCC is wrong. Check that the proper parity is selected on the keyboard switch and the proper BCC is strapped on the data comm card.

To receive the printer status, poll terminal DA: send

```
S S S E P S S S S           E P
Y Y Y O A Y Y Y Y           N A
N N N T D N N N N DD AA Q D
```

After turnaround, you may receive from the terminal

```
S S S S E
Y Y Y Y O
N N N N T
```

This means the terminal is not yet ready to send status. Poll again until you receive it:

```
S S S S S           E           E B
Y Y Y Y T           S           C L G T C
N N N N X DA C/P 021 R F S X C

Terminal Status Cartridge Block
ID       Header Status Bytes Terminator
         for
         Multi-
         point
```

You should check the BCC the terminal sends to assure the integrity of the data.

If it checks and no parity errors were detected, send the terminal an ACK1 after you turn the line around:

```
S S S S D
Y Y Y Y L 1
N N N N E
```

After flipping to receive mode, the program will get:

```
S S S S E
Y Y Y Y O
N N N N T
```

from the terminal indicating it has no more data to send for the moment. Now the program may examine the status byte to glean whatever information it desires from them.

If the data comm had been asynchronous, the procedure would be identical except that the sync characters are not required to be sent to the terminal (although doing so is harmless) and no sync headers would be sent by the terminal (unless strap J05 were open). So except for hardware considerations, the process flows in much the same fashion in both communication methods.

Incidentally, the programmer would not normally need to perform such tasks as turning the communication lines from send to receive and vice versa, placing the STX, ETX framing characters on the messages, calculating BCC or sending select and poll sequences. At the higher level, the program simply makes output and input statements. However, this example indicates the complete process that occurs for such a two-way communication as a status request. Transactions like cursor sensing would follow exactly the same lines; only escape sequences specific to the operation differ.

## FORMS BUILDING















Special purpose forms can be built using the terminals alternate character sets and data fields. The following technique builds forms in three phases using the soft keys and tape cartridges.

- Phase 1 — Outline the form and define protected/unprotected fields.
- Phase 2 — Detail the form.
- Phase 3 — Assign field checking parameters and record the form.

This example uses one tape cartridge to hold codes to be assigned to the soft keys. A second tape cartridge is used to hold the completed form. This example assumes that the Line Drawing character set is in position "C".

## PROGRAMMING THE SOFT KEYS

Figures A-1 through A-3 show the escape sequences required to program the soft keys for each of the phases of form building. To record the escape sequences on the first tape cartridge, proceed as follows:

1. Press   (or Esc j).
2. Program each key as shown in figure A-1. Mark a key overlay.
3. Place a tape cartridge in the right tape slot.
4. Press  (Esc & pF). This stores phase 1 key assignments in file 1 on the tape.
5. Press , , .
6. Program the keys for phase 2 as shown in figure A-2.
7. Press  (Esc & pF). Press , , . This stores phase 2 key assignments in file 2.
8. Program the keys for phase 3 using figure A-3.
9. Press  (Esc & pF).
10. Press , ,  (Esc & p2u0C). After the tape is rewound, remove the tape.

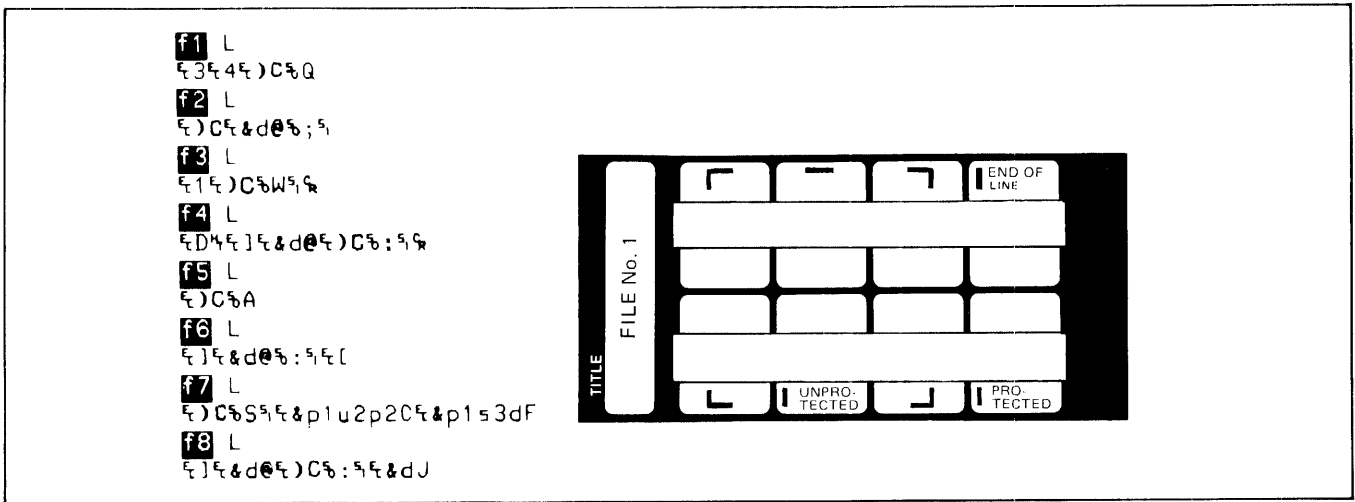


Figure A-1. Soft Key Programming and Soft Key Overlay for File 1

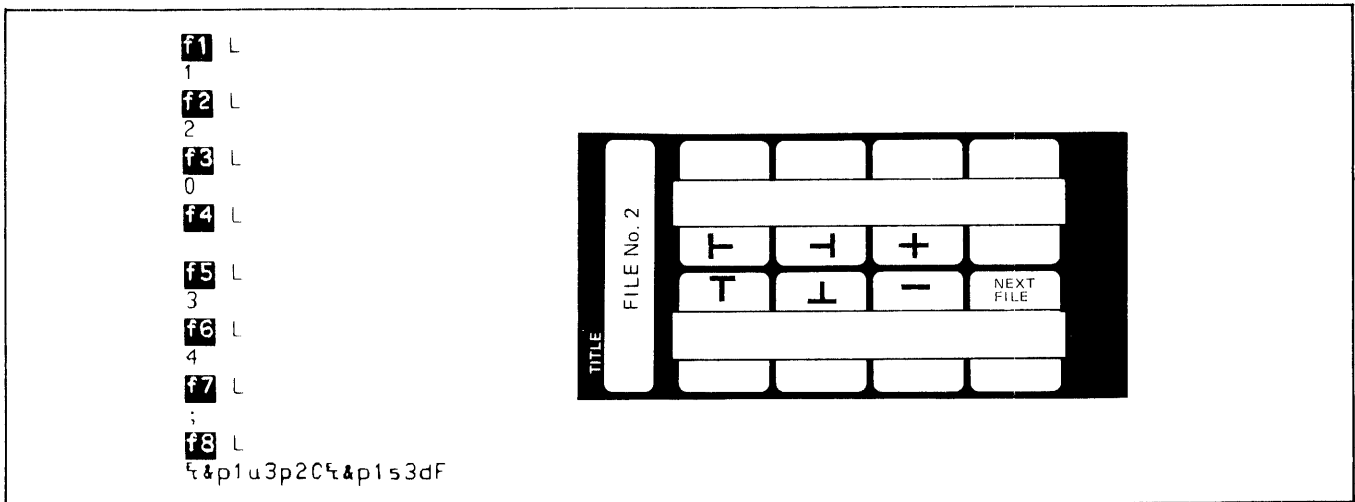


Figure A-2. Soft Key Programming and Soft Key Overlay for File 2

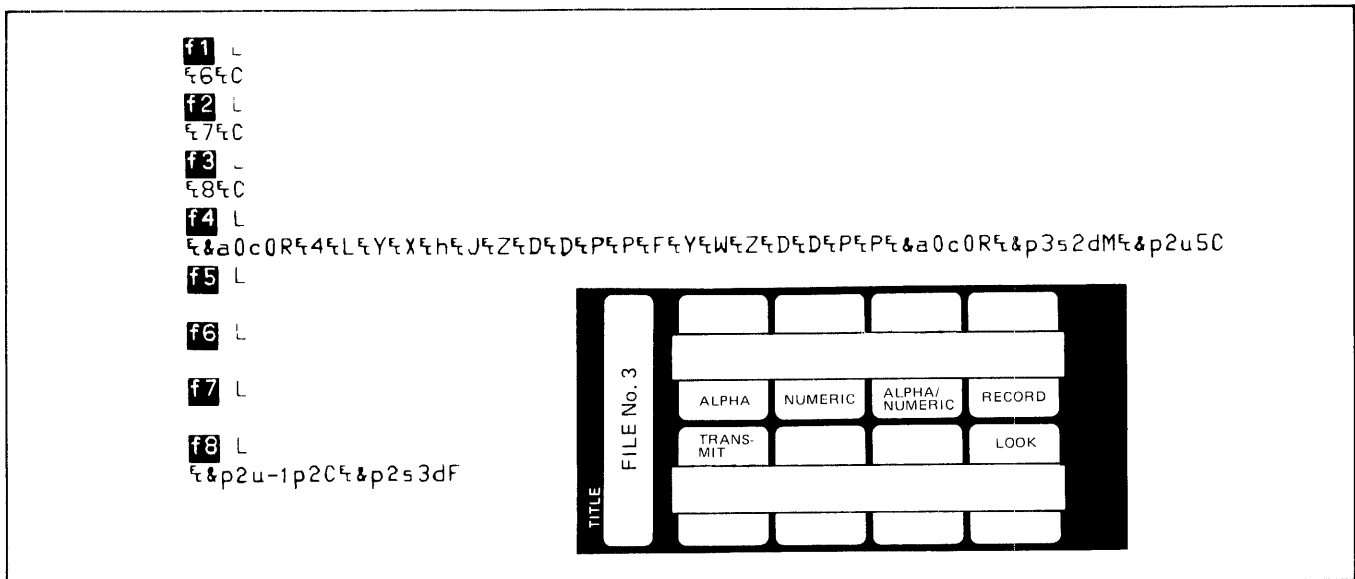


Figure A-3. Soft Key Programming and Soft Key Overlay for File 3

## Building The Form

Figure A-4 contains a simple form which will be used as an example. Insert your soft key program tape cartridge into the left tape slot and a blank tape cartridge into the right tape slot. Make sure that the device assignments are "from" left tape and "to" right tape. Forms can now be built as follows:

### PHASE 1.

1. Press READ. This loads the soft keys with file 1. Place the soft key overlay for file 1 over the soft keys.
2. Starting at the top left of your form, proceed to build your form — left to right — top to bottom. The soft key overlay will help you by giving the function of each key. Figure A-5 shows the details of building the sample form line-by-line during the first phase. The field headings (Name, Address, etc.) are "protected", and the fields to be filled by an operator are "unprotected".

After the length of the first line is defined (the top of the form), the cursor automatically tabs to the end of the first line when the END OF LINE key ( f4 ) is pressed.

When the **■** ( f7 ) key is pressed (finishing the outline phase of the form), file 2 is automatically read to load the soft keys for phase 2.

Name			
Address		State	ZIP
Telephone			

Figure A-4. Sample Form

### PHASE 2.

1. Place the soft key overlay for file 2 over the soft keys.
2. Move the cursor to each line intersection, and press the appropriate soft key. Figure A-6 shows the soft key used at each line intersection in the sample form.
3. When you have finished with the line intersection, press NEXT (f8). This will automatically load the soft keys with file 3.

### PHASE 3.

1. Place the soft key overlay for file 3 over the soft keys.
2. Starting at the top of the form, move the cursor to beginning of the first unprotected field. (In this case, it would be where the operator will fill in the name.) Pressing ALPHA (f1), defines the first space as an alpha-only field. You should press f1 as many times as necessary to fill the name field. This will prevent numbers from being entered erroneously in this field.
3. Move the cursor to the beginning of each of the remaining unprotected fields, and define each as ALPHA, NUMERIC, ALPHANUMERIC, or undefined, as applicable. Figure A-7 shows the definition of each field.
4. After each field has been defined, the form is complete. Now, press RECORD (f4) to store the form on the right tape cartridge. LOOK (f8) can be used to recall the form from the right tape cartridge to insure that it has been recorded correctly.

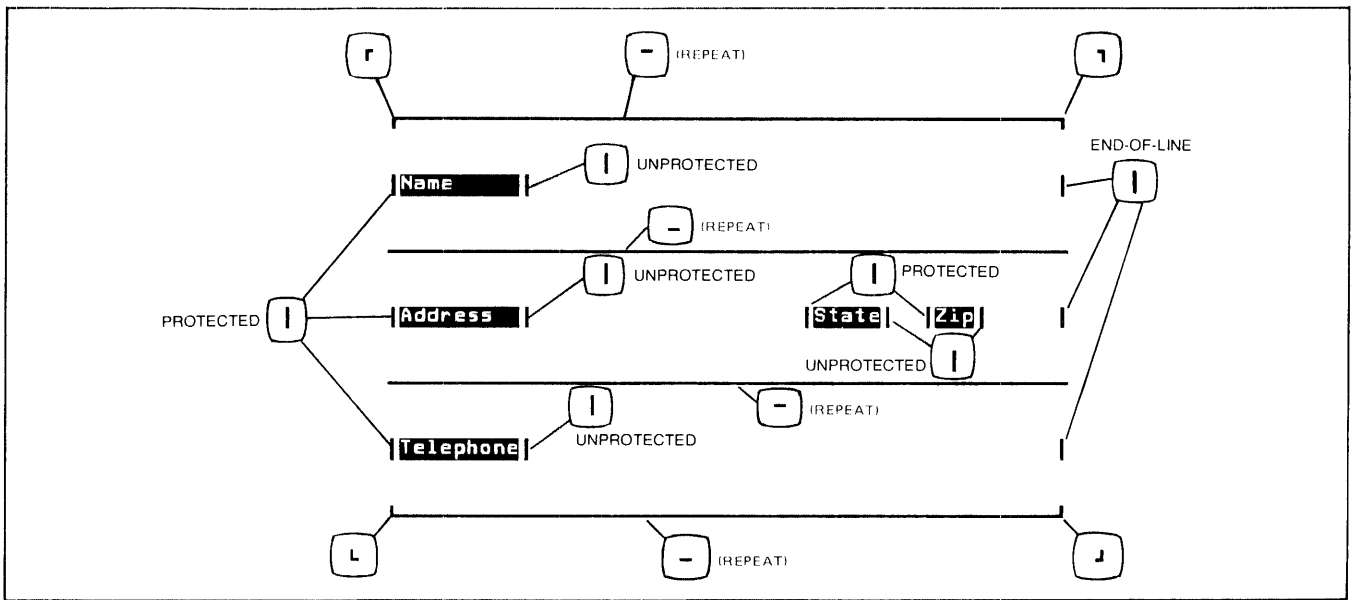


Figure A-5. Building a Form — Phase 1

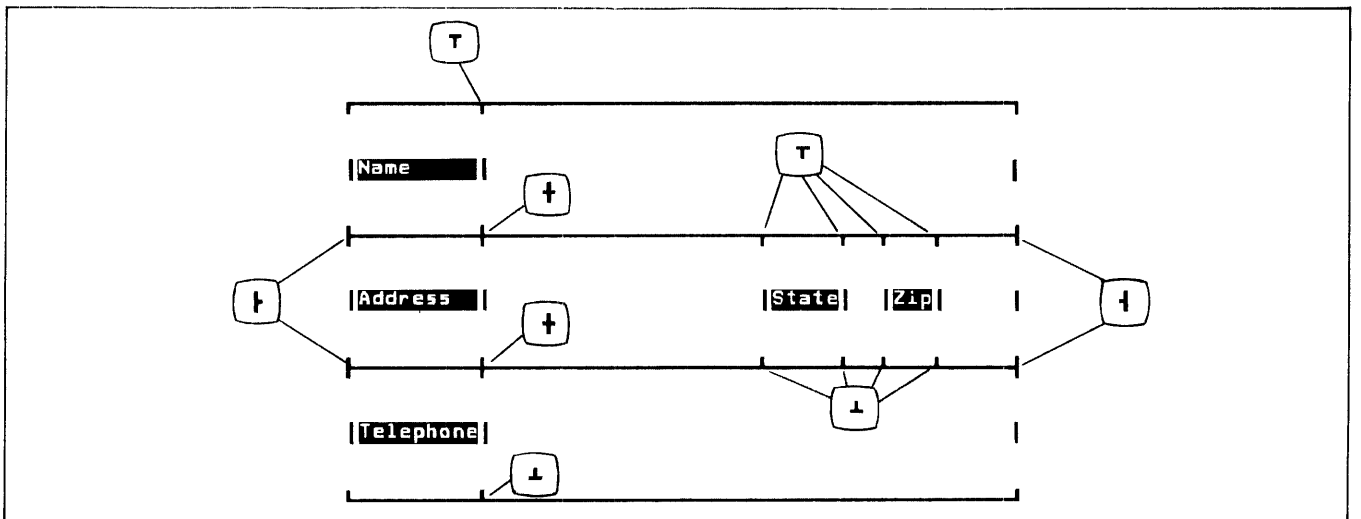


Figure A-6. Building a Form — Phase 2

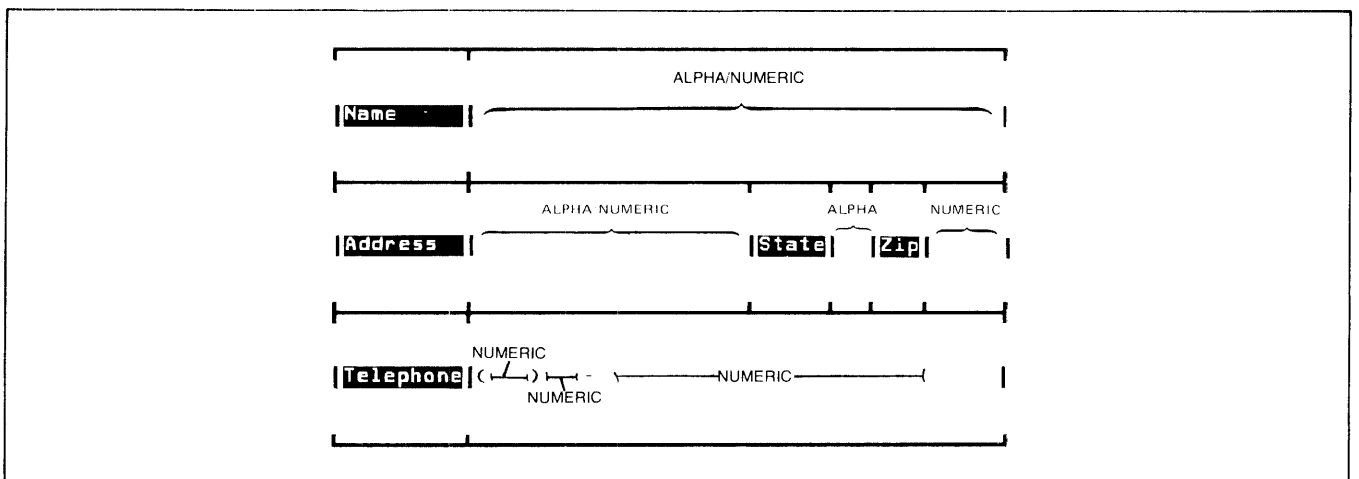


Figure A-7. Building a Form — Phase 3

## LARGE CHARACTER SET UTILITY

Since using the Large Character Set requires using up to nine character codes for each large character, it is desirable to use a utility program. A sample utility program is given in figure A-8. The program is written in BASIC/3000 but can be adopted for use in other languages. The utility accepts an entire line or string of characters and generates the necessary codes to generate the entire line in Large Characters.

```

9000 REM SUBROUTINE TO PRINT STRING IN L1$ IN LARGE CHAR. SET
9010 REM USES VARIABLES L1$ THRU L4$, L1 THRU L3
9020 DIM L1$(26),L2$(11),L3$(128,9),L4$(1)
9030 IF UND(L1)=1 THEN DO
9040 PRINT CTL(208);'27"C";
9050 L3$(33)=" 0 E E "
9060 L3$(34)=" ## "
9070 L3$(35)="C&C0 0C&C"
9080 L3$(36)="C+GC+GCL"
9090 L3$(37)="^ #3<DE ^"
9100 L3$(38)=" * & 4 "
9110 L3$(39)=" E "
9120 L3$(40)="!& 0 G& "
9130 L3$(41)=" &+ 0 &L"
9140 L3$(42)=" ( K "
9150 L3$(43)=" C "
9160 L3$(44)=" L "
9170 L3$(45)=" & "
9180 L3$(46)=" E "
9190 L3$(47)=" #3<DE "
9200 L3$(48)="!&+0 0G&L"
9210 L3$(49)=" - 0 E "
9220 L3$(50)="!&+!&LF&,"
9230 L3$(51)="!&+ &OG&L"
9240 L3$(52)=" #F&C E"
9250 L3$(53)='34"&,F&+G&L"
9260 L3$(54)="!&+!&+G&L"
9270 L3$(55)=" %&. >D E "
9280 L3$(56)="!&+5&OG&L"
9290 L3$(57)="!&+G&?G&L"
9300 L3$(58)=" E E "
9310 L3$(59)=" E L "
9320 L3$(60)=" 3 2 "
9330 L3$(61)=" & & "
9340 L3$(62)=" ) D "
9350 L3$(63)="!&+ >D V "
9360 L3$(65)="!&+!&?E E"
9370 L3$(66)='34"&+!&OF&L"
9380 L3$(67)="!&+0 G&L"
9390 L3$(68)='34"&+0 0F&L"
9400 L3$(69)='34"&, /& F&,"
9410 L3$(70)='34"&, /& E "
9420 L3$(71)="!&+0 .G&L"
9430 L3$(72)=" # /&?E E"
9440 L3$(73)=" ' 0 I "
9450 L3$(74)=" # 0G&L"
9460 L3$(75)=" # /6AE E"
9470 L3$(76)=" # 0 F&,"
9480 L3$(77)=" #(-070E E"
9490 L3$(78)=" #) #08BE E"
9500 L3$(79)='34"&.0 0F&M"
9510 L3$(80)='34"&+ /&LE "
9520 L3$(81)="!&+0 0G&N"
9530 L3$(82)='34"&+ /&OE E"
9540 L3$(83)="!&+G&+G&L"
9550 L3$(84)=" %', 0 E "
9560 L3$(85)=" # #0 0G&L"
9570 L3$(86)=" # #0 02JD"
9580 L3$(87)=" # #090HKD"
9590 L3$(88)=" # #1:AE E"
9600 L3$(89)=" # #2;D E "
9610 L3$(90)='34"&.3<DF&M"
9620 L3$(91)='34"& 0 F& "
9630 L3$(93)=" &. 0 &M"
9640 L3$(95)=" &&&"
9650 DOEND
9660 FOR L1=1 TO LEN(L1$)
9670 L2=NUM(UPS$(L1$(L1,L1)))
9680 IF L3$(L2)="" THEN PRINT CTL(208);'27"&a+3C";
9690 ELSE DO
9700 PRINT CTL(208);'14+L3$(L2,1,3)+'27"&a-3c+1R"'+14+L3$(L2,4,6)+'
'27"&a-3c+1R"'+14+L3$(L2,7,9)+'27"&a-2R";
9710 DOEND
9720 NEXT L1
9730 PRINT CTL(208);'27"&a+3R"'/13'10;
9740 RETURN

```

Figure A-8. Large Character Set Utility Routine

# REFERENCE TABLES

APPENDIX

B

## INTRODUCTION

This appendix contains the following reference information for each model terminal covered by this manual:

- Programmer's Reference Table
- Character Code Chart

- List of Options and Accessories
- List of Specifications

A Large Character Set coding table for forming characters is included at the end of this appendix.



Table B-1. Specifications

**GENERAL**

Screen Size: 127 mm (5 inches) X 254 mm (10 inches)

Screen Capacity: 24 lines X 80 columns (alphanumeric)  
720 rows X 360 dots (graphics)

Character Generation: 7 X 9 enhanced (alphanumeric)  
9 X 15 dot character cell  
Non-interlaced raster scan

Character Size: 2.46 mm (.097 inches) X 3.175 mm  
(.125 inches) (alphanumeric)  
5 X 7 dot character cell (graphics)

Character Set: 128 character (alphanumeric)

Cursor: Blinking-Underline (alphanumeric)  
Blinking-Crosshair (graphics)

Display Modes: White on Black; Black on White  
(Inverse Video)

Refresh Rate: 60 Hz (50 Hz optional)

Tube Phosphor: P4

Implosion Protection: Bonded implosion panel

Memory:  
Alphanumeric: 37 lines of 80 characters (less  
enhancements)  
Graphics: 720 dots by 360 rows of displayable points

Option Slots: 4 available

Keyboard: Detachable, bit pairing; User-defined soft keys,  
18 control and editing keys; Graphics pad; cursor pad;  
auto-repeat, n-key rollover; 1.2 m (4 foot) cable.

Cartridge Tape (option): Two mechanisms  
Read/Write Speed: 10 ips  
Search/Rewind Speed: 60 ips  
Recording: 800 bpi  
Mini Cartridge: 110 kilobyte capacity (maximum per  
cartridge)

**DATA COMMUNICATIONS**

Data Rate: 110, 150, 300, 1200, 2400, 4800, 9600 baud,  
and external. Switch selectable. (110 selects two stop  
bits). Operation above 2400 baud may require nulls or  
handshake protocol to insure data integrity.

Vector Drawing Time (9600 baud, typical):  
7 ms — half screen  
9 ms — full screen

Standard Asynchronous Communications Interface: EIA  
standard RS232C; fully compatible with Bell 103A  
modems; compatible with Bell 202C/D/S/T modems.  
Choice of main channel or reverse channel line turn-  
around for half duplex operation.

Optional Communications Interfaces (see 13260A/B/C/D  
Communications data sheet for details):

- Current loop, split speed, custom baud rates
- Asynchronous Multipoint Communications
- Synchronous Multipoint Communications — Bisync

Transmission Modes: Full or half duplex, asynchronous

Operating Modes: On-Line; Off-Line; Character, Block

Parity: Switch selectable; Even, Odd, None

**ENVIRONMENTAL CONDITIONS**

Temperature, Free Space Ambient:  
Non-Operating: -40 to +75°C (-40 to +167°F)  
Operating: 0 to 55°C (+32 to +131°F)

Temperature, Free Space Ambient (Tape):  
Non-Operating: -10 to 60°C (-15 to +140°F)  
Operating: 5 to 40°C (+41 to +104°F)

Humidity: 5 to 95% (non-condensing)

Humidity (Tape): 20 to 80% (non-condensing)

Altitude:  
Non-Operating: Sea level to 7620 metres (25,000 feet)  
Operating: Sea level to 4572 metres (15,000 feet)

Vibration and Shock (Type tested to qualify for normal  
shipping and handling in original shipping carton):  
Vibration: .37 mm (0.015") pp, 10 to 55 Hz, 3 axis  
Shock: 30g, 11ms, 1/2 sine

**PHYSICAL SPECIFICATIONS**

Display Monitor Weight: 19.6kg (43 pounds)

Keyboard Weight: 3.2kg (7 pounds)

Display Monitor Dimensions: 444 mmW X 457 mmD  
X 342 mmH (17.5"W X 18"D X 13.5"H)  
(648mmD (25.5"D) including keyboard)

Keyboard Dimensions: 444 mmW X 216 mmD X  
90 mmH (17.5"W X 8.5"D X 3.5"H)

**POWER REQUIREMENTS**

Input Voltage: 115 (+10% -23%) at 60 Hz (±0.2%)  
230 (+10% -23%) at 50 Hz (±0.2%)

Power Consumption: 115 W to 150 W max.

**PRODUCT SAFETY**

Product meets:  
UL Requirements for: EDP equipment, office  
appliances, teaching equipment  
CSA Requirements for: EDP equipment

U.L. and CSA labels are applied to equipment shipped to  
the U.S. and Canada.

**PRODUCT SUPPORT****WARRANTY**

90 day on-site parts and labor

**HP SYSTEMS SUPPORT**

Refer to appropriate HP system data sheet for use and  
support of 2648A in systems. If this product is used in  
a customer-assembled system, the overall operational  
responsibility of the system rests with the customer.

**HARDWARE SUPPLIED**

2648A Graphics Terminal

**DOCUMENTATION SUPPLIED**

2648A User's Manual  
2648A Reference Manual

Table B-2. Programmer's Reference Table

KEY	CODE	FUNCTION	KEY	CODE	FUNCTION
<b>ALPHANUMERIC DISPLAY CONTROL</b>					
	ESC A	Cursor up		ESC 6	Alphabetic only field
	ESC B	Cursor down		ESC 7	Numeric only field
	ESC C	Cursor right		ESC 8	Alphanumeric field
	ESC D	Cursor left	<b>EDITING</b>		
	BS (H <sup>8</sup> )	Cursor left one space		ESC L	Insert a blank line
	ESC F	Cursor home down		ESC M	Delete line containing cursor
	ESC G	Cursor return		ESC P	Delete character at cursor
	ESC h	Cursor home (excluding transmit-only fields)		ESC O	Delete character with wraparound from next line
	ESC H	Home cursor (including transmit-only fields)		ESC Q (on) ESC R (off)	Insert succeeding inputs at cursor
	CR (M <sup>6</sup> )	Move cursor to left margin		ESC N (on) ESC R (off)	Character Wraparound Mode. Insert succeeding inputs at cursor with wraparound to next line.
	LF (J <sup>5</sup> )	Move cursor down one line		-----	Toggles EDIT mode
	HT (I <sup>5</sup> ) ESC I	Forward cursor to next tab position	<b>TERMINAL CONTROL GROUP</b>		
	ESC J	Back tab	ESC	ESC (I <sup>7</sup> )	Leads off an ASCII escape sequence
	ESC K	Set tab at the current cursor column		-----	Used to generate ASCII control codes and alternate key functions
	ESC L	Clear the tab at the current cursor column		ESC&k0C(off) ESC&k1C(on)	Upper-case alphabetical lock
	ESC M	Clear all tabs		ESC I (on) ESC m (off)	Memory overflow protect; display lock
	ESC N	Set left margin		ESC&k0A(off) ESC&k1A(on)	Line Feed with each terminal carriage return
	ESC O	Set right margin		ESC&k0R(off) ESC&k1R(on)	Remote (on-line) operations; otherwise, off-line operation
	ESC P	Clear memory from cursor position to end of memory		ESC&k0B(off) ESC&k1B(on)	Block Mode: data displayed but not transmitted until requested; otherwise, terminal is in Character Mode and data transmitted as typed
	ESC Q	Clear line from the cursor to end of line		-----	Enables block transfers
	ESC R	Scroll the display up one line		-----	Transmits BREAK signal to interrupt computer
	ESC S	Scroll the display down one line	TRANSMIT indicator	-----	Data link exists
	ESC T	Display the next 24 lines of memory		ESC Y (on) ESC Z (off)	Control functions disabled and displayed
	ESC U	Display the previous 24 lines of memory		ESC y (on) ESC Z (off)	Monitor Mode: display all codes received from data comm lines
	ESC V	Turn on display enhance		ESC g	(First press): frees the keyboard and clears I/O operations
	ESC & d	Start an unprotected field		ESC E	(Second press): sets the terminal to power-on state
	ESC I	End an unprotected field or transmit-only field		ESC x	Data Comm Self-Test
	ESC J	Turn format mode on. Only unprotected fields can be modified.		ESC z	Terminal Self-Test (no tape test)
	ESC W (on)	Turn format mode off.			
	ESC X (off)	Start transmit-only field			
	ESC {				

KEY	CODE	FUNCTION
<b>ADDITIONAL FUNCTIONS</b>		
-----	ENQ (E <sup>c</sup> )	Enquiry from the computer
-----	ACK (F <sup>c</sup> )	Acknowledge — response to ENQ
-----	BEL (G <sup>c</sup> )	Bell
-----	ESC )	Define alternate character set: (@, A, B, C
-----	SO (N <sup>c</sup> )	Turn on alternate character set
-----	SI (O <sup>c</sup> )	Turn off alternate character set
-----	DC1 (Q <sup>c</sup> )	Block transfer trigger
-----	DC2 (R <sup>c</sup> )	Block transfer enable from terminal
-----	RS (A <sup>c</sup> )	Record separator
-----	US (U <sup>c</sup> )	Unit separator
-----	ESC @	Delay one second
-----	ESC ^	Cursor sensing (screen relative)
-----	ESC a	Cursor sensing (absolute)
-----	ESC b	Keyboard enable
-----	ESC c	Keyboard disable
-----	ESC d	Block transfer enable from computer (See DC2)
-----	ESC e	Fast binary read
-----	ESC f	Modem hang-up
-----	ESC j (on) ESC k (off)	Display user-defined soft keys
-----	ESC ^	Terminal status
-----	ESC ~	Extended status request
-----	ESC & a <parameters>	Cursor addressing

Example: Cursor to 12th row 35th column (+, - for relative addressing)  
 $\text{ESC} \& \text{a} 12\text{r} 35\text{C}$

-----

ESC & b <parameters> or ESC & c <parameters>	HP diagnostics ONLY
ESC & d <enhancement>	Turn on display enhancement

where:  
enhancement = @ through O

Example: Select half-bright, blinking, and underline.  
 $\text{ESC} \& \text{d} \text{M}$

	Enhancement Character															
	@	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
Underline				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	x	x	x	x	x	x	x	x
End Enhancement	x															

KEY	CODE	FUNCTION
-----	ESC & f <parameters>	Define soft keys
where:	0 (normal) <parameters> = {0-8}k 1 (local) a {1-80} <text string> 2 (transmit only)	
Example:	Assign the string "HELLO-MYCANT" to the <b>f1</b> key. The key should function as normal keyboard input. $\text{ESC} \& \text{f} 1 \text{k} 1 \text{a} 13 \text{L} \text{HELLO-MYACNT}$	
-----	ESC & f <key #>E	Execute the soft key. Key # = 0-8

KEY	DEFAULT	PROGRAMMABLE
<b>RETURN</b>	CR	
<b>f1</b>	ESC p to computer	
<b>f2</b>	ESC q to computer	
<b>f3</b>	ESC r to computer	
<b>f4</b>	ESC s to computer	Up to 80-character sequence for each key (local, transmit or both)
<b>f5</b>	ESC t to computer	
<b>f6</b>	ESC u to computer	
<b>f7</b>	ESC v to computer	
<b>f8</b>	ESC w to computer	

KEY	CODE	FUNCTION
-----	ESC & g <parameters>	Simulate PA, PF keys (see Reference Manual)
-----	ESC & k <parameters>	Define latching keys
where:	0 (up) <parameter> = 1 (down)	$\left. \begin{matrix} \text{a} \\ \text{b} \\ \text{c} \\ \text{r} \end{matrix} \right\}$
Example:	Block Mode up Remote up Auto LF down Caps Lock down $\text{ESC} \& \text{k} 1 \text{a} 0 \text{b} 1 \text{c} 0 \text{R}$	

KEY	CODE	FUNCTION	KEY	CODE	FUNCTION	
<b>DEVICE CONTROL</b>						
ESC & p <parameters>						
(GOLD)	f1	ESC & p 1S Assigns LEFT TAPE as source device	(GREEN)	f5, ±n, or f6	ESC & p (±n)p 1u 2C Positions LEFT TAPE to a relative (±n) or absolute (n) FILE	
(GOLD)	f2	ESC & p 2S Assigns RIGHT TAPE as source device	(GREEN)	ENTL f1 or f2 or f3	ESC & p 1M Compare data between source and destination (All)	
(GOLD)	f3	ESC & p 3S Assigns DISPLAY as source device	(GREEN)	f1 or f2 or f3	ESC & p 1F (File)	
(GOLD)	f5	ESC & p 1D Assigns LEFT TAPE as destination device	(GREEN)	f3	ESC & p 1B (Line)	
(GOLD)	f6	ESC & p 2D Assigns RIGHT TAPE as destination device	ENTL READ	-----	Read page without DC handshake	
(GOLD)	f7	ESC & p 3D Assigns DISPLAY as destination device	(GREEN)	ENTL READ	-----	Read tape beyond end-of-data mark
(GOLD)	f8	ESC & p 4D Assigns PRINTER as destination device	ENTL TAPE TEST	f5 or f6	ESC & p 1u 4C Conditions left or right tape	
(GOLD)	INSERT CHAR	ESC & p 5D Assigns PRINTER on HP-IB as destination device	(GREEN)	TAPE TEST	ESC & p 1u 7C Tape Self-Test (tests cartridges)	
NOTE: One source and multiple destinations can be set up with the same sequence.						
Example: (GOLD) f5, f7, f8 ESC & p 1s3d 4D						
(GREEN)	f1	ESC & p xs xd M All files (current position) from source device are transferred to destination device	(GREEN)	f5 INSERT CHAR	ESC & p 5u 0C Copy graphics memory to destination device	
(GREEN)	f1	ESC & p xs xd F One File (current position) from source device is transferred to destination device.	(GREEN)	f6 INSERT CHAR	ESC & p 5u 5C Copy from source device to graphics memory	
(GREEN)	f3	ESC & p xs xd B One line (current position) from source device is transferred to destination device.	(GREEN)	f7 INSERT CHAR	ESC & p 5u address, (address)p 1C Select HP-IB TALK address	
NOTE: x is variable to indicate source and destination device codes. 1 = Left Tape, 3 = Display, 5 = HP-IB 2 = Right Tape, 4 = Printer						
Example: Copy all files from right tape to left tape and display. ESC & p 2s 1d 3d M						
(GREEN)	INSERT LINE	ESC & p 9C Turn off tape Write-Backspace-Read Mode	(GREEN)	f8 INSERT CHAR	ESC & p 5u address, (address)p 2C Select HP-IB LISTEN address	
(GREEN)	DELETE LINE	ESC & p 10C Turn on tape Write-Backspace-Read Mode	-----	-----	ESC & p 5u 3C Enable HP-IB timeout function.	
(GREEN)	f5 or f6	ESC & p 1u 0C Rewinds LEFT TAPE	-----	-----	ESC & p 5u 4C Disable HP-IB timeout function.	
(GREEN)	f5 or f6	ESC & p 2u 0C Rewinds RIGHT TAPE	-----	-----	ESC & p 5u 7C Initialize HP-IB.	
(GREEN)	f5 or f6	ESC & p 1u 5C Write a FILE MARK on LEFT TAPE	-----	-----	ESC & p 5u <>p 6C Read byte count on HP-IB.	
(GREEN)	f5 or f6	ESC & p 2u 5C Write a FILE MARK on RIGHT TAPE	(GREEN)	SPACE BAR	-----	Display file, inches remaining for each tape.
(GREEN)	TAPE TEST	ESC & p 1u 7C Tests left tape unit ESC & p 2u 7C Tests right tape unit	READ	-----	In REMOTE, transfers data from source device to computer. In LOCAL, transfers one file from source device to DISPLAY.	
(GREEN)	f5 or f6	ESC & p (±n)p 1u 1C Positions LEFT TAPE to a relative (±n) LINE	RECORD	-----	In REMOTE, transfers data from computer to destination device. In LOCAL, transfers one file from DISPLAY to destination device.	
(GREEN)	f5 or f6	ESC & p (±n)p 2u 1C Positions RIGHT TAPE to a relative (±n) LINE	ENTER	-----	In REMOTE, enables block transfers. In LOCAL, operates same as RECORD	
Example: Remove Q strap, install P strap. ESC & s 0q 1P						
Define strap settings ESC & s <parameters>						

**GRAPHICS CONTROL SEQUENCES**

ESC \* <control sequence>  
 a = Autoplot  
 d = Display control  
 l = Graphics text label  
 m = Mode control

p = Plot control  
 s = Status  
 t = Compatibility mode  
 r, b = Raster dump

**AUTO PLOT**

ESC \* a <parameters>

KEY	CODE	FUNCTION
	a	Turn AUTO PLOT on.
	b	Turn AUTO PLOT off.
	c	Draw AUTO PLOT axes.
-----	d	Clear AUTO PLOT menu.
	f	Turn AUTO PLOT menu on.
	g	Turn AUTO PLOT menu off.
-----	<# of cols> h	Load # of cols.
-----	<x col> i	Load x column #
-----	<y col> j	Load y column #
-----	<line type> k	Load line type (1-9)
-----	<min x> l	Load minimum x value
-----	<max x> m	Load maximum x value
-----	<min y> n	Load minimum y value
-----	<max y> o	Load maximum y value
-----	<x labels> p	Load x label increment
-----	<x tics> q	Load x tic increment
-----	<y labels> r	Load y label increment
-----	<y tics> s	Load y tic increment
-----	<# of lines> t	Load heading lines to be skipped
-----	<# of points> u	Load # of points to plot
-----	<grid?> v	Load grid command (1/0)
-----	<display?> w	Load display plot command (1/0)
-----	z	NOP

**AUTO PLOT**

**A. PLOT SPECIFICATION**

1. NO. OF COLUMNS 2
2. X IS COLUMN 1
3. Y IS COLUMN 2
4. LINE TYPE (1-9) 1
5. MIN X 0
6. MAX X 100
7. MIN Y 0
8. MAX Y 100

**B. AXES SPECIFICATION**

1. UNITS BETWEEN X LABELS 10
2. UNITS BETWEEN X TICS 10
3. UNITS BETWEEN Y LABELS 10
4. UNITS BETWEEN Y TICS 10

**C. PLOT OPTIONS**

1. SKIP FIRST LINES OF TEXT 10
2. STOP AFTER POINTS 10
3. DRAW GRID? 1
4. FROM DSPLY? 0

Example: The following example programs the terminal to accept and plot 10 points (in the range 1 to 100) sent from the computer.

```
ESC * a d 2h 11 2j 1k 11 100m 1n 100o 20p 10q 20r 10s 10u 1v 0w c A
```

**KEY**

**CODE**

**FUNCTION**

**DISPLAY CONTROL**

ESC \* d <parameters>

	a	Clear graphics memory
-----	b	Set graphics memory
	c	Turn on graphics display
	d	Turn off graphics display
	e	Turn on alphanumeric display
	f	Turn off alphanumeric display
	g	Turn on zoom
	h	Turn off zoom
	<size> i	Set zoom size (1-16)
-----	<x,y> j	Set zoom position
	k	Turn on graphics cursor
	l	Turn off graphics cursor
	m	Turn on rubber band line
	n	Turn off rubber band line
-----	<x,y> o	Move graphics cursor absolute
	<x,y> p	Move graphics cursor incremental
-----	q	Turn on alphanumeric cursor
-----	r	Turn off alphanumeric cursor
	s	Turn on graphics text mode
	t	Turn off graphics text mode
-----	z	NOP

Example: Clear the graphics display, position the cursor at x=100, y=100, turn the cursor on, and zoom to 4 times.

```
ESC * d a 100,100o k 4i G
```

**GRAPHICS LABEL**

ESC \* l <text label> <A, B, C, or D>

Example: Send the text "X=TIME, Y=TEMP"

```
ESC * l X=TIME, Y=TEMP A
```

**KEY CODE FUNCTION**

**VECTOR DRAWING MODE**

ESC \* m <parameters>

-----	<mode>	a	Select drawing mode (0-4)*
-----	<line type>	b	Select line type (1-11)**
-----	<pattern> <scale>	c	Define line pattern (2 bytes)
-----	<pattern>	d	Define area shading pattern (8 bytes)
-----	<x1,y1,x2,y2>	e	Fill area, absolute
-----	<x1,y1,x2,y2>	f	Fill area, relocatable
-----	<x,y>	j	Set relocatable origin
-----		k	Set relocatable origin to current pen position
-----		l	Set relocatable origin to graphics cursor position
-----	<size>	m	Set graphics text size (1-8)
-----	<rotation>	n	Set graphics text orientation (1-4)
-----		o	Turn on text slant
-----		p	Turn off text slant
-----	<1-9>	q	Set graphics text origin
-----		r	Set graphics defaults
-----		z	NOP



\* 0 (no effect), 1 (set), 2 (clear), 3 (complement), 4 (jam)

** 1 (solid line)	4 (line #1)	7 (line #4)	10 (line #7)
2 (user line pattern)	5 (line #2)	8 (line #5)	11 (point plot)
3 (user area pattern)	6 (line #3)	9 (line #6)	

Example: Select the set drawing mode, a graphics text size of 2 and slanted. Set the text to be center justified.

```
ESC * m 1 a 2 m o 4 Q
```

**PLOTTING COMMANDS**

ESC \* p <parameters>

-----	a	Lift the pen
-----	b	Lower the pen
-----	c	Use graphics cursor as new point
-----	d	Draw a point at the current pen position and lift the pen
-----	e	Set relocatable origin to the current pen position
-----	f	Data is ASCII absolute
-----	g	Data is ASCII incremental
-----	h	Data is ASCII relocatable
-----	i	Data is absolute
-----	j	Data is short incremental
-----	k	Data is incremental
-----	l	Data is relocatable
-----	z	NOP

Example: Draw a box 25 units wide and 10 units high, beginning at x=100, y=50.

```
ESC * p a f 100 50 g 25, 0 0, 10 -25, 0 0, -10 Z
```

**KEY CODE FUNCTION**

**GRAPHICS STATUS**

ESC \* s <parameter> ^

-----	1	Read device I.D.
-----	2	Read pen position
-----	3	Read graphics cursor position
-----	4	Read cursor position and wait for key
-----	5	Read display size
-----	6	Read graphics capabilities
-----	7	Read graphics text status
-----	8	Read zoom status
-----	9	Read relocatable origin
-----	10	Read reset status
-----	11	Read area shading
-----	12	Read dynamics

Example: Read text status.

```
ESC * s 7 ^ DC1
```

**COMPATIBILITY MODE**

ESC \* t <parameter>

-----	<0/1/2>	a	Set graphics input terminator (0=CR, 1=CR EOT, 2=none)
-----	<0/1>	b	Set Page Full Break strap (0=out, 1=in)
-----	<0/1>	c	Set Page Full Busy strap (0=out, 1=in)
-----		z	NOP

Keyboard Interface switches:

P open = Scaled compatibility mode.  
Q open = Unscaled compatibility mode.

Example: Select a CR input terminator and set the Page Full Busy strap.

```
ESC * t 0 a 1 C
```

**RASTER DUMP**

ESC \* r <parameter>

-----	a	Start transfer
-----	b	End transfer
-----	c	Erase screen
-----	d	Turn on video

ESC \* b <parameter>

-----	<no. of data bytes>	w	Write specified number of bytes into graphics memory scan line.
-------	---------------------	---	---

Table B-3. Character Code Chart

BIT 4321	CONTROL (CNTL) CHARACTERS				DISPLAYABLE CHARACTERS				ESCAPE SENT FIRST										
	0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1					
0000	@	NUL	P	16	SP	0	@	P	\	p		SP	0	DELAY 1 SEC	@	P	CURSOR RELATIVE SENSE	f <sub>1</sub>	p
0001	A	SOH	Q	17	!	1	A	Q	a	q		SET TAB	1	CURSOR UP	A	Q	CURSOR ABSOLUTE SENSE	f <sub>2</sub>	q
0010	B	STX	R	18	"	2	B	R	b	r		CLEAR TAB	2	CURSOR DOWN	B	R	KEYBOARD ENABLE	f <sub>3</sub>	r
0011	C	ETX	S	19	#	3	C	S	c	s		CLEAR ALL TABS	3	CURSOR RIGHT	C	S	KEYBOARD DISABLE	f <sub>4</sub>	s
0100	D	EOT	T	20	\$	4	D	T	d	t		SET LEFT MARGIN	4	CURSOR LEFT	D	T	ENTER	f <sub>5</sub>	t
0101	E	ENQ	U	21	%	5	E	U	e	u		SET RIGHT MARGIN	5	RESET TERMINAL	E	U	BINARY READ	f <sub>6</sub>	u
0110	F	ACK	V	22	&	6	F	V	f	v		PARAMETER SEQUENCE	6	CURSOR HOME DOWN	F	V	MODEM DISCONNECT	f <sub>7</sub>	v
0111	G	BEL	W	23	'	7	G	W	g	w		START NUMERIC FIELD	7	CURSOR RETURN	G	W	SOFT RESET	f <sub>8</sub>	w
1000	H	BS	X	24	(	8	H	X	h	x		START ALPHNUM FIELD	8	HOME CURSOR (SEE NOTE 3)	H	X	HOME CURSOR (SEE NOTE 3)	f <sub>9</sub>	x
1001	I	HT	Y	25	)	9	I	Y	i	y		DEFINE CHAR SET	9	HORIZONTAL TAB	I	Y	BACK TAB	f <sub>10</sub>	y
1010	J	LF	Z	26	*	:	J	Z	j	z		GRAPHICS SEQUENCE	:	CLEAR DISPLY	J	Z	SOFT KEY DISPLAY ON	f <sub>11</sub>	z
1011	K	VT	[	27	+	;	K	[	k	{			;	ERASE TO END OF LINE	K	[	SOFT KEY DISPLAY OFF	f <sub>12</sub>	{
1100	L	FF	\	28	,	<	L	\	l	!			<	INSERT LINE	L	\	MEMORY LOCK ON	f <sub>13</sub>	!
1101	M	CR	]	29	=	=	M	]	m	}			=	DELETE LINE	M	]	MEMORY LOCK OFF	f <sub>14</sub>	}
1110	N	SO	^	30	>	>	N	^	n	~			>	INSERT CHAR W/WRAP	N	^	SEND SECONDARY STATUS	f <sub>15</sub>	~
1111	O	SI	_	31	/	?	O	_	o	▀			/	DELETE CHAR W/WRAP	O	_	INSERT NON-DISP TERMINATR	f <sub>16</sub>	▀

Example: J is bits 1001010; Control J is LF line feed; Escape (ESC) followed by J is CLEAR DISPLAY

**LEGEND**

**NOTES:**

1. LOWER CASE LETTER, LOWER CASE SYMBOL, AND CONTROL CHARACTER CODES ARE GENERATED BY STANDARD TERMINAL, BUT ASSOCIATED CHARACTERS ARE NOT DISPLAYED ON THE SCREEN. PRESS TAPE TEST KEY FOR DISPLAYABLE CHARACTER SET.
2. SINGLE CHARACTER ESCAPE SEQUENCES AND CONTROL CODES NOT LISTED WITH A FUNCTION ARE NEITHER ACTED UPON NOR DISPLAYED.
3. ESC H HOMES CURSOR INCLUDING TRANSMIT-ONLY FIELDS. ESC h HOMES CURSOR EXCLUDING TRANSMIT-ONLY FIELDS.

- A<sub>k</sub> - ACKNOWLEDGE
- B - BELL
- B<sub>s</sub> - BACKSPACE
- C<sub>h</sub> - CANCEL LINE
- C<sub>r</sub> - CARRIAGE RETURN
- D<sub>1</sub> - DEVICE CONTROL 1
- D<sub>2</sub> - DEVICE CONTROL 2
- D<sub>3</sub> - DEVICE CONTROL 3
- D<sub>4</sub> - DEVICE CONTROL 4
- DEL - DELETE
- DL - DATA LINK ESCAPE
- EM - END OF MEDIUM
- ENQ - ENQUIRY
- ET - END OF TRANSMISSION
- ESC - ESCAPE
- EB - END OF TRANSMISSION BLOCK
- ET - END OF TEXT
- FF - FORM FEED
- F<sub>s</sub> - FILE SEPARATOR
- G<sub>s</sub> - GROUP SEPARATOR
- H<sub>t</sub> - HORIZONTAL TABULATION
- LF - LINE FEED
- N<sub>k</sub> - NEGATIVE ACKNOWLEDGE
- R<sub>s</sub> - RECORD SEPARATOR
- S<sub>i</sub> - SHIFT IN
- S<sub>o</sub> - SHIFT OUT
- SP - SPACE
- S<sub>h</sub> - START OF HEADING
- S<sub>t</sub> - START OF TEXT
- S<sub>s</sub> - SUBSTITUTE
- S<sub>i</sub> - SYNCHRONOUS IDLE
- U<sub>s</sub> - UNIT SEPARATOR
- V<sub>t</sub> - VERTICAL TABULATION

**Control Characters Legend:**

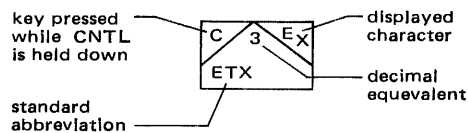


Table B-4. Options and Accessories

PRODUCT NUMBER	DESCRIPTION
2648A	<p>GRAPHICS TERMINAL 720 × 360 dot graphics image memory and random access alphanumeric memory (expandable by 4K bytes), 128 character Roman, inverse video, editing, 9 user-defined soft keys, RS232C, 4 option slots. <i>Note: No interface cable included.</i></p>
-007	<p>Integrated Dual Cartridge Tape — Mini Data Station Adds two built-in cartridge tape transports and electronics to provide Mini Data Station capabilities (requires 2 option slots). Includes device support firmware.</p>
-013	5 Mini Cartridges
-015	50 Hertz, 330V
-016	60 Hertz, 115V
-030	<p>Delete Standard Asynchronous Communications <i>Note: One of the 13260 data communications accessories must be ordered when option 030 is ordered.</i></p>
	<b>ACCESSORIES</b>
13231A	<p>DISPLAY ENHANCEMENTS Adds blinking, half-bright and underline, includes line drawing character set, and provides for addition of two more 128 character sets. (Requires 1 option slot).</p>
-201	64 Character Mathematic Symbol Set. Adds display of integral signs, Greek letters, etc.
-203	Large Character Set
13234A	<p>TERMINAL MEMORY MODULE (+4K) Adds 4096 bytes of user RAM memory (requires 1 option slot).</p>
13236B	<p>INTEGRATED DUAL CARTRIDGE TAPE UPGRADE KIT Field upgrade for adding two built-in cartridge tape transports and electronics to provide Mini Data Station capabilities (requires 2 option slots). Includes installation. <i>Note: 13261A-003 also required.</i></p>
13250B	<p>SERIAL PRINTER INTERFACE Adds interface for connecting RS232C serial printing devices (requires 1 option slot). No interface cable included. <i>Note: 13261A-003 also required on tapeless 2648A's.</i></p>
13254A	<p>VIDEO OUTPUT INTERFACE Provides video signal for a compatible monitor or hardcopy unit (requires one option slot).</p>
13260A	<p>STANDARD ASYNCHRONOUS COMMUNICATIONS Upgrade which provides standard RS232C communications interface for the 2648A. <i>Note: This is identical to the capability deleted by 2648A-030.</i></p>
-003	Adds data communications firmware compatible with 2648A.
13260B	<p>EXTENDED ASYNCHRONOUS COMMUNICATIONS Provides either an RS232C or 20mA current loop communication interface for the 2648A. Has split speed and custom baud rates. <i>Note: 2648A-030 must be ordered to delete the Standard Asynchronous interface.</i></p>
-003	Adds data communications firmware compatible with 2648A.
13261A	<p>DEVICE SUPPORT FIRMWARE Required by tapeless 2648A's to support printers, tape upgrade or other I/O devices.</p>
-003	
13296A	<p>PERIPHERAL INTERFACE Provides interface capability to compatible HP-IB devices. Recommended peripherals are: 2631A, 2631G, and 7245A with their appropriate HP-IB compatible configuration.</p>
-048	Provides 2648A compatible firmware for raster format output to compatible hardcopy device or tape cartridge.
9162-0061	MINI CARTRIDGE (purchased from Corporate Parts Center)
	<b>CABLES</b>
13232C	RS232C Cable, female, 5 ft.
13232F	Current Loop Connector Kit, four wire, 5 ft.
13232K	Video Cable. For connection to compatible video hardcopy.
13232L	Video Cable. For connection to compatible video monitor.
13232M	European Modem Cable, male RS232C, 15 ft.
13232N	Modem Cable, male RS232C, 15 ft.

**ORDERING EXAMPLE**

This is an example of ordering a 2648A Graphics Terminal with cartridge tape, extra cartridges, video hardcopy interface and cable, to be used with a modem with autodisconnect.

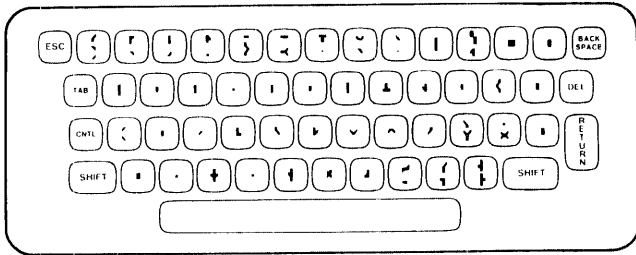
NOTE: With this configuration there are no available option slots.

2648A Graphics Terminal  
 -007 Adds cartridge tape units  
 -013 Adds five cartridges  
 -030 Deletes standard data communications  
 13232N 103/202 Modem Cable  
 13260B Extended Async Data Communications  
 13254A Video Interface  
 13232K Cable, Compatible video hardcopy

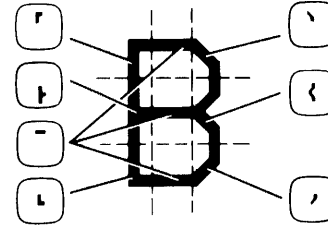


Table B-5. Coding the Large Character Set

The elements of the Large Character Set are associated with the keyboard as pictured below:



Each large character is actually made up of nine character segments. An example of constructing the letter "B" using the Large Character Set follows:



!	0 S	-	%&	9	!&+ G&? G&L	E	"&, /& F&,	Q	!&+ 0 0 G&N	J	%. 0 %M	i	. E	U	" FM
"	"	.	Z	:	Z	F	"&, /& E	R	"&+ /&@ E E	^	9 K	j	. GL	U	" GL
#	CC CC	÷	. %&, 4	;	Y L	G	!&+ 0 . G&L	S	!&+ G&+ G&L	-	...	k	./@ EE	W	" HD
\$	!C+ GC+ GCL	0	!&+ 0 0 G&L	<	3 2	H	" /&? E E	T	%, 0 E	,	+	l	./@ E	X	5@ EE
%	P P 3<D Y Y	1	- 0 E	=	%&, %&, %&,	I	./@ I	U	" 0 0 G&L	a	!. GM	m	\$- EE	4	" G?
&	!+ 5IC G&L	2	!&+ !&L F&,	>	) D	J	./@ L	V	" 0 0 2JD	b	./+ FL	n	" EE	2	%. F,
'	'	3	!&+ /&@ G&L	?	!&+ >D S	K	"/@A E E	W	" 090 HKD	c	!, G,	o	!+ GL	{	!, @ G,
(	!, 0 G,	4	"/@ F&C E	@	!&+ !0 GIL	L	./@ F&,	X	!: 1:A E E	d	!? GM	p	" /L		Q U
)	%+ 0 %L	5	"&, F&+ G&L	A	!&+ /&? E E	M	\$(- 070 E E	Y	" 2;D E	e	!+ G,	q	!. G?	}	%+ 5 %L
x	!: 1:A	6	!&+ /&+ G&L	B	"&+ /&@ F&L	N	\$)/ 08B E E	Z	"&. 3<D F&M	f	!+ C E	r	!, E	~	!&L
+	. %C, 4	7	%&. >D E	C	!&+ 0 G&L	O	"&. 0 0 F&M	[	" 0 F,	q	!. G?	s	!, %L	■	... ... ...
,	L	8	!&+ 5&@ G&L	D	"&+ 0 0 F&L	P	"&+ /&L E	x	./@ 2:) E	h	./+ EE	t	. C GL		

# COMMUNICATIONS FLOWCHARTS

APPENDIX

C

This appendix contains reference information on terminal communication functions. This material consists of the following flowcharts and tables:

- ASCII code table
- ASCII to EBCDIC code conversion table
- Overall point-to-point communications flowchart
- Keyboard communication switches

Table C-1 is a list of the ASCII characters and their decimal equivalents. Tables C-2 and C-3 contain information for converting data between the ASCII and EBCDIC character sets.

The flowchart in figure C-1 illustrates the overall point-to-point communication function. The various configuration parameters (switches) are included in the diagram. Detailed descriptions of the switches are given in Sections V and VII. Figure C-2 illustrates the way the terminal responds to various keyboard switches.

## MULTIPOINT COMPATIBILITY

Earlier versions of the Multipoint data communications code provide slightly different features and require a different configuration procedure. The configuration switches affected are on the Keyboard Interface and Multipoint Communications Interface PCA's.

The following backdating information is required for functional compatibility if your terminal uses any of the following ROM circuit part numbers:

1818-0214	1818-0433
1818-0261	1818-0434
1818-0288	1818-0435

The following multipoint features are not available in the earlier code versions:

- Space Compression
- Internal Data Set Ready
- Transparency

Tables C-4 and C-5 contain the switch definitions for the older code versions. The definitions for switches not shown are unchanged. Use these definitions instead of those given in Sections V and VII.

Table C-1. ASCII Character Set

DECIMAL VALUE	GRAPHIC	COMMENTS	ALTERNATE CHARACTER	DECIMAL VALUE	GRAPHIC	COMMENTS
0	$\backslash 0$	Null	@ <sup>c</sup>	64	@	Commercial at
1	$\backslash 1$	Start of heading	A <sup>c</sup>	65	A	Uppercase A
2	$\backslash 2$	Start of text	B <sup>c</sup>	66	B	Uppercase B
3	$\backslash 3$	End of text	C <sup>c</sup>	67	C	Uppercase C
4	$\backslash 4$	End of transmission	D <sup>c</sup>	68	D	Uppercase D
5	$\backslash 5$	Enquiry	E <sup>c</sup>	69	E	Uppercase E
6	$\backslash 6$	Acknowledge	F <sup>c</sup>	70	F	Uppercase F
7	$\backslash 7$	Bell	G <sup>c</sup>	71	G	Uppercase G
8	$\backslash 8$	Backspace	H <sup>c</sup>	72	H	Uppercase H
9	$\backslash 9$	Horizontal tabulation	I <sup>c</sup>	73	I	Uppercase I
10	$\backslash 10$	Line feed	J <sup>c</sup>	74	J	Uppercase J
11	$\backslash 11$	Vertical tabulation	K <sup>c</sup>	75	K	Uppercase K
12	$\backslash 12$	Form feed	L <sup>c</sup>	76	L	Uppercase L
13	$\backslash 13$	Carriage return	M <sup>c</sup>	77	M	Uppercase M
14	$\backslash 14$	Shift out	N <sup>c</sup>	78	N	Uppercase N
15	$\backslash 15$	Shift in	O <sup>c</sup>	79	O	Uppercase O
16	$\backslash 16$	Data link escape	P <sup>c</sup>	80	P	Uppercase P
17	$\backslash 17$	Device control 1 (X-ON)	Q <sup>c</sup>	81	Q	Uppercase Q
18	$\backslash 18$	Device control 2	R <sup>c</sup>	82	R	Uppercase R
19	$\backslash 19$	Device control 3 (X-OFF)	S <sup>c</sup>	83	S	Uppercase S
20	$\backslash 20$	Device control 4	T <sup>c</sup>	84	T	Uppercase T
21	$\backslash 21$	Negative acknowledge	U <sup>c</sup>	85	U	Uppercase U
22	$\backslash 22$	Synchronous idle	V <sup>c</sup>	86	V	Uppercase V
23	$\backslash 23$	End of transmission block	W <sup>c</sup>	87	W	Uppercase W
24	$\backslash 24$	Cancel	X <sup>c</sup>	88	X	Uppercase X
25	$\backslash 25$	End of medium	Y <sup>c</sup>	89	Y	Uppercase Y
26	$\backslash 26$	Substitute	Z <sup>c</sup>	90	Z	Uppercase Z
27	$\backslash 27$	Escape	[ <sup>c</sup>	<sup>1</sup> 91	[	Opening bracket
28	$\backslash 28$	File separator	\ <sup>c</sup>	<sup>2</sup> 92	\	Reverse slant
29	$\backslash 29$	Group separator	] <sup>c</sup>	<sup>1</sup> 93	]	Closing bracket
30	$\backslash 30$	Record separator	^ <sup>c</sup>	<sup>1</sup> 94	^	Circumflex
31	$\backslash 31$	Unit separator	_ <sup>c</sup>	<sup>2</sup> 95	_	Underscore
32		Space (Blank)		96	`	Grave accent
<sup>1</sup> 33	!	Exclamation point		97	a	Lowercase a
34	"	Quotation mark		98	b	Lowercase b
35	#	Number sign		99	c	Lowercase c
36	\$	Dollar sign		100	d	Lowercase d
37	%	Percent sign		101	e	Lowercase e
38	&	Ampersand		102	f	Lowercase f
39	'	Apostrophe		103	g	Lowercase g
40	(	Opening parenthesis		104	h	Lowercase h
41	)	Closing parenthesis		105	i	Lowercase i
42	*	Asterisk		106	j	Lowercase j
43	+	Plus		107	k	Lowercase k
44	,	Comma		108	l	Lowercase l
45	-	Hyphen (Minus)		109	m	Lowercase m
46	.	Period (Decimal)		110	n	Lowercase n
47	/	Slant		111	o	Lowercase o
48	0	Zero		112	p	Lowercase p
49	1	One		113	q	Lowercase q
50	2	Two		114	r	Lowercase r
51	3	Three		115	s	Lowercase s
52	4	Four		116	t	Lowercase t
53	5	Five		117	u	Lowercase u
54	6	Six		118	v	Lowercase v
55	7	Seven		119	w	Lowercase w
56	8	Eight		120	x	Lowercase x
57	9	Nine		121	y	Lowercase y
58	:	Colon		122	z	Lowercase z
59	;	Semicolon		<sup>2</sup> 123	{	Opening (left) brace
60	<	Less than		<sup>2</sup> 124		Vertical line
61	=	Equals		<sup>2</sup> 125	}	Closing (right) brace
62	>	Greater than		<sup>2</sup> 126	~	Tilde
63	?	Question mark		127		Delete

Notes: 1. The equivalent EBCDIC character uses a different graphic.  
2. No equivalent character exists in EBCDIC.

Table C-3. EBCDIC Character Codes

GRAPHIC	DEC	OCT	HEX
NUL	0	0	0
SOH	1	1	1
STX	2	2	2
ETX	3	3	3
PF	4	4	4
HT	5	5	5
DEL	6	6	6
	7	7	7
	8	10	8
	9	11	9
	10	12	A
VT	11	13	B
FF	12	14	C
CR	13	15	D
SO	14	16	E
SI	15	17	F
DLE	16	20	10
DC1	17	21	11
DC2	18	22	12
TM	19	23	13
RES	20	24	14
NL	21	25	15
BS	22	26	16
IL	23	27	17
CAN	24	30	18
EM	25	31	19
CC	26	32	1A
CU1	27	33	1B
IFS	28	34	1C
IGS	29	35	1D
IRS	30	36	1E
IUS	31	37	1F
DS	32	40	20
SOS	33	41	21
FS	34	42	22
	35	43	23
BYP	36	44	24
LF	37	45	25
ETB	38	46	26
ESC	39	47	27
	40	50	28
	41	51	29
SM	42	52	2A
CU2	43	53	2B
	44	54	2C
ENQ	45	55	2D
ACK	46	56	2E
BEL	47	57	2F
	48	60	30
	49	61	31
SYN	50	62	32
	51	63	33
PN	52	64	34
RS	53	65	35
UC	54	66	36
EOT	55	67	37
	56	70	38
	57	71	39
	58	72	3A
CU3	59	73	3B
DC4	60	74	3C
NAK	61	75	3D
	62	76	3E
SUB	63	77	3F

GRAPHIC	DEC	OCT	HEX
SP	64	100	40
	65	101	41
	66	102	42
	67	103	43
	68	104	44
	69	105	45
	70	106	46
	71	107	47
	72	110	48
	73	111	49
	74	112	4A
	75	113	4B
.	76	114	4C
<	77	115	4D
(	78	116	4E
+	79	117	4F
BB	80	120	50
&	81	121	51
	82	122	52
	83	123	53
	84	124	54
	85	125	55
	86	126	56
	87	127	57
	88	130	58
	89	131	59
!	90	132	5A
\$	91	133	5B
*	92	134	5C
)	93	135	5D
;	94	136	5E
7	95	137	5F
-	96	140	60
/	97	141	61
	98	142	62
	99	143	63
	100	144	64
	101	145	65
	102	146	66
	103	147	67
	104	150	68
	105	151	69
!	106	152	6A
,	107	153	6B
%	108	154	6C
-	109	155	6D
>	110	156	6E
?	111	157	6F
	112	160	70
	113	161	71
	114	162	72
	115	163	73
	116	164	74
	117	165	75
	118	166	76
	119	167	77
	120	170	78
	121	171	79
:	122	172	7A
#	123	173	7B
@	124	174	7C
'	125	175	7D
=	126	176	7E
"	127	177	7F

Table C-3. EBCDIC Character Codes (Continued)

GRAPHIC	DEC	OCT	HEX
	128	200	80
a	129	201	81
b	130	202	82
c	131	203	83
d	132	204	84
e	133	205	85
f	134	206	86
g	135	207	87
h	136	210	88
i	137	211	89
	138	212	8A
	139	213	8B
	140	214	8C
	141	215	8D
	142	216	8E
	143	217	8F
	144	220	90
j	145	221	91
k	146	222	92
l	147	223	93
m	148	224	94
n	149	225	95
o	150	226	96
p	151	227	97
q	152	230	98
r	153	231	99
	154	232	9A
	155	233	9B
	156	234	9C
	157	235	9D
	158	236	9E
	159	237	9F
	160	240	A0
~	161	241	A1
s	162	242	A2
t	163	243	A3
u	164	244	A4
v	165	245	A5
w	166	246	A6
x	167	247	A7
y	168	250	A8
z	169	251	A9
	170	252	AA
	171	253	AB
	172	254	AC
[	173	255	AD
	174	256	AE
	175	257	AF
	176	260	B0
	177	261	B1
	178	262	B2
	179	263	B3
	180	264	B4
	181	265	B5
	182	266	B6
	183	267	B7
	184	270	B8
	185	271	B9
	186	272	BA
	187	273	BB
	188	274	BC
]	189	275	BD
	190	276	BE
	191	277	BF

GRAPHIC	DEC	OCT	HEX
{	192	300	C0
A	193	301	C1
B	194	302	C2
C	195	303	C3
D	196	304	C4
E	197	305	C5
F	198	306	C6
G	199	307	C7
H	200	310	C8
I	201	311	C9
	202	312	CA
	203	313	CB
	204	314	CC
	205	315	CD
	206	316	CE
	207	317	CF
}	208	320	D0
J	209	321	D1
K	210	322	D2
L	211	323	D3
M	212	324	D4
N	213	325	D5
O	214	326	D6
P	215	327	D7
Q	216	330	D8
R	217	331	D9
	218	332	DA
	219	333	DB
	220	334	DC
	221	335	DD
	222	336	DE
	223	337	DF
\	224	340	E0
	225	341	E1
S	226	342	E2
T	227	343	E3
U	228	344	E4
V	229	345	E5
W	230	346	E6
X	231	347	E7
Y	232	350	E8
Z	233	351	E9
	234	352	EA
	235	353	EB
	236	354	EC
	237	355	ED
	238	356	EE
	239	357	EF
0	240	360	F0
1	241	361	F1
2	242	362	F2
3	243	363	F3
4	244	364	F4
5	245	365	F5
6	246	366	F6
7	247	367	F7
8	248	370	F8
9	249	371	F9
	250	372	FA
	251	373	FB
	252	374	FC
	253	375	FD
	254	376	FE
	255	377	FF

Table C-2. ASCII (7-Bit) Character Codes

GRAPHIC	DEC	OCT	HEX
NUL	0	0	0
SOH	1	1	1
STX	2	2	2
ETX	3	3	3
EOT	4	4	4
ENQ	5	5	5
ACK	6	6	6
BEL	7	7	7
BS	8	10	8
HT	9	11	9
LF	10	12	A
VT	11	13	B
FF	12	14	C
CR	13	15	D
SO	14	16	E
SI	15	17	F
DLE	16	20	10
DC1	17	21	11
DC2	18	22	12
DC3	19	23	13
DC4	20	24	14
NAK	21	25	15
SYN	22	26	16
ETB	23	27	17
CAN	24	30	18
EM	25	31	19
SUB	26	32	1A
ESC	27	33	1B
FS	28	34	1C
GS	29	35	1D
RS	30	36	1E
US	31	37	1F
SP	32	40	20
!	33	41	21
"	34	42	22
#	35	43	23
\$	36	44	24
%	37	45	25
&	38	46	26
'	39	47	27
(	40	50	28
)	41	51	29
*	42	52	2A
+	43	53	2B
,	44	54	2C
-	45	55	2D
.	46	56	2E
/	47	57	2F
0	48	60	30
1	49	61	31
2	50	62	32
3	51	63	33
4	52	64	34
5	53	65	35
6	54	66	36
7	55	67	37
8	56	70	38
9	57	71	39
:	58	72	3A
;	59	73	3B
<	60	74	3C
=	61	75	3D
>	62	76	3E
?	63	77	3F

GRAPHIC	DEC	OCT	HEX
@	64	100	40
A	65	101	41
B	66	102	42
C	67	103	43
D	68	104	44
E	69	105	45
F	70	106	46
G	71	107	47
H	72	110	48
I	73	111	49
J	74	112	4A
K	75	113	4B
L	76	114	4C
M	77	115	4D
N	78	116	4E
O	79	117	4F
P	80	120	50
Q	81	121	51
R	82	122	52
S	83	123	53
T	84	124	54
U	85	125	55
V	86	126	56
W	87	127	57
X	88	130	58
Y	89	131	59
Z	90	132	5A
[	91	133	5B
\	92	134	5C
]	93	135	5D
^	94	136	5E
_	95	137	5F
`	96	140	60
a	97	141	61
b	98	142	62
c	99	143	63
d	100	144	64
e	101	145	65
f	102	146	66
g	103	147	67
h	104	150	68
i	105	151	69
j	106	152	6A
k	107	153	6B
l	108	154	6C
m	109	155	6D
n	110	156	6E
o	111	157	6F
p	112	160	70
q	113	161	71
r	114	162	72
s	115	163	73
t	116	164	74
u	117	165	75
v	118	166	76
w	119	167	77
x	120	170	78
y	121	171	79
z	122	172	7A
{	123	173	7B
	124	174	7C
}	125	175	7D
~	126	176	7E
•	127	177	7F

Table C-4. Keyboard Interface Switch Definitions for Earlier Multipoint Code

STRAP	STRAPPING OPTION	NORMAL OPERATION (SWITCH CLOSED)	OPERATION WITH STRAPPING OPTION (SWITCH OPEN)
R	Set Trailing Pad	If in ASCII mode (switch J07 closed on multipoint PCA), sets pad to 177 (octal) + parity. If in EBCDIC mode (switch J07 open on multipoint PCA), sets pad to 377 (octal).	Sets pad to 377 (octal) if any of the following conditions are present: (1) PARITY switch on keyboard is set to NONE. (2) Switch Z on this PCA is open. (3) CRC-16 is selected (switch J06 on multipoint PCA is closed).
S	(not used)		
V	Continuous Carrier	Continuous carrier off indicates that the modem does not have continuous carrier.	Continuous carrier on indicates that the modem does have continuous carrier. Allows firmware to abort operation.
Z	Parity	<p>The PARITY switch on the terminal keyboard is affected as follows:</p> <p><b>None:</b> (Force 0). Send 8 bits and receive 8 bits. Force bit 8 to zero. Check for parity error.†</p> <p><b>Odd Parity:</b> Send 7 bits + odd parity. Receive 7 bits + odd parity. Check for parity error.</p> <p><b>Even Parity:</b> Send 7 bits + even parity. Receive 7 bits + even parity. Check for parity error.</p> <p>†Allows Transparency Mode.</p>	<p><b>None:</b> (Force 1). Send 8 bits and receive 8 bits. Force bit 8 to one. Check for parity error.†</p> <p><b>Odd Parity:</b> Send 7 bits + odd parity. Receive 7 bits + odd parity. Check for parity error.†</p> <p><b>Even Parity:</b> Send 7 bits + even parity. Receive 7 bits + even parity. Check for parity error.†</p>

Table C-5. Multipoint Communications Interface Switch Definitions for Earlier Multipoint Code

STRAP	STRAPPING OPTION	DESCRIPTION
J05	Sync Mode (Asynchronous Interface Only)	<p><b>Open:</b> Enables the insertion and deletion of sync characters to be compatible with a single, generalized data communications driver.</p> <p><b>Closed:</b> Sync Mode disabled.</p>
J15	Block Mode	<p><b>Open:</b> An entire input block must be received correctly before being processed by the terminal firmware. (See Block Check Character.)</p> <p><b>Closed:</b> Each character is processed by the terminal firmware as it is received from the computer.</p>

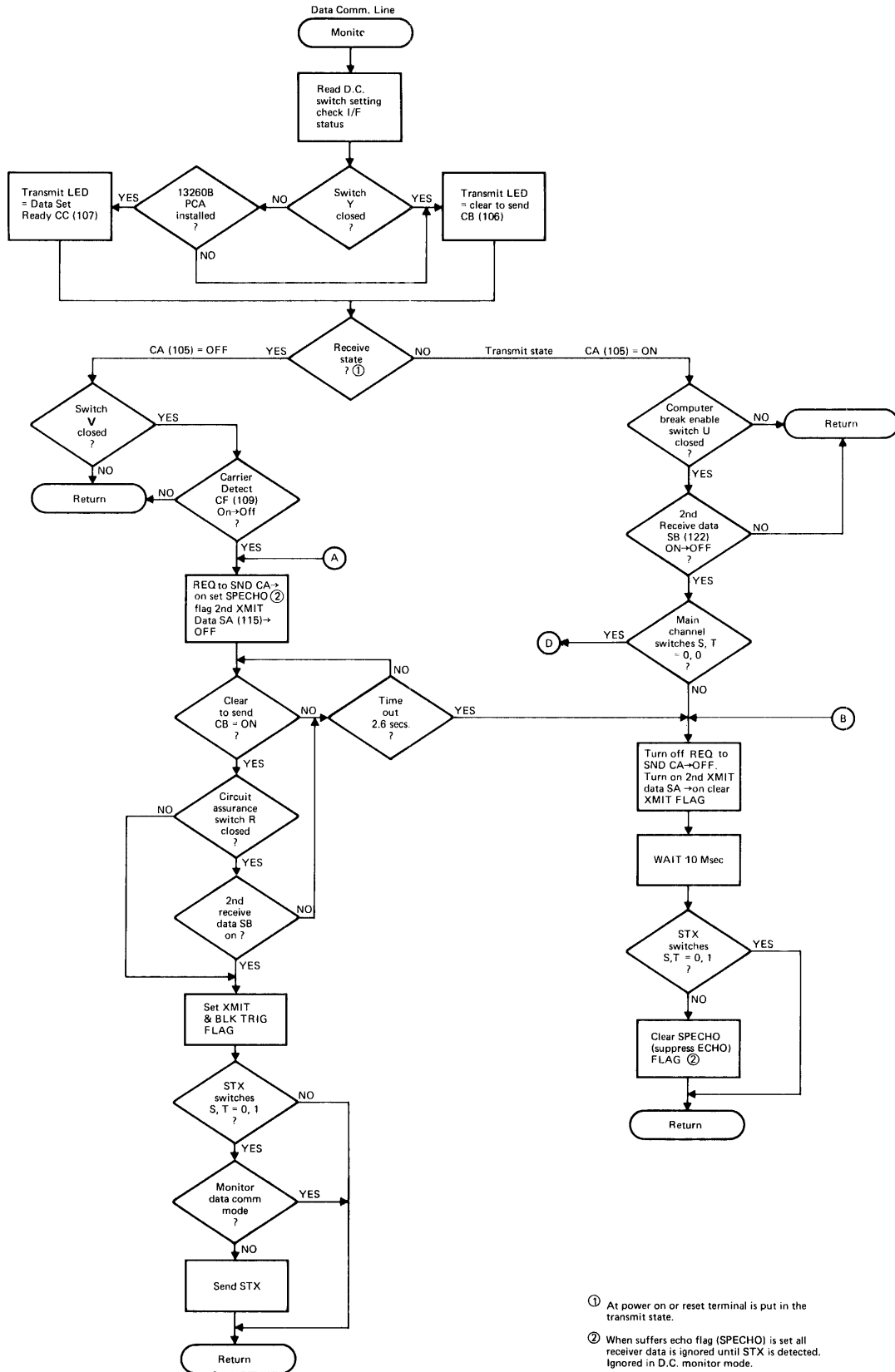


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 1 of 3)



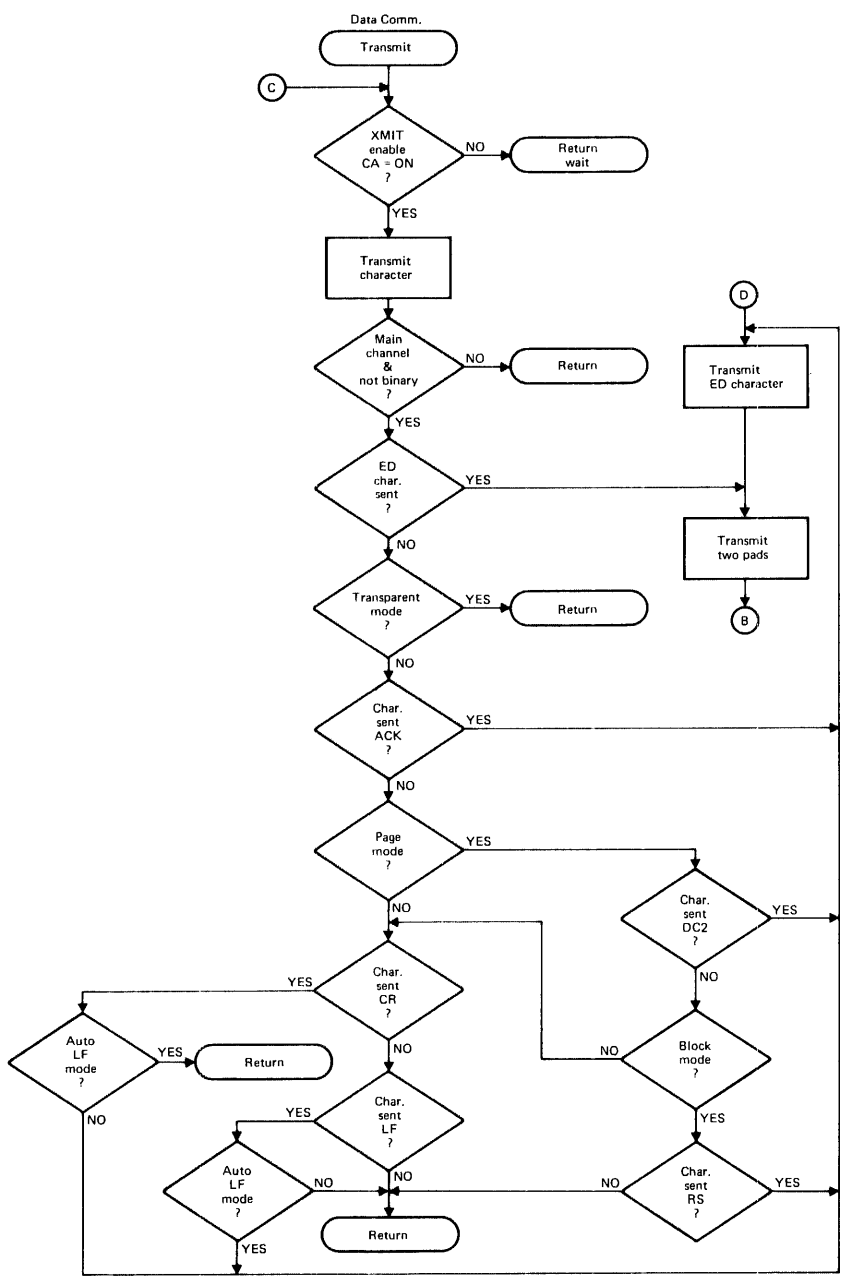


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 2 of 3)

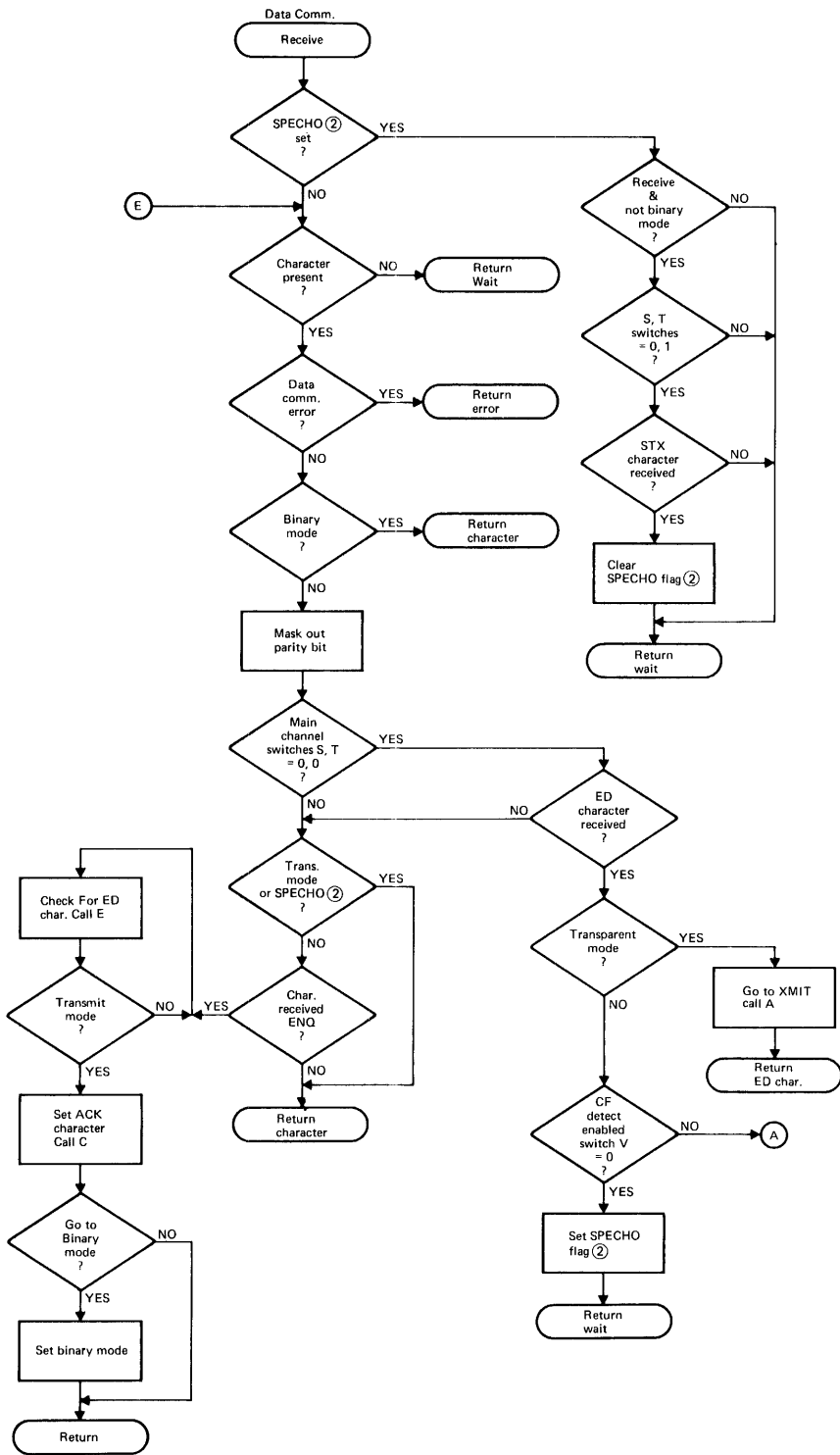


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 3 of 3)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

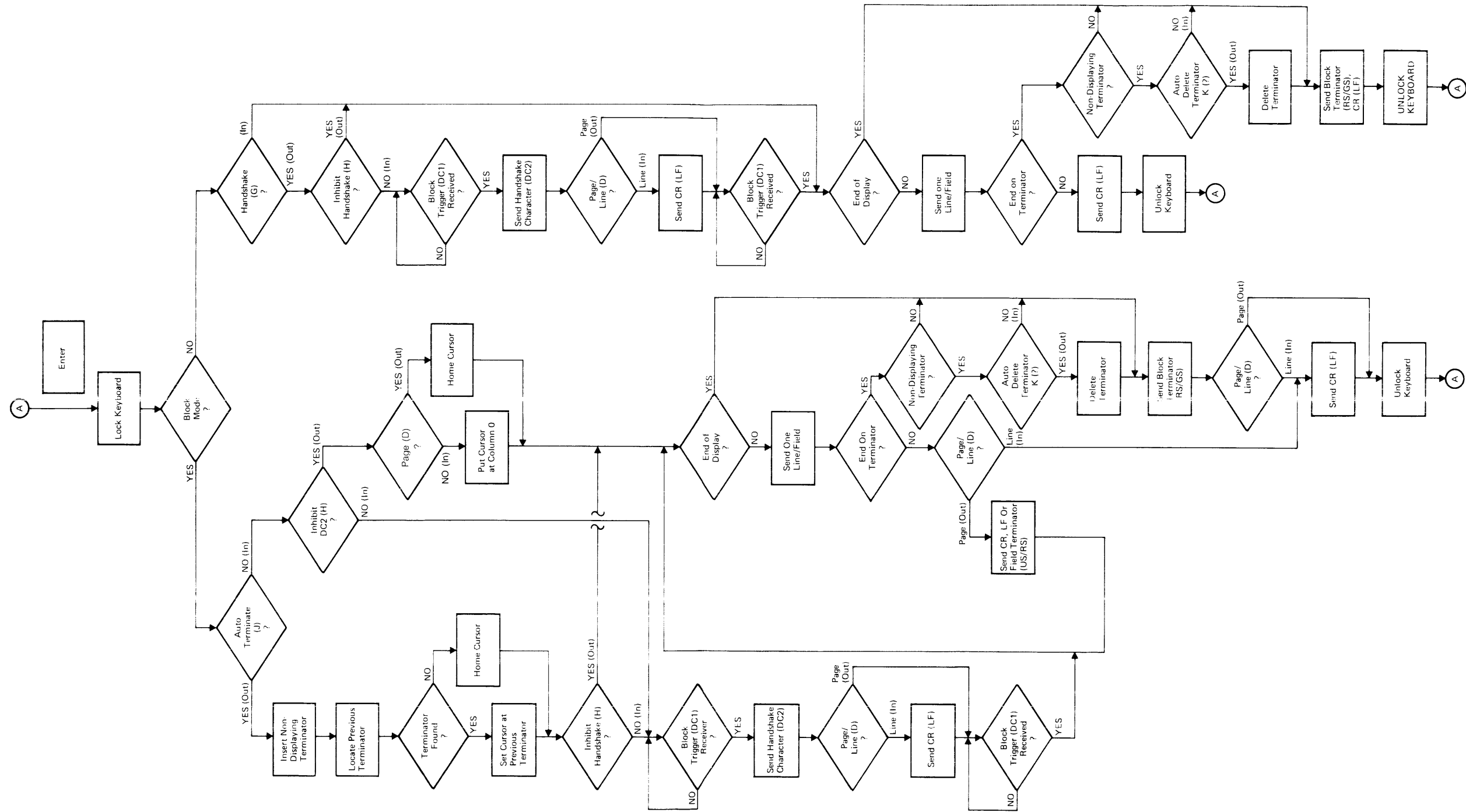


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 1 of 4)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

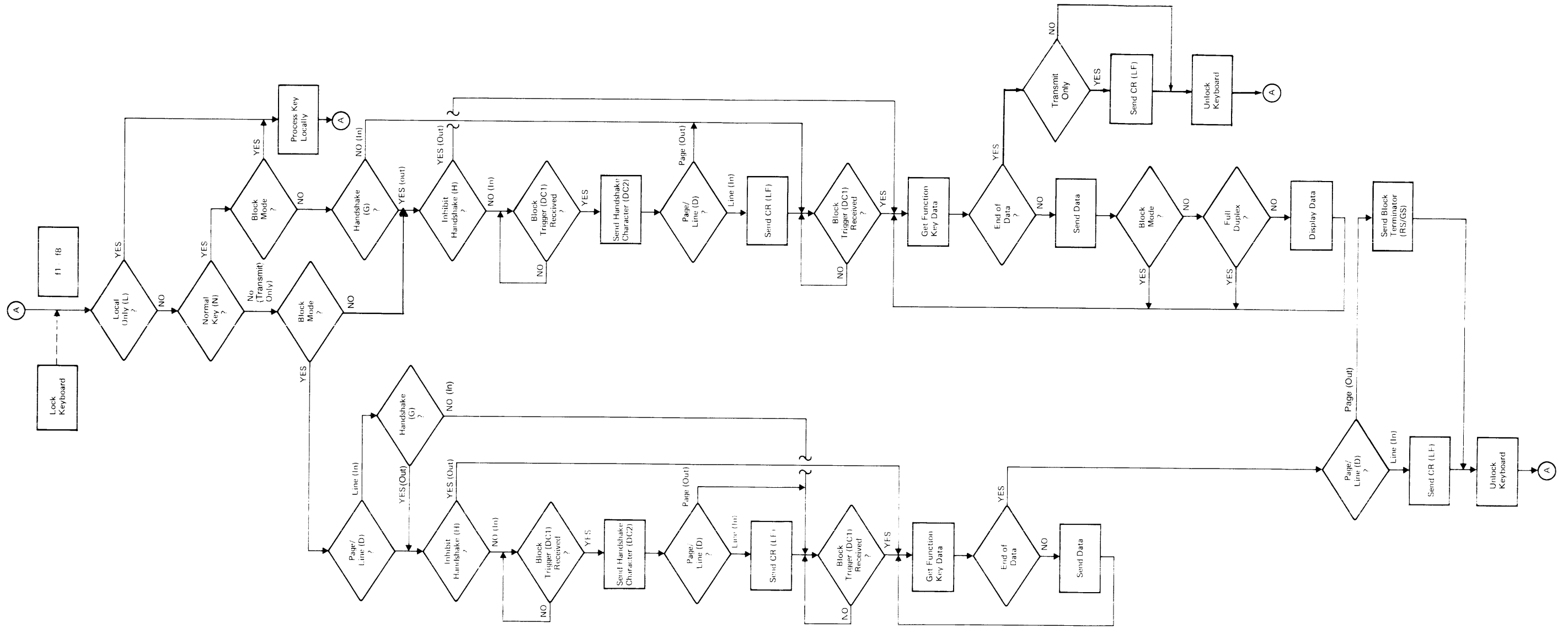


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 2 of 4)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

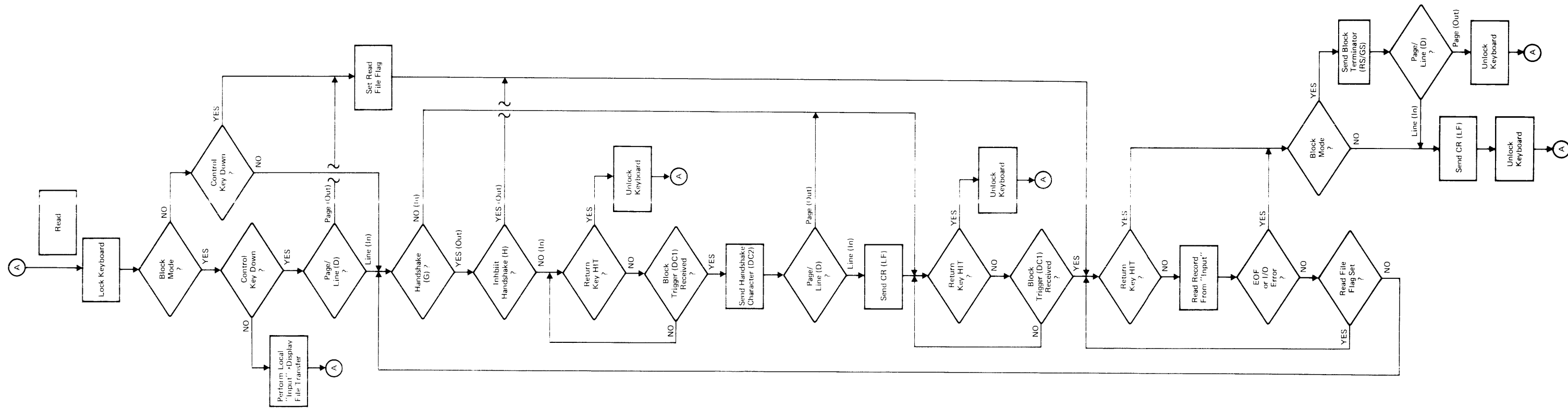


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 3 of 4)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

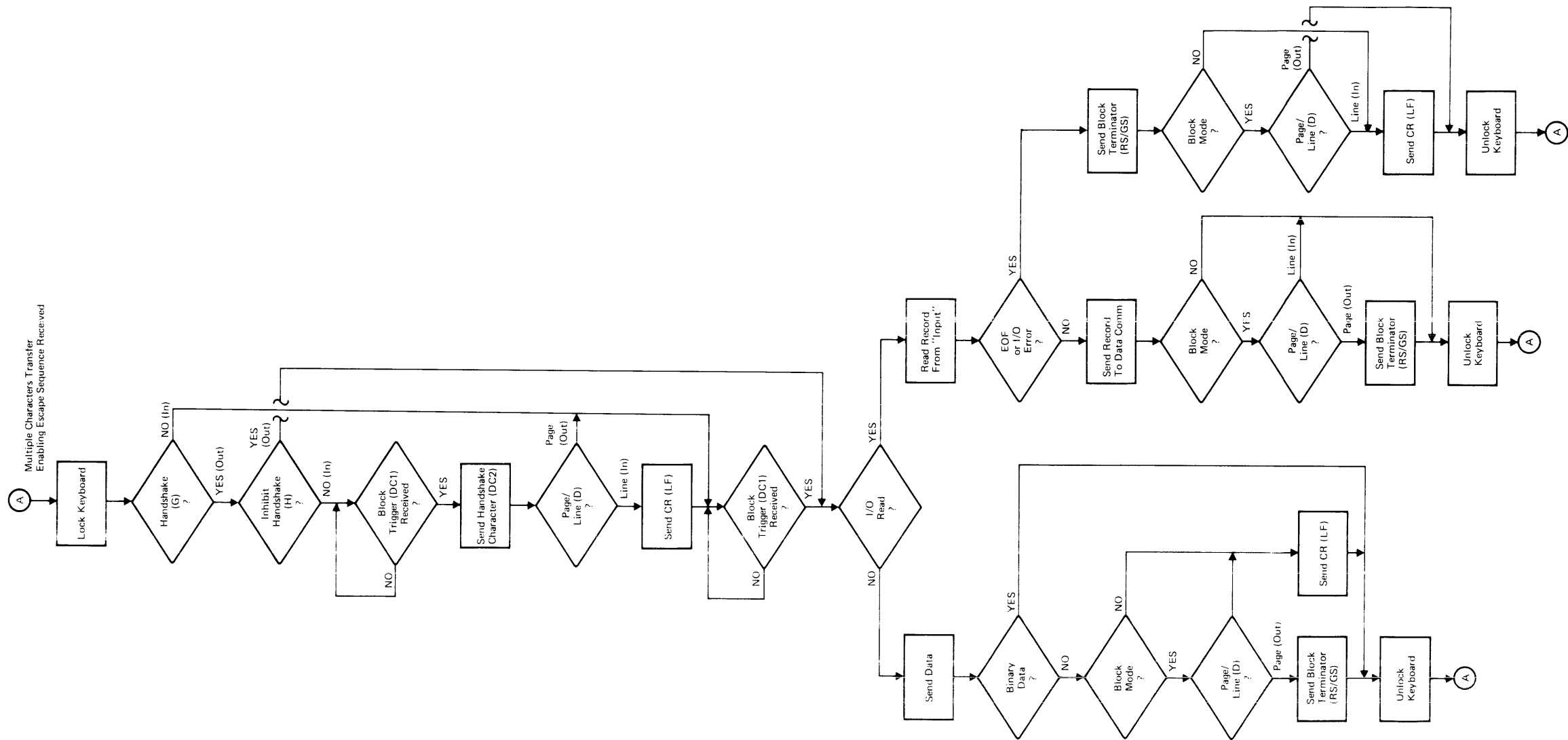


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 4 of 4)

# TAPE CARTRIDGE RETHREADING

APPENDIX

D

Tape rethreading is difficult and is not recommended unless the data recorded on the runoff tape must be recovered. Instead, when tape runoff occurs, it is recommended to replace the entire tape cartridge. The rethreading procedures contained in this paragraph are for rethreading tape onto the tape cartridge's left tape hub. If a tape run-off condition occurs from the right tape hub, use the left tape hub rethreading instructions except interchange all right-hand and left-hand instructions and change all counter-clockwise directions to clockwise directions. This procedure requires the use of a small Phillips-head screwdriver. Rethread tape onto the left tape hub as follows:

## CAUTION

Whenever the tape cartridge top cover is removed, the spring-loaded door and spring can easily slide off the door pivot post. To prevent loss of parts, ensure that door is always completely seated on its pivot post as long as the tape cartridge top cover and backplate are separated.

- Remove tape cartridge top cover by removing four screws from backplate with Phillips-head screwdriver.
- As shown in figure D-1, view A, rethread loose end of tape around right tape guide, through tape cleaner (use tweezers, if necessary), past belt drive puck, outside guide pin, and around left tape guide so that approximately 1-3/4 inches of tape is clear of guide.
- Hold tape cartridge as shown in figure D-1, view B, so that right hand can be used to rotate belt drive puck and left hand can be used to maintain tape tension at left tape guide.
- Moisten inside surface of free end of tape and, while maintaining tape tension at left tape guide, rotate belt drive puck counterclockwise to wrap free end of tape around left tape hub until tape reaches point where drive belt touches tape hub.
- While maintaining tape tension, use any small round-tipped tool to trap free end of tape between drive belt and left tape hub as shown in figure D-1, view C.
- Rotate belt drive puck counterclockwise until tape is wrapped several times around left tape hub past first set of tape holes (approximately two feet).

- Replace tape cartridge top cover on backplate and secure in place with four screws.
- Condition tape ( **CNTL** **TAPE TEST** , **f5** or **f6** ).

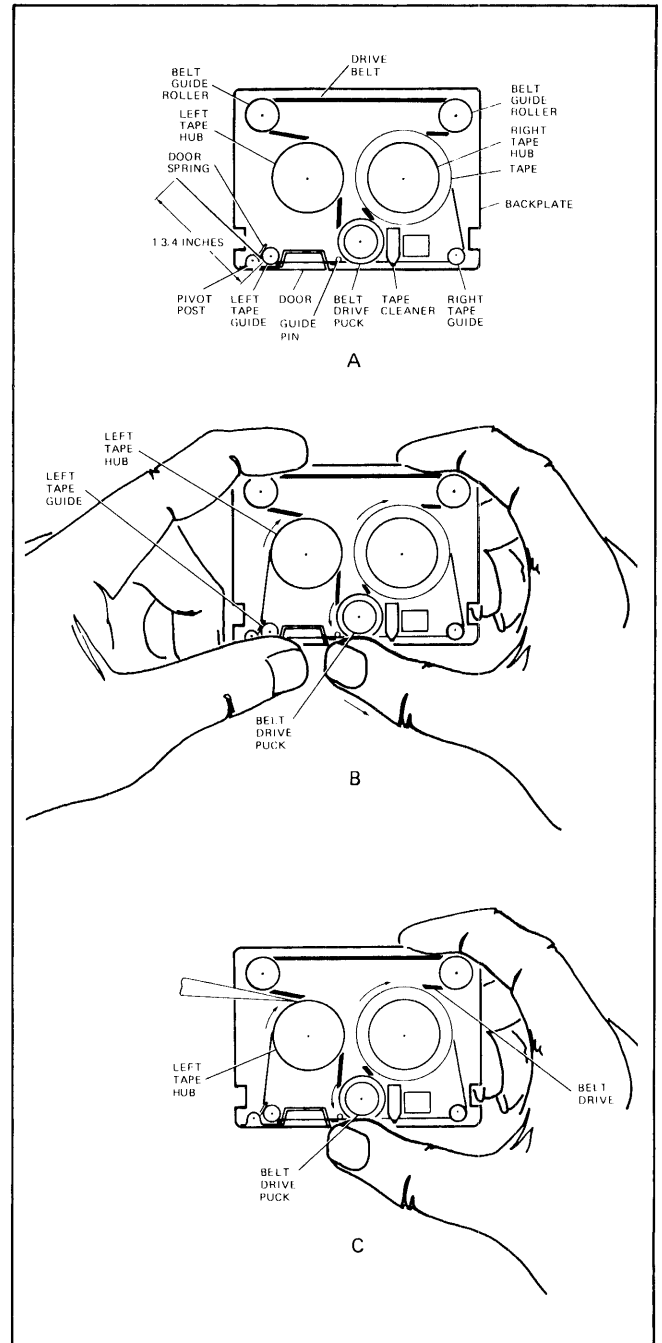


Figure D-1. Tape Cartridge Rethreading

13231 Display Enhancements	7-6
13232 Cable Assemblies	7-9
13234 Memory Module (4K)	7-10
13236 Tape Installation	7-12
13238 Terminal Duplex Register Installation	7-12
13245 Character Set Generation Kit Installation	7-12
13250 Serial Printer Interface Installation	7-13
13254 Video Interface Installation	7-13
13260 Data Communications Interfaces, description	5-1, 5-3
13260 Data Communications Interfaces, installation	7-13
13261 Device Support Firmware installation	7-15
13296 Shared Peripheral Interface installation	7-16
13349 Printer Subsystem installation	7-17
202 Modem compatibility	5-9
9871 Printer installation	7-17

## A

absolute, ASCII	3-16
absolute, binary	3-16
accessory installation	7-6
ACK	5-6
ACK1	5-24, 5-27, 5-30, 5-37, 5-38
address, terminal	5-35, 7-24, 7-25
addressing, absolute	2-2
addressing, column	2-2
addressing, cursor relative	2-2
addressing, screen relative	2-2
AID character	5-25
alpha field checking	2-7
alphanumeric cursor operations	2-4
alternate character sets	2-8
area fill, absolute and relocatable	3-12
area pattern, definition	3-10
area pattern	3-8
ASCII (7-bit) read operations	4-9
ASCII (7-bit) write operations	4-12
ASCII absolute	3-16
ASCII character set table	C-2
ASCII code selection	5-24, 7-24, 7-25
ASCII incremental	3-16
ASCII relocatable	3-16
ASCII to EBCDIC code conversion table	C-3
ASCII	3-15
Asynchronous Multipoint Interface (13260C) straps	5-40, 7-24
Attention ID (AID)	5-25
AUTO LF key programming	2-11

## B

back tab	2-4
backspace	2-4
baud rate selection (13260B)	5-19, 5-22
BAUD RATE switch (point-to-point)	5-4

baud rate switches (block)	5-33
BCC	5-23, 5-25, 7-24, 7-25
bell	2-15
binary (8-bit) read operations	4-11
binary (8-bit) write operations	4-13
binary absolute	3-17
binary incremental	3-17
binary operation (block)	5-26
binary relocatable	3-18
binary short incremental	3-17
blinking	2-8
block check character (BCC)	5-24, 5-25, 7-24, 7-25
block check mode	5-25
block checking	5-24
BLOCK MODE key programming	2-11
BLOCK MODE key	5-4
Block Mode, ENTER key	5-9
Block Mode, G and H straps	5-9
block mode	5-9, 5-10
block moves (text)	2-6
block operation	5-23
block protocol control characters operation	5-29
block protocol control characters summary	5-31
block protocols	5-6, 5-23
block, control mode	5-29
block, text-input mode	5-30
block, text-out mode	5-29
BREAK key	5-4, 5-29, 5-30
buffer overflow error (EOT)	5-28, 5-30, 7-45
buffer size	5-25, 5-33, 7-25, 7-25
buffer size	5-25, 5-33, 7-25, 7-25
buffer, data communications	5-7
bus, general description of	1-3

## C

cable fabrication	7-34
cable lengths, data communications	7-28
cable lengths	7-34
cable types	7-26
cables	7-8
CAPS LOCK key programming	2-11
cartridge tape control operations	4-3
cartridge tape operations, Compatibility Mode	3-36
cartridge tape rethreading	D-1
cartridge tape, plotting from	3-29
CCA character	5-25
CCITT signals	7-27
centering graphics text	3-22
character cell, general description of	1-3
character checking	5-24
character code chart	B-8
character delete	2-5
Character Mode (block)	5-23
Character Mode (point-to-point)	5-7
character protocols	5-6
Character Set Generator installation	7-12



character set selection ..... 2-8  
 character sets, general description of ..... 1-4  
 character sets ..... 2-8  
 checkerboard pattern ..... 3-10  
 clear display ..... 2-4  
 clear line ..... 2-4  
 clear mode ..... 3-6  
 clear tab ..... 2-4  
 COBOL program data (autoplot) ..... 3-27  
 code selection (ASCII/EBCDIC) ..... 5-24  
 code, device ..... 4-1  
 coding the Large Character Set ..... B-10  
 commands, graphics ..... 3-3  
 communication protocols ..... 5-6  
 communication test procedure ..... 7-43  
 communications, point-to-point ..... 5-6  
 communications ..... 5-1  
 compare all ..... 4-7  
 compare files ..... 4-7  
 compare record ..... 4-6  
 Compatibility Mode graphics data coding ..... 3-37  
 Compatibility Mode strapping ..... 3-33  
 Compatibility Mode ..... 3-32  
 complement mode, graphics ..... 3-6  
 computers, connecting to ..... 5-1  
 condition tape ..... 4-4  
 configuration (block) ..... 5-39  
 configuration (point-to-point) ..... 5-14  
 configuration status, block ..... 5-39  
 connecting terminals ..... 5-1  
 control character protocol ..... 5-15  
 control codes, graphics ..... 3-3  
 control functions, terminal ..... 2-11  
 control sequences (block) ..... 5-27, 5-29  
 controlling the display ..... 2-1  
 copy all ..... 4-6  
 copy file ..... 4-5  
 copy record ..... 4-5  
 CRC ..... 5-24  
 Current Cursor Address (CCA) ..... 5-25  
 current loop cable (13232F) ..... 7-8, 7-29  
 current loop PCA ..... (refer to 13260B Accessory)  
 current loop self test ..... 7-44  
 current loop signal names ..... 7-35  
 cursor addressing ..... 2-2  
 cursor control, graphics ..... 3-4  
 cursor movements, general description of ..... 1-5  
 cursor positioning ..... 2-2  
 cursor sensing ..... 2-2  
 cursor/display operations ..... 2-4  
 curve shading, autoplot ..... 3-30

## D

data checking (communications) ..... 5-24  
 data checking (fields) ..... 2-7  
 data comm error ..... 7-43  
 data comm, plotting from ..... 3-29  
 data communications, general description of ..... 1-5  
 data communications ..... 5-1  
 data format, ASCII absolute ..... 3-16

data format, ASCII incremental ..... 3-16  
 data format, ASCII relocatable ..... 3-16  
 data format, autoplot ..... 3-27  
 data format, binary absolute ..... 3-16  
 data format, binary incremental ..... 3-17  
 data format, binary relocatable ..... 3-18  
 data format, binary short incremental ..... 3-17  
 data format, binary ..... 3-16  
 data format, Compatibility Mode ..... 3-35  
 data format, default ..... 3-15  
 data format, packed (see binary) ..... 3-16  
 data overflow (block) ..... 5-30, 5-44, 5-45  
 data transfers (block) ..... 5-8, 5-36  
 data transfers, HP-IB to terminal devices ..... 4-16  
 data transfers, computer to HP-IB device ..... 4-16  
 data transfers, datacomm-to-device ..... 4-12  
 data transfers, device-to-datacomm ..... 4-8  
 data transfers, graphics ..... 4-15  
 data transfers, HP-IB device to computer ..... 4-16  
 datacomm flowcharts ..... C-1  
 datacomm flowcharts ..... C-1  
 datacomm interface installation ..... 7-13  
 datacomm self-test ..... 7-43  
 datacomm self test, multipoint ..... 7-47  
 datacomm self test, point-to-point ..... 7-44  
 datacomm self test disable ..... 5-13, 5-33  
 DC1 character inhibit ..... 5-12  
 DC1 character ..... 5-7, 5-8  
 DC2 character inhibit ..... 5-12  
 DC2 character ..... 5-7, 5-8  
 default data format ..... 3-15  
 default parameters, graphics ..... 3-14  
 defining soft keys ..... 2-13  
 delays in graphics operations ..... 3-6  
 delete character ..... 2-5  
 delete line ..... 2-5  
 device code ..... 4-1  
 device control completion codes ..... 4-2  
 device control operation, interruptable ..... 4-2  
 device control operation, non-interruptable ..... 4-2  
 device control ..... 4-1  
 Device ID ..... 5-35  
 device selection, defaults ..... 4-3  
 device selection ..... 4-2  
 device status ..... 6-10  
 Device Support Firmware (13261) installation ..... 7-15  
 device-to-device operations ..... 4-5  
 diagonal lines, define pattern ..... 3-10  
 diagonal lines ..... 3-8  
 direction, graphics text ..... 3-21  
 display clear ..... 2-4  
 display control, graphics ..... 3-4  
 display control ..... 2-1  
 Display Enhancements (13231A), installation ..... 7-6  
 display enhancements and graphics data ..... 3-1  
 Display Enhancements ..... 2-8, 7-39  
 display memory functions ..... 2-1  
 display on/off, graphics ..... 3-5  
 display, alphanumeric, general description of ..... 1-1  
 display, graphics, addressing points ..... 3-1  
 display, graphics, general description of ..... 1-1  
 display, transfer to computer (block) ..... 2-15

distance, cabling limits ..... 7-34  
 DLE ..... 5-27  
 drawing modes, graphics ..... 3-6  
 Driver Mode ..... 5-46  
 Duplex Register installation ..... 7-12  
 DUPLEX switch ..... 5-4

## E

EBCDIC code selection ..... 5-24, 7-24, 7-25  
 edit operations ..... 2-5  
 editing in Forms Mode ..... 2-7  
 end-of-data mark, writing ..... 4-4  
 end-of-data, locating ..... 4-4  
 ENQ (block) ..... 5-27  
 ENQ (point-to-point) ..... 5-6  
 ENTER key ..... 5-4  
 EOT ..... 5-28  
 error tests ..... 7-43  
 errors, autoplots ..... 3-31  
 escape sequence, device control ..... 4-1  
 ETB ..... 5-28  
 ETX ..... 5-28  
 exponential (autoplots menu data) ..... 3-26  
 Extended Asynchronous  
   Interface (13260B) straps ..... 5-19, 7-23  
 Extended Text features ..... 5-38

## F

f1-f8 keys, programming of ..... 2-12  
 fail ..... 7-43  
 field checking ..... 2-7  
 fields ..... 2-7  
 file mark, recording ..... 4-4  
 files, comparing ..... 4-7  
 files, copying ..... 4-5  
 files, finding ..... 4-4  
 files, locating ..... 4-4  
 files, skipping ..... 4-4  
 fill area, absolute and relocatable ..... 3-12  
 firmware, general description of ..... 1-4  
 floating point (autoplots menu data) ..... 3-26  
 form, sample ..... 2-11  
 Format Mode ..... 2-6  
 Forms Mode ..... 2-6  
 forms, building ..... A-3  
 forms, creating ..... 2-6, 2-8  
 from device selection ..... 4-2  
 full duplex operation ..... 5-9

## G

graphics commands, length of ..... 3-15  
 graphics commands ..... 3-3  
 graphics cursor control ..... 3-4  
 graphics data transfers ..... 3-19, 4-15  
 graphics display on/off ..... 3-5  
 graphics display set/clear ..... 3-5  
 graphics display, addressing points ..... 3-1  
 graphics drawing modes ..... 3-6

graphics functions, keyboard ..... 3-1  
 graphics functions, recording of ..... 3-19  
 graphics hard copy operations ..... 4-15  
 graphics input terminator ..... 3-33  
 graphics keys, description of ..... 3-2  
 graphics keys, location of ..... 3-1  
 graphics memory control ..... 3-5  
 graphics operations, inserting delays ..... 3-6  
 graphics sequence termination ..... 3-15  
 graphics sequence types ..... 3-3  
 graphics status ..... 6-6  
 graphics text centering ..... 3-22  
 graphics text direction ..... 3-21  
 graphics text margin ..... 3-22  
 graphics text on/off ..... 3-22  
 graphics text size ..... 3-21  
 graphics text slant ..... 3-21  
 graphics text, justifying ..... 3-22  
 graphics text ..... 3-20  
 grid, autoplots ..... 3-25  
 grounding ..... 7-5  
 Group ID ..... 5-35  
 Group Poll, character distortion in ..... 5-45  
 group polling ..... 5-37  
 group select ..... 5-37

## H

half bright ..... 2-8  
 half duplex operation ..... 5-9  
 hardcopy operations ..... 3-19  
 high-speed operation ..... 5-6  
 home down cursor ..... 2-4  
 home up cursor ..... 2-4  
 horizontal lines, define pattern ..... 3-8, 3-10  
 HP-IB addressing ..... 4-15  
 HP-IB cabling considerations ..... 8-2  
 HP-IB configurations ..... 8-1  
 HP-IB device addressing ..... 8-3, 8-4  
 HP-IB Initialization ..... 4-16  
 HP-IB Interface Adapter ..... 8-1  
 HP-IB Loads ..... 8-2  
 HP-IB operational considerations ..... 8-3  
 HP-IB operations, program control ..... 4-15, 4-16  
 HP-IB status ..... 4-16  
 HP-IB testing ..... 8-4  
 HP-IB timeout ..... 4-16

## I

ID Number ..... 5-35  
 I/O subsystem, general description of ..... 1-5  
 incremental, ASCII ..... 3-16  
 incremental, binary ..... 3-17  
 input buffer (communications) ..... 5-25, 7-24, 7-25  
 input/output modules, general description of ..... 1-3  
 insert character ..... 2-5  
 insert line ..... 2-5  
 installation, 13296 Shared Peripheral Interface ..... 7-16  
 installation, accessory ..... 7-6  
 installation ..... 7-1

## Index

integer (autoplot menu data) ..... 3-26  
interface cable signals ..... 7-26  
interface signals ..... 5-3  
interfaces, data communications ..... 5-1, 7-13  
inverse video ..... 2-8

## J

jam mode, graphics ..... 3-6  
justifying graphics text ..... 3-22

## K

keyboard disable ..... 2-15  
keyboard enable ..... 2-15  
keyboard graphics functions ..... 3-1  
Keyboard Interface PCA strapping,  
multipoint ..... 5-40, 5-42, 7-21  
Keyboard Interface PCA strapping,  
point-to-point ..... 5-12, 5-20, 7-19  
Keyboard Interface PCA switch summary ..... 5-3  
Keyboard Interface PCA switches (block) ..... 5-32  
Keyboard Interface switches, programming ..... 2-12  
keyboard overlay installation ..... 7-16  
keyboard, general description of ..... 1-4

## L

labeling autoplot graphs ..... 3-29  
labels, graphics text ..... 3-23  
Large Character Set coding ..... B-10  
Large Character Set installation ..... 7-6  
Large Character Set utility program ..... A-7  
Large Character Set ..... 2-10  
latching keys, programming ..... 2-11  
line delete ..... 2-5  
Line Drawing character set ..... 2-10  
Line Drawing Set installation ..... 7-6  
line feed ..... 2-4  
line pattern, defining ..... 3-9  
Line Select ..... 5-38  
line strap ..... 5-10  
line type ..... 3-8  
locate end-of-data ..... 4-4  
logic levels ..... 7-26  
LRC ..... 5-24

## M

Main Channel protocol ..... 5-14, 5-15  
margin, graphics text ..... 3-22  
margins, Compatibility Mode ..... 3-32  
margins ..... 2-3, 2-4  
Math Character Set installation ..... 7-6  
Math Character Set ..... 2-10  
memory addressing ..... 2-1  
memory configuration ..... 7-11  
memory control, graphics ..... 3-5  
memory link structure,  
general description of ..... 1-6

memory lock ..... 2-4  
memory management, general description of ..... 1-5  
memory, general description of ..... 1-3  
menu, autoplot ..... 3-24  
message blocks ..... 5-23  
modem disconnect command ..... 2-16  
modems ..... 5-5  
modes, graphics drawing ..... 3-6  
Monitor Mode, sample data transfers ..... 5-45  
Monitor Mode ..... 5-14, 5-44, 7-15  
moving text blocks ..... 2-6  
multicharacter transfers (block) ..... 5-23  
multicharacter transfers (point-to-point) ..... 5-7  
Multiplot ..... 3-24  
multipoint communications ..... 5-1  
multipoint example ..... A-2  
multipoint strapping flowchart ..... 5-40  
multipoint ..... 5-34

## N

NAK ..... 5-27  
network configurations ..... 5-2  
networks ..... 5-1  
next page ..... 2-4  
NOP, graphics ..... 3-15  
null characters ..... 5-7  
numeric field checking ..... 2-7

## O

opening the terminal ..... 7-2  
options and accessories ..... B-9  
origin, relocatable ..... 3-13  
output buffer (communications) ..... 5-25

## P

PA key ..... 5-34  
PAD ..... 5-28  
Page Full Break ..... 3-33  
Page Full Busy ..... 3-33  
page strap ..... 5-10  
page strapping in Format Mode ..... 5-11  
page ..... 2-4  
parameters, graphics default ..... 3-14  
parameters, graphics ..... 3-3  
PARITY switch (block) ..... 5-33  
PARITY switch (point-to-point) ..... 5-4  
pattern, area ..... 3-8  
pattern, checkerboard ..... 3-10  
pattern, defining line ..... 3-9  
pattern, predefined ..... 3-8  
pattern, user defined ..... 3-8  
patterns, graphics drawing ..... 3-8  
pen control ..... 3-15  
PF key ..... 5-34  
plot options, autoplot ..... 3-25  
plot specifications, autoplot ..... 3-24  
Plotter (9872A), sample program ..... A-1  
plotting commands summary ..... 3-14

plotting tabular data ..... 3-24  
 point plot (line type pattern) ..... 3-8  
 point-to-point configuration ..... 5-14  
 Point-to-Point strapping flowchart ..... 5-19  
 point-to-point ..... 5-6  
 polling ..... 5-36  
 positioning the cursor ..... 2-2  
 Power Down Protect Cable ..... 7-29  
 power supply adjustment ..... 7-17  
 precision (autoplot) ..... 3-27  
 predefined pattern ..... 3-8  
 previous page ..... 2-4  
 primary terminal status ..... 6-2  
 printer control operations ..... 4-8  
 printer installation (9871) ..... 7-17  
 printer status ..... 6-11  
 processor, general description of ..... 1-1  
 program load command ..... 2-16  
 programmable graphics functions ..... 3-1  
 programmable keys ..... 2-12  
 Programmers Reference Table ..... B-3  
 programming Keyboard Interface switches ..... 2-12  
 programming the latching keys ..... 2-11  
 programming the soft keys ..... 2-12  
 protected fields ..... 2-7  
 protocol, block ..... 5-23  
 protocol, Main Channel ..... 5-14  
 protocol, Reverse Channel ..... 5-14  
 protocols ..... 5-6

## R

RAM error ..... 7-39  
 RANGE switch ..... 5-4  
 range, autoplot ..... 3-27  
 range, data ..... 3-16  
 raster dump commands ..... 3-19  
 raster dump examples ..... 3-19  
 raster dump output format ..... 3-19  
 raster dump ROM installation ..... 7-16  
 raster dump speed ..... 3-19  
 raster dump, tape used in ..... 3-19  
 raster dump ..... 3-19, 4-15  
 raster scan, general description of ..... 1-3  
 read operations, ASCII (7-bit) ..... 4-9  
 read operations, binary ..... 4-11  
 read operations, fast binary (program load) ..... 4-12  
 records, comparing ..... 4-6  
 records, copying ..... 4-5  
 records, skipping ..... 4-3  
 relocatable binary ..... 3-18  
 relocatable origin ..... 3-13  
 relocatable, ASCII ..... 3-16  
 REMOTE key programming ..... 2-11  
 REMOTE key ..... 5-4  
 reset terminal (full) ..... 2-15  
 reset terminal (soft) ..... 2-15  
 reset ..... 2-15  
 rethreading tapes ..... D-1  
 RETURN key programming ..... 2-13  
 RETURN ..... 2-4

Reverse Channel Protocol ..... 5-14, 5-17  
 rewinding tapes ..... 4-3  
 roll down ..... 2-4  
 roll up ..... 2-4  
 ROM error ..... 7-39  
 RS232C signals ..... 7-27  
 rubber band line ..... 3-14  
 RVI ..... 5-28

## S

scale switch ..... 5-4  
 scale, line pattern ..... 3-9  
 scaled Compatibility Mode ..... 3-33  
 secondary terminal status ..... 6-4  
 selection (multipoint) ..... 5-37  
 selection of character sets ..... 2-8  
 Self-Test command ..... 2-16  
 Self-Test disable, datacomm ..... 5-13, 5-33  
 Self-Test procedure ..... 7-39  
 send display ..... 2-15  
 sensing the cursor position ..... 2-2  
 Serial Printer Interface installation ..... 7-13  
 set mode, graphics ..... 3-6  
 set tab ..... 2-4  
 short incremental, binary ..... 3-17  
 signal levels ..... 7-27  
 signal line control ..... 5-9  
 signals interface ..... 7-26  
 signals, interface ..... 5-3  
 soft key applications ..... 2-14  
 soft key labels ..... 2-14  
 soft key programming ..... 2-12  
 soft key triggering ..... 2-14  
 soft reset ..... 2-15  
 space compression ..... 5-26  
 specifications ..... B-8  
 speed switches (block) ..... 5-33  
 status, block configuration ..... 5-38  
 status, device ..... 6-10  
 status, graphics ..... 6-6  
 status, interpretation of ..... 6-1  
 status, terminal primary ..... 6-2  
 status, terminal secondary ..... 6-4  
 status, terminal ..... 6-1  
 status ..... 6-1  
 strap options (block) ..... 5-24  
 strapping option selection ..... 7-17  
 strapping, Asynchronous Multipoint  
     Interface (13260C) ..... 7-24  
 strapping, Comaptibility Mode ..... 3-33  
 strapping, Extended Asynchronous  
     Interface (13260B) ..... 7-23  
 strapping, G and H straps ..... 5-9  
 strapping, Line/Page ..... 5-10  
 strapping, multipoint flowchart ..... 5-40  
 strapping, multipoint, Keyboard Interface PCA ..... 5-42  
 strapping, Page in Format Mode ..... 5-11  
 strapping, Point-to-Point ..... 5-12, 5-19  
 strapping, Shared Peripheral Interface (HP-IB) ... 7-16

strapping, Synchronous Multipoint  
 Interface (13260D) ..... 7-25  
 STX ..... 5-27  
 switch settings, data communications ..... 5-4, 5-32  
 SYN ..... 5-28  
 sync characters ..... 5-26  
 synchronization, graphics ..... 3-15  
 synchronous compatibility (block) ..... 5-38  
 Synchronous Multipoint  
 Interface (13260D) straps ..... 7-25

**T**

tab ..... 2-4  
 tabbing in Forms Mode ..... 2-7  
 tabs, clearing ..... 2-4  
 tabs, setting ..... 2-3, 2-4  
 tabs, using ..... 2-3, 2-4  
 tabs ..... 2-3, 2-4  
 tabular data, plotting ..... 3-24  
 tape conditioning ..... 4-4  
 tape installation ..... 7-12  
 tape rethreading ..... D-1  
 tape runoff ..... D-1  
 tape status ..... 6-11  
 tape storage capacity ..... B-2  
 tape test procedure ..... 7-43  
 tape test ..... 4-4  
 tape units, general description of ..... 1-6  
 tapes, rewinding ..... 4-3  
 teletype compatible operation ..... 5-9  
 terminal address ..... 5-35  
 terminal architecture, general description of ..... 1-2  
 terminal control functions ..... 2-11  
 terminal ID ..... 5-35  
 terminal mainframe, general description of ..... 1-1  
 terminal networks ..... 5-1, 7-9  
 terminal self-test command ..... 2-16  
 terminal status ..... 6-1  
 terminals, connecting ..... 5-1  
 terminating graphics sequences ..... 3-15  
 test patterns ..... 7-39  
 testing tapes from computer program ..... 4-4  
 text moving ..... 2-6

text slant, graphics ..... 3-21  
 text, Compatibility Mode ..... 3-36  
 text, graphics ..... 3-20  
 to device selection ..... 4-2  
 TRANSMIT indicator ..... 5-4  
 transmit only fields ..... 2-7  
 Transparency Mode (binary operation) ..... 5-26

**U**

underline ..... 2-8  
 unprotected fields ..... 2-6  
 unscaled Compatibility Mode ..... 3-33  
 user defined area pattern ..... 3-10  
 user defined line pattern ..... 3-9  
 user defined pattern ..... 3-8

**V**

vectors ..... 3-15  
 vertical lines, define pattern ..... 3-8, 3-10  
 video hardcopy ..... 3-19  
 Video Interface installation ..... 7-13  
 voltage selection ..... 7-5  
 VRC ..... 5-24

**W**

WACK ..... 5-27  
 wait command ..... 2-15  
 window, graphics ..... 3-13  
 wraparound ..... 2-5  
 write operations, ASCII (7-bit) ..... 4-12  
 write operations, binary (8-bit) ..... 4-13  
 Write-Backspace-Read Mode ..... 4-5  
 WRU ..... 5-38

**Z**

zoom ..... 3-5