

reference manual

intelligent  
graphics

2647A

HEWLETT  PACKARD

reference manual

# intelligent graphics

## 2647A



---

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 2647A Intelligent 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 2647A Intelligent Graphics Terminal User's Manual* (02647-90001). The *HP 2647A Service Manual* (02647-90003) provides a discussion of troubleshooting, repair, and theory of operation.

The HP 2647A Terminal is capable of executing BASIC programs directly without the need of a host computer system. A complete description of how to access BASIC is given in the *HP 2647A User's Manual* (part no. 02647-90001). A description of the BASIC language is contained in the *Terminal BASIC Manual* (part number 02647-90005).

## 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 describes how to configure terminal networks using the Hewlett-Packard Interface Bus.



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: `␣ & p [ < "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: `Gc = bell character =  .`

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	Terminal Control Functions	2-13
<b>GENERAL DESCRIPTION</b>		Latching Keys	2-13
Introduction	1-1	Keyboard Interface Switches	2-14
Mainframe	1-1	Programmable Soft Keys	2-14
Microprocessor	1-1	Controlling the Soft Key Menu	2-14
Display	1-1	Defining Soft Keys	2-15
Terminal Bus	1-3	Displaying the Soft Key Labels	2-16
Terminal Memory	1-3	Triggering Soft Keys	2-16
Input/Output Modules	1-3	Soft Key Applications	2-16
CRT Monitor	1-3	Additional Control Functions	2-17
The Raster	1-3	Bell	2-17
Alphanumeric and Microvector Character Sets	1-4	Send Display	2-17
Display Features and Alternate		Wait	2-17
Character Sets	1-4	Keyboard Disable/Enable	2-17
Keyboard	1-4	Reset Terminal	2-17
The Firmware	1-4	Monitor Mode	2-18
System Monitor	1-4	Self-Test	2-18
Keyboard and I/O Subsystem	1-5	Modem Disconnect	2-18
Cursor Movement	1-5	Program Down Load	2-18
Display Memory Management	1-5	Display Message, Command or Soft Key	
Data Communications	1-5	Label Line	2-18
The Tape Units	1-6		
 Section II	Page	 Section III	Page
<b>DISPLAY MEMORY FUNCTIONS</b>		<b>GRAPHICS CONTROL FUNCTIONS</b>	
Introduction	2-1	Introduction	3-1
Display Memory Functions	2-1	Graphics Display	3-1
Display Control	2-1	Keyboard Graphics Functions	3-1
Display Window Control	2-1	Programming Graphics Functions by	
Display Workspace	2-1	Escape Sequences	3-1
Message, Command, Softkey Label Line	2-3	Control Codes	3-3
Memory Addressing Scheme	2-3	Commands	3-3
Cursor Sensing	2-4	Parameters	3-3
Cursor Positioning	2-4	Graphics Display Control	3-4
Other Cursor Operations	2-5	Graphics Cursor Control	3-4
Tabs	2-5	Zoom	3-5
Margins	2-5	Graphics Memory Control	3-5
Alphanumeric Display On/Off	2-5	Inserting Delays In Graphics Operations	3-6
Edit Operations	2-7	Graphics Drawing Mode Parameters	3-6
Insert Character With Wraparound	2-7	Drawing Modes	3-6
Delete Character With Wraparound	2-7	Drawing Patterns	3-8
Moving Text Blocks	2-8	Area Fill	3-12
Forms Mode (Format Mode)	2-8	Relocatable Origin	3-13
Protected Fields	2-8	Rubber Band Line	3-14
Unprotected Fields	2-8	Selecting the Graphics Default Parameters	3-14
Transmit Only Fields	2-9	Plotting Sequences	3-14
Data Checking	2-9	Pen Control	3-15
Tabbing	2-9	Vectors	3-15
Editing	2-9	ASCII Formats	3-15
Building the Form	2-10	Binary Formats	3-16
Sending Data to the Computer	2-10	Recording Graphics Functions	3-19
Display Enhancements	2-10	Raster Data Transfers	3-19
Alternate Character Sets	2-10	Raster Output Format	3-20
Selecting Alternate Sets	2-10	Sending Image Data from CPU to the 2647	3-20
Using Alternate Sets	2-11	Reading the Image Memory	3-23
		Summary of Commands	3-24



Section III (Continued)	Page
Graphics Hardcopy Operations .....	3-24
Graphics Text .....	3-25
Keyboard Control of Graphics Text .....	3-25
Program Control of Graphics Text .....	3-26
Multiplot .....	3-29
Plotting Pie Charts .....	3-29
Plotting Bar Charts .....	3-31
Plotting Linear Charts .....	3-33
Compatibility Mode .....	3-35
Compatibility Mode Straps .....	3-36
Graphics Data .....	3-36
Graphics Data Format .....	3-38
Text .....	3-39
Cartridge Tape Operation In	
Compatibility Mode .....	3-39

#### Section IV

### DEVICE CONTROL

Introduction .....	4-1
Using the $\text{F}_{\text{c}}$ Generalized Escape Sequence .....	4-1
Assign .....	4-4
Bye .....	4-4
Close .....	4-4
Compare .....	4-5
Condition .....	4-5
Copy .....	4-5
Enable/Disable (Edit Mode, Record Mode, or	
Verify Mode .....	4-5
Display .....	4-6
Execute .....	4-6
Exit .....	4-6
Find .....	4-6
Hello .....	4-7
Mark .....	4-7
Report .....	4-7
Resume .....	4-7
Rewind .....	4-7
Set .....	4-7
Show .....	4-8
Skip .....	4-8
Suspend .....	4-8
Tell .....	4-8
Test .....	4-8
Transfer .....	4-9
Using the $\text{F}_{\text{cp}}$ Generalized Escape Sequence .....	4-10
Selecting Input/Output Devices .....	4-11
Controlling the Devices .....	4-11
Cartridge Tapes .....	4-11
Display .....	4-12
Printer .....	4-12
Transferring Data From Device to Computer .....	4-12
ASCII Transfers .....	4-12
Binary Transfers .....	4-12
Transferring Data From Computer to Device .....	4-13
Copying a Record .....	4-13
Copying a File .....	4-13
Copying to End of Medium .....	4-14
Fast Binary Read (Program Loading) .....	4-14
Indicating Successful Completion of a	
Program-Controlled Function .....	4-14

#### 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
Device Status .....	6-6
Graphics Status .....	6-8
Read Device I.D. ....	6-8
Read Current Pen Position .....	6-8
Read Graphics Cursor Position .....	6-9
Read Cursor Position with Wait .....	6-9
Read Display Size .....	6-9
Read Device Capabilities .....	6-9
Read Graphics Text Status .....	6-10
Read Zoom Status .....	6-10
Read Relocatable Origin .....	6-10
Read Reset Status .....	6-10
Read Area Shading Capability .....	6-11
Read Graphic Modification Capabilities .....	6-11
Any Other Parameters .....	6-11
Command Status .....	6-12
File System/User Interface Errors .....	6-12
BASIC Interpreter Errors .....	6-17
AGL Errors .....	6-21

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
32K RAM Memory PCAs .....	7-10
32K Control Memory PCAs .....	7-10
HP 13238A Terminal Duplex Register .....	7-12
HP 13246A/B Printer Subsystem (9866) .....	7-12
HP 13250B Serial Printer Interface .....	7-12
HP 13254A Video Interface .....	7-12
HP 13260A,B,C,D Data Communications	
Accessories .....	7-13
13296A Shared Peripheral Interface .....	7-14
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
HP-IB Self-Test .....	7-49
Terminal-to-Terminal Loop-Back Test .....	7-49

Section VIII	Page
<b>HP-IB CONFIGURATIONS</b>	
Introduction .....	8-1
Cabling Considerations .....	8-2
Cabling Limitations .....	8-3
Operational Considerations .....	8-3
Local HP-IB Testing .....	8-3
Remote HP-IB Testing .....	8-3
Device Considerations .....	8-4
Sample Configurations .....	8-4

Appendix A	Page
<b>APPLICATIONS</b> .....	A-1
Introduction .....	A-1
Multipoint Example .....	A-1
Digitizer Program Example .....	A-2
Forms Building .....	A-4
Programming Soft Keys .....	A-4
Building the Form .....	A-4

Appendix B	Page
<b>REFERENCE TABLES</b> .....	B-1

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

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



# ILLUSTRATIONS

Title	Page	Title	Page
Terminal Architecture .....	1-2	Sample Data Transfers Using	
Character Cell .....	1-3	Reverse Channel Protocol .....	5-18
System Monitor Basic Loop .....	1-4	Point-to-Point Data Communications	
Display Memory Linked List Structure .....	1-6	Configuration Flowchart .....	5-19
Display Features .....	2-2	Examples of Block Transmissions .....	5-23
Three Examples of Display Memory Allocation		Operation of Block Protocol	
to the Four Display Workspaces .....	2-2	Control Characters .....	5-29
Row Addressing .....	2-4	Block Keyboard Data Communication	
Column Addressing .....	2-4	Switches .....	5-33
Character Delete With Margins .....	2-7	Terminal Addressing .....	5-36
Character Delete With Wraparound .....	2-7	Typical Configuration Status Request	
Character Set Locations .....	2-11	and Response Sequence .....	5-38
Example Using the Math Set .....	2-11	Configuration Status Byte Contents .....	5-39
Example Using the Large Character Set .....	2-11	Multipoint Data Communications	
Math Set Elements .....	2-12	Configuration Flowchart .....	5-40
Large Character Set Elements .....	2-12	Communication Line Using a Monitor .....	5-44
Line Drawing Set .....	2-12	Sample Data Transfers Displayed in	
Sample Form .....	2-13	Monitor Mode .....	5-45
Location of Graphics Keys .....	3-1	Character Distortion in Group Poll .....	5-45
Examples of Drawing Modes .....	3-7	Data Overflow Indication .....	5-45
Predefined Line Type Patterns .....	3-8	Driver Mode Configuration .....	5-46
Using Area Patterns As Line Types .....	3-9	Sample Select Sequence Using Driver Mode .....	5-47
Examples of User Defined Line Patterns .....	3-9	Control Character Display On Driver Terminal .....	5-47
Area Pattern Examples .....	3-11	Terminal Input .....	5-47
Relocatable Origin .....	3-13	Primary Terminal Status Example .....	6-2
Current Pen Position and New End Point .....	3-15	Secondary Terminal Status Example .....	6-4
Recording Graphics Sequences .....	3-19	Device Status Example .....	6-6
Raster Data Format .....	3-20	Opening the Terminal .....	7-2
Raster Data X and Y Offsets .....	3-21	Mainframe Bottom Part Locations .....	7-3
Raster Data Positioning .....	3-22	Mainframe Top Part Locations .....	7-4
Raster Data Windowing .....	3-23	Fuse Positions for 115 VAC and 230 VAC	
Graphics Text Characters .....	3-25	Line Voltage .....	7-5
Graphics Text Sizes .....	3-26	Sample Character Set Configuration .....	7-6
Graphics Text Direction .....	3-26	Display Enhancement PCA Jumper and ROM	
Graphics Text Justification .....	3-28	Socket Locations .....	7-7
Blank Pie Chart Menu .....	3-29	Top Plane Assembly Removal .....	7-7
Completed Pie Chart Menu .....	3-30	32K RAM PCA Jumper Settings .....	7-10
Blank Bar Chart Menu .....	3-31	Control Memory PCA Jumper Settings .....	7-11
Completed Bar Chart Menu .....	3-32	Typical Memory Map .....	7-11
Blank Linear Chart Menu .....	3-33	Terminal Duplex Register PCA	
Completed Linear Chart Menu .....	3-34	Jumper Configuration .....	7-12
Turning on Compatibility Mode .....	3-37	Control Memory PCA Data Comm Firmware	
Comparison of a Terminal With 1024 × 780 Display		IC Locations .....	7-14
and the HP 2648A .....	3-37	Installing Keyboard Overlays .....	7-15
Scaled Data .....	3-38	Typical Strapping Option Switch Assembly .....	7-18
Unscaled Data .....	3-38	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

Title	Page
Asynchronous Multipoint Cabling .....	7-30
Synchronous Multipoint Cabling .....	7-31
Power-Down-Protect Cabling for Asynchronous Multipoint Configuration .....	7-32
Power-Down-Protect Cabling for Synchronous Multipoint Configuration .....	7-33
Assembling the PCA Hood Connector .....	7-36
Assembling the RS232C Connector .....	7-37
Assembling the Multipoint Connector .....	7-38
Basic Self-Test Patterns .....	7-39
Basic Terminal Self-Test Flowchart .....	7-40
Basic Data Comm Self-Test Flowchart .....	7-45
Multipoint Data Comm Self-Test Flowchart .....	7-48
HP-IB Interconnecting Cable .....	8-1
HP-IB Interface Adapter .....	8-1
Selecting Additional Device Loads .....	8-3
Terminal-to-Plotter Configuration .....	8-4

Title	Page
Shared Plotter/Line Printer Configuration .....	8-5
3-terminal, Shared Plotter/Printer Configuration ...	8-5
5-terminal, Multiple Plotter/Printer Configuration ..	8-5
Sample Form .....	A-1
Soft Key Programming and Soft Key Labels for File 1 .....	A-5
Soft Key Programming and Soft Key Labels for File 2 .....	A-5
Soft Key Programming and Soft Key Labels for File 3 .....	A-5
Building a Form — Phase 1 .....	A-6
Building a Form — Phase 2 .....	A-6
Building a Form — Phase 3 .....	A-6
Point-to-Point Communications Flowcharts .....	C-6
Keyboard Communication Switches Flowcharts .....	C-9
Tape Cartridge Rethreading .....	D-1



# TABLES

Title	Page
Cursor/Display Operations .....	2-6
Edit Operations .....	2-7
Keyboard Interface Switch Summary .....	2-14
Graphics Control Keys .....	3-2
Summary of Graphics Sequence Types .....	3-3
Graphics Display Control Functions .....	3-4
Graphics Mode Commands .....	3-6
Graphics Parameter Default Values .....	3-14
Graphics Plotting Control Functions .....	3-14
Characters Used In Packed Data Formats .....	3-17
Absolute Format Addressing Bytes .....	3-18
Incremental (Short) Vector Bytes .....	3-19
Graphics Control Sequences Used in	
Record Operations .....	3-19
Summary of Raster Dump Commands .....	3-24
Graphics Text Keyboard Functions .....	3-25
Pie Chart Escape Sequences .....	3-31
Bar Chart Escape Sequences .....	3-33
Linear Chart Escape Sequences .....	3-34
Compatibility Mode Control Sequences .....	3-35
Commands for Selecting Compatibility Mode .....	3-36
Coding of Compatibility Mode Graphics Data .....	3-40
Esc Generalized Escape Sequence Syntax .....	4-2
Command Syntax Abbreviations .....	4-4
EscP Device Control Escape Sequences .....	4-10
Data Communication Interfaces .....	5-1
Data Communication Interface Capabilities .....	5-3
Keyboard Interface Switch Summary .....	5-3
Keyboard Communications Switches .....	5-4
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

Title	Page
ASCII Status Characters .....	6-1
Graphics Status Requests .....	6-8
Display Enhancement PCA Jumper Protocol .....	7-7
HP 13232 Cable Assemblies .....	7-8
Contents of HP 13260 Data Communications	
Accessories .....	7-13
HP-IB Interface Switch Settings .....	7-16
Keyboard Interface PCA Strapping	
Options for Point-to-Point .....	7-19
Keyboard Interface PCA Strapping	
Options for Multipoint .....	7-21
Extended Asynchronous Communications	
Interface Strapping Options .....	7-23
Asynchronous Multipoint Communications	
Interface Strapping Options .....	7-24
Synchronous Multipoint Communications	
Interface Strapping Options .....	7-25
EIA RS232C and CCITTV24 Interface	
Data and Control Signals .....	7-27
Data Communications Signal Levels .....	7-27
Multipoint Data Communication Cables .....	7-28
Parts for Fabricating Your Custom	
Data Communications Cable .....	7-34
13260 Series Data Communications PCA	
Signal Names .....	7-35
Data Communications Self-Test Connectors .....	7-44
Point-to-Point Data Communications	
Self-Test Procedure .....	7-44
Multipoint Data Communications	
Self-Test Procedure .....	7-47
HP 9874A Digitizer Driver Using	
Rubber Band Line .....	A-3
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
ASCII Character Set .....	C-2
ASCII (7-Bit) Character Codes .....	C-3
EBCDIC Character Codes .....	C-4

# 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. Program control may be either the local BASIC language (which may be loaded into the terminal from cartridge tape) or a program running in a remote computer. 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, an integrated mass storage capability of up to 220 kilobytes of data using two tape cartridges is provided. 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 BASIC running in the terminal or 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 4800 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

one slot 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 5,000 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.

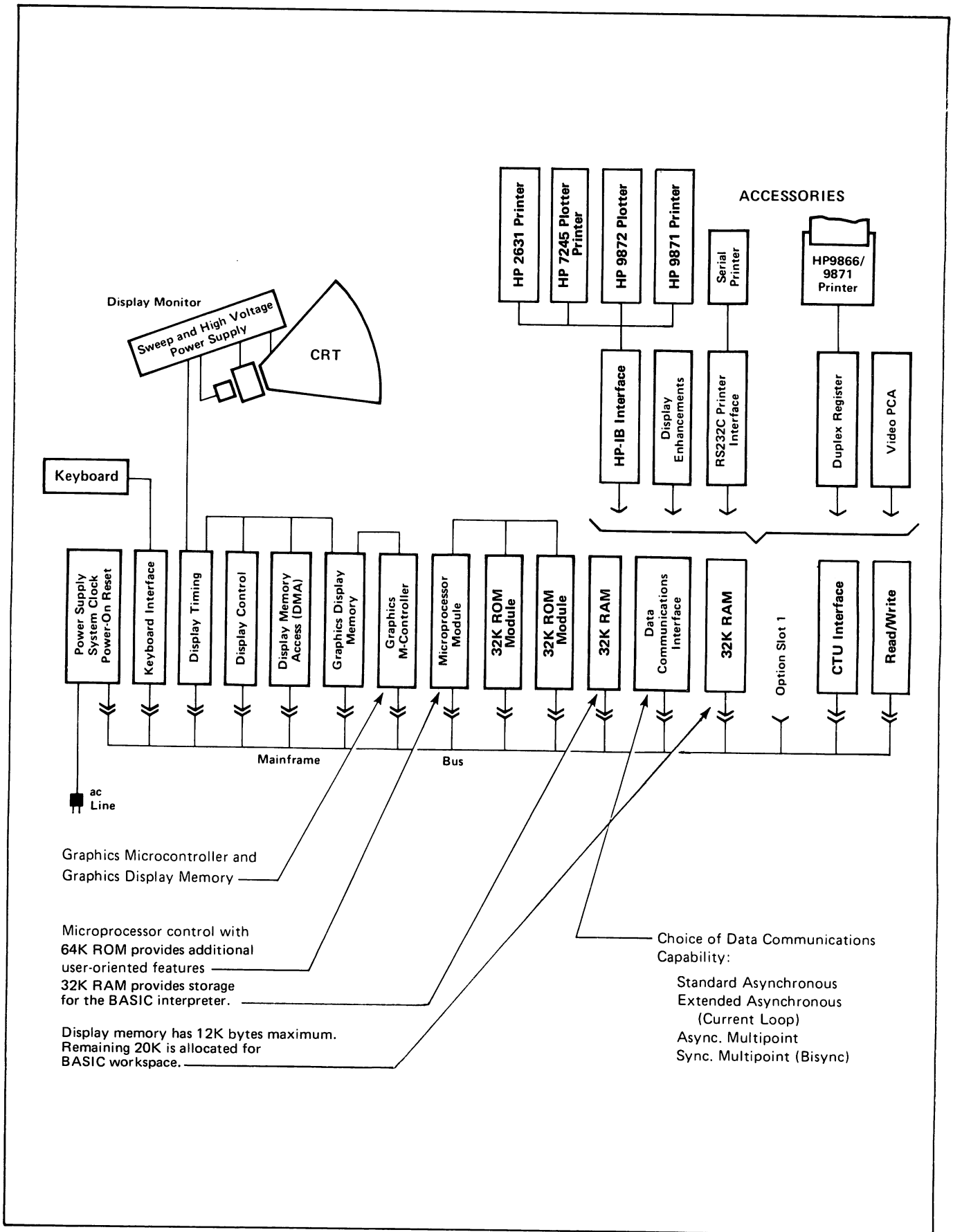


Figure 1-1. Terminal Architecture

## 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 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). Also, the BASIC Interpreter and BASIC workspace resides in RAM. Terminal programs are called firmware because the ROM makes them more permanent than software but less permanent than hardware. In the terminal one-half of the available memory is dedicated to ROM or program memory and the remaining memory locations are RAM and can be used for data and BASIC. All of the terminal memory is MOS semiconductor memory. A separate RAM memory is used to store the  $720 \times 360$  graphics 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/B and HP 9871A Printer interface), the optional serial printer interface PCA, and the HP-IB interface PCA (used for interfacing the HP 9871A-001 Printer, HP 2631A-046 Printer, HP 2631G Printer, HP 7245A-001 Plotter/Printer, HP 9872A Plotter, and other terminals).

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).

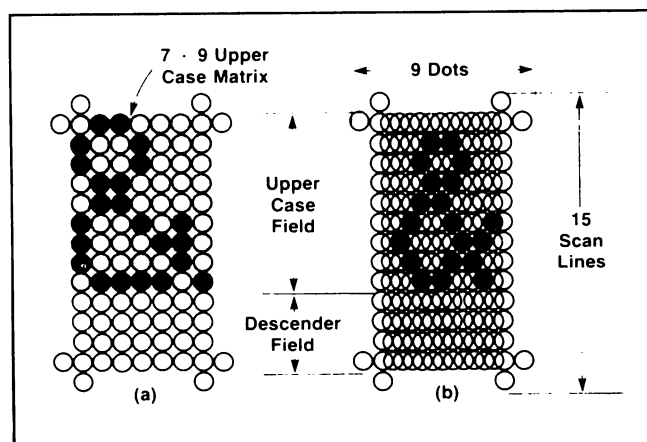


Figure 1-2. Character Cell

## 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  $7 \times 9$  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 optional 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.

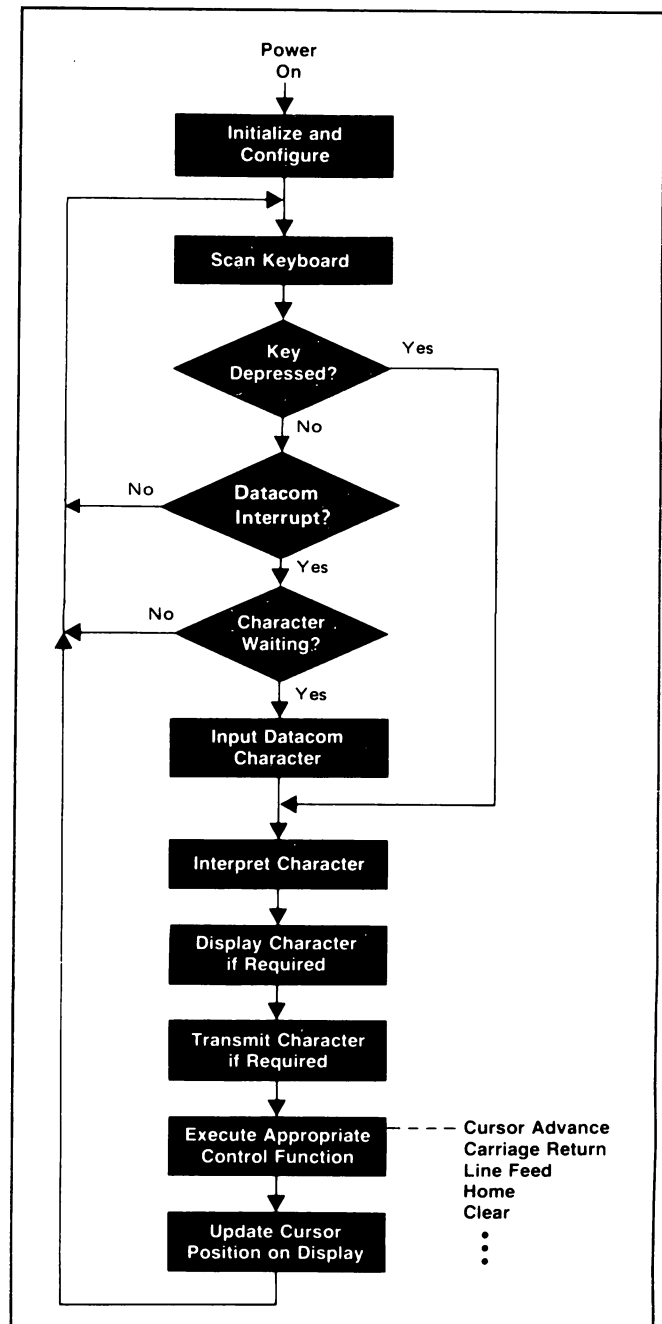


Figure 1-3. System Monitor Basic Loop

## 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

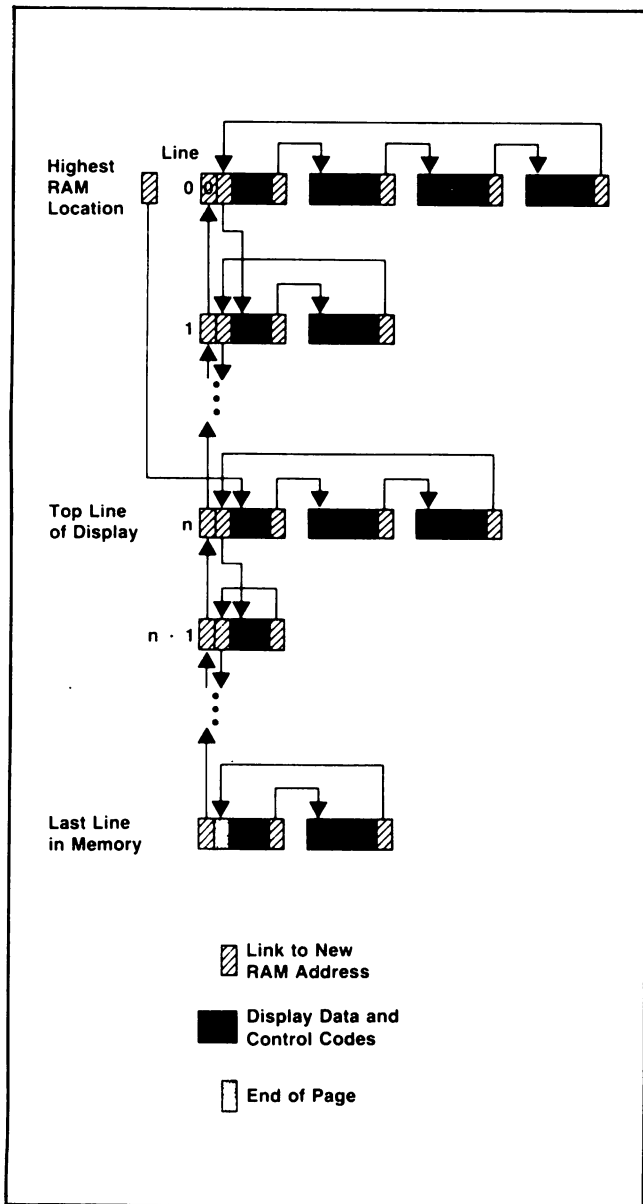


Figure 1-4. Display Memory Linked List Structure

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.



# 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 within each display workspace. 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

Cursor and display positioning operations can be programmed within each of four display workspaces. Only the alphanumeric display functions are described here. Refer to Section III for a description of the graphic display functions. The programmable functions available are as follows:

- Display Window Control
  - Message Line
  - Command Line
  - Soft Key Label Line
- 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)

**Display Window Control.** Alphanumeric display memory is shared by four dynamic display workspaces (window#1 through #4), a message line (window#5), a command line (window#6), and a soft key label line (window#7). These "windows" are controlled through the command channel. (Figure 2-1 shows the display features.)

`␣,cDIsply Window#<n>␣` displays the selected window  
`␣,cCLDse Window#<n>` removes selected window #5, #6, or #7

**Display Workspace.** Only one display workspace can be displayed at a time. Cursor and display positioning operations operate independently in each workspace (i.e., you can roll up, roll down, next page, previous page, set margins and tabs, and position the cursor independently in each workspace.) As data is entered into a display workspace, it uses some of the alphanumeric display memory. If needed, the present display workspace will "steal" lines from the other workspaces that are not memory locked. Therefore, to preserve information in another workspace, it must be memory locked. At power on or full reset, windows #2 through #4 are memory locked. (Figure 2-2 shows examples of memory allocation for the four display memory workspaces.)

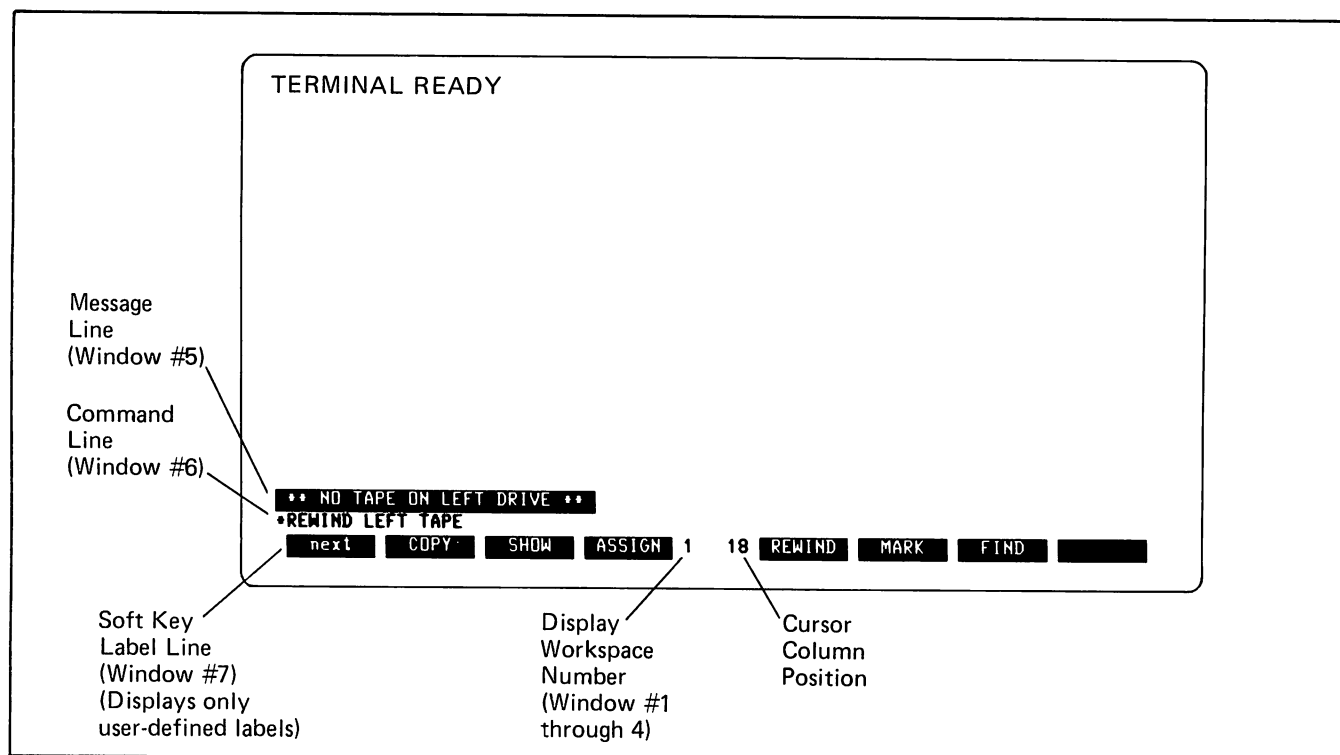


Figure 2-1. Display Features

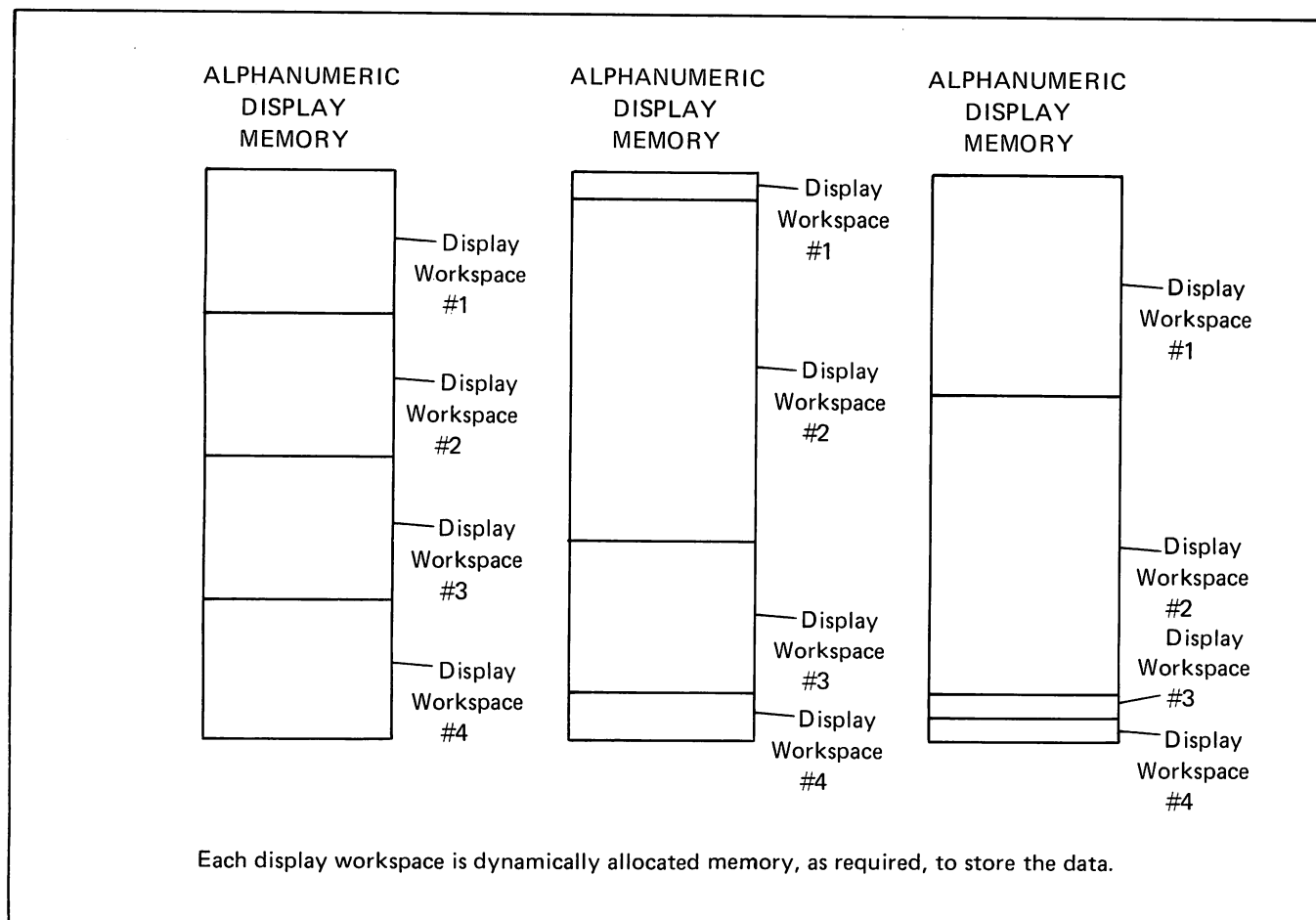


Figure 2-2. Three Examples of Display Memory Allocation to the Four Display Workspaces

**Message Line.** The message line (window#5) is automatically displayed when an incorrect operation is performed. All messages that the terminal generates are given in the *User's Manual*. You may use the message window in a program to instruct the user to perform an operation (such as, insert a tape, or enter some data from the keyboard) by sending:

```
␣,cDisplay Window#5␣ <message string>␣
```

To remove the message:

```
␣,c CLose Window#5␣
```

**Command Line.** The command line is displayed to allow the user to enter commands into the command channel while under program control.

```
␣,cDisplay Window#6␣
```

To remove the command line:

```
␣,cCLose Window#6␣
```

**Soft Key Label Line.** The soft key label line displays the user-defined function labels for the soft keys (f1 through f8). Predefined user functions may be loaded into the soft keys under program control (see "Defining Soft Keys" explained later in this section). The labels may then be displayed to allow the user to select one of the predefined functions while running under program control:

```
␣,cDisplay Window#7␣
```

To remove the soft key label line:

```
␣,cCLose Window#7␣
```

**MEMORY ADDRESSING SCHEME.** Display workspace positions can be addressed using absolute or relative coordinate values. Each display workspace is made up of

80 columns (0-79) and a number of rows determined by the size of the workspace. There can be as many as 71 lines of 80 characters (3 screens). The types of addressing available are:

- Absolute
- Screen Relative
- Cursor Relative

**Row Addressing.** Figure 2-3 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 absolute column addressing. Figure 2-4 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 2-3c 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 2-4a and -5 to 74 in figure 2-4b). The cursor cannot be wrapped around from column 0 to column 79 by specifying large negative values for relative column positions.

Note the difference between this type of addressing (0-79) and the column indicator in the soft key label line (1-80).

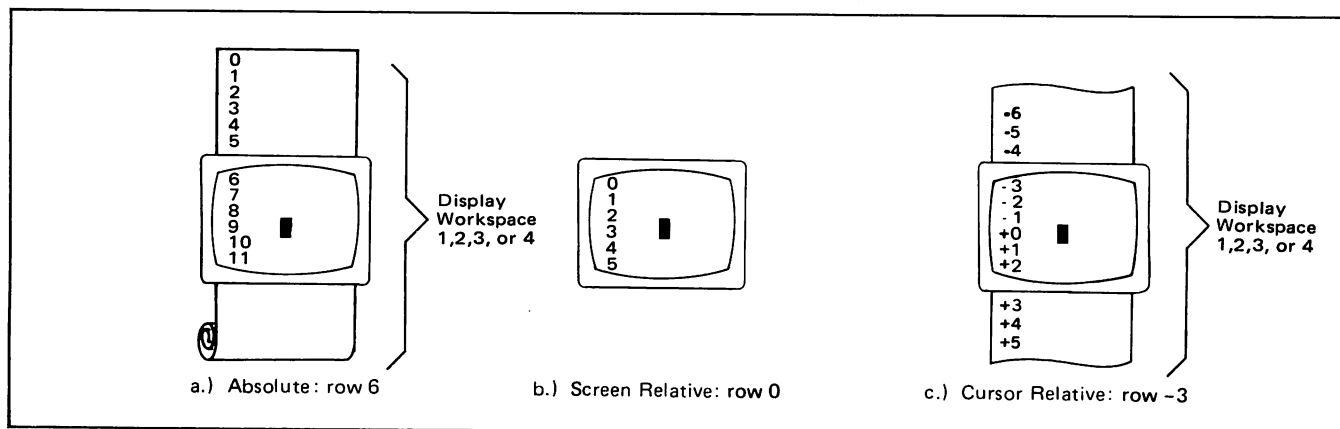


Figure 2-3. Row Addressing

**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 2-4. 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.

```

^ & a +2r -4C
      or
^ & 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.

```

^ & a +8r 60C
      or
^ & a 60c +8R

```

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

```

^ & a 4y -15C
      or
^ & 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:

```
^ * d q
```

Turn Alphanumeric  
Cursor Off:

```
^ * 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 independently in each of the workspaces.

**Setting Tabs.** To set a tab, move the cursor to the desired column and send  $\text{^t1}$ . 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  $\text{^t1}$ . 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  $\text{^t2}$ . All of the tabs in a workspace can be cleared at once without having to position the cursor by sending  $\text{^t3}$ .

**MARGINS.** You can set the left and right margins in each workspace 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 in each of the workspaces. 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  $\text{^t4}$ .

Set Left Margin:  $\text{^t4}$

**Right Margin:** Move the cursor to the desired right margin setting. Send  $\text{^t5}$ .

Set Right Margin:  $\text{^t5}$

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.

```

^ & a 20C      ^ t4      ^ & a 59C      ^ t5
 \ position /   \ set / \ position /   \ set /
  cursor      margin cursor      margin

```

column numbers				
2	3	4	5	6
0	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:

```
^ * d e
```

Turn Alphanumeric  
Display Off:

```
^ * d f
```

Table 2-1. Cursor/Display Operations

FUNCTION	CODE	DESCRIPTION
<b>Cursor</b>		
	␣*dq	Turn cursor on.
	␣*dr	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.
	␣ 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.
Cursor Up	␣ A	Move the cursor up one row. If the cursor is in row 0, it wraps around to row 23.
Cursor Down	␣ B	Move the cursor down one row. If the cursor is in row 23, it wraps around to row 0.
Cursor Right	␣ 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.
Cursor Left	␣ 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.
Home Up	␣ h	Move the cursor to the beginning of the first line of the current display workspace (excluding transmit-only fields in Format Mode.)
Home Up	␣ H	Move cursor to the beginning of the first line in the current display workspace (including transmit-only fields in Format Mode).
Home Down	␣ F	Move the cursor to the beginning of the line following the last data in the current display workspace.
<b>Tabs</b>		
Tab	HT (I <sup>c</sup> ) or ␣ I	Move the cursor forward to the next tab position.
Back Tab	␣ i	Move the cursor back to the previous tab position.
Set Tab	␣ 1	Place a tab at the current cursor column.
Clear Tab	␣ 2	Clear the tab at the current cursor column.
Clear All Tabs	␣ 3	Clear all tabs in the current display workspace.

FUNCTION	CODE	DESCRIPTION
<b>Margins</b>		
Set Left	␣ 4	Set the left margin at the current cursor column.
Set Right	␣ 5	Set the right margin at the current cursor column.
<b>Display</b>		
	␣*de	Turn on current alphanumeric display workspace.
	␣*df	Turn off current alphanumeric display workspace.
Select Display Workspace, Message, Command, or Soft Key Label Line	␣,c DIsplay Win- dow# <n> <sup>c</sup> r	Selects display workspace n (where n is 1, 2, 3, or 4), message line (n×5), command line (n×6), or soft key label line (n×7).
Close Window 5, 6, or 7	␣,c CLose Window # 5, 6, or 7	Removes message, command, or soft key label line from display.
Clear Display	␣ J	Clear the current display workspace from the cursor position to the end of the workspace.
Clear to End of Line	␣ K	Clear current line beginning at the column containing the cursor.
Roll Up	␣ 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	␣ 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	␣ U	Display the next 24 rows of the current display workspace (until the last row of the current display workspace 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	␣ V	Displays the previous 24 rows of the current display workspace (until the first row of the current display workspace 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	␣ l	Turn on memory lock (overflow protect). Note that when Memory Lock is on, only screen relative addressing can be used.
	␣ m	Turn off memory lock. Refer to the <i>User Manual</i> 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	$\text{ESC L}$	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	$\text{ESC M}$	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	$\text{ESC Q}$	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	$\text{ESC N}$	Turn on Insert Character with Wrap-around (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.
	$\text{ESC R}$	Turn off Insert Character Mode (and indicator).
Delete Character	$\text{ESC P}$	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 2-3).
Delete Character with Wraparound	$\text{ESC O}$	The character at the cursor is deleted. Characters from the next lines are wrapped to the end of the current line (see figure 2-4).

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

Turn on Insert with Wrap:  $\text{ESC N}$

Turn off Insert with Wrap:  $\text{ESC R}$

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 WRAP-AROUND.** When characters are deleted using  $\text{ESC O}$ , 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:  $\text{ESC O}$

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 2-5 and 2-6.

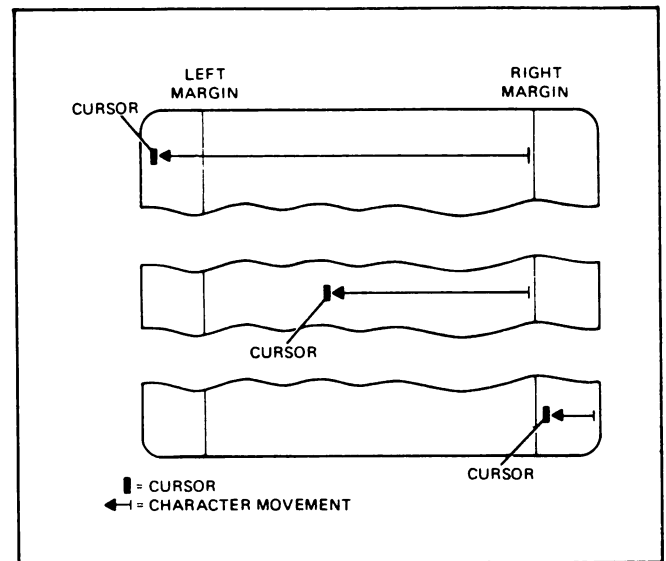


Figure 2-5. Character Delete With Margins

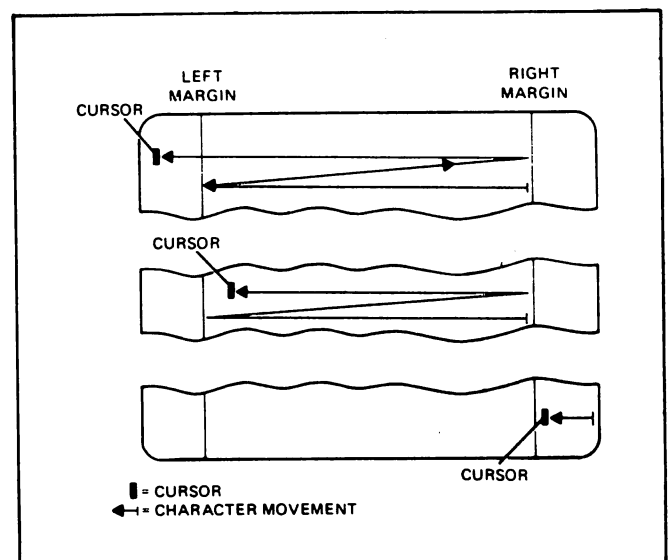


Figure 2-6. Character Delete With Wraparound



**MOVING TEXT BLOCKS.** You can move blocks of text or data in any of the display workspaces 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)

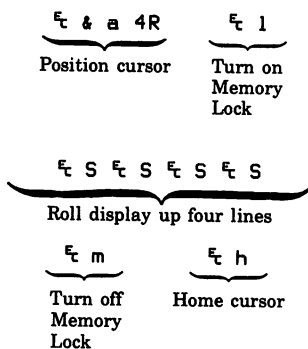
**Step 1.** Position the cursor in the first line of paragraph 2.

**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.



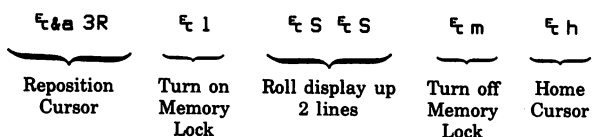
The display should appear as follows:

- (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 can be turned on in any of the four workspaces independently. 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  $\text{F} \ \text{W}$  (the cursor is homed to the beginning of the first unprotected field). Normal operation is returned with  $\text{F} \ \text{X}$  (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.  $\text{F} \ 1$  causes the cursor to be positioned at the beginning of the previous unprotected field. Fields are defined as "unprotected" by using  $\text{F} \ 1$  at the start of the field.  $\text{F} \ 1$  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				Unit Price	Total		
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								
HOMER L. FALGOUT			H C DOUGLAS				DEC 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  $\text{F} \ 1$ .

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

**Step 4.** Send  $\text{F} \ 1$ .

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

**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  $\text{F} \text{ t}$  at the beginning of the field.  $\text{F} \text{ l}$  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 ( $\text{F} \text{ X}$ ).

**Step 2.** Position the cursor at column 11 line 3.

**Step 3.** Send  $\text{F} \text{ t}$ .

**Step 4.** Move the cursor to column 15 in line 3.

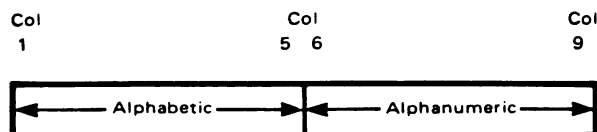
**Step 5.** Send  $\text{F} \text{ l}$ .

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  $\text{RETURN}$ . You can then continue entering data. Data checking fields are defined by beginning the field with one of the following sequences:

- $\text{F} \text{ 6}$  — begin alphabetic field (A thru Z, a thru z, space only).
- $\text{F} \text{ 7}$  — begin numeric field (space, 0 thru 9, -, +, ., and ,)
- $\text{F} \text{ 8}$  — 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 ( $\text{F} \text{ X}$ ).

**Step 2.** Position the cursor at column 1 of line 4.

**Step 3.** Send  $\text{F} \text{ t}$  to define the beginning of an unprotected field.

**Step 4.** Send  $\text{F} \text{ 6}$  to define an alphabetic field.

**Step 5.** Move the cursor to column 6 of line 4.

**Step 6.** Send  $\text{F} \text{ 8}$  to define normal alphanumeric data.

**Step 7.** Move the cursor to column 10 of line 4.

**Step 8.** Send  $\text{F} \text{ l}$  to end the unprotected field.

**Step 9.** Turn on forms mode ( $\text{F} \text{ W}$ ).

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  $\text{RETURN}$  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  $\text{F} \text{ J}$  (CLEAR DISPLAY) or  $\text{F} \text{ K}$  (CLEAR DISPLAY) 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 ( $\text{F} \text{ Q}$ ). 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 ( $\text{F} \text{ P}$ ). 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  $\text{E}K$  (clear to end of line). Also, characters may be deleted from the current cursor position to the end of the last field by sending  $\text{E}J$  (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.

### NOTE

The 13231A Display Enhancement accessory is not normally present in graphics terminals because the two available PCA slots are used for data communications and peripheral interfacing. However, either the data communications PCA or the peripheral interface PCA may be removed to accommodate the display enhancement accessory.

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

$\text{E} \& d \langle \text{enhancement character} \rangle$

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  $\text{E} \& d @$ .

	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
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  $\text{E} \& d C$

**Step 3.** Move the cursor to column 15 in line 5.

**Step 4.** Send  $\text{E} \& d @$  (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.)

1 1  
0 5  
↓ ↓  
TERMINAL

## 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:

$\text{E} \& \langle \text{set} \rangle$

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

**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> (5).

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

An alternate set is selected with the following sequence:

To display  $A\alpha B\beta$  send the following sequence:

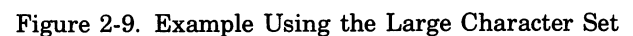
The screen should display  $A\alpha B\beta$ :

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 `␣ <set>` sequence must be sent.

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 2-10. An example of the use of the Math Set is shown in figure 2-8.

Figure 2-8. 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 2-11. An example of how to use the Large Character Set to build the character “B” is shown in figure 2-9. Table B-3 in Appendix B shows the keys required to build each character. Appendix A gives a program to build each character.



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 2-12. Figure 2-13 shows how the line drawing set can be used to build a data entry form.

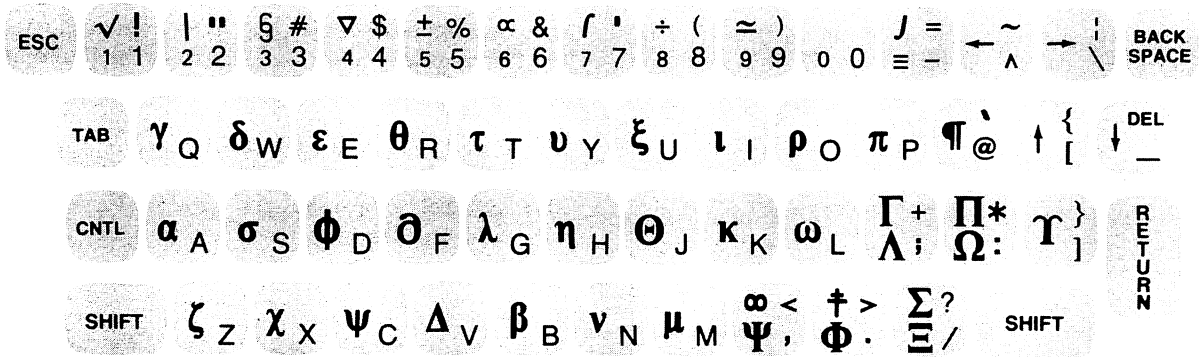


Figure 2-10. Math Set Elements

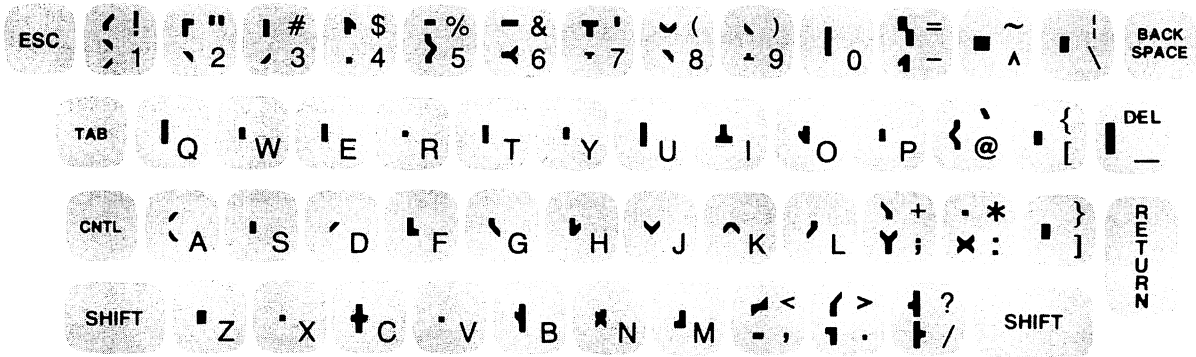


Figure 2-11. Large Character Set Elements

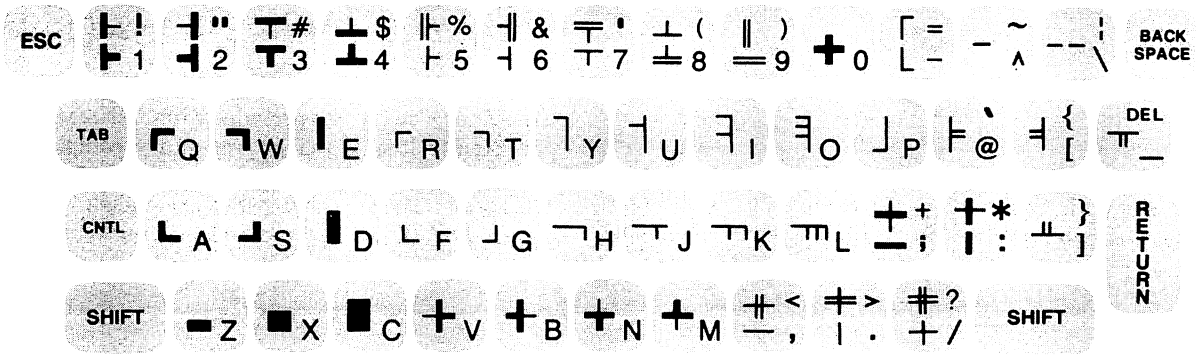



Figure 2-12. Line Drawing Set

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

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.

E 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  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:

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

where:                    0 (closed)  
                          <state> is            or  
   1 (open)  
  
                          <switch> is        A through Z  
   less I and O

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.

```
^ & 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  — . In addition, the  key can also be assigned a string value. The  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 label of up to 8 characters and 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 opera-


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

tion. The keys can be used in application programs to create "menu" lists of special commands or in the case of  to create a terminator for communications protocols.

Controlling the Soft Key Menu

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

Display soft keys menu: 

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 label  
string
```

where:    "#" is            the key number (0-8)  
  
          "type" is        L (local only)  
                          N (normal keyboard operation)  
                          T (transmit only)  
  
          "label" is      is any series of up to 8 characters  
   in the base character set  
  
          "string" is     any series of up to 80 characters



The soft key assignments are displayed in place of the normal screen display. Data in the display workspaces 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 menu:

`ESC k`

Defining Soft Keys

The key assignment operation displays the current key assignments in format mode. The attribute, label, 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:

<code>RETURN</code>	=	<code>ESC</code> (normal)
<code>f1</code>	=	<code>ESC p</code> (transmit only)
<code>f2</code>	=	<code>ESC q</code> (transmit only)
<code>f3</code>	=	<code>ESC r</code> (transmit only)
<code>f4</code>	=	<code>ESC s</code> (transmit only)
<code>f5</code>	=	<code>ESC t</code> (transmit only)
<code>f6</code>	=	<code>ESC u</code> (transmit only)
<code>f7</code>	=	<code>ESC v</code> (transmit only)
<code>f8</code>	=	<code>ESC w</code> (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:

`ESC & f <attribute> <key> <label length>  
<string length> <label> <string> . .`

where:

`<attribute>` =  $\left. \begin{matrix} 0 \text{ (normal)} \\ 1 \text{ (local only)} \\ 2 \text{ (transmit only)} \end{matrix} \right\} \text{a (0 is default)}$

`<key>` is 0-8 k (1 is default)

`<label length>` is 0-8 d (0 is default)

`<string length>` is 1-80 l (1 is default)

`<label>` is the character sequence for the label

`<string>` is the character sequence to be assigned

The label length parameter "d" specifies the number of characters at the beginning of the string that will be assigned to the key label. If incorrectly specified, characters will be added or deleted from the function string.

**Example:** Assign LOG-ON as a label for `f5` and HELLO USER.ACCOUNT as a function.

`ESC & f 0a 5k 6d 19L LOG-ONHELLO USER.ACCOUNTESC`

The softkey loading for `f5` would be:

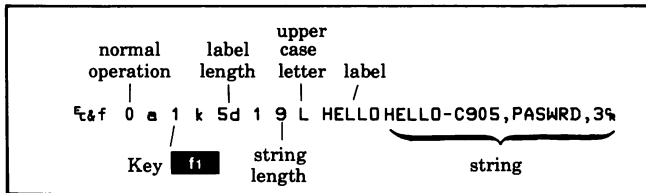
`f5` N LOG-ON  
HELLO USER.ACCOUNT

If a carriage return (`ESC`) is included in the string portion of the soft key definition, the `ESC` will be translated to a `ESC` if the `AUTO LF` key is down. The `ESC` used in assigning values to keys `f1`—`f8` is unaffected by the string assignment made to the `ESC` key. The `ESC` key will only generate the `ESC` 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 `f1` key.



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

`f0` N RETURN  
`ESC`  
`f1` N HELLO  
HELLO-C905,PASWRD,3  
`f2` T `f2`  
`ESC` q  
`f3` T `f3`  
`ESC` r  
`f4` T `f4`  
`ESC` s  
`f5` T `f5`  
`ESC` t  
`f6` T `f6`  
`ESC` u  
`f7` T `f7`  
`ESC` v  
`f8` T `f8`  
`ESC` w

CAUTION

Do not include line feeds (`LF`) or block terminators (`ESC`/`ESC`) 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.

## Displaying the Soft Key Labels

The assigned soft key labels may be displayed to allow the user to select one of the predefined functions while running under program control:

Display Soft Key  
Label Line:

```
␣,cDisplay Window#7␣
```

Remove Soft Key  
Label Line:

```
␣,cClose Window#7␣
```

## Triggering Soft Keys

Soft keys can be triggered from a program. The escape sequence used can be alone or a part of a longer sequence.

Soft key

trigger sequence:

```
␣ & f <0-8> E
```

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

<b>return</b>	= 0		
<b>f1</b>	= 1	<b>f5</b>	= 5
<b>f2</b>	= 2	<b>f6</b>	= 6
<b>f3</b>	= 3	<b>f7</b>	= 7
<b>f4</b>	= 4	<b>f8</b>	= 8

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

```
␣ & f 1 E
```

## Soft Key Applications

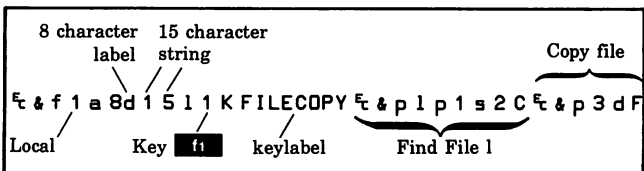
There are many applications of the soft keys. One application is the creation of "menu" operations. For example, the keys can be loaded with the escape sequences to find and read various files on a tape cartridge.

**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:

```
␣ & pnp1s2C ␣ & p3dF
```

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

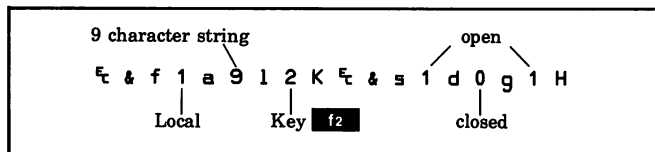
To program the **f1** key to find and display file number 1 and label the key "FILECOPY":



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.

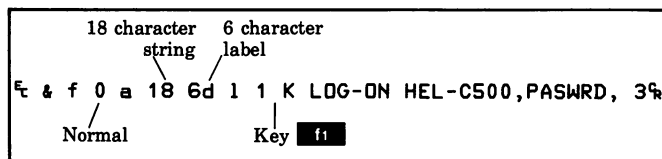
**Example:** To program the **f2** key to change the settings of Switches D, G, and H to open, closed, and open, the following sequence could be used (no label):



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.

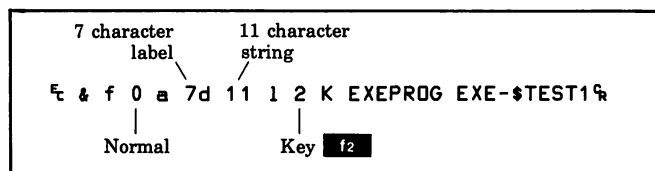
**Example:** A sample key assignment to log-on to the Hewlett-Packard 2000 Series Timeshare System might be as follows:

```
f1 = HEL-C500,PASWRD,3 ␣
```



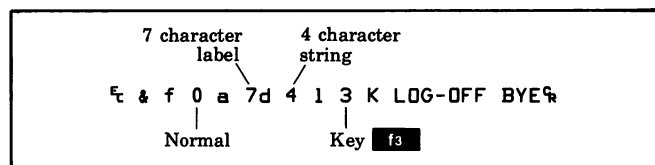
A second key could be used to call and execute a library program:

```
f2 = EXE-$TEST1
```



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

```
f3 = BYE ␣
```



## 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 —  $\text{E} \text{ d}$
- Wait —  $\text{E} \text{ @}$
- Keyboard Disable —  $\text{E} \text{ c}$
- Keyboard Enable —  $\text{E} \text{ b}$
- Reset Terminal (Soft) —  $\text{E} \text{ g}$
- Reset Terminal (Full) —  $\text{E} \text{ E}$
- Turn On Monitor Mode —  $\text{E} \text{ y}$
- Terminal Self-Test —  $\text{E} \text{ z}$
- Data Comm Self-Test —  $\text{E} \text{ x}$
- Modem Disconnect —  $\text{E} \text{ f}$
- Program Down Loading —  $\text{E} \text{ & b}$  or  $\text{E} \text{ & c}$
- Display message, command, or soft key label line —  $\text{E}, \text{c} <\text{command sequence}>$

### 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  $\text{E} \text{ d}$  sequence causes the terminal to send a block of the current workspace 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  $\text{E} \text{ @}$ . Multiple commands can be used to obtain any desired time period.

### Keyboard Disable/Enable

The terminal keyboard can be locked by sending an  $\text{E} \text{ c}$ . It must then be unlocked by sending an  $\text{E} \text{ b}$  or by pressing the RESET TERMINAL key.

## Reset Terminal

A programmatic “Soft Reset” can be made by sending an  $\text{E} \text{ g}$  to the terminal. A “Full Reset” can be made using  $\text{E} \text{ E}$ .

**Soft Reset ( $\text{E} \text{ g}$ ).** 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 ( $\text{E} \text{ E}$ ).** 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 in workspace 1. 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 one second after issuing  $\text{E} \text{ E}$  before sending additional data.

## Monitor Mode

Monitor Mode can be turned on by sending  $\text{E} y$ . Refer to Section V for a discussion of Monitor Mode.

## Self-Test

The Terminal Self-Test can be executed by sending an  $\text{E} z$ .  
The Data Comm Self-Test can be executed by sending an  $\text{E} x$ . 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  $\text{E} f$ . 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  $\text{E} b$  and  $\text{E} c$  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  $\text{E} c$  functions the same as  $\text{E} b$  except that the LOADER message is not displayed.

## Display Message, Command, or Soft Key Label Line

Any or all of the message, command, or soft key label lines may be displayed by sending  $\text{E}, c \text{ Display Window} \langle n \rangle$

where  $n$     5 — message line  
              6 — command line  
              7 — soft key label line

# GRAPHICS CONTROL FUNCTIONS

SECTION

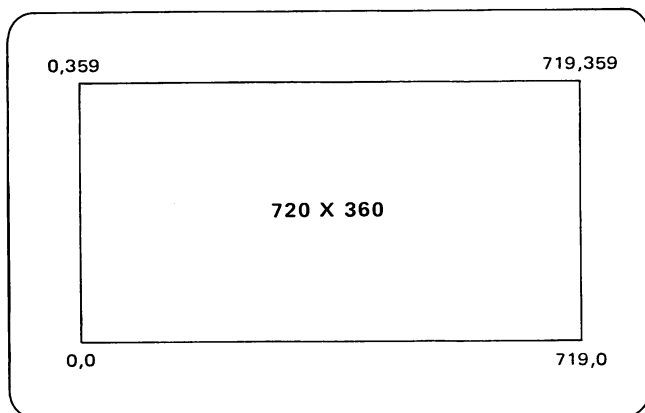
III

## 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 through escape sequences. Controlling the graphics functions through AGL (A Graphics Language) is described in *BASIC For Terminals Reference Manual*, part number 02647-90005. Additional information on how to use "stand alone" graphics features such as MULTILOT 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.

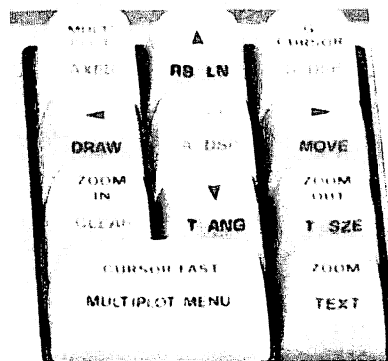


Figure 3-1. Location of Graphics Keys

## 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 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.

## PROGRAMMING GRAPHICS FUNCTIONS BY ESCAPE SEQUENCES

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 multiplot on. If plot from display has been selected, or a point count specified, multiplot will turn itself off. Otherwise, the STOP key must be used to terminate multiplot. (See Multiplot.)		Toggles the alphanumeric display.
	Toggles the graphics cursor on and off.		Erases the graphics image memory.
	Move the graphics cursor. More than one can be pressed for diagonal motion.		Draws a vector from the current pen position to the graphics cursor. Only works if the cursor is on.
	Speeds up the graphics cursor if pressed in conjunction with the cursor keys. The rate returns to normal when released.		Moves the pen to the graphics cursor without drawing a vector. The graphics cursor must be 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.		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 Compatibility Mode the drawing mode is unchanged. (Both Drawing Mode and Compatibility Mode are discussed later in this section.)
	Increments 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.	
	Decrements the zoom magnification.		
	Turns multiplot and graphics text mode off. (See Multiplot.)		
	Key held down selects the alternate key functions.		
	Draws the axes, tic marks, and labels specified in the multiplot menu. (See Multiplot.)		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.
	Toggles the menu for multiplot parameters. (See Multiplot.)		Sets the character orientation (multiples of 90 degrees) and turns slant on or off.
	Toggles the rubber band line connecting the current pen position and the graphics cursor.		
	Toggles the graphics display, to inhibit the graphics image without erasing.		

Table 3-2. Summary of Graphics Sequence Types

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

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
4321	0	0	0	0	0	1	1
0000	NUL	DL	SP	@	P	\	p
0001	SOH	DC1	!	1	A	Q	q
0010	STX	DC2	"	2	B	R	r
0011	ETX	DC3	#	3	C	S	s
0100	EOT	DC4	\$	4	D	T	t
0101	ENQ	NAK	%	5	E	U	u
0110	ACK	SYN	&	6	F	V	v
0111	BEL	ETB	'	7	G	W	w
1000	BS	CAN	(	8	H	X	x
1001	HT	EM	)	9	I	Y	y
1010	LF	SUB	*	:	J	Z	z
1011	VT	ESC	+	;	K	[	{
1100	FF	FS	,	<	L	\	!
1101	CR	GS	-	=	M	]	}
1110	SO	RS	.	>	N	^	~
1111	SI	US	/	?	O	_	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 a-z). 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 the 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.

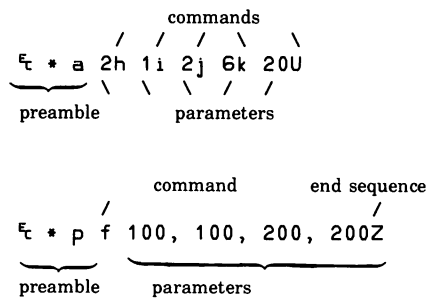
Parameters

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
- Multiplot
- Compatibility Mode
- Raster Dump

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	<code>ESC*dK</code>	Turn on the graphics cursor.
Cursor Off	<code>ESC*dI</code>	Turn off the graphics cursor.
Move Absolute	<code>ESC*d&lt;x,y&gt;o</code>	Position the graphics cursor.
Move Relative	<code>ESC*d&lt;x,y&gt;p</code>	Position the graphics cursor.
<b>Zoom</b>		
Zoom On	<code>ESC*dG</code>	Turn on the zoom function.
Zoom Off	<code>ESC*dH</code>	Turn off the zoom function.
Zoom Size	<code>ESC*d&lt;size&gt;i</code>	Set the zoom size.
Zoom Position	<code>ESC*d&lt;x,y&gt;j</code>	Set the zoom position.
<b>Graphics Memory Control</b>		
Clear Memory	<code>ESC*da</code>	Turn off all dots in graphics memory.
Set Memory	<code>ESC*db</code>	Turn on all dots in graphics memory.
Display On	<code>ESC*dc</code>	Enable the graphics display.
Display Off	<code>ESC*dd</code>	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:

`ESC*dK`

Graphics Cursor Off:

`ESC*dI`

**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.

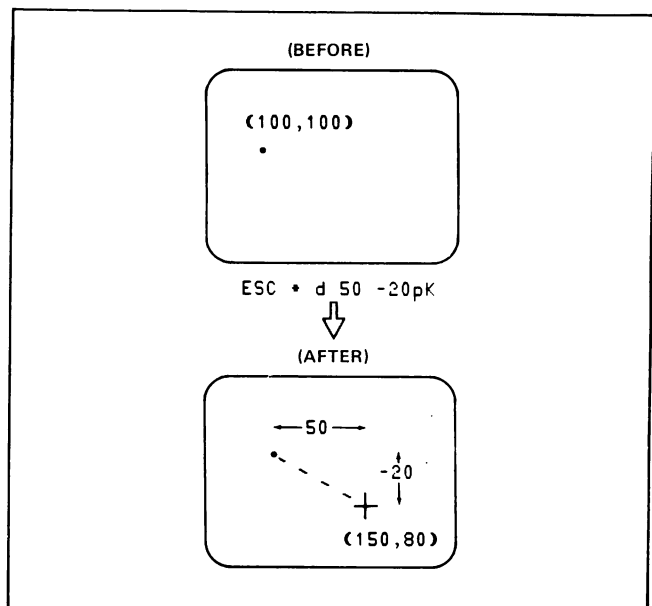
Position Graphics  
Cursor Absolute:

`ESC*d<X,Y>o`

Position Graphics  
Cursor Relative:

`ESC*d<X,Y>p`

**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.)

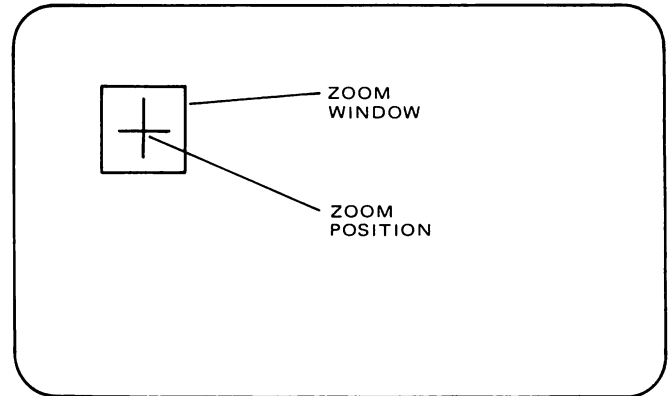
**Example:** Step the zoom position between (50,50), (50,250), (250,250), and (250,50) at 16 ms intervals.

```
␣ * 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 8i 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`

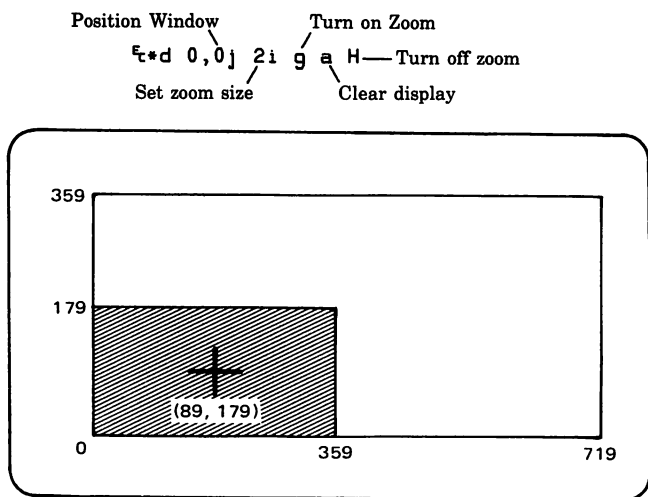
Graphics Display Off: `␣*dd`

**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	t*dc
Graphics Display Off	t*dd
Alphanumeric Display On	t*de
Alphanumeric Display Off	t*df
Zoom On	t*dg
Zoom Off	t*dh
Set Zoom Size	t*di
Set Zoom Position	t*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 t \* 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

t * 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:

t \* 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.

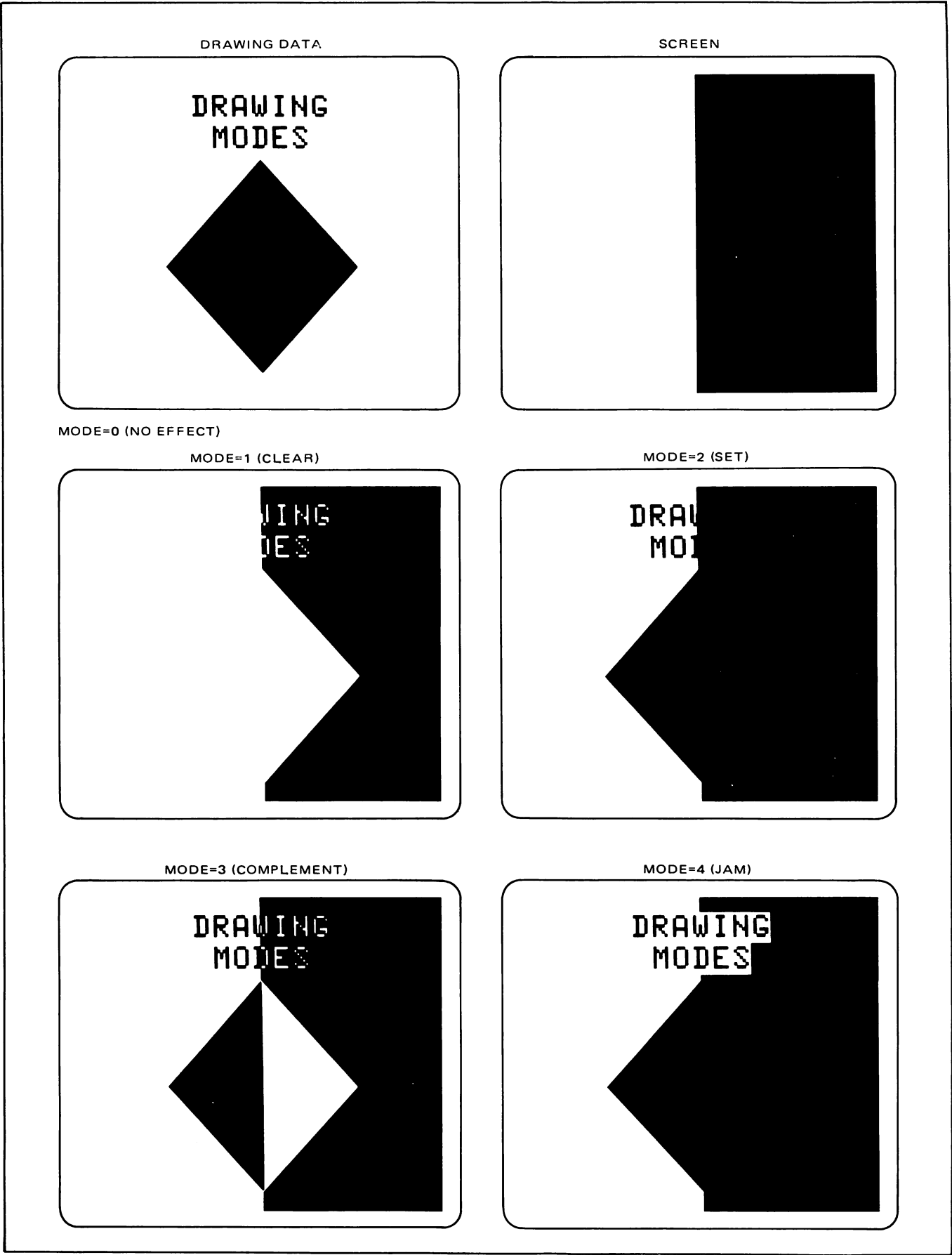


Figure 3-2. Examples Of Drawing Modes

**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

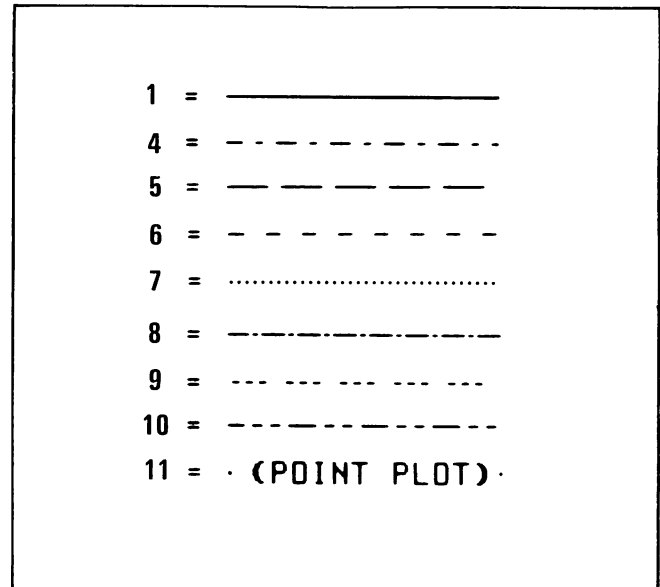


Figure 3-3. Predefined Line Type Patterns

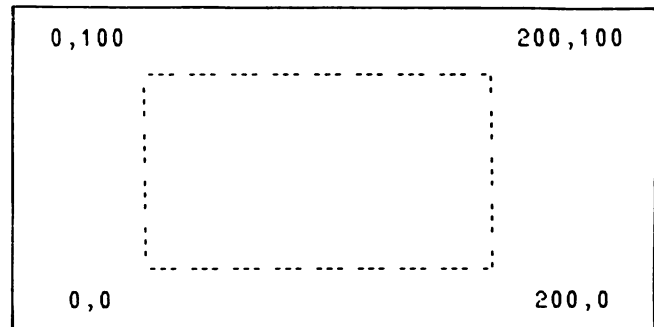
Point plot causes a single point to be plotted at the coordinates specified by the data. This line type is useful for generating "scattergram" type graphs. If user defined area shading is selected (type = 3) the line patterns used are selected from the eight lines making up the area fill pattern (refer 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.

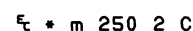
```

% * m 3 B

```


$$E = m \langle \text{pattern} \rangle \langle \text{scale} \rangle c$$

...



**DEFINE AREA PATTERN.** An 8×8 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 8×8 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:

```

E * m <row 0> <row 1> ... <row 7> d
    
```

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

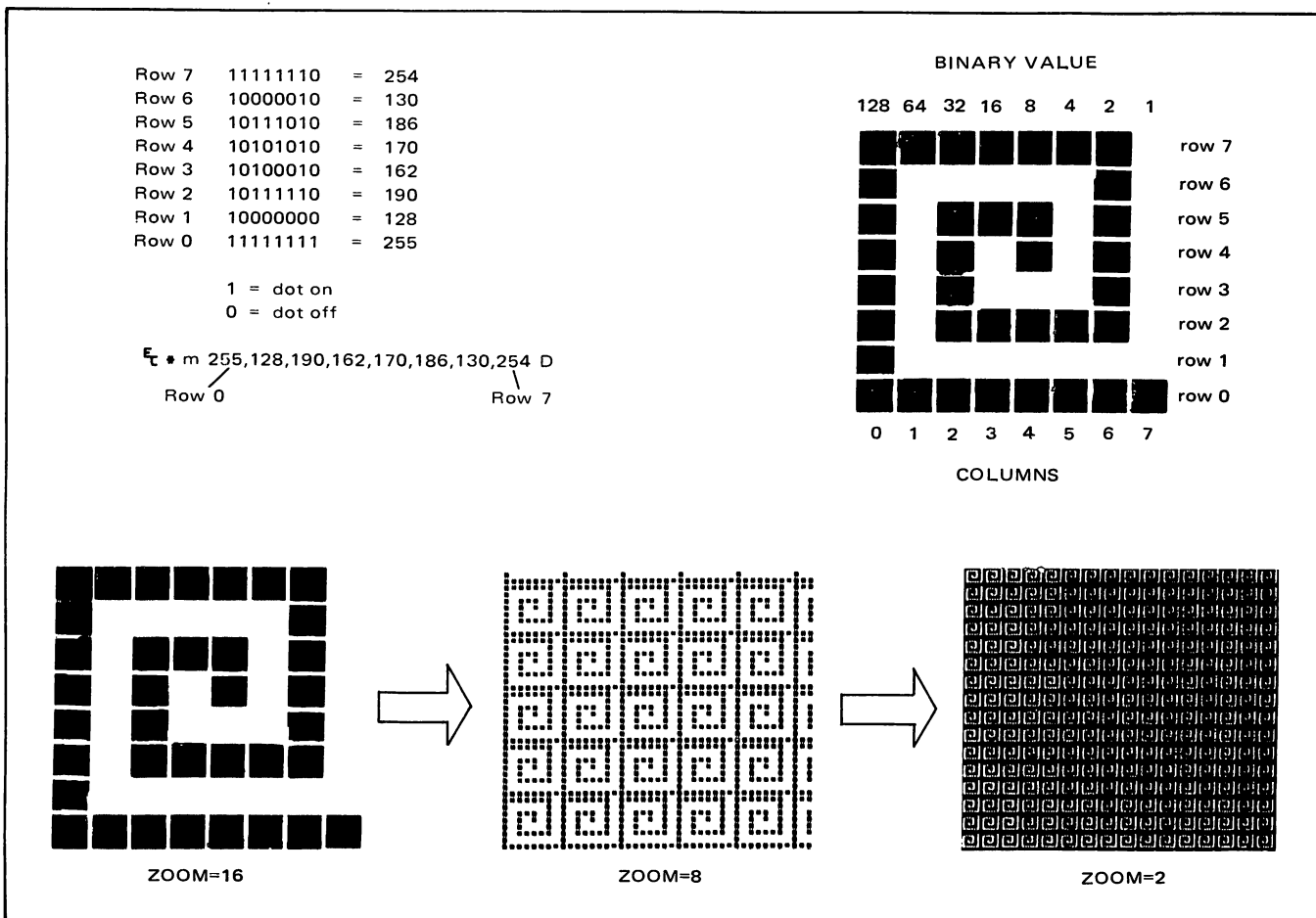
⋮

<row 7> is the 8-bit pattern for row 7

A simple checkerboard pattern would be defined as follows:

```

E * m 170 85 170 85 170 85 170 85 D
      |               |
      Row 0           Row 7
    
```



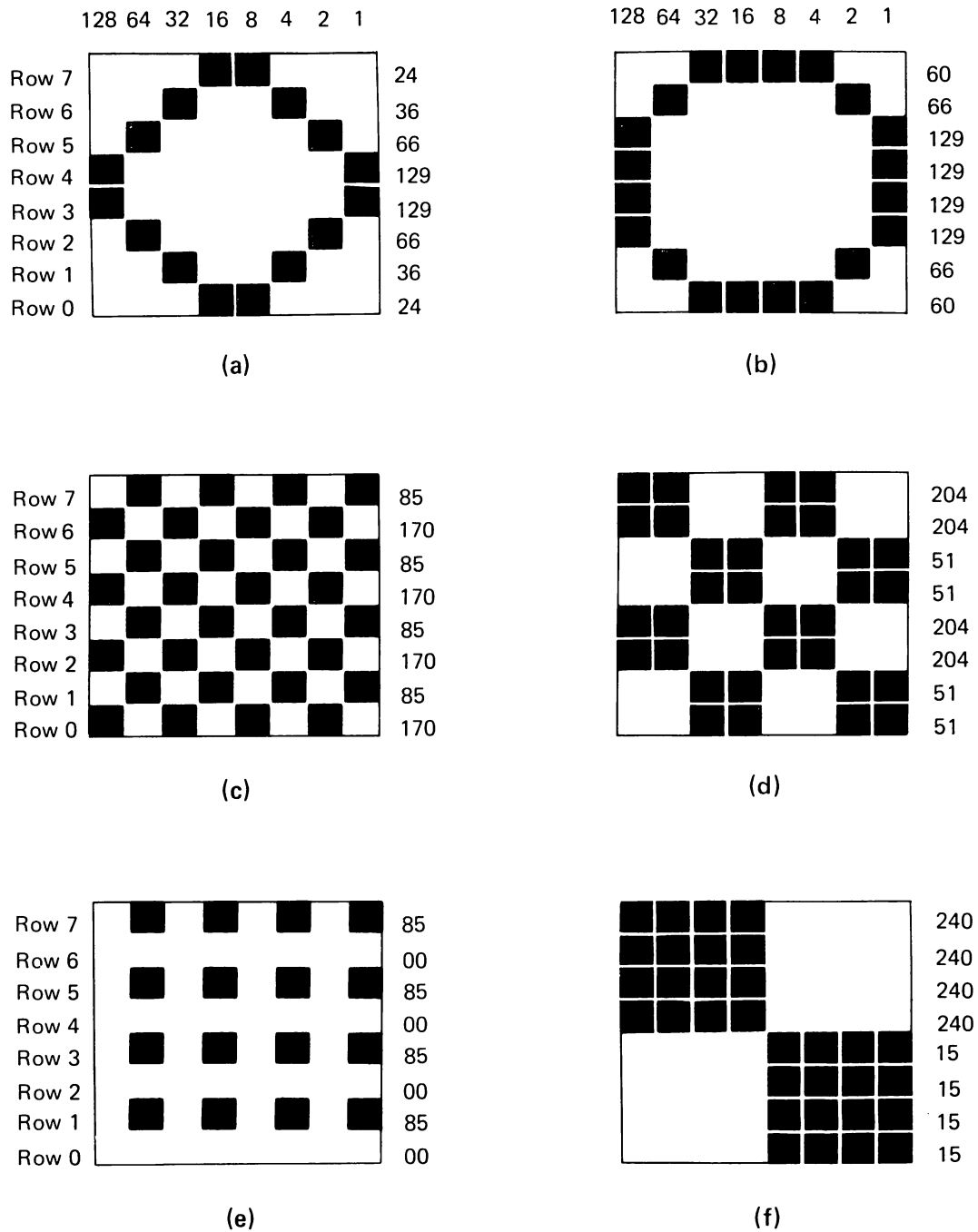


Figure 3-6. Area Pattern Examples



## 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 coordinates.

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

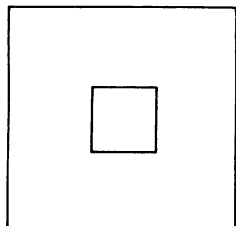
Area Fill,  
Absolute:

```
␣ * m <XLL,YLL> <XUR,YLR> e
```

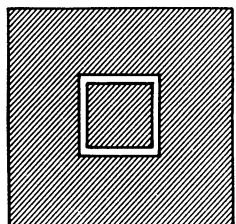
where: <XLL,YLL> and <XUR,YUR> are the absolute coordinates of the lower left and upper right corners of the area to be filled.

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

```
␣*p a f 150 150 200 150 200 200 150 200 150 150 Z
```



```
␣ * 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.

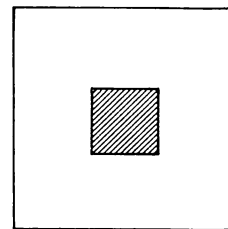
```
␣ * m 1 a 1 b 0,0 100,100 E
```

← 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.

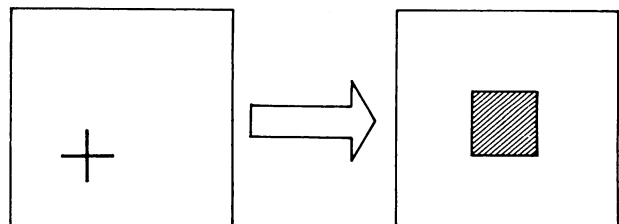
Area Fill,  
Relocatable:

```
␣ * 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
```



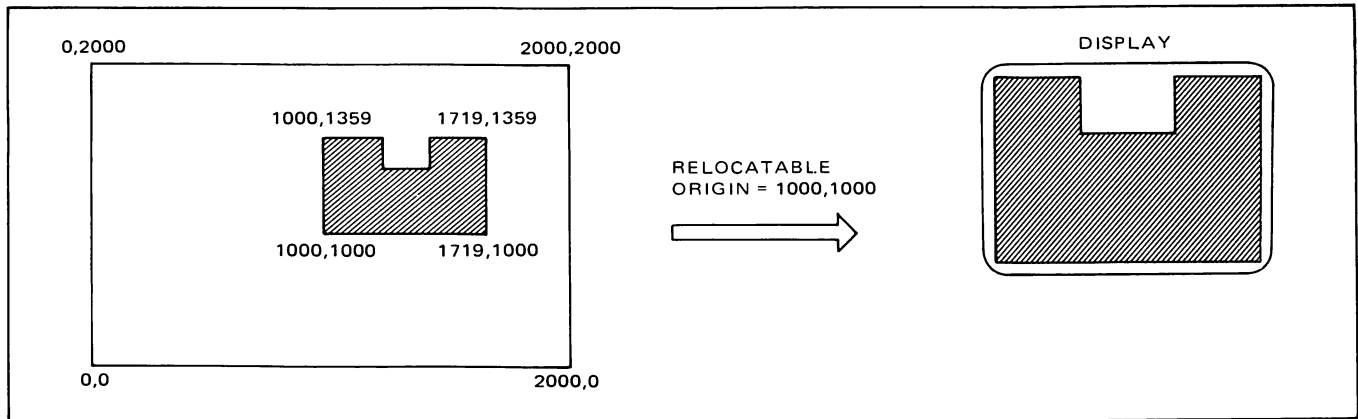


Figure 3-7. Relocatable Origin

### 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.

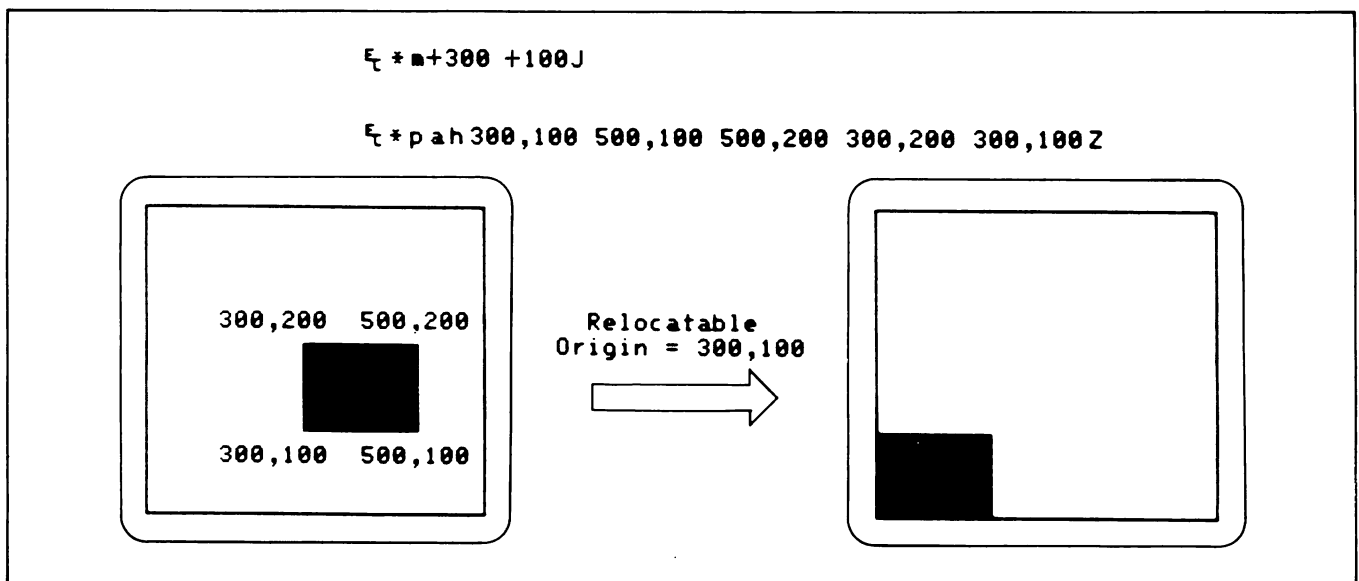
**SET RELOCATABLE ORIGIN ABSOLUTE.** The relocatable origin can be set to any absolute coordinates using ASCII absolute format (–16384 to 16383).

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:

ESC \* m k

**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:

ESC \* m l

## 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:

ESC \* d m

Rubber Band Line Off:

ESC \* d n

## 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:

ESC \* m r

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	0
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
Compatibility Mode	
Page Full Strap	1 (in)
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.

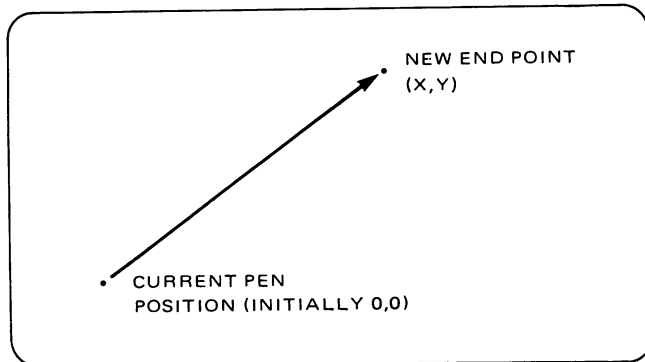


Figure 3-8. Current Pen Position And New End Point

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:

```
ESC * p a
```

Lower Pen:

```
ESC * p b
```

## 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

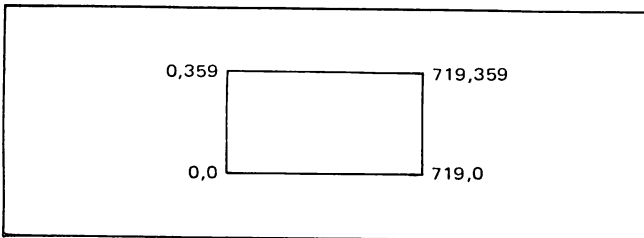
In the ASCII formats, coordinates are specified with ASCII characters 0 through 9. This means that numeric characters generated by a simple print statement can be used to specify X,Y pairs. The first value is used as the X coordinate, and the second as the Y coordinate.

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:

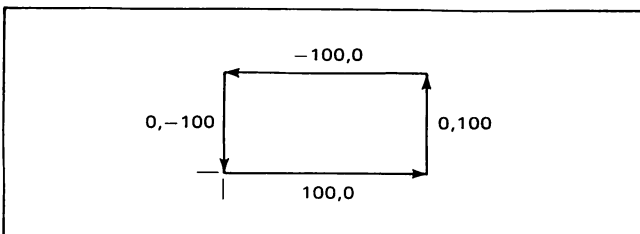
␣ \* p a 0,0 719,0 719,359,0,359,0,0Z



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:

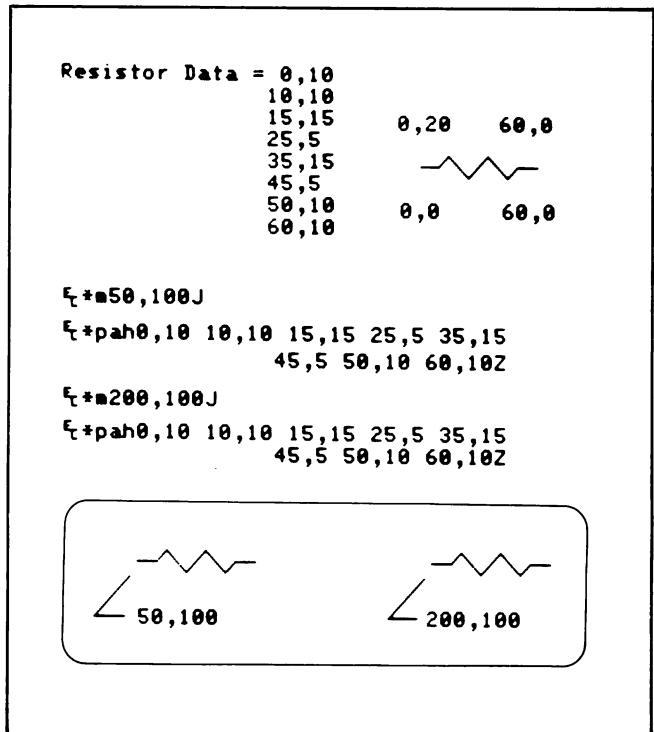
␣ \* p g 100 0 0 100 -100 0 0 -100 Z



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.



## Binary Formats

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 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  $X = 360 - 12 = 350$ ,  $y = 180 + 6 = 186$ .

```

ESC * p i a + ( % 4 j 4 & <byte1><byte2>...Z
      Short Incremental          Short Incremental vectors

```

**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	00	01	02	03	04	05	06	07	08	09
10	10	11	12	13	14	15	16	17	18	19
20	20	21	22	23	24	25	26	27	28	29
30	30	31	32	33	34	35	36	37	38	39
40	40	41	42	43	44	45	46	47	48	49
50	50	51	52	53	54	55	56	57	58	59
60	60	61	62	63	64	65	66	67	68	69
70	70	71	72	73	74	75	76	77	78	79
80	80	81	82	83	84	85	86	87	88	89
90	90	91	92	93	94	95	96	97	98	99
100	100	101	102	103	104	105	106	107	108	109
110	110	111	112	113	114	115	116	117	118	119
120	120	121	122	123	124	125	126	127	128	129
130	130	131	132	133	134	135	136	137	138	139
140	140	141	142	143	144	145	146	147	148	149
150	150	151	152	153	154	155	156	157	158	159
160	160	161	162	163	164	165	166	167	168	169
170	170	171	172	173	174	175	176	177	178	179
180	180	181	182	183	184	185	186	187	188	189
190	190	191	192	193	194	195	196	197	198	199
200	200	201	202	203	204	205	206	207	208	209
210	210	211	212	213	214	215	216	217	218	219
220	220	221	222	223	224	225	226	227	228	229
230	230	231	232	233	234	235	236	237	238	239
240	240	241	242	243	244	245	246	247	248	249
250	250	251	252	253	254	255	256	257	258	259
260	260	261	262	263	264	265	266	267	268	269
270	270	271	272	273	274	275	276	277	278	279
280	280	281	282	283	284	285	286	287	288	289
290	290	291	292	293	294	295	296	297	298	299
300	300	301	302	303	304	305	306	307	308	309
310	310	311	312	313	314	315	316	317	318	319
320	320	321	322	323	324	325	326	327	328	329
330	330	331	332	333	334	335	336	337	338	339
340	340	341	342	343	344	345	346	347	348	349

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

	0	1	2	3	4	5	6	7	8	9
350	350	351	352	353	354	355	356	357	358	359
360	360	361	362	363	364	365	366	367	368	369
370	370	371	372	373	374	375	376	377	378	379
380	380	381	382	383	384	385	386	387	388	389
390	390	391	392	393	394	395	396	397	398	399
400	400	401	402	403	404	405	406	407	408	409
410	410	411	412	413	414	415	416	417	418	419
420	420	421	422	423	424	425	426	427	428	429
430	430	431	432	433	434	435	436	437	438	439
440	440	441	442	443	444	445	446	447	448	449
450	450	451	452	453	454	455	456	457	458	459
460	460	461	462	463	464	465	466	467	468	469
470	470	471	472	473	474	475	476	477	478	479
480	480	481	482	483	484	485	486	487	488	489
490	490	491	492	493	494	495	496	497	498	499
500	500	501	502	503	504	505	506	507	508	509
510	510	511	512	513	514	515	516	517	518	519
520	520	521	522	523	524	525	526	527	528	529
530	530	531	532	533	534	535	536	537	538	539
540	540	541	542	543	544	545	546	547	548	549
550	550	551	552	553	554	555	556	557	558	559
560	560	561	562	563	564	565	566	567	568	569
570	570	571	572	573	574	575	576	577	578	579
580	580	581	582	583	584	585	586	587	588	589
590	590	591	592	593	594	595	596	597	598	599
600	600	601	602	603	604	605	606	607	608	609
610	610	611	612	613	614	615	616	617	618	619
620	620	621	622	623	624	625	626	627	628	629
630	630	631	632	633	634	635	636	637	638	639
640	640	641	642	643	644	645	646	647	648	649
650	650	651	652	653	654	655	656	657	658	659
660	660	661	662	663	664	665	666	667	668	669
670	670	671	672	673	674	675	676	677	678	679
680	680	681	682	683	684	685	686	687	688	689
690	690	691	692	693	694	695	696	697	698	699
700	700	701	702	703	704	705	706	707	708	709
710	710	711	712	713	714	715	716	717	718	719

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
	none	Graphics cursor controls
	none	Graphics cursor fast
	⌘ * dM ⌘ * dN	
	⌘ * aB ⌘ * dT	
	⌘ * dnI	"n" is the new zoom size
	⌘ * dnI	"n" is the new zoom size
	⌘ * dH ⌘ * dG	
	⌘ * dL ⌘ * dK	
	⌘ * dC ⌘ * dD	
	⌘ * dA	
	⌘ * dF ⌘ * dE	
	⌘ * dS ⌘ * dK ⌘ * m4A	Turns on cursor, jam pattern (if not in scaled Compatibility Mode), and turns on Graphics Text Mode.
	⌘ * mnM	"n" is the text size
	⌘ * 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.

RASTER DATA TRANSFERS

Image data can be transferred directly from the graphics memory to cartridge tape, compatible printer (such as the HP 2631G or HP 7245A) or datacomm. In addition, a cartridge tape or a remote CPU can transfer image data directly to the graphics memory. The following paragraphs discuss the data format for such transfers, as well as additional escape sequences that can be used to control the placement of image data on the 2647's display.

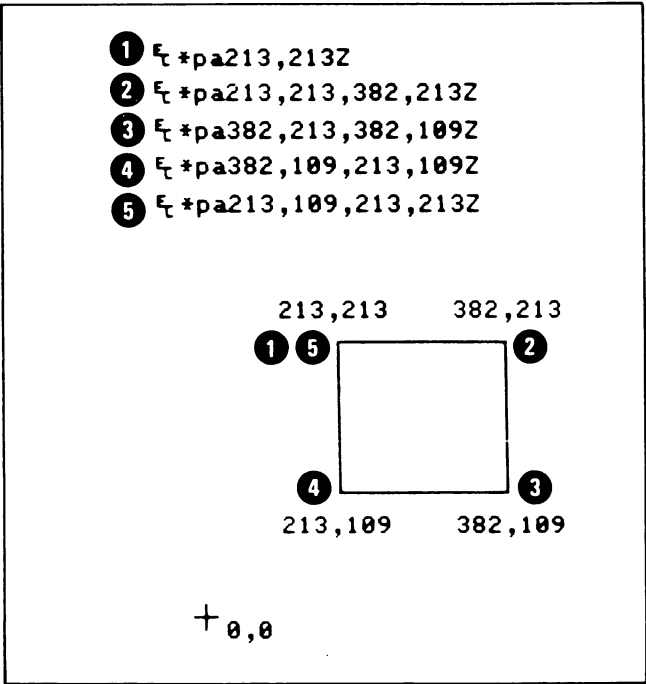


Figure 3-9. Recording Graphics Sequences



## Raster Output Format

The 2647 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 left to right. (See figure 3-10.) The display is scanned top to bottom, so that the first data record represents the top-most 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.

When transferring the raster data into the graphics memory, the current drawing mode is used. Jam mode copies the data exactly, complement mode complements the memory, etc.

The S and T parameters inform the receiving device of the size of the picture. Some devices can use this information to center the picture. If the destination device is the datacom, and the G and H straps on the keyboard are closed (DC1 handshake with standard datacom), a DC1 control character must be sent by the receiving device on a record by record basis to initiate the transfer of each record. That is, each time the receiving CPU sends DC1, one raster record will be sent. If this handshake is not enabled, the entire display is sent without stopping.

## Sending Image Data from CPU to the 2647

Using a format similar to the example above, a remote CPU can send image data to the 2647.  $\text{ESC} * r A$  is first sent to initialize the terminal for a raster transfer. Before the binary data for each data record is sent, an ENQ/ACK handshake must be performed as follows, to allow time for the terminal to go into binary mode.

1. Send  $\text{ESC} * b 90 W \text{ESC}$ .
2. When the terminal sends back  $\text{ACK}$ , send the 90 data bytes.

This handshake is similar to that used when writing binary data to tape. If no data is sent ( $\text{ESC} * b 0 W$  for a blank line) the ENQ/ACK handshake need not be done. Fewer than 90 bytes can be sent. As each line is sent, it is drawn below the previous line, so the data must be sent left to right, top to bottom. Commands described in the next paragraph can be used to position the image anywhere on the screen.

**POSITIONING COMMANDS.** When the 2647 copies the graphics memory to tape or other destination, the entire graphics memory is always transferred. When sending data to the terminal, however, it is possible to transfer less than a full screen. The data sent can be positioned anywhere on the CRT. It is also possible to select and display only a subset of the data sent to the terminal ("windowing"). (See figure 3-13.)

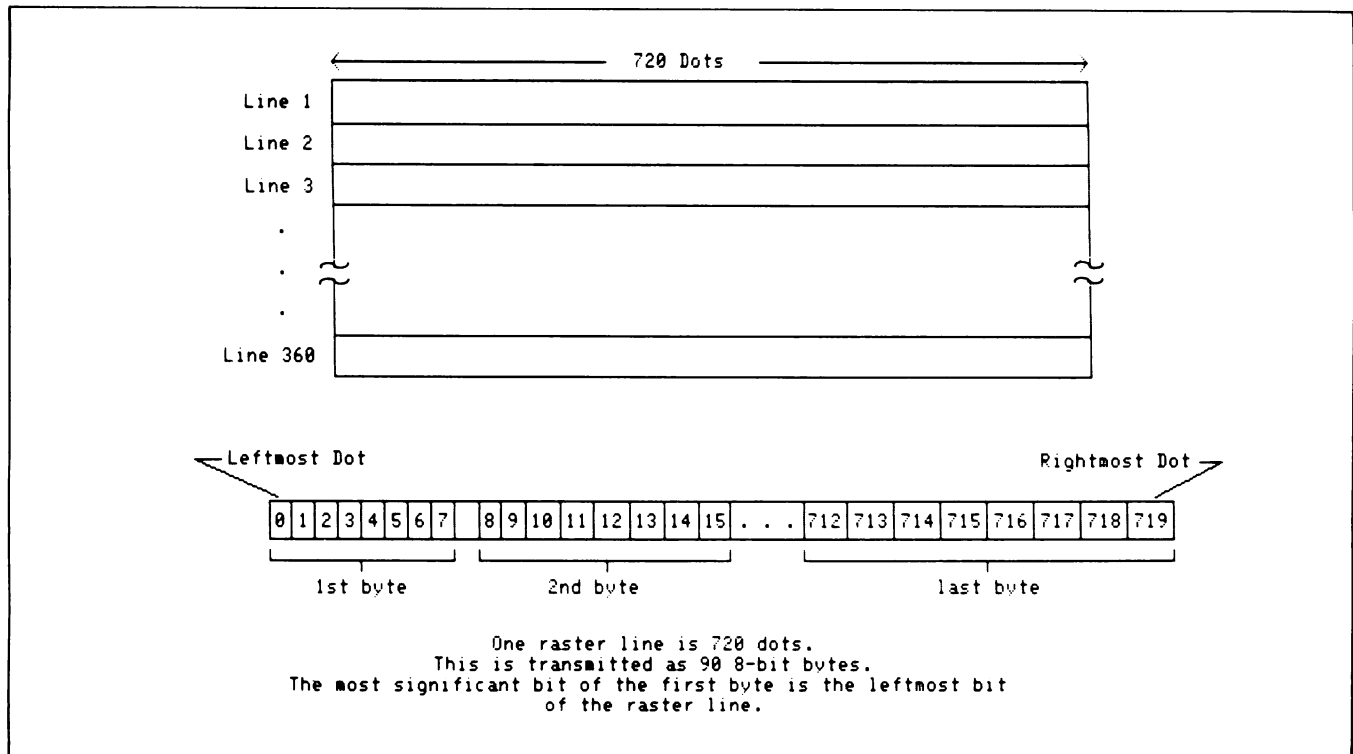


Figure 3-10. Raster Data Format

**X and Y Offsets.** Ordinarily, the starting point for drawing raster data is the upper left corner of the screen. That is, when `ESC * r A` is received, the pen position is initialized to that point. It is possible to offset that point as follows:

```
ESC * r <x offset> x <y offset> y A
```

For example, if it was desired to move the starting point down 100 dots from the top, and in 40 dots from the left side (see figure 3-11), then the following sequence would be used to initialize the transfer:

```
ESC * r 40x 100y A
```

Note that the 'A' command must follow these or any other windowing commands. Also, the X offset is truncated to a multiple of 8 dots. An X offset of 50, for example, would be interpreted as an offset of 48.

If 90 bytes (720 dots) were sent in the above example, each raster line would exceed the right side of the screen by 40 dots. This overflow is ignored.

An example to illustrate the transfer of a small raster image to a specific point on the screen is shown in figure 3-12.

**Windowing Commands.** It is possible to set up the terminal to ignore specific raster lines and data bytes before a transfer is initiated. This allows the user to window out a selected portion of the data.

These escape sequences can be entered locally or can be sent before the raster transfer begins. The commands are as follows:

- |  |   |
|--|---|
| <code>ESC * r</code>                             |   |
| <code>&lt;horizontal window address&gt; m</code> | Gives the number of dots to ignore at the start of each line.         |
| <code>&lt;vertical window address&gt; n</code>   | Gives the number of raster lines to skip at the start of the picture. |
| <code>&lt;horiz. window dimension&gt; p</code>   | Gives the number of dots to draw in each line.                        |
| <code>&lt;vert. window dimension&gt; q</code>    | Gives the total number of raster lines to draw.                       |

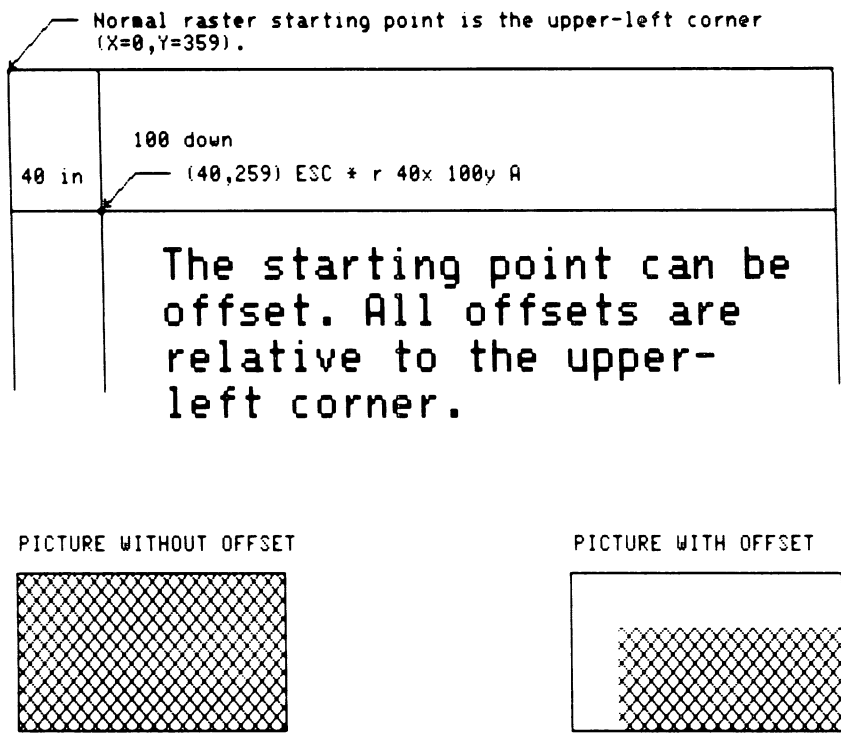
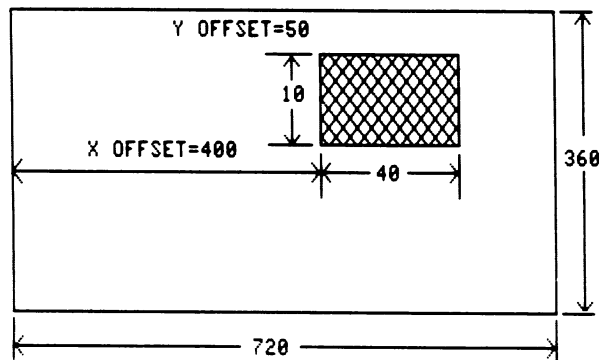


Figure 3-11. Raster Data X and Y Offsets

A smaller than full screen image can be positioned anywhere on the display.



```

ESC * r 400x 50 y A
ESC * b 5W ... (5 data bytes = 40 dots)
ESC * b 5W
10 ESC * b 5W
lines .
.
ESC * b 5W
ESC * r B

```

Figure 3-12. Raster Data Positioning

These commands will be explained by the use of an example (see figure 3-13). The escape sequence  $\text{ESC} * r 40m 10n 400p 1000q$  would be interpreted as follows:

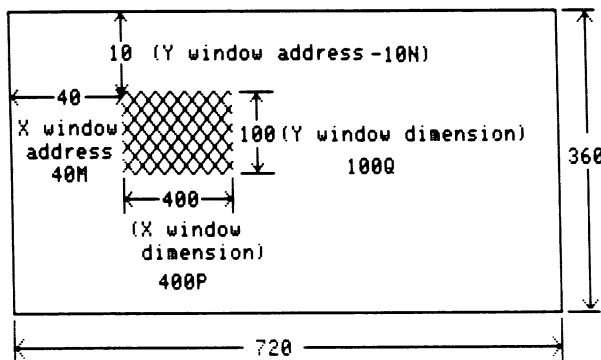
1. The first 10 raster lines would be ignored (10N).
2. Only the next 100 raster lines would be displayed (100Q). Any succeeding data would be ignored until the next raster dump started, as signaled by  $\text{ESC} * r A$ .
3. On the 100 raster lines which would be drawn, the first 40 dots (40M) would be ignored.
4. After the first 40 dots are skipped, only the next 400 dots are drawn (400P). Any data on the raster line after that is ignored.

The result of this would be to display a 400 by 100 dot subset of the original 720 by 360 picture. The P and Q commands select the size of the subset, and the M and N commands determine where the subset is located in the original data.

As with the X offset parameter, the X window address (M) and the X window size (P) are truncated to the nearest multiple of 8 dots.

Offset and window parameters stay in effect for succeeding pictures until they are explicitly changed or the terminal is (hard) reset.

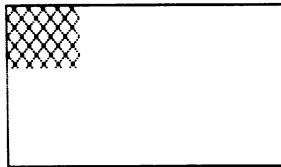
A "window" of raster data can be displayed.



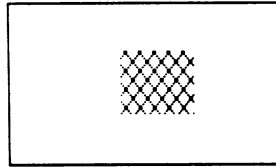
By selecting a window, only a specified area of the raster data will be displayed, and all other raster data will be ignored.

```
Esc * r 40m 10n 400p 100Q
(Selects the window)
```

```
Esc * r 430x 120y A
(Specifies the offset
and starts the transfer
into graphics memory)
```



If no offset is specified, the windowed data is drawn in the upper left corner.



X and Y offsets can be used to position the windowed data.

Figure 3-13. Raster Data Windowing

## Reading the Image Memory

A remote CPU can read the data stored in the 2647A's graphics memory. To read the entire display, the command sequence 'TRANSFER ALL FROM GRAPHICS TO DATA COMM' can be used, as previously described. It is also possible to read subset of the display, as follows:

1. An X and a Y offset are specified to select the upper left corner of the area to be read.
2. The sequence  $\text{Esc} * r A$  is sent to initialize the terminal for reading.
3. The sequence  $\text{Esc} * b \langle \text{count} \rangle R$  (DC1) is sent to read the desired number of data bytes in each scan line.

### Example:

The CPU sends  $\text{Esc} * r 80x 100y A$  to select the area to be read.

The CPU sends  $\text{Esc} * b 10 R$  to read 10 bytes (80 dots).

The terminal sends back 10 bytes.

The CPU sends  $\text{Esc} * b 10 R$  to read 10 bytes in the next (lower) scan line, etc.

Summary of Commands

As with all parameterized sequences, as many lower case commands as desired can be concentrated together. An upper case command terminates the sequence. All of the commands in table 3-11 begin with  $\text{E}^*r$ .

Any raster formatting parameters should be placed before the "A" in the sequence; otherwise, the current parameters will be in effect when the "A" (prepare for raster dump) is executed.

THIS

NOT THIS

$\text{E}^*r$  10m 15n 525p 50q A       $\text{E}^*r$  a 10m 15n 525p 50Q

GRAPHICS HARDCOPY OPERATIONS

An HP 2631G Printer or HP 7245A Plotter Printer can print the contents of graphics memory. The printer is interfaced to the terminal using the HP 13296A Shared Peripheral Interface accessory. Procedures for configuring and installing this accessory are contained in Section VII, Installation. The printer must be assigned as the "Destination" device (refer to Section IV), then the raster data is transferred as described previously (refer to "Raster Data Transfers").

A video hardcopy subsystem is also 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 (Mark File Header on External printer) can be entered locally at the terminal or sent from a computer. Copies can also be initiated manually from the hard copy unit itself.

Table 3-11. Summary of Raster Dump Commands

$E_t * r$ Sequence	Description																
A	Prepare for raster dump.																
B	End of raster dump.																
C	Erase screen.																
D	Turn video on.																
E	Return raster status. Upon receipt of DC1 (if G and H closed), the terminal returns one byte of status, interpreted as follows: <div><table><tr><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1/0</td><td>1/0</td><td>1/0</td><td>1/0</td><td>1/0</td></tr></table><div><div>ENQ/ACK 1 = ENQ/ACK handshake enabled</div><div>ERROR 1 = Error has occurred</div><div>RESET 1 = Terminal has been reset</div><div>READY 1 = Terminal is ready for raster data</div><div>REPEAT 1 = Repeat the transfer</div></div></div>	8	7	6	5	4	3	2	1	0	1	0	1/0	1/0	1/0	1/0	1/0
8	7	6	5	4	3	2	1										
0	1	0	1/0	1/0	1/0	1/0	1/0										
F	Not used.																
G	Not used.																
H	Not used.																
I	Set all parameters to default value.																
J	Return raster size status. The terminal returns the string '720,360'.																
K	Return model number. The string '2647A' is returned.																
L	Not used.																
M	Horizontal window address.																
N	Vertical window address.																
O	Not used.																
P	Window horizontal dimension.																
Q	Window vertical dimension.																
R	Not used.																
S	Horizontal size of picture.																
T	Vertical size of picture.																
U	Not used.																
V	Not used.																
W	Not used.																
X	X offset.																
Y	Y offset.																
Z	NOP.																

## 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 (96 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 3-14 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 3-12. Additional information on keyboard text entry is contained in the User's Manual.

Table 3-12. 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:	
 ,  ,  ,  ,  ,  , 	

```

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
!"#$%&'()*+,-./0123456789:
;<=>?@[\\]^_`{|}~

```

Figure 3-14. 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 3-15). A size of "1" is the default.

Set Graphics  
Text Size:

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

**TEXT DIRECTION.** This command uses the ASCII characters 1 through 4 to specify the text orientation (see figure 3-16). 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
```

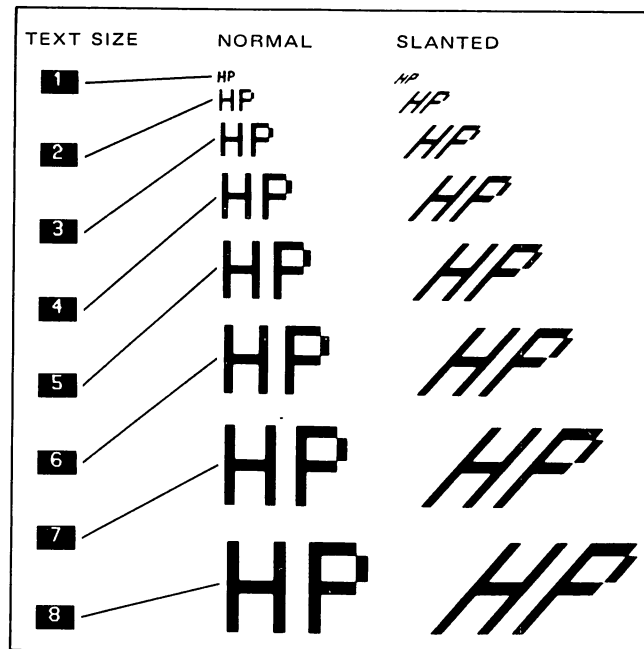


Figure 3-15. Graphics Text Sizes

**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
```

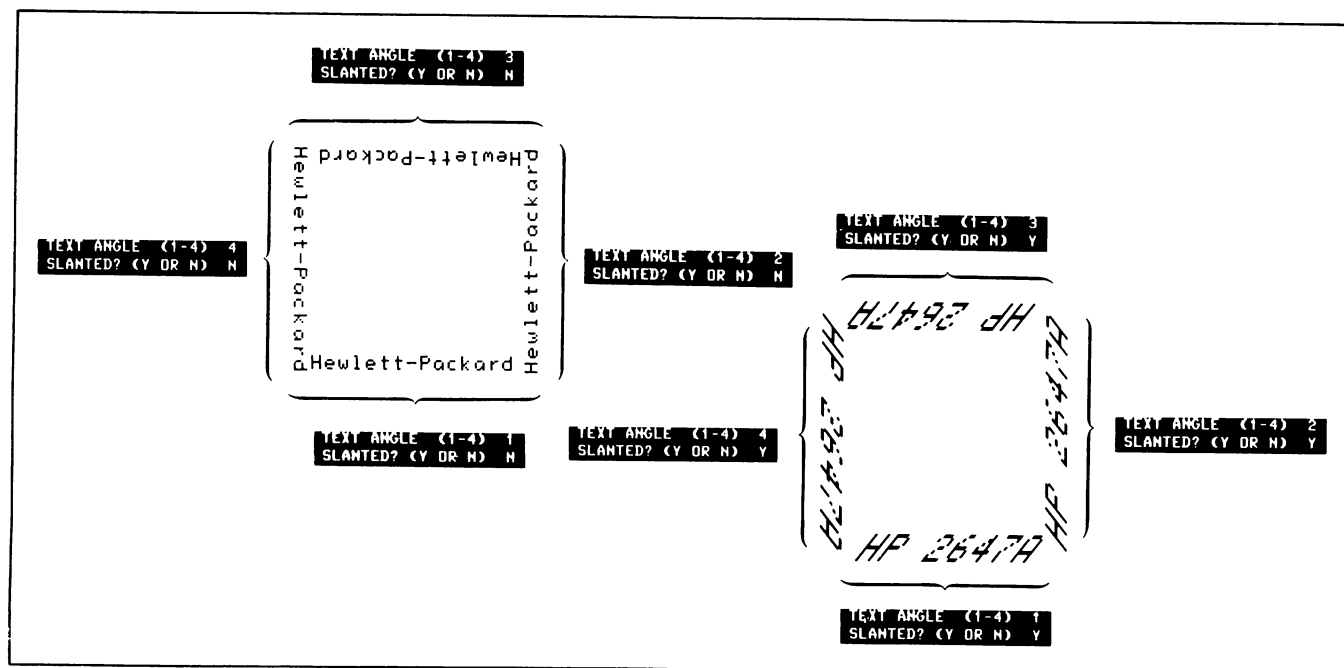


Figure 3-16. Graphics Text Direction

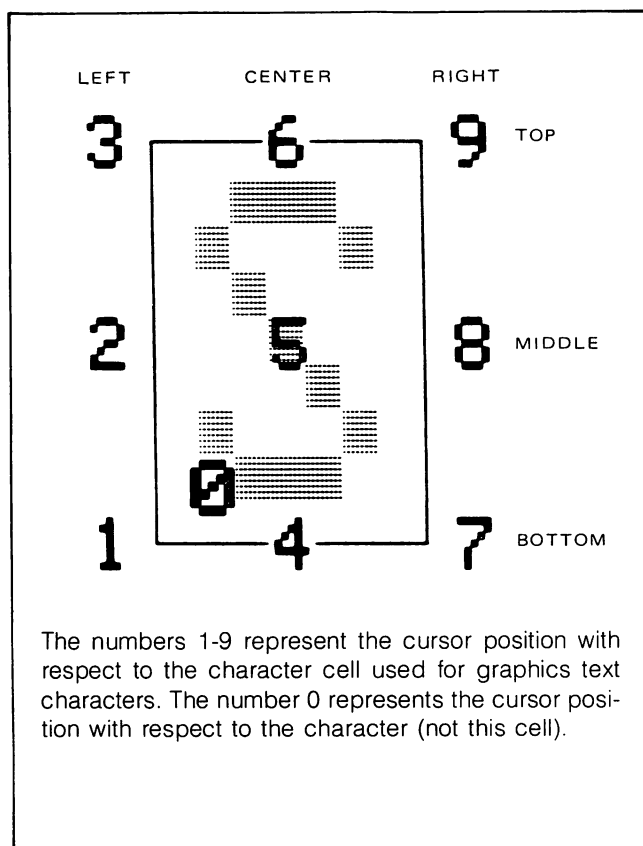
**JUSTIFICATION/ORIGIN.** Text strings can be automatically right or left justified, or centered about a specified point. An ASCII character 0 through 9 indicates the origin (justification and base line) for characters with respect to the current pen position. This function is useful when drawing labels. (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 3-17 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 boundary 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.

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.



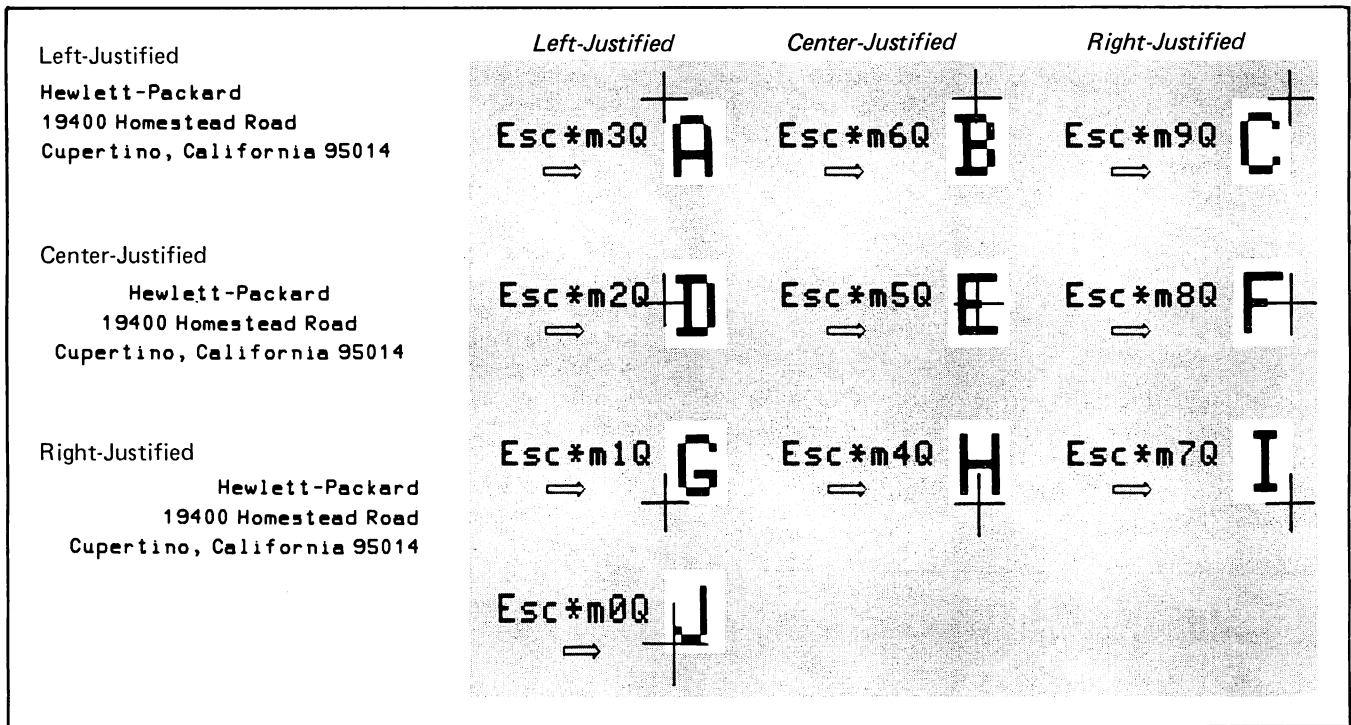


Figure 3-17. Graphics Text Justification

**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*1This is a sample labelEsc(LF)`

## MULTILOT

MULTILOT is a set of three menu-driven graphics plotting programs that allow you to quickly and easily plot numeric data in the form of Pie charts, Bar charts, Linear charts, or Logarithmic charts. The MULTILOT programs, which execute within the terminal itself under control of 2647A BASIC, all reside on a single data cartridge (HP part no. 02647-13301). Once the data cartridge is inserted in the left tape slot of the terminal, the MULTILOT programs can be run either locally by the terminal user or remotely by a program executing in a host computer.

Instructions for running the MULTILOT programs locally are presented in Section 7, Using MULTILOT In Your Terminal, of the *HP 2647A Intelligent Graphics Terminal User's Manual*.

In order for a program executing in a host computer to run any of the MULTILOT programs remotely, the MULTILOT data cartridge must be inserted in the left tape slot of the terminal (your program should display a message on the terminal's screen asking the terminal user to do so). Also, the keyboard *DUPLEX* switch must be set to *HALF*. Your program then loads the terminal's softkeys

from the tape by issuing the following escape sequence:

```
ESC,c COPY FILE FROM LEFT TAPE TO DISPLAY CR
or
ESC,c C F L D I CR
```

Wait 10 seconds, then you are ready to run the MULTILOT programs remotely.

### Plotting Pie Charts

To load the Pie chart program from the data cartridge into the terminal, issue the following escape sequence:

```
ESC,f1E CR
```

When loaded, the Pie chart program displays the Pie chart menu in window #2 of the terminal's screen and waits. Figure 3-18 shows a blank Pie chart menu. At this point your program transmits the menu data in order line-by-line, left to right within each line. If a particular data item does not entirely fill the associated menu field, transmit a Horizontal Tab code (H) to cause the next data item to be directed to the next menu field. Similarly, to leave a particular field blank and skip to the next menu field, transmit a Tab code (T).

**Pie Charts**

**Title** \_\_\_\_\_ **Subtitle** \_\_\_\_\_

**Label** \_\_\_\_\_ **Value** \_\_\_\_\_ **Explode** \_\_\_\_\_ **Shade** \_\_\_\_\_ **Pen** \_\_\_\_\_

**Plotter ?** ☐ **AUTO-SORT?** ☐

Figure 3-18. Blank Pie Chart Menu

Pie Charts				
Title		Subtitle		
XYZ Company - Fiscal 1977 Summary		Earnings by Product Line		
Label	Value	Explode	Shade	Pen
Computers	71			
Test & Measurement	74			
Calculators	52			
Medical	19		1	
Peripherals	28	Y		
Plotter ? <input type="checkbox"/> AUTO-SORT? <input type="checkbox"/>				

Figure 3-19. Completed Pie Chart Menu

The following sequence fills in a Pie chart menu with the data shown in figure 3-19:





```

Esc,c DI W#2 Esc Esc W Esc J XYZ Company - Fiscal 1977
Summary Esc Earnings by Product Line Esc Computers
Esc 71 Esc Esc Esc Esc Test & Measurement Esc 74 Esc Esc Esc Esc
Calculators Esc 52 Esc Esc Esc Esc Medical Esc 19 Esc Esc Esc 1 Esc
Peripherals Esc 28 Esc Y Esc Esc Esc
  
```

Once the Pie chart menu has been filled in, the following sequence plots the Pie chart:

```
Esc Esc f8E
```

To alter the content of a selected field in a completed Pie chart menu, your program first homes the cursor (Esc H) and then moves the cursor to the start of the desired field using any combination of the following:

1. Cursor relative addressing escape sequences (such as Esc &a+7r+23C).
2. Cursor control escape sequences (Esc A for , Esc B for , Esc C for , and Esc D for ).
3. Tab codes (Esc H) and Back Tab escape sequences (Esc i).

For example, to alter the content of the "Value" field associated with the fifth "Label" in a completed Pie chart menu, you would do as follows:

1. Home the cursor using an Esc H sequence.
2. Move the cursor down seven lines (using either seven Esc B sequences or an Esc &a+7r0C sequence).
3. Tab the cursor to the second field of the line using a Esc H code.
4. Transmit the new "Value" field data.






The entire escape sequence for the above example would be as follows:

```
Esc H Esc Esc &a+7r0C Esc ...new data...
```

Note that whenever you fill in a menu field completely the cursor automatically tabs to the start of the next subsequent field (this will always happen when you fill in a single-character field). When the cursor automatically advances in this manner, you must remember not to transmit the Tab code unless you wish to skip over that field.

Table 3-13 summarizes the various escape sequences that apply to Pie charts.

Table 3-13. Pie Chart Escape Sequences

KEY	SEQUENCE	FUNCTION
	␣&f8E	Turn MULTIPLY on.
 	␣,c DI W#2 ␣	Turn menu on.
 	␣,c DI W#1 ␣	Turn menu off.
	␣,c DI W#2 ␣ ␣ W ␣ J	Clear the menu.

Plotting Bar Charts

To load the Bar chart program from the data cartridge into the terminal, issue the following escape sequence:

␣&f2E ␣

When loaded, the Bar chart program displays the Bar chart menu in window #2 of the terminal's screen and waits. Figure 3-20 shows a blank Bar chart menu. At this point your program transmits the menu data in order line-by-line, left to right within each line. If a particular data item does not entirely fill the associated menu field, transmit a Horizontal Tab code (␣) to cause the next data item to be directed to the next menu field. Similarly, to leave a particular field blank and skip to the next menu field, transmit a Tab code (␣).

Bar Charts

Type ☐ (Normal,☐ Stacked,☐ Comparative)

Titles

Main

Sub

X Axis

Y Axis

Labels

Value 1

Value 2

Value 3

Value 4

Value 5

Pen ☐

Shade ☐

Pen ☐

Shade ☐

Pen ☐

Shade ☐

Pen ☐

Shade ☐

Pen ☐

Shade ☐

LEGENDS? ☐

GRID? ☐

Plotter? ☐

Y MIN

Y MAX

Y LABEL SPACING

Figure 3-20. Blank Bar Chart Menu

Bar Charts		Type <input checked="" type="checkbox"/> Normal <input checked="" type="checkbox"/> Stacked <input checked="" type="checkbox"/> Comparative										
Titles												
Main	XYZ Company - 1978 Annual Report										X Axis	Product Line
Sub	Earnings by Product Line										Y Axis	Millions of Dollars
Labels	Value 1		Value 2		Value 3		Value 4		Value 5			
	Pen	Shade	Pen	Shade	Pen	Shade	Pen	Shade	Pen	Shade		
Computers	72		64		51		43					
Test & Meas.	74		69		62		58					
Calculators	52		47		39		19					
Medical	34		31		29		25					
Peripherals	28		22		18		14					
LEGENDS?		<input checked="" type="checkbox"/>	1977		1976		1975		1974			
GRID?		<input checked="" type="checkbox"/>										
Plotter?		<input type="checkbox"/>	Y MIN		Y MAX		Y LABEL SPACING					

Figure 3-21. Completed Bar Chart Menu

The following sequence fills in a Bar chart menu with the data shown in figure 3-21:

```

Esc,c DI W#2  Esc W Esc J N XYZ Company - 1978 Annual
Report  Product Line  Earnings by Product
Line  Millions of Dollars  7  5  4  1
Computers  72  64  51  43  Test & Meas.  74
69  62  58  Calculators  52  47  39  19
Medical  34  31  29  25  Peripherals  28
22  18  14  Esc &a+9R  Esc i  Esc i  Esc i  Esc i  Y 1977  1976
  1975  1974  Esc  Y

```

Once the Bar chart menu has been filled in, the following sequence plots the Bar chart:

```
Esc &f8E
```

To alter the content of a selected field in a completed Bar chart menu, your program first homes the cursor (Esc H) and then moves the cursor to the start of the desired field using any combination of the following:

1. Cursor relative addressing escape sequences (such as Esc &a+7r+23C).

2. Cursor control escape sequences (Esc A for ☒, Esc B for ☒, Esc C for ☒, and Esc D for ☒.

3. Tab codes (Esc) and Back Tab escape sequences (Esc i).

For example, to alter the content of the "Shade" field associated with the "Value 2" column in a completed Pie chart menu, you would do as follows:

1. Home the cursor using an Esc H sequence.
2. Move the cursor down six lines (using either six Esc B sequences or an Esc &a+7R sequence).
3. Back Tab the cursor twice using two Esc i sequences.
4. Transmit the new "Shade" field data.

The entire escape sequence for the above example would be as follows:

```
Esc H Esc &a+7R Esc i Esc i ...new data...
```

Table 3-14 summarizes the various escape sequences that apply to Bar charts.

KEY	SEQUENCE	FUNCTION
	␣&f8E	Turn MULTIPLOT on.
	␣,c DI W#2 ␣	Turn menu on.
	␣,c DI W#1 ␣	Turn menu off.
	␣,c DI W#2 ␣ ␣ H ␣ J	Clear the menu.

Table 3-14. Bar Chart Escape Sequences

Plotting Linear Charts

To load the Linear chart program from the data cartridge into the terminal, issue the following escape sequence:

␣&f3E ␣

When loaded, the Linear chart program displays the Linear chart menu in window #2 of the terminal's screen and waits. Figure 3-22 shows a blank Linear chart menu. At this point your program transmits the menu data in order line-by-line, left to right within each line. If a particular data item does not entirely fill the associated menu field, transmit a Horizontal Tab code (␣) to cause the next data item to be directed to the next menu field. Similarly, to leave a particular field blank and skip to the next menu field, transmit a Tab code (␣).

linear Charts

A. PLOT SPECIFICATION

NO. OF COLUMNS

X IS COLUMN(S)

Y IS COLUMN(S)

PEN & LINE TYPE

SKIP FIRST

STOP AFTER

LINES

POINTS

ID. DEVICE SPECIFICATION

:

SOURCE

:

PLOTTER ?

:

:

:

:

:

B. AXES SPECIFICATION

X AXES

Y AXES

MIN X

MAX X

MIN Y

MAX Y

Linear

Linear

GRID(Y OR N)

UNITS BETWEEN X LABELS

UNITS BETWEEN X TICS

UNITS BETWEEN Y LABELS

UNITS BETWEEN Y TICS

C. ANNOTATION

Titles

Main

Sub

Xaxis

Yaxis

LEGENDS

Figure 3-22. Blank Linear Chart Menu

Linear Charts

A. PLOT SPECIFICATION

NO. OF COLUMNS

4

X IS COLUMN(S)

1

Y IS COLUMN(S)

2

PEN & LINE TYPE

1

SKIP FIRST

STOP AFTER

LINES

POINTS

ID. DEVICE SPECIFICATION

SOURCE

D1

PLOTTER ?

B. AXES SPECIFICATION

X AXES

Linear

Y AXES

Linear

MIN X

0

MAX X

100

MIN Y

0

MAX Y

100

GRID(Y OR N)

UNITS BETWEEN X LABELS

10

UNITS BETWEEN X TICS

5

UNITS BETWEEN Y LABELS

10

UNITS BETWEEN Y TICS

5

C. ANNOTATION

Titles

Main

EXAMPLE OF LINEAR PLOT

Sub

HP 2647A Multiplot

Xaxis

X-Axis Label

Yaxis

Y-Axis Label

LEGENDS

Legend 1

Legend 2

Figure 3-23. Completed Linear Chart Menu

The following sequence fills in a Linear chart menu with the data shown in figure 3-23:





```
␣,c DI W#2 ␣ ␣ W ␣ J 4 ␣ DI 1 ␣ ␣ 2 ␣ 1 ␣ ␣ ␣ Linear
Linear Y ␣ 0 ␣ 10 ␣ 100 ␣ 5 ␣ 0 ␣ 10 ␣ 100 ␣ 5 ␣
EXAMPLE OF LINEAR PLOT ␣ X-Axis Label ␣ HP 2647A
Multiplot ␣ Y-Axis Label ␣ Legend 1 ␣ Legend 2
```

Once the Linear chart menu has been filled in, the following sequences draw the chart axes and plot the chart, respectively:

```
␣&f7E (Note that this sequence may need to be followed by a time-delay depending upon the complexity of the axes to be drawn.)

␣&f8E
```

To alter the content of a selected field in a completed Linear chart menu, your program first homes the cursor (␣ H) and then moves the cursor to the start of the desired field using any combination of the following:

- 1. Cursor relative addressing escape sequences (such as ␣&a+7r+23C).
- 2. Cursor control escape sequences (␣ A for , ␣ B for , ␣ C for , and ␣ D for ).
- 3. Tab codes (␣) and Back Tab escape sequences (␣ i).

For example, to alter the content of the "MAX Y" field in a completed Linear chart menu, you would do as follows:








- 1. Home the cursor using an H sequence.
- 2. Move the cursor down 13 lines (using either 13 ␣ B sequences or an ␣&a+13R sequence).
- 3. Back tab the cursor once using an ␣ i sequence.
- 4. Transmit the new "MAX Y" field data.

The entire escape sequence for the above example would be as follows:

```
␣ H ␣&a+13R ␣ i ...new data...
```

Table 3-15 summarizes the various escape sequences that apply to Linear charts.

Table 3-15. Linear Chart Escape Sequences

KEY	SEQUENCE	FUNCTION
	␣&f8E	Turn MULTIPLY on.
 	␣&f7E	Draw MULTIPLY axes.
 	␣,c DI W#2 ␣	Turn menu on.
 	␣,c DI W#1 ␣	Turn menu off.
	␣,c DI W#2 ␣ ␣ W ␣ J	Clear the menu.

## 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  $\text{Esc} \text{ t}$ . 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
$\text{Esc} \text{ S}$	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;"> <p>1 0 1 1/0 0/1 0/1 1</p> <p>Hard Copy Unit 0 = not ready 1 = ready</p> <p>Linear Interpolation (off)</p> <p>Mode 0 = Graphics Mode 1 = Alpha Mode</p> <p>Auxiliary Device (inactive)</p> <p>Margin 1 = margin 1 0 = margin 2</p> </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>		
$\text{Esc} \text{ B}$ (20 ms delay) $\text{Esc} \text{ S}$	Read graphics cursor position	$\langle \text{HI X} \rangle \langle \text{LO X} \rangle \langle \text{HI Y} \rangle \langle \text{LO Y} \rangle \langle \text{terminator} \rangle$
$\text{Esc} \text{ B}$	Read graphics cursor position when key struck	$\langle \text{KEY} \rangle \langle \text{HI X} \rangle \langle \text{LO X} \rangle \langle \text{HI Y} \rangle \langle \text{LO Y} \rangle \langle \text{terminator} \rangle$
$\text{Esc} \text{ S}$	Make hardcopy	
$\text{Esc} \text{ F}$	End graphics mode, clear screen, and home cursor	
$\text{Esc}$	Go into graphics mode (draw vectors)	
$\text{Esc}$	Go into alpha mode	
$\text{Esc}$	Backspace ( $\text{H}^c$ ). Moves 1 space left (14 units)	
$\text{Esc}$	Horizontal Tab ( $\text{I}^c$ ). Moves 1 space right (14 units)	
$\text{Esc}$	End graphics mode	
$\text{Esc}$	Line Feed ( $\text{J}^c$ ). Moves 1 line down (22 units)	
$\text{Esc}$	Vertical Tab ( $\text{K}^c$ ). Moves 1 line up (22 units)	
<p style="text-align: center;"><b>NOTES</b></p> <p>The terminal will normally respond with an <math>\text{Esc}</math> character when an <math>\text{Esc}</math> character is received. Compatibility Mode disables the terminal's <math>\text{Esc}/\text{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 "⌘ & 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 3-17. 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 3-25.

Table 3-17. 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 1 — In (Normal position)	⌘ * t <byte1> c
NOP	z

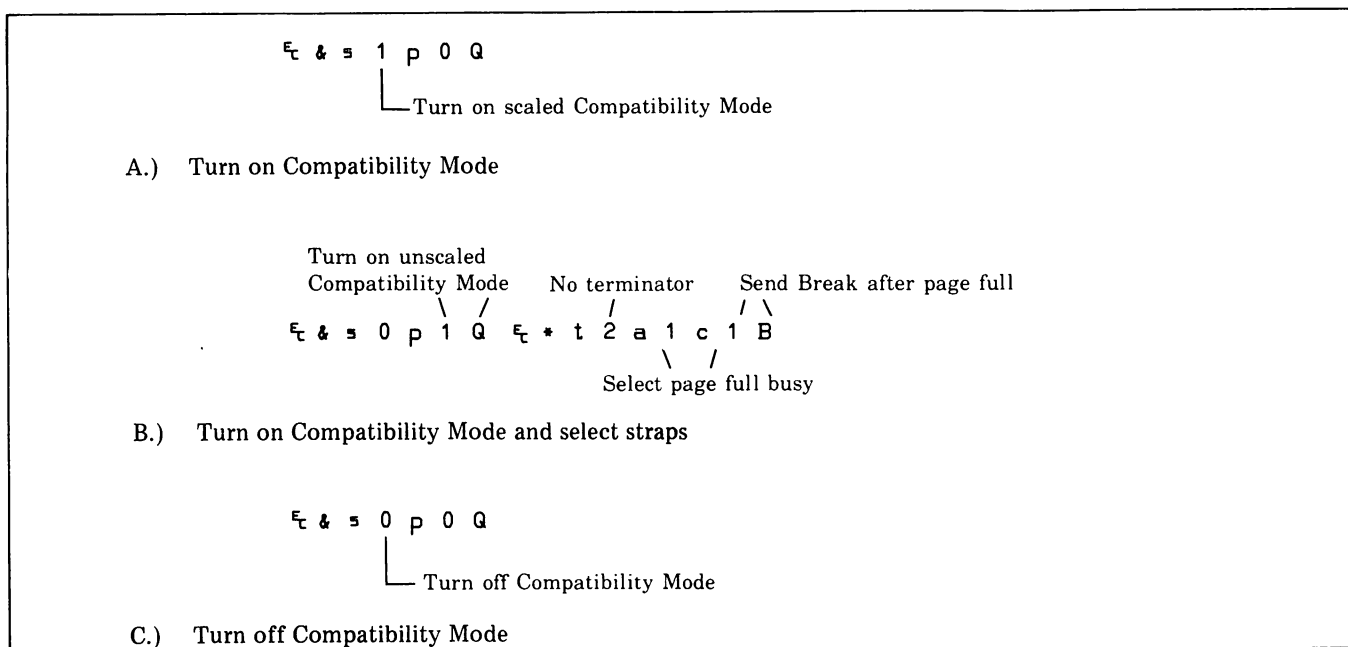


Figure 3-24. Turning on Compatibility Mode

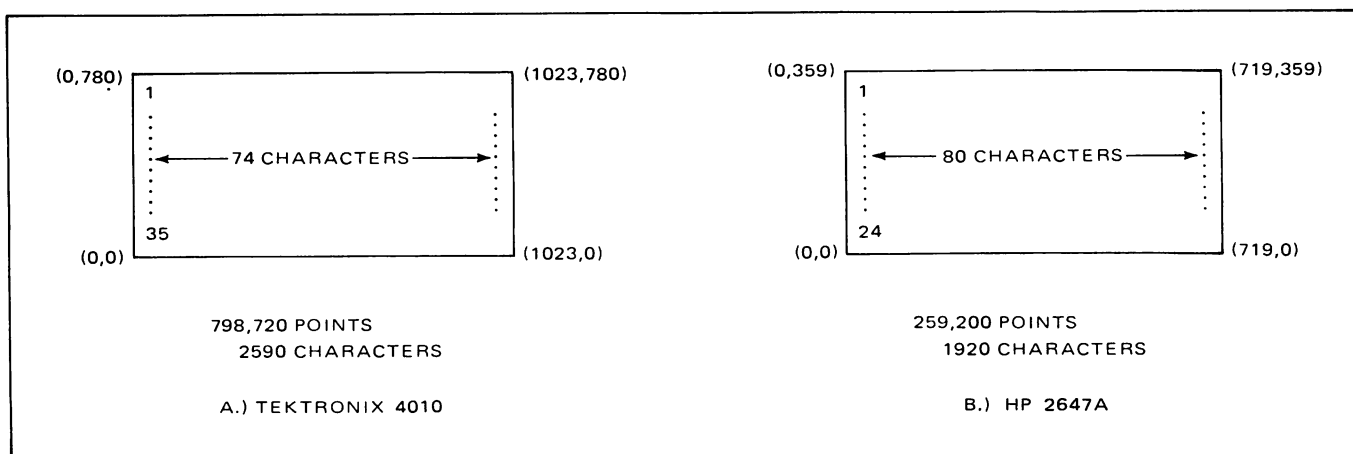


Figure 3-25. Comparison of a Terminal with 1024 × 780 Display and the HP 2647A

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 3-26).

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 3-27).

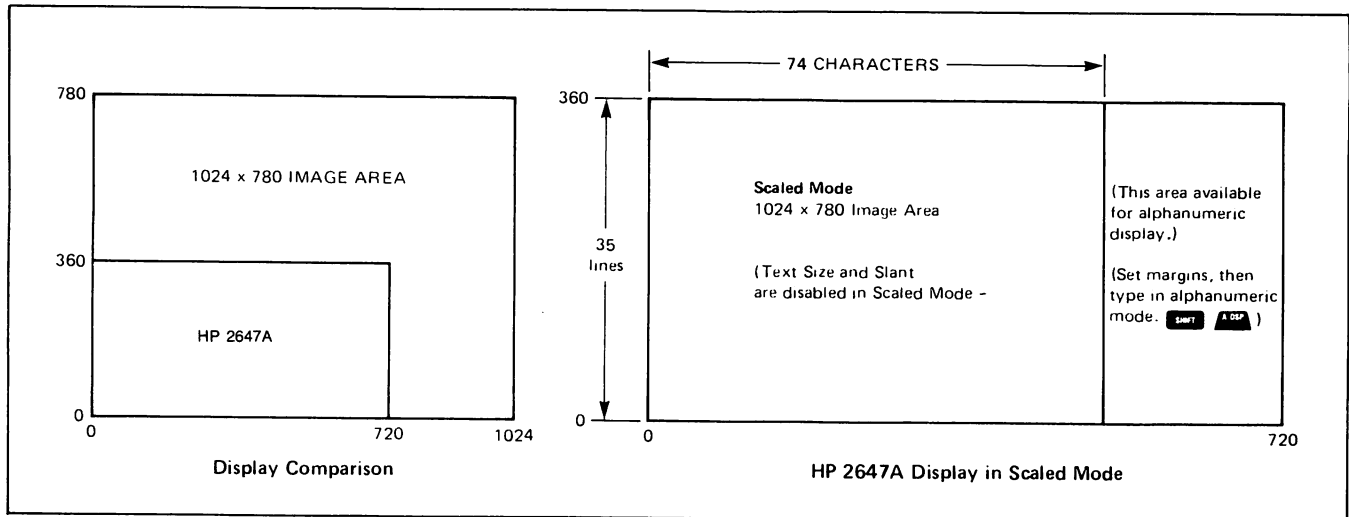


Figure 3-26. Scaled Data

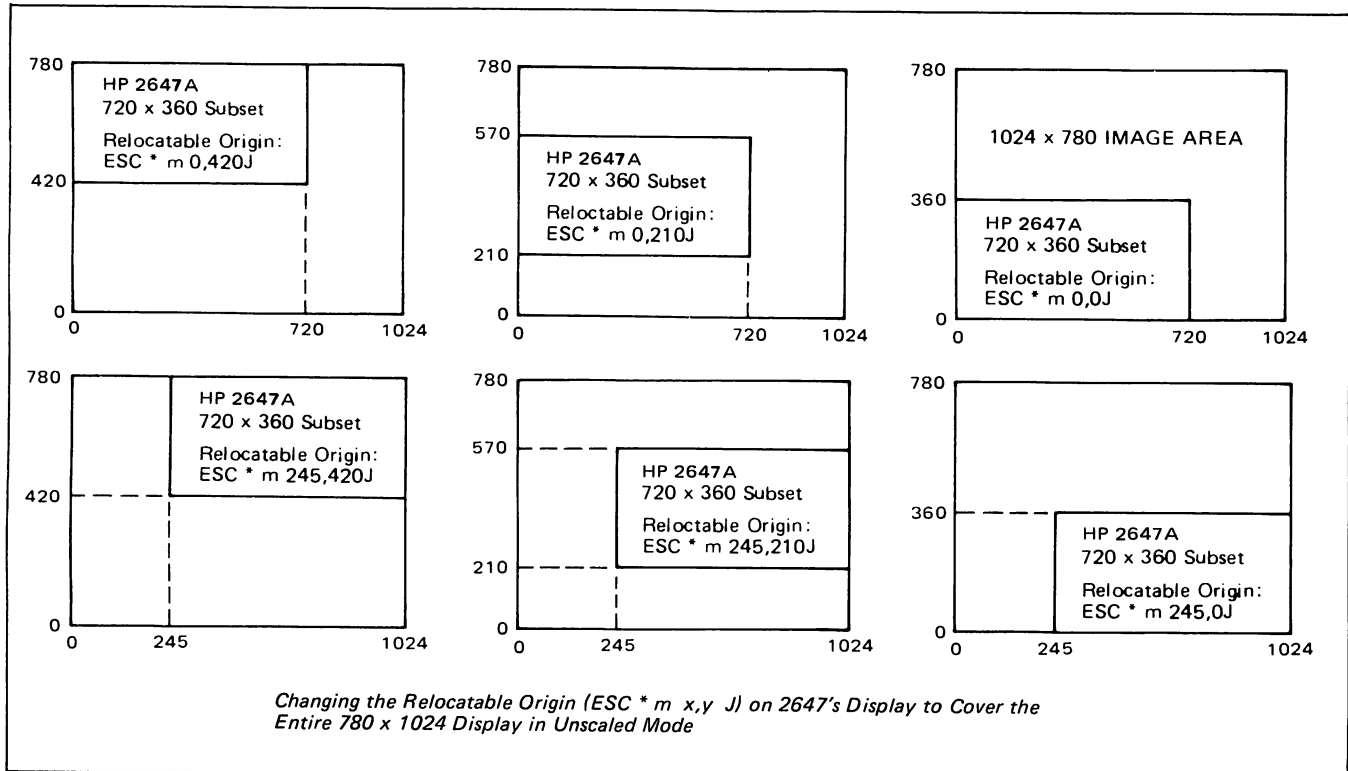


Figure 3-27. Unscaled Data

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 3-27).

## 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>.

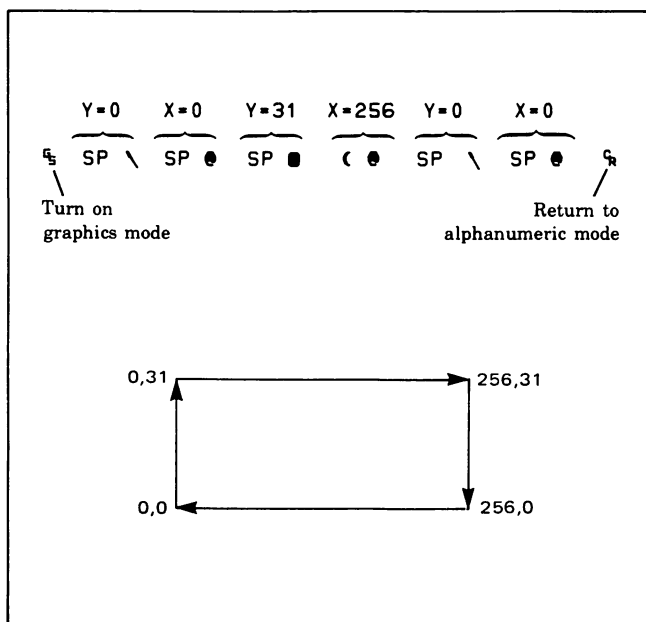
Data bytes sent to the terminal use bits 6 and 7 to indicate the byte is an Upper byte, a lower Y, or a lower X. Bit 8 (parity) is not used.

Bits		
7	6	
0	0	Upper X or Y byte
0	1	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 3-18 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 ( $\text{Ctrl}+\text{ds}$ ), 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 graphics mode while reading graphics 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.

Table 3-18. Coding of Compatibility Mode Graphics Data

X or Y Coordinate																Low Order Y		Low Order X	
																DEC.	ASCII	DEC.	ASCII
0	32	64	96	128	160	192	224	256	288	320	352	384	416	448	480	96	\	64	@
1	33	65	97	129	161	193	225	257	289	321	353	385	417	449	481	97	a	65	A
2	34	66	98	130	162	194	226	258	290	322	354	386	418	450	482	98	b	66	B
3	35	67	99	131	163	195	227	259	291	323	355	387	419	451	483	99	c	67	C
4	36	68	100	132	164	196	228	260	292	324	356	388	420	452	484	100	d	68	D
5	37	69	101	133	165	197	229	261	293	325	357	389	421	453	485	101	e	69	E
6	38	70	102	134	166	198	230	262	294	326	358	390	422	454	486	102	f	70	F
7	39	71	103	135	167	199	231	263	295	327	359	391	423	455	487	103	g	71	G
8	40	72	104	136	168	200	232	264	296	328	360	392	424	456	488	104	h	72	H
9	41	73	105	137	169	201	233	265	297	329	361	393	425	457	489	105	i	73	I
10	42	74	106	138	170	202	234	266	298	330	362	394	426	458	490	106	j	74	J
11	43	75	107	139	171	203	235	267	299	331	363	395	427	459	491	107	k	75	K
12	44	76	108	140	172	204	236	268	300	332	364	396	428	460	492	108	l	76	L
13	45	77	109	141	173	205	237	269	301	333	365	397	429	461	493	109	m	77	M
14	46	78	110	142	174	206	238	270	302	334	366	398	430	462	494	110	n	78	N
15	47	79	111	143	175	207	239	271	303	335	367	399	431	463	495	111	o	79	O
16	48	80	112	144	176	208	240	272	304	336	368	400	432	464	496	112	p	80	P
17	49	81	113	145	177	209	241	273	305	337	369	401	433	465	497	113	q	81	Q
18	50	82	114	146	178	210	242	274	306	338	370	402	434	466	498	114	r	82	R
19	51	83	115	147	179	211	243	275	307	339	371	403	435	467	499	115	s	83	S
20	52	84	116	148	180	212	244	276	308	340	372	404	436	468	500	116	t	84	T
21	53	85	117	149	181	213	245	277	309	341	373	405	437	469	501	117	u	85	U
22	54	86	118	150	182	214	246	278	310	342	374	406	438	470	502	118	v	86	V
23	55	87	119	151	183	215	247	279	311	343	375	407	439	471	503	119	w	87	W
24	56	88	120	152	184	216	248	280	312	344	376	408	440	472	504	120	x	88	X
25	57	89	121	153	185	217	249	281	313	345	377	409	441	473	505	121	y	89	Y
26	58	90	122	154	186	218	250	282	314	346	378	410	442	474	506	122	z	90	Z
27	59	91	123	155	187	219	251	283	315	347	379	411	443	475	507	123	(	91	[
28	60	92	124	156	188	220	252	284	316	348	380	412	444	476	508	124	)	92	^
29	61	93	125	157	189	221	253	285	317	349	381	413	445	477	509	125	~	93	]
30	62	94	126	158	190	222	254	286	318	350	382	414	446	478	510	126	~	94	^
31	63	95	127	159	191	223	255	287	319	351	383	415	447	479	511	127	~	95	_
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	← DEC.			
SP	..	#	\$	%	&			(	)	°	+	.	-	.	/	← ASCII			
High Order X & Y																			
X or Y Coordinate																Low Order Y		Low Order X	
																DEC.	ASCII	DEC.	ASCII
512	544	576	608	640	672	704	736	768	800	832	864	896	928	960	992	96	\	64	@
513	545	577	609	641	673	705	737	769	801	833	865	897	929	961	993	97	a	65	A
514	546	578	610	642	674	706	738	770	802	834	866	898	930	962	954	98	b	66	B
515	547	579	611	643	675	707	739	771	803	835	867	899	931	963	995	99	c	67	C
516	548	580	612	644	676	708	740	772	804	836	868	900	932	964	996	100	d	68	D
517	549	581	613	645	677	709	741	773	805	837	869	901	933	965	997	101	e	69	E
518	550	582	614	646	678	710	742	774	806	838	870	902	934	966	998	102	f	70	F
519	551	583	615	647	679	711	743	775	807	839	871	903	935	967	999	103	g	71	G
520	552	584	616	648	680	712	744	776	808	840	872	904	936	968	1000	104	h	72	H
521	553	585	617	649	681	713	745	777	809	841	873	905	937	969	1001	105	i	73	I
522	554	586	618	650	682	714	746	778	810	842	874	906	938	970	1002	106	j	74	J
523	555	587	619	651	683	715	747	779	811	843	875	907	939	971	1003	107	k	75	K
524	556	588	620	652	684	716	748	780	812	844	876	908	940	972	1004	108	l	76	L
525	557	589	621	653	685	717	749	781	813	845	877	909	941	973	1005	109	m	77	M
526	558	590	622	654	686	718	750	782	814	846	878	910	942	974	1006	110	n	78	N
527	559	591	623	655	687	719	751	783	815	847	879	911	943	975	1007	111	o	79	O
528	560	592	624	656	688	720	752	784	816	848	880	912	944	976	1008	112	p	80	P
529	561	593	625	657	689	721	753	785	817	849	881	913	945	977	1009	113	q	81	Q
530	562	594	626	658	690	722	754	786	818	850	882	914	946	978	1010	114	r	82	R
531	563	595	627	659	691	723	755	787	819	851	883	915	947	979	1011	115	s	83	S
532	564	596	628	660	692	724	756	788	820	852	884	916	948	980	1012	116	t	84	T
533	565	597	629	661	693	725	757	789	821	853	885	917	949	981	1013	117	u	85	U
534	566	598	630	662	694	726	758	790	822	854	886	918	950	982	1014	118	v	86	V
535	567	599	631	663	695	727	759	791	823	855	887	919	951	983	1015	119	w	87	W
536	568	600	632	664	696	728	760	792	824	856	888	920	952	984	1016	120	x	88	X
537	569	601	633	665	697	729	761	793	825	857	889	921	953	985	1017	121	y	89	Y
538	570	602	634	666	698	730	762	794	826	858	890	922	954	986	1018	122	z	90	Z
539	571	603	635	667	699	731	763	795	827	859	891	923	955	987	1019	123	(	91	[
540	572	604	636	668	700	732	764	796	828	860	892	924	956	988	1020	124	)	92	^
541	573	605	637	669	701	733	765	797	829	861	893	925	957	989	1021	125	~	93	]
542	574	606	638	670	702	734	766	798	830	862	894	926	958	990	1022	126	~	94	^
543	575	607	639	671	703	735	767	799	831	863	895	927	959	991	1023	127	~	95	_
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	← DEC			
0	1	2	3	4	5	6	7	8	9	:	:	<	=	>	?	← ASCII			
High Order X & Y																			

Example: 340Y,70X is found as follows:

340Y = 42 (upper Y) 116 (Lower Y) 70X = 34 (Upper X) 70 (Lower X)

340Y,70X → \* t " F

## INTRODUCTION

The terminal's I/O devices (display, cartridge tape units, printer, and devices connected to the HP-IB) can be program-controlled from the user-defined softkeys, cartridge tapes, or a computer through the use of two types of generalized escape sequences.

- $\text{\textasciitilde{c}}$  <command sequence> $\text{\textasciitilde{}}$
- $\text{\textasciitilde{p}}$  <parameter sequence>

The  $\text{\textasciitilde{c}}$  method allows English-like commands to be used in the command sequence to perform a given function. It is recommended that you use  $\text{\textasciitilde{c}}$  sequences, unless backward compatibility to HP 2645 or HP 2648 terminals is required.

The  $\text{\textasciitilde{p}}$  method uses special codes of numerical and alphabetical characters to perform a given function. This method is compatible with HP 2645 and HP 2648 terminals.

The following are examples of the escape sequences used to control a device and/or transfer information.

$\text{\textasciitilde{c}}$ Copy All from Right tape to Display $\text{\textasciitilde{}}$ or $\text{\textasciitilde{p}}$ 2s 3d M	Transfer all information stored on the right car- tridge tape unit to the display.
$\text{\textasciitilde{c}}$ REwind Left tape $\text{\textasciitilde{}}$ or $\text{\textasciitilde{p}}$ 1u C	Rewind the left cartridge tape unit.
$\text{\textasciitilde{p}}$ 1^ (no equivalent $\text{\textasciitilde{c}}$ sequence)	Fetch the status of the left cartridge tape unit.
$\text{\textasciitilde{p}}$ 2d 25W (no equivalent $\text{\textasciitilde{c}}$ sequence)	Write the next 25 bytes sent from the computer on the right cartridge tape unit.

## NOTE

Data can be sent directly from cartridge tape to a computer by the READ key. Also, data can be sent directly from a computer to cartridge tape, printer or HP-IB device by the **RECORD** key. (See Section 11 of the *HP 2647A User's Manual* for a description of these features.)

## USING THE $\text{\textasciitilde{c}}$ GENERALIZED ESCAPE SEQUENCE

The command syntax is given in table 4-1. An abbreviated form of the command syntax may be used by sending only the uppercase characters given in the command sequence. Also, words in braces "{ }" may be omitted from the syntax. All escape sequences must be terminated by a  $\text{\textasciitilde{}}$ .

Table 4-2 lists the command syntax abbreviations (and their meanings) in alphabetical order. (You may find this table helpful when debugging programs that contain the abbreviated syntax.)

**Example:** Assign the right tape as the source device.

From the Assign command syntax given in table 4-1:

$\text{\textasciitilde{c}}$ Assign Source to Right tape $\text{\textasciitilde{}}$

Using only the uppercase characters in the syntax:

$\text{\textasciitilde{c}}$ A S R  $\text{\textasciitilde{}}$

A cross-reference list of command functions using either  $\text{\textasciitilde{c}}$  or  $\text{\textasciitilde{p}}$  is given in the programmer's reference table in Appendix B.

Table 4-1.  $\epsilon, c$  Generalized Escape Sequence Syntax

<p> <b>** &lt;device&gt; defined as</b>            Left tape            Right tape            EXternal printer (A non HP-IB printer)            SHared printer#&lt;n&gt; (SH#5) (An HP-IB printer)            Display            TErминаl#&lt;p&gt;            Hp-ib#&lt;p&gt;[#&lt;s&gt;[#&lt;m&gt;]]            where:                p = primary address                s = secondary address                m = module address            DATacomm            Null            Graphics         </p> <p> <b>** &lt;name&gt; defined as</b>            Source            Destination            LOG            user specified name         </p>	
Assign	Source {to} <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>
BYE — <CR>	
CLOse — Window#<n> — <CR>	
COmpare	All {of} <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>
CONDition <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>	
Copy	All {from} <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>
Disable — Edit {mode} — <CR> or     Record {mode} — Enable     Verify {mode} —	
Display — Window#<n> — <CR>	
EXECUTE <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>	
EXIT <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">Command file</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">Application</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>	
Find	File — <[+!-] n> {on} <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">&lt;device&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div> <div style="display: inline-block; vertical-align: middle;"> <div style="display: inline-block; vertical-align: middle;">End {of} Data</div> <div style="display: inline-block; vertical-align: middle;">&lt;name&gt;</div> <div style="display: inline-block; vertical-align: middle;">&lt;CR&gt;</div> </div>

Table 4-1.  $\text{\textcircled{E}}$ , c Generalized Escape Sequence Syntax (Continued)

HELLO ——— $\langle$ user.group $\rangle$ ——— $\langle$ CR $\rangle$	
Mark ——— File Header ——— {on} ———	$\langle$ device $\rangle$ ——— $\langle$ CR $\rangle$ $\langle$ name $\rangle$ ——— , 
REPort Status {of} ——— Command ——— $\langle$ CR $\rangle$	
RESume ———	Command file ——— $\langle$ CR $\rangle$ Application ———
REWind ———	$\langle$ name ——— $\langle$ CR $\rangle$ $\langle$ device $\rangle$ ——— , 
SET ———	Time $\langle$ n $\rangle$ : $\langle$ n $\rangle$ {: $\langle$ n $\rangle$ } ——— AM ——— $\langle$ CR $\rangle$ PM Date " $\langle$ string $\rangle$ " — $\langle$ CR $\rangle$
SHow ———	Assignments ——— $\langle$ CR $\rangle$ Volumes ——— Tapes ——— Time ——— Date ———
SKip ———	$\langle$ [+ -] n $\rangle$ ——— {lines on} ——— $\langle$ name $\rangle$ ——— $\langle$ CR $\rangle$ End {of} ——— Data {on} ——— $\langle$ device $\rangle$ ——— , Page {on} ———
SUSpend ———	Command file ——— $\langle$ CR $\rangle$ Application ———
TEll ———	TERminal# $\langle$ n $\rangle$ ——— " $\langle$ string $\rangle$ " $\langle$ CR $\rangle$ $\langle$ device $\rangle$ ——— $\langle$ name $\rangle$ ——— , 
Test ———	$\langle$ CR $\rangle$ TERminal# $\langle$ n $\rangle$ ——— Hp-ib# $\langle$ p $\rangle$ [# $\langle$ s $\rangle$ [# $\langle$ m $\rangle$ ]] ——— DATacomm ——— Tapes ———
TRansfer ———	All ——— {from} ——— $\langle$ device $\rangle$ ——— {to} ——— $\langle$ device $\rangle$ ——— $\langle$ CR $\rangle$ File ——— $\langle$ name $\rangle$ ——— Line ——— $\langle$ CR $\rangle$ $\langle$ CR $\rangle$ ——— $\langle$ name $\rangle$ ——— , $\langle$ CR $\rangle$ ——— $\langle$ CR $\rangle$



Table 4-2. Command Syntax Abbreviations

All	Destination	LOG	SHow
AM	Disable	Mark	SKip
Application	Edit	Name	Source
Assign	Enable	Null	Status
Assignments	End	Page	SUspend
CLOse	EXecute	PM	Tapes
Command File	EXIT	Record	TEll
COmpare	EXternal printer	REPort	TErминаl#<p>
CONdition	File	RESume	Test
Copy	Find	Rewind	Time
DISplay	Graphics	Right tape	Window#<n>
Data	Hp-ib#<p>[#<s>[#<m>]]	SET	Verify
DATacomm	Left tape	SHared printer#<n>	Volumes
Date	Line		

## Assign

The Assign command sequence specifies which device(s) will be assigned to the logical filenames Source, Destination, LOG, or your own filename. The default values are:

Source = Left tape  
Destination = Right tape  
LOG = Display

Multiple devices may be assigned to Destination and LOG; a comma and a space is required to separate the device names. Your own filename may be up to 11 characters. These may be alphabetic, numeric, single quote, or underscore.

**Example:** Assign the left tape and local printer as the Destination devices.

```
Ⓔ,c Assign Destination {to} Right tape, External
printer␣
Ⓔ,cA D R EX␣
```

**Example:** Assign the display workspaces as the Source device.

```
Ⓔ,c Assign Source {to} Display ␣
Ⓔ,cA S DI␣
```

**Example:** Assign the graphics memory as the source device.

```
Ⓔ,c Assign Source {to} Graphics ␣
Ⓔ,cA S G␣
```

**Example:** Assign the data communications line to the host computer as the destination for data.

```
Ⓔ,c Assign Destination {to} Datacomm ␣
Ⓔ,cA D DA␣
```

**Example:** Assign "PLOT" as a user-assigned name to the left tape.

```
Ⓔ,c Assign Name PLOT {to} Left tape ␣
Ⓔ,cA N PLOT L␣
```

**Example:** Assign "PLOTTER" as a user-assigned name to the HP 7245 Plotter Printer connected to the HP-IB (device#4).

```
Ⓔ,c Assign Name PLOTTER {to} Hp-ib#4 ␣
Ⓔ,cA N PLOTTER H#4␣
```

## Bye

The BYE command sequence terminates use of the current user.group assigned by the HELLO command sequence on the shared printer listings.

**Example:** Ⓔ,c BYE␣

## CLOse

The CLOse command turns off any of the terminal display lines (message, command, or softkey labels) on the screen. (For turn-on, see DISplay command.)

**Example:** Turn off the message line on the display.

```
Ⓔ,c CLOse Window#5 ␣
Ⓔ,cCLO W#5␣
```

**Example:** Turn off the command line on the display.

```
Ⓔ,c CLOse Window#6 ␣
Ⓔ,cCLO W#6␣
```

**Example:** Turn off the softkey label line on the display.

```
Ⓔ,c CLOse Window#7 ␣
Ⓔ,cCLO W#7␣
```

## COmpare

The COmpare command sequence compares records between two devices. If a compare is not successful in any record, a message will appear on the message line, giving the byte number in the record that did not compare. Nothing is sent to the computer system to indicate successful or unsuccessful comparison.

The defaults are: File, Source, Destination.

**Example:** Compare all of the data on the left cartridge tape to the data on the right cartridge tape.

```
␣,c COmpare All {of} Left tape {to} Right tape ␣
␣,cCO A L R␣
```

**Example:** Compare the current file of the source device to the destination device. (The defaults are: File, Source, Destination.)

```
␣,c COmpare ␣
␣,cCO␣
```

**Example:** Compare the current line of the device with a user-assigned name of "INPUT" to the device with a user-assigned name of "OUTPUT".

```
␣,c COmpare Line {of} INPUT {to} OUTPUT ␣
␣,cCO L INPUT OUTPUT␣
```

## CONdition

The condition command sequence causes either the left tape or right tape to run forward to EOT (end-of-tape) then run the tape backware to BOT (beginning-of-tape). (For more information on tape conditioning, see the *User's Manual*.)

**Example:** Condition the left cartridge tape.

```
␣,c CONdition Left tape ␣
␣,cCON L␣
```

## Copy

The Copy command sequence copies 7--bit ASCII data (a line, a file, or all) from the specified source to the specified destination. The defaults are Source, Destination, and File. If the computer system sent ␣,c Copy␣, a file would be copied from Source to Destination. Multiple devices may be specified as destinations by separating the device names with a comma and a space, or by equating the devices to Destination in the Assign command sequence.

### NOTE

To copy 8-bit binary data (e.g., graphics memory), the TRansfer command sequence must be used.

**Example:** Copy a file from the Source device to the Destination device.

```
␣,c Copy ␣
␣,cC␣
```

**Example:** Copy all data on the Source device (assume the display) to the local printer and the right tape.

```
␣,c Copy All {from} Source {to} EXternal printer,
Right tape ␣
␣,cC A S EX, R␣
```

**Example:** Copy the contents of the left tape to the HP-IB device#4 (e.g., HP 7245A Plotter Printer). Data must be ASCII to use the COPY command sequence.

```
␣,c Copy File {from} Left tape {to} Hp-ib#4 ␣
␣,cC F L H#4␣
```

## Enable/Disable (Edit Mode, Record Mode, or Verify Mode)

The Enable and Disable command sequences turn on and off Edit Mode, Record Mode, and Verify Mode.

**Edit Mode.** In local mode (**REMOTE** key up), the assigned Source and Destination devices are the source and destination for data, respectively. The display workspace presently on the screen displays the data read from Source. The READ and RECORD keys operate normally. (See the *User's Manual* for a description of Edit Mode.)

In remote mode (**REMOTE** key down), Data Logging Mode is enabled. (See the *User's Manual* for a description of Data Logging Mode.)

**Example:** Turn on Edit Mode.

```
␣,c Enable Edit {mode} ␣
␣,cE E␣
```

**Record Mode.** Enabling Record Mode has the same effect as pressing the **RECORD** key.

**Example:** Turn off Record Mode.

```
␣,c Disable Record {mode} ␣
␣,cD R␣
```

**Verify Mode:** The Enable/Disable Verify command sequence turns the write-backspace-read function on and off. This function assures the integrity of data sent to a cartridge tape unit.

**Example:** Turn on Verify Mode.

```
␣,c Enable Verify {mode} ␣
␣,cE V␣
```

## Display

The DIspay command sequence controls the four display workspaces and the three terminal control lines (message, command, and softkeys). Also, the DIspay command sequence may be used to specify the display workspace (1, 2, 3, or 4), message line (5), or command line (6) as the destination for data that is to follow.

**Example:** Send "Change Left tape to Right tape in the command line" to the message line. (To turn off the message line, use the CLOse command described previously.)

```
␣,c DIspay Window#5 ␣ Change Left tape to Right
    tape in the command line ␣
```

```
␣,cDI W#5␣Change Left tape to Right tape in the com-
    mand line␣
```

**Example:** Turn on memory lock in workspace 3, clear workspace 1, and display workspace 2 on the screen.

```
␣,c DIspay Window#3 ␣ ␣1
␣,c DIspay Window#1 ␣ ␣H␣J
␣,c DIspay Window#2 ␣
```

```
␣,cDI W#3␣␣1
␣,cDI W#1␣␣H␣J
␣,cDI W#2␣
```

**Example:** Display the command line.

```
␣,c DIspay Window#6 ␣
␣,cDI W#6␣
```

## EXecute

The EXecute command sequence allows command sequences to be executed from a device rather than from the computer system. Each command sequence must be on a separate line or record. If user intervention is required at any point in the execution (such as inserting a tape), the SUSpend command sequence may be used. (Also see "EXIT" and "REsume".)

**Example:** Execute a list of commands stored in display workspace 3.

```
␣,c DIspay Window#3 ␣ or ␣,cDI W#3␣
␣,c EXecute Display ␣ ␣,cEX DI␣
```

The commands to be executed do the following:

- Read file 2 on the right cartridge tape to the display workspace 1.

- Suspend execution for user editing of the displayed data.
- Copy the edited data on the display to the printer.

The commands are:

```
Find File 2 {on} Right tape
Display Window #1
Copy File {from} Right tape {to} Display
SUSpend Command file
    (The user would use the RESUME command here to
    continue execution.)
Copy File <from> Display <to> EXternal printer
EXIT Command file
```

or in abbreviated form:

```
F F 2 R
DI W..#1
C F R DI
SU C
    (User uses RESUME command here.)
C F DI EX
EXIT
```

## EXIT

The EXIT command sequence terminates execution from a file or application program. (See "EXecute" above.)

## Find

The Find command sequence positions the device to an absolute or relative (+, -) file number, or to the end-of-data mark. Some printers position to top-of-form when a Find file command is sent. Also, some printers do not respond to line positioning.

**Example:** Find file 4 on the left cartridge tape, then position the left tape forward 6 lines.

```
␣,c Find File 4 {on} Left tape ␣
␣,c SKip +6 {lines} {on} Left tape ␣
```

or

```
␣,cF F 4 L␣
␣,cSK +6 L␣
```

**Example:** Find the end-of-data mark on the right tape.

```
␣,c Find End {of} Data {on} Right tape ␣
```

or

```
␣,cF E D R␣
```

## HELLO

The HELLO command sequence allows the user to assign a "user.group" to the listings printed by the shared printer. Up to eight characters may be assigned to each field.

**Example:** Specify "PROJECT3.CHEMLAB" to be printed on all shared printer listings.

```
␣,c HELLO PROJECT3.CHEMLAB ␣
```

## Mark

The Mark command sequence writes file marks on the specified device. Some printers position to top-of-form when a Mark command is sent.

**Example:** Write a file mark on the right cartridge tape.

```
␣,c Mark File Header {on} Right tape ␣
or
␣,cM F H R␣
```

## REPort

The REPort command sequence sends the successful/unsuccessful completion status of the previous command execution to the datacomm line. The status consists of a 5-digit number; the meaning of this number is given in a table at the end of section 6, Status.

**Example:**

```
␣,cREPort Status {of} Command ␣
or
␣,cREP S C␣
```

## RESume

The RESume command sequence returns control to the command file or the application if it had been suspended. (Also see SUSpend and EXecute.)

**Example:**

```
␣,c RESume Command file ␣
or
␣,cRES C␣
```

## REwind

The REwind command sequence rewinds the specified tape to BOT (beginning of tape). A user-assigned device name may be used.

**Example:** Rewind the left tape.␣,c RE-  
wind Left tape ␣  
or  
␣,cRE L␣

**Example:** Rewind the tape unit with the user-assigned device name "BILLING".

```
␣,c REwind BILLING ␣
or
␣,cRE BILLING␣
```

## SET

The SET command sequence allows the user to set the current time in hours, minutes, and seconds (optional) and to set the current date in the terminal. When power is turned on, or a hard reset is performed, the time is set to 8:00:00 AM and the date string is set to 30 asterisks (\*). Once the time is set, it will not require resetting until the terminal is turned off or a hard reset is performed. However, the date will require resetting each day. To set the date, the user may select any data string up to 30 characters.

**Example:** Set the terminal time to 10 hours, 15 minutes, 23 seconds AM.

```
␣,c SET Time 10:15:23 AM ␣
or
␣,cSET TI 10:15:23 AM␣
```

**Example:** Set the terminal date to "FRIDAY, JULY 14, 1978".

```
␣,c SET Date "FRIDAY, JULY 14, 1978" ␣
or
␣,cSET D "FRIDAY, JULY 14, 1978"␣
```

## SHow

The SHow command sequence lists the device assignments, current file number and space remaining on each tape, current terminal time, or current terminal date string on the LOG device.

**Example:** Show the current device assignments for Source, Destination, LOG, and user-defined assignments.

```
⌘,c SHow Assignments ␣
      or
⌘,cSH A␣
```

**Example:** Show the current file number and space remaining for each tape unit on the display.

```
⌘,c Assign LOG {to} Display ␣
⌘,c SHow Tapes ␣
      or
⌘,cA LOG DI␣
⌘,cSH T␣
```

**Example:** Show the current terminal time.

```
⌘,c SHow Time ␣
      or
⌘,cSH TI␣
```

**Example:** Example: Show current terminal date string.

```
⌘,c SHow Date ␣
      or
⌘,cSH D␣
```

## SKip

The SKip command sequence positions a device to a relative line, top-of-form, or beyond an end-of-data mark.

**Example:** Position the tape with the user-assigned name of "MEMO" backward six lines.

```
⌘,c SKIP -6 {lines} {on} MEMO ␣
      or
⌘,cSK -6 MEMO␣
```

**Example:** Position the shared printer to next top-of-form.

```
⌘,c SKip Page {on} SHared#5 ␣
      or
⌘,cSK P SH#5␣
```

**Example:** Position the left tape beyond the end-of-data mark. (The tape must be positioned to the end-of-data mark, first.)

```
⌘,c SKip End {of} Data {on} Left tape ␣
      or
⌘,cSK E D L␣
```

## SUsuspend

The SUSpend command sequence allows user intervention when executing command sequences from a file or application. A RESume command sequence returns control to the file containing the command sequences. An EXIT command sequence terminates execution. (See EXecute command.)

## TEll

The TELL command sequence sends up to 80 characters to the specified device or a terminal in a network. The character string must be enclosed in quotes.

**Example:** Send "This is Display Workspace 3" to workspace 3.

```
⌘,c Display Window#3 ␣
⌘,c TELL Display "This is Display Workspace 3" ␣
      or
⌘,cDI W#3␣
⌘,cTE DI "This is Display Workspace 3"␣
```

**Example:** Send "Insert GAS LAW CURVE DATA into left tape slot" to LOG device and terminals 3, 5, and 7 in the terminal network.

```
⌘,c TELL LOG,TErmi#3,TErmi#5,TErmi-
      nal#7 "Insert GAS LAW CURVE DATA into left tape
      slot" ␣
      or
⌘,cTE LOG,TE#3,TE#5,TE#7 &Insert GAS LAW CURVE
      DATA into left tape slot"␣
```

## Test

The Test command sequence tests the following:

- terminal (excluding tape units and data paths)
- left and right cartridge tape units.
- datacomm line (the test connector must be installed — see section 7.)
- Data path to a specified HP-IB device.
- Data path to a specified terminal in a network.

Further explanation of self tests, interpretation of results, and appropriate action is given in "Self Test" contained in section 7.

**Terminal.** The terminal test verifies the operation of the terminal (such as the display, ROM, character sets, etc.).

**Example:** Test the terminal.

```
⌘,c Test ␣
      or
⌘,cT␣
```

**Tape Units.** The tape unit test writes a worst case data pattern ("%Z") repeated 128 times to form a 256-character record) on the left tape unit, backspaces to the beginning of the record, reads and verifies the record, and writes and verifies a file mark. A worst case test pattern and file mark are written on the right tape unit and verified. Another terminal test is performed.

If a fault is detected during the Tape Test, the test will stop and one of the following error messages will be displayed:

NO TAPE, RUNOFF, DATA PROTECTED, FAIL, WRITE FAIL, STALL, or END OF TAPE.

(See the *User's Manual* for further information on tape test.)

**Example:** Test the tape units.

```
␣,c Test Tapes ␣
```

or

```
␣,cT T␣
```

**Datacomm:** The datacomm test verifies the data communications PCA in the terminal. The test connector, part no. 02645-60002, must be installed on the PCA to run the test. (See "Self Test" in section 7 for further information.)

**Example:** Test the data communications PCA.

```
␣,c Test DATacomm ␣
```

or

```
␣,cT DA␣
```

**Terminal#<n>.** The data path between terminals in a network may be tested by specifying the assigned terminal number (its primary address).

**Example:** Test terminal#5.

```
␣,c Test TErминаl#5 ␣
```

or

```
␣,cT TE#5␣
```

**HP-IB.** The HP-IB printed circuit assembly in the terminal may be tested. (See "Self Test" in section 7 for further information.)

**Example:** Test the HP-IB printed circuit assembly at module address 4.

```
␣,c Test Hp-ib ␣
```

or

```
␣,cT H␣
```

## TRansfer

The TRansfer command sequence copies 8-bit binary data (a line, a file, or all) from the specified source to the specified destination. The defaults are Source, Destination, and File. If the computer system sent ␣,c TRansfer <CR>, a file would be copied from Source to Destination. Multiple devices may be specified as destinations by separating the device names with a comma and a space, or by equating the devices to Destination in the Assign command sequence. (The TRansfer command sequence is useful for copying graphics memory data.)

**Example:** Copy a file from the Source device to the Destination device (e.g., left tape is the Source and Graphics is the Destination).

```
␣,c TRansfer ␣
```

or

```
␣,c TRansfer File {from} Source {to} Destination ␣
```

**Example:** Copy all data on the Source device (assume the right tape) to the HP-IB device #6 (e.g., HP 7245A Plotter/Printer).

```
␣,c TRansfer All {from} Source {to} Hp-ib#6 ␣
```

or

```
␣,c TRASH#6␣
```

**Example:** Copy the contents of graphics memory to the HP-IB device#7 (e.g., HP 2631G Printer).

```
␣,c TRansfer File {from} Graphics {to} Hp-ib#7 ␣
```

or

```
␣,cTRFGH#7␣
```

## USING THE $\text{\textbackslash}$ &p GENERALIZED ESCAPE SEQUENCE.

The generalized escape sequence for I/O device control is as shown in table 4-3. Items in angle brackets (< >) are replaced by an appropriate numerical value. Items in square brackets ([ ]) are optional.

The device control escape sequence is initiated by the characters  $\text{\textbackslash}$ &p and *terminated by an upper case character* (B, C, D, F, M, P, R, S, U, W, or ^).

The characters b, c, f, m, r, w, and ~ (lower case ^) indicate a command is to be performed. All other letters define parameters for the commands. For a given escape sequence, only one command character may be specified. Also, a device operation (other than a status request) should not be initiated before the previous device operation has been completed.

For example, after initiating a read command, the data record must be read by the CPU before another device operation is initiated. Otherwise, the read operation may not be executed properly.

During the execution of a command, input from the data communications interface is ignored and the keyboard is locked out except for the **RETURN** key during device-to-device transfer and read file operations. Pressing the **RETURN** key will terminate the operation in progress, set a flag to indicate user interrupt to the CPU, and unlock the keyboard. Other operations (such as rewind, condition tape, etc.), cannot be terminated by the return key.

Any errors in the escape sequence will cause the entire sequence to be ignored by the terminal. This may cause the CPU to go into a wait loop if a response is expected from the escape sequence. A programmed time-out can be used to counteract this problem.

Table 4-3.  $\text{\textbackslash}$ &p Device Control Escape Sequences

		$\text{\textbackslash}$ &p	commands
S	=	Source Device Assignment	[<"from" device code>]s]
D	=	Destination Device Assignment	[<"to" device code>d]
P,U,C	=	Device Command	[<control parameter>p][<device code>u]<control code>c
^	=	Status	[<device code>]^
R	=	Read	[<read control byte>]r
W	=	Write	[<byte count>]w
B	=	Copy (Compare) Record	[<control bit>]b
F	=	Copy (Compare) File	[<control bit>]f
M	=	Copy (Compare) All	[<control bit>]m

<b>where:</b>		control parameter (p) is:	
device codes (s, d, u) are:		a positive (+n), negative (-n), or (unsigned) (n) integer, specifying the number of records or files for (c) control codes 1 and 2.	
1 = left cartridge tape unit		(If no value is specified, +1 is assumed.)	
2 = right cartridge tape unit			
3 = display			
4 = external printer			
(If no value is specified, previous device assignments are in effect.)			
control code (c) is:		read control byte (r) is:	
Control Code (c)	Default Device	Function	0 = transmit next record
0	"from"	Rewind	1 = retransmit last record only
1	"from"	Space "p" records	2 = send byte count before transmitting next record
2	"from"	Space "p" files	= send byte count before retransmitting last record read
3	"from"	Locate end-of-data mark	4 = transmit file
4	"from"	Condition tape	6 = transmit file with byte count before each record
5	"to"	Record file mark	(If no value is specified, 0 is assumed.)
6	"to"	Record end-of-data mark	
7	"to"	Test cartridge tape unit	byte count (w) is:
8	"to"	Skip "p" records immediately without recording end-of-data mark	if no value is specified, ASCII is assumed. Data is received until a Line Feed (^) character or 256 characters (maximum) are received.
9		Turn on write-backspace-read mode	if a value is specified, binary data is assumed.
10		Turn off write-backspace-read mode	control bits (b, f, m) are:
(If no value is specified, 0 is assumed.)		0 or no integer = copy record (b), file (f), or all (m) from source device to destination device.	
		1 = compare record (b), file (f), or all (m) on source and destination devices.	

## SELECTING INPUT/OUTPUT DEVICES

The devices to be controlled are selected by the following escape sequence format:

```
Esc & p      [<"from" device code>s]
              [<"to" device code>d]
```

where device codes are:

- 1 = left cartridge tape unit
- 2 = right cartridge tape unit
- 3 = display
- 4 = external printer

### Example:

```
Esc & p 2s 1d 4D
```

(Specifies the right cartridge tape unit as the source of the information, and the left cartridge tape unit and printer as destinations for the information).

Only one "from" device may be specified for a given escape sequence. Multiple "to" devices may be specified.

At power on or hard reset, the "preset" assignments are left tape unit for "from" device and right tape unit for "to" device.

## CONTROLLING THE DEVICES

The device functions are controlled by escape sequences in the following format:

```
Esc & p      [<"from" device code>s]
              [<"to" device code>d]
              [<control parameter>p][<device
              code>u]
              [<control code>c]
```

### Examples:

Esc & p 2u 0C	Rewind the right cartridge tape unit.
Esc & p 2u - 1p 1C	Backspace one record on the right cartridge tape unit.
Esc & p 1u + 3p 2C	Forward space three files on the left cartridge tape unit.
Esc & p 1u 6p 2C	Find the sixth file on the left cartridge tape unit.
Esc & p 2u 3C	Locate end-of-data mark on the right cartridge tape unit.
Esc & p 1u 4C	Condition the tape on the left cartridge tape unit.

Esc & p 2u 5C	Record a file mark on the right cartridge tape unit.
Esc & p 2u 6C	Record end-of-data mark on the right cartridge tape unit.
Esc & p 1u 7C	Perform a cartridge tape test on the left cartridge tape unit.
Esc & p -5p 8C	Backspace five records immediately without recording end-of-data mark on the "to" device.
Esc & p 9C	Turn on write-backspace-read mode.
Esc & p 10C	Turn off write-backspace-read mode.

If the (p) parameter is omitted (control code 1, 2, or 8) or zero is specified, a default value of +1 is assumed.

## Cartridge Tapes

For the skip record functions (1 or 8), all movements are relative. Backspacing is indicated with a minus (-) sign preceding the p parameter number, while forward spacing is indicated by a plus (+) sign or no sign preceding the number. If a file mark is the last record encountered while backspacing, the tape is spaced forward so that the tape is positioned immediately after the file mark (i.e., just before the first record of the file). Also, the end-of-file mark status bit is set (bit 4 of cartridge tape unit status byte 0). In order to backspace past a file mark, you must specify at least 2 records.

For the locate file function (2), the (p) parameter may be either an absolute file number, or a relative file count indicated by a plus (+) or minus (-) sign preceding the number. The tape is positioned before the first record of the specified file (i.e., after the file mark of the previous file). Files are numbered from 1 to 255.

Skip/locate functions (1, 2, 3, or 8) are limited to the bounds of load point and end-of-data (or end of tape). Any attempt to exceed these bounds will cause the command to be aborted, and the appropriate bits in the device status will be set. To append a new file on a cartridge, first find the end-of-data mark (3), then record a file mark (5) to terminate the last file before starting a new file. If a file mark is not written, the new data will be appended to the end of the last file.

Unless the "skip p records immediately" function (8) is used, an end-of-data mark will be written before a skip, locate, rewind, or condition tape operation (0-4) is performed, if the last function performed on the cartridge was a record operation. "Skip p records immediately" inhibits the writing of the end-of-data mark and is intended primarily for write verification in a write-backspace-read



operation sequence. After using the "skip p records immediately" function, a file mark must be written on the tape before rewinding the tape. This function should not be used to skip forward on a cartridge on which a record function was the last operation.

## Display

The display ignores all control functions. Any control functions applied to the display will be flagged as executing successfully.

## Printer

All functions, except the skip lines functions (1), cause one ASCII Form Feed character (octal 14) to be sent to the The Form Feed character will cause some printers to skip to the top of the next page. The skip lines function will cause the printer to skip p lines using the absolute value of p.

Generally, the terminator sent by the terminal in response to the I/O control escape sequence may be a CR(LF), RS, or GS depending on the communications protocol and terminal configuration (refer to section V). Whenever the terminator is specified, the characters CR(LF)/RS/GS will be used to denote the above conditions.

## TRANSFERRING DATA FROM DEVICE TO COMPUTER

Data may be transferred from the cartridge tape units or display to the computer by the following escape sequence.

```
Esc p      [ <"from" device code> s ]
           [ <read control byte> ] r
```

### Examples:

```
Esc p 2s 2R    Right tape unit is selected as the new
                "from" device; send byte count before
                sending next record
```

```
Esc p 0R      Send next record from the "from" device
```


The read control byte has the following meanings:

- 0 = Transmit next record with no byte count
- 1 = Retransmit last record only
- 2 = Send byte count before transmitting next record
- 3 = Send byte count before retransmitting last record read
- 4 = Transmit file
- 6 = Transmit file with byte count before each record

## ASCII Transfers

An ASCII transfer is specified by a read *without* byte

count. For reads without byte count (0R, 1R, 4R) an enabling multicharacter transfer trigger (DC1) from your program (following the escape sequence) causes one record to be read and transmitted to the computer.

For reads without byte count (0R, 1R), an enabling block transfer from your program (following the escape sequence) causes one record to be read and transmitted to the computer. A CR(LF)/RS/GS terminator is appended to the end of the record. Any Line Feed characters in a record will not be transmitted if the  key is not latched down. When a file mark is read, the terminal sends an RS (Record Separator) or GS (Group Separator) followed by CR(LF). If the terminal is in BLOCK MODE strapped for page, only RS or GS is sent. The escape sequence must be repeated to read each record from a device.

If a byte count is specified in the escape sequence (2R, 3R, or 6R), the information is sent in two steps:

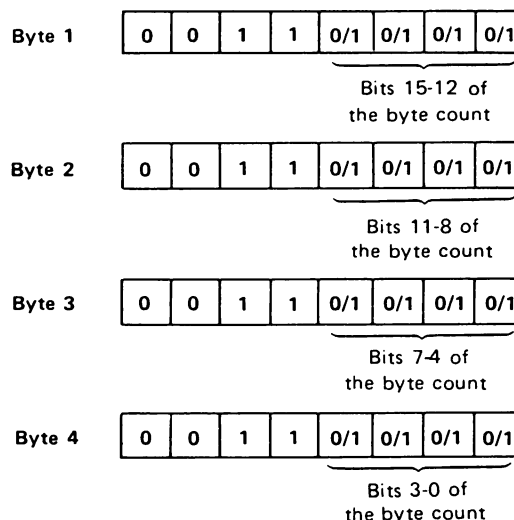
**STEP 1.** When your program issues a block transfer enable (following the escape sequence), the byte count (the number of bytes in the record to be sent) will be transmitted to the computer.

**STEP 2.** When your program enables the next block transfer from the terminal, the record will be sent to the computer. All characters within the record will be sent (including LFs). No record terminator will be appended to the record. (After the record has been sent, the Request to Send (CA) line from the terminal will be dropped for about 5 milliseconds. This may be used as an interrupt condition for the computer.)

## Binary Transfers

The byte count is sent in binary as four bytes followed by a CR(LF), or an RS/GS if the terminal is in BLOCK MODE, strapped for page.

If retransmit is specified (1R or 3R), the previous record read is transmitted. Only the previous record can be retransmitted. Intervening read or write operations are not allowed.



To transfer binary data, the read control byte in the escape sequence must specify a byte count (2R or 3R), and the PARITY switch on the terminal must be set to NONE. If a non-recoverable error occurs, the terminal will send an RS/GS as if a file mark were detected. The type of error can be determined by inspecting the device status.

## TRANSFERRING DATA FROM COMPUTER TO DEVICE

A record of data may be transferred from the computer to the cartridge tape units, display, and optional printer by the following sequence:

```

^&p      [<"to" device code>d]
          [<byte count>]w

```

### Example:

```

^&p 15W      Send the next 15 data bytes from the
              computer to all "to" devices

```

The byte count must consist of ASCII numerals. The maximum value is 256. If no byte count is specified, data is accepted by the terminal until a Line Feed character is received or a maximum of 256 characters are received. If a byte count is specified, an ENquiry character (octal 5) must be sent after the escape sequence in point-to-point operation, but before the data bytes. When the terminal responds with an ACKnowledge character (octal 6), then the data bytes may be sent. For multipoint operation, refer to section V.

During the transmission of the data byte, nulls and rub-outs will not be stripped out of the data byte stream, and the terminal will not respond to an ENquiry character from the computer with an ACKnowledge character.

To use all eight bits of each byte for binary data, no parity (NONE) should be selected for both terminal and the computer.

The keyboard will be locked out until the record has been transferred to all destination devices. Upon successful completion of the operation, the terminal will respond with an S followed by CR(LF)/RS/GS after receiving a block transfer enable. Any non-recoverable write errors terminate the escape sequence immediately, and the terminal will respond with an F followed by CR(LF)/RS/GS instead.

## COPYING A RECORD

A record may be copied from one terminal device to

another. The escape sequence format is as follows:

```

^&p      [<"from" device codes>s]
          [<"to" device code>d]
          b

```

### Example:

```

^&p B      Copy one record from the "from" device
           to all "to" devices.

```

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 where the display is the "from" device.)

An error condition results if an attempt is made to copy a record beyond the available data space of a "to" device (for example, end of tape). Also an error condition results if the "from" device is located at end-of-data.

Upon successful completion of the transfer, the terminal sends an S followed by CR(LF)/RS/GS after receiving a block transfer enable. If an error occurred during the transfer, an F followed by CR(LF)/RS/GS is sent instead.

## COPYING A FILE

A file may be copied from one terminal device to another. The escape sequence format is as follows:

```

^&p      [<"from" device code>s]
          [<"to" device code>d]
          f

```

### Example:

```

^&p 2s 4d F  Copy one file from the right cartridge
              tape unit to the printer. The right car-
              tridge tape unit is selected as the new
              "from" device; and the printer is selected
              as the new "to" device.

```

The file copy operation starts from the current position on the from device and copies one record at a time until a file or end-of-data mark is detected. Upon completion, the mark is sent to all "to" devices. If the data space is exceeded on a "to" device (for example, end of tape), the transfer is terminated and an error condition results.

Upon successful completion of the transfer, the terminal sends an S followed by CR(LF)/RS/GS after receiving a block transfer enable. If an error occurred during the transfer, an F followed by CR(LF)/RS/GS is sent instead.

The terminal operator may interrupt this operation by pressing the **RETURN** key. In this case, the termination response is U followed by CR(LF)/RS/GS.

## COPYING TO END OF MEDIUM

All files on a from device may be copied to one or more to devices by using the following escape sequence format:

```

^&p      [<"from" device code>s]
          [<"to"device code>d]
          m

```


### Example:

```

^&p 1s 4d M      Copy all data from the left tape unit to
                  the printer.

```

The end of medium copy operation starts from the current position on the "from" device to the end of medium (end-of-data mark on the cartridge tape unit, or end of display memory). If the data space is exceeded on the "to" device, the copy operation is terminated and an error condition results.

Upon successful completion of the transfer, the terminal sends an S followed by CR(LF)/RS/GS after receiving a block transfer enable. If an error occurred during the transfer, an F followed by CR(LF)/RS/GS is sent instead. The terminal operator may interrupt this operation by pressing the  key. In this case, the termination response is U followed by CR(LF)/RS/GS.

## FAST BINARY READ (PROGRAM LOADING)

Binary data can be read directly into the computer without the normal handshake process by using:


```
^&e
```

The principal use of this escape sequence is for loading of binary data. When the sequence is issued to the terminal, parity is turned off, and transmission begins immediately without waiting for a block transfer enable from the computer. Transmission continues until a file mark is read.

Data is transmitted as read from the source device. No terminators (that is, CR, LF, RS, GS) are appended to the end of record. The mark does not cause an RS (Record Separator) to be transmitted; it serves only to terminate transmission. Instead, the reading of a file mark or end-of-data mark causes two null bytes (all zeros) to be transmitted. If an I/O error occurs, the binary read operation is terminated. Two "all ones" bytes will be sent if the tape is already positioned past the end-of-data mark when the fast binary read operation is invoked. If the Fast Binary Read Strapping Option is set (see page 5-12), the baud rate of the 2647 will automatically switch to 9600 baud. This is valid only if the CPU is capable of receiving at 9600 baud and the CPU's interface is clocked by the 2647.

## INDICATING SUCCESSFUL COMPLETION OF A PROGRAM-CONTROLLED FUNCTION

Completion of a device control or transfer of information should be tested by your program as follows:

- Initiate a block transfer from the terminal. (The manner of initiating this block transfer from the terminal varies according to the data communications protocol — see section V.)
- After the terminal has successfully completed the function, it responds to the computer program with an S character followed by a CR(LF)/RS/GS. If the function was a data read operation, (^&p R) successful completion is the data.
- If the operation failed, or an error occurred in the process, the terminal responds with an F character followed by CR(LF)/RS/GS. If the function was a data read operation, an I/O failure or end-of-file is indicated by a response of RS, CR(LF), GS.
- If a device-to-device operation was interrupted by you (by pressing ) , the terminal responds with a U character followed by CR(LF)/RS/GS.

## 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

Basic Communications (Point-to-Point)		
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.
Multipoint Communications		
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.

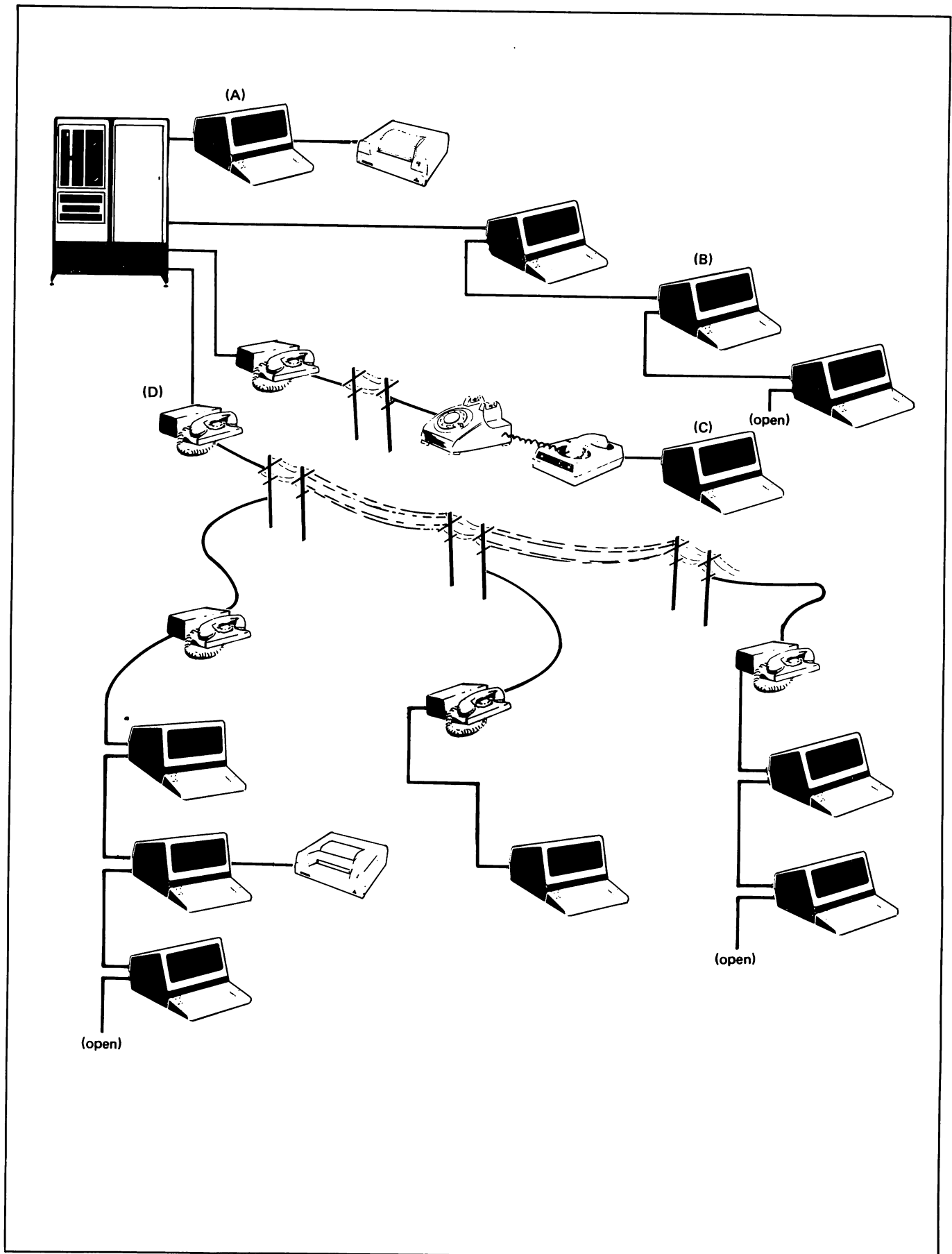


Figure 5-1. Terminal Network Configurations

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.

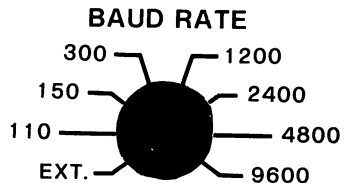
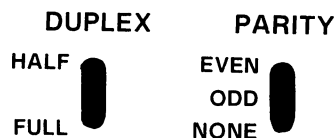
Table 5-3. Keyboard Interface (PCA) Switch Summary

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	same
V	Carrier detect	Synch Mode for Asynchronous Operation
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




### BASIC COMMUNICATIONS DATA COMM SWITCHES



**DUPLEX Switch.** 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.

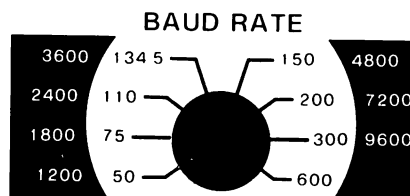
**RANGE Switch.** This switch is used to select ranges for the BAUD RATE switch (multipoint only).

**PARITY Switch.** When set to EVEN/ODD/NONE, even/odd/no parity is transmitted for each character. Incorrect parity: a "■" is displayed.

**BAUD RATE Switch.** 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.



 When down, the terminal is in Remote (on-line) operation. Otherwise, the terminal is in local (off-line) operation.

 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.



### MULTIPOINT DATA COMM SWITCHES

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.)

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.)

The indicator will be lighted when a data link exists for transmission between the terminal and the computer.

### On-Line Mode

- Character Mode, Format Off. The entire line containing the cursor is transmitted as a block.
- 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.
- 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>
- 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>

<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

Single Terminal (Character Mode Protocols)	
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.
Multiple Terminal (Block Mode Protocols)	
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. (With the 2048 byte buffer, the standard terminal can store 37 lines of 80 characters.) 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 Control 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)	800
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
* These times will increase if BASIC is active.	

## 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.

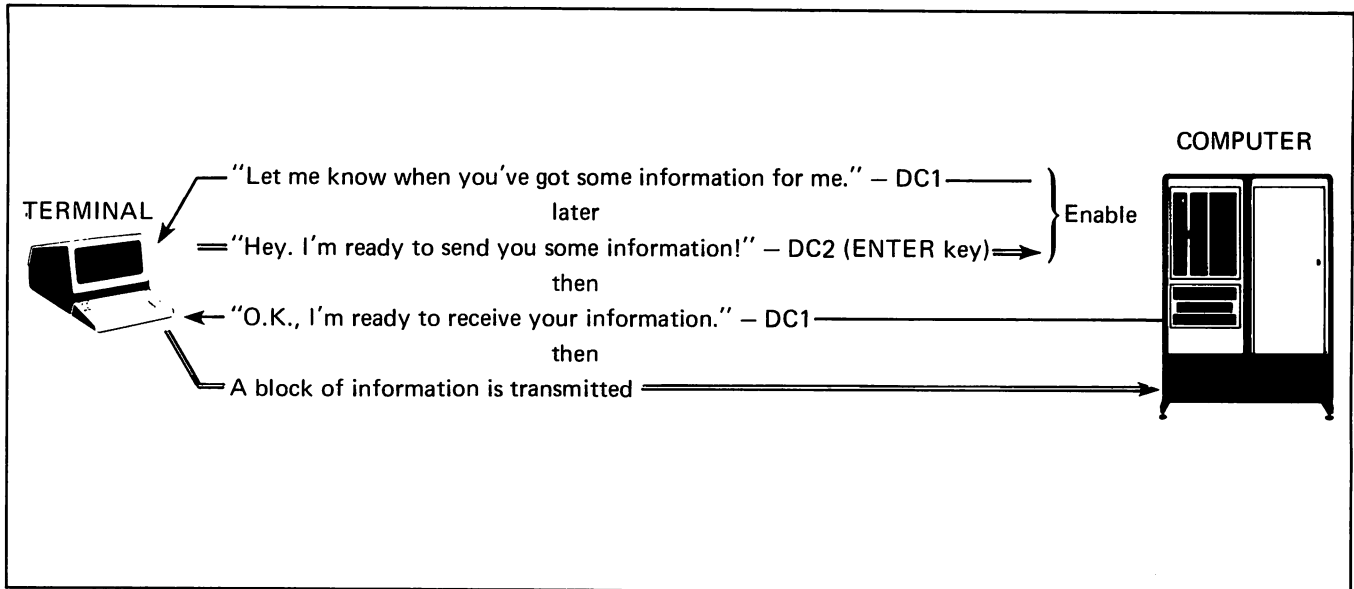


Figure 5-2. Block Transfer Enabled By The ENTER Key

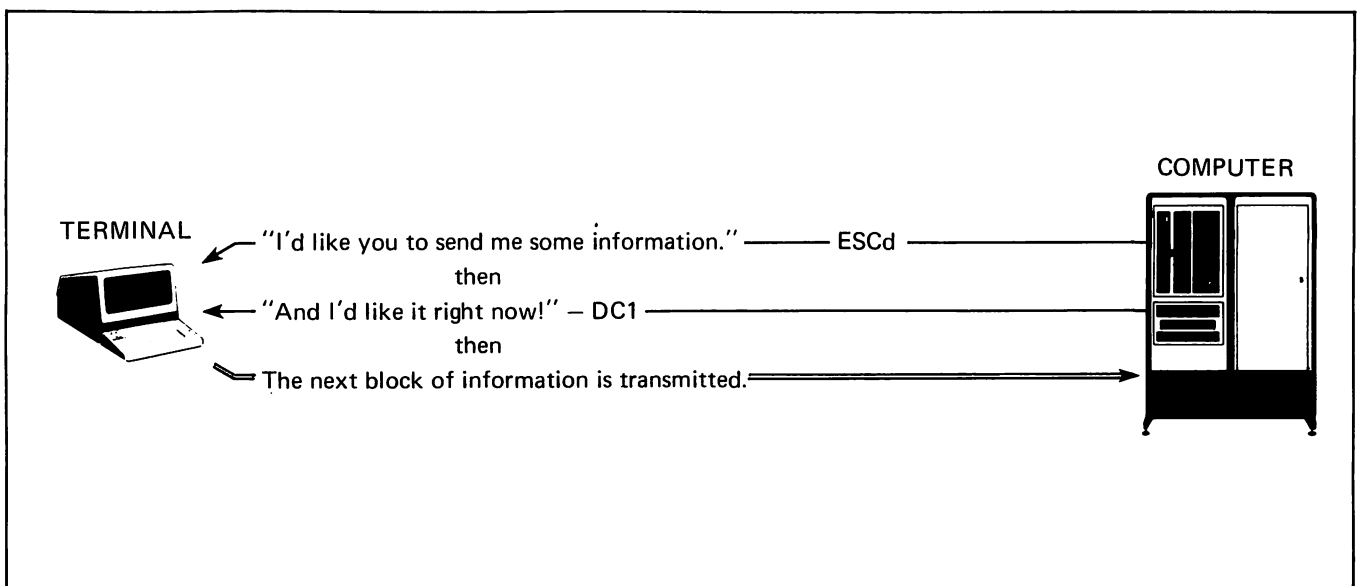


Figure 5-3. Block Transfer Enabled By The Computer

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.

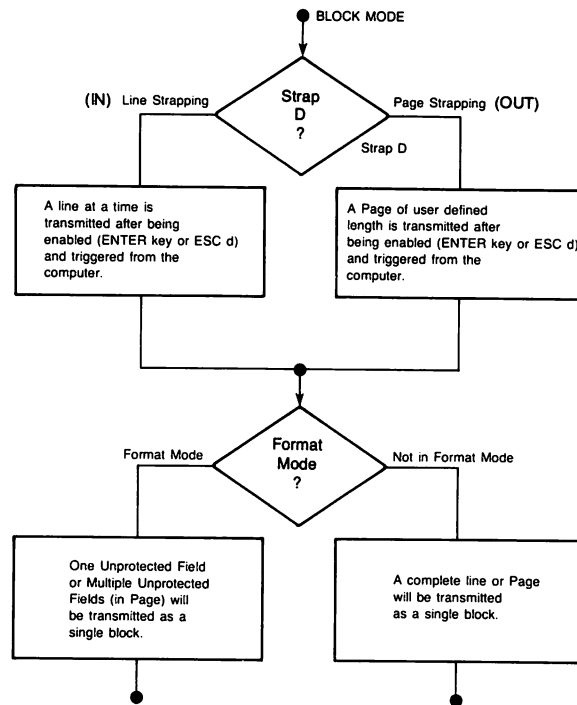
The terminal then receives the processed data until one of the following occurs:

- 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****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).)

**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.

**STRAPPED FOR PAGE**

- 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.

- 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

01-2345	HEWLETT-PACKARD	11000 WOLFE ROAD			
03/14/75	01-23-456-789012	DUPERTINO	CA	95014	
0123456789	SCREW DRIVER	\$509.99	++10	\$5099.99	AB000
789000123	SOCKET WRENCH	\$8.00	++5	\$40.00	AB000
456789012	PRECISION COMPASS	\$12.95	++10	\$129.50	FGH10

**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 **RESET** 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 VFD</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>CTZ</b> 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>CNTL</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 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).	Normal operation
		P-closed, Q-closed P-closed, Q-open P-open, Q-closed P-open, Q-open	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 (122) (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(122) (Secondary Receive Data) line. The terminal responds by turning off CA (106) (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.

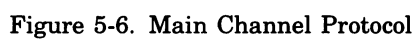
**OTHER PROTOCOLS.** In addition to the Main and Reverse Channel protocols you can select various features of both to configure a custom protocol to suit your own requirements. A flowchart of the overall Basic Communications function including the Half-Duplex settings is given in Appendix C. You can 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.



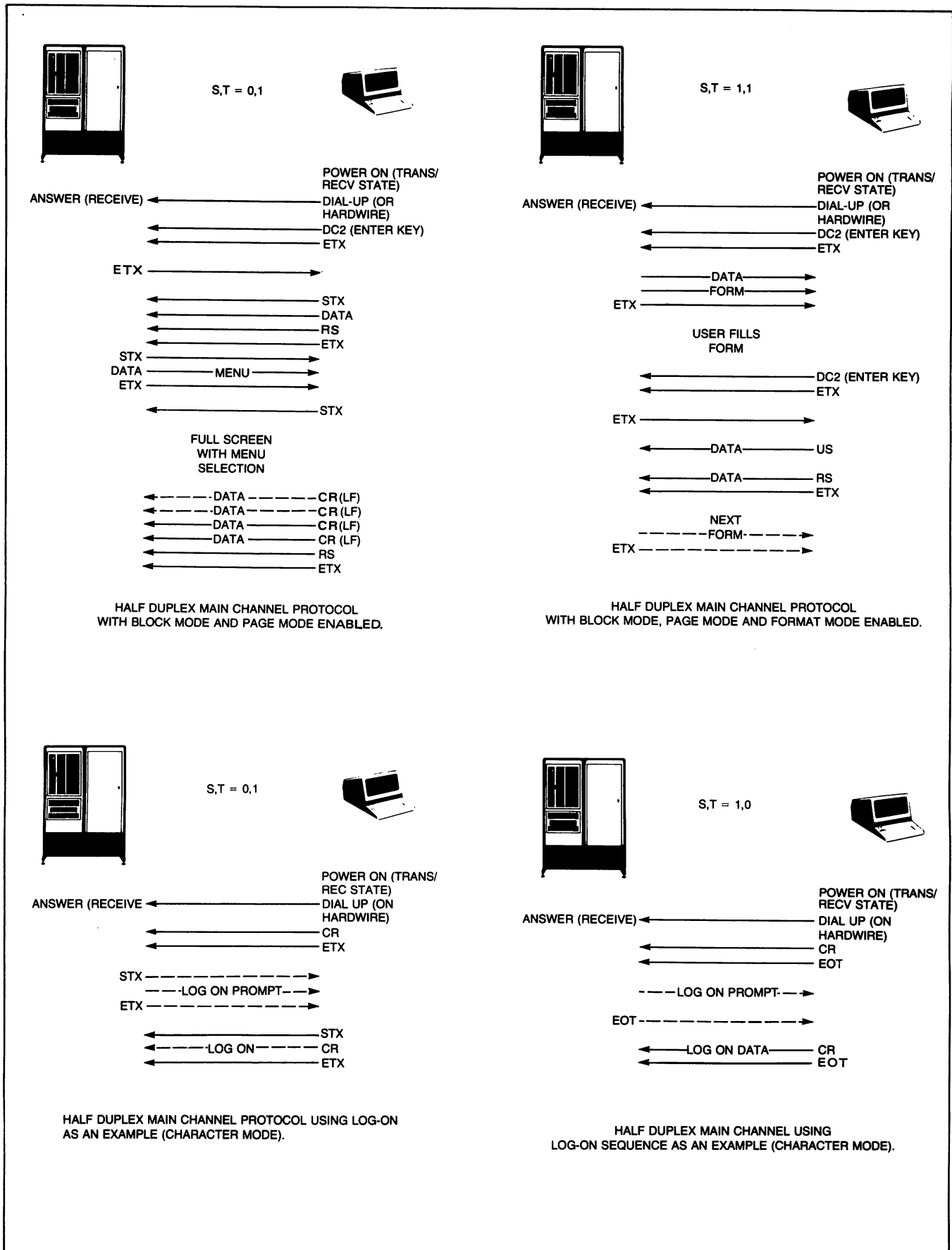


Figure 5-7. Sample Data Transfers Using Main Channel Protocol

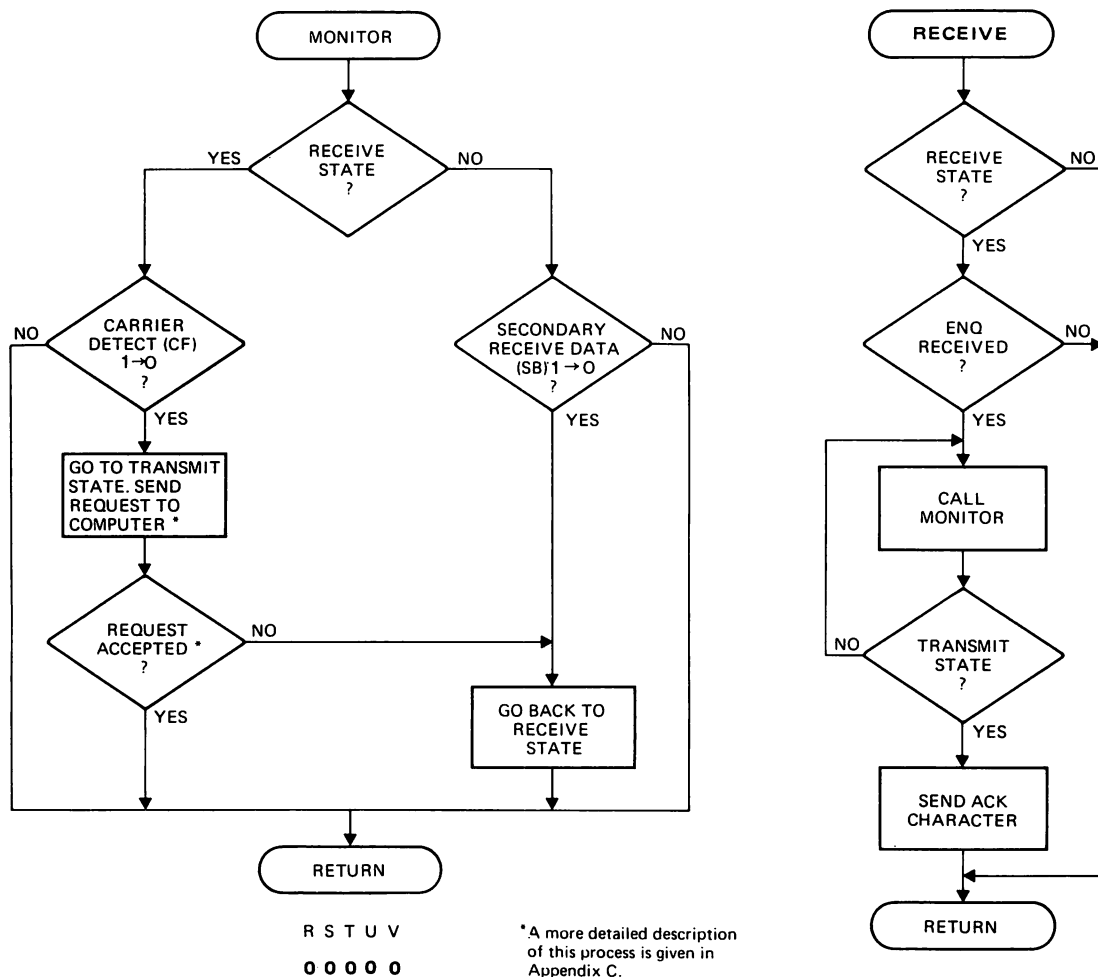


Figure 5-8. Reverse Channel Protocol

# CHARACTER MODE HALF-DUPLEX REVERSE CHANNEL



POWER ON (TRANS/RECV STATE)

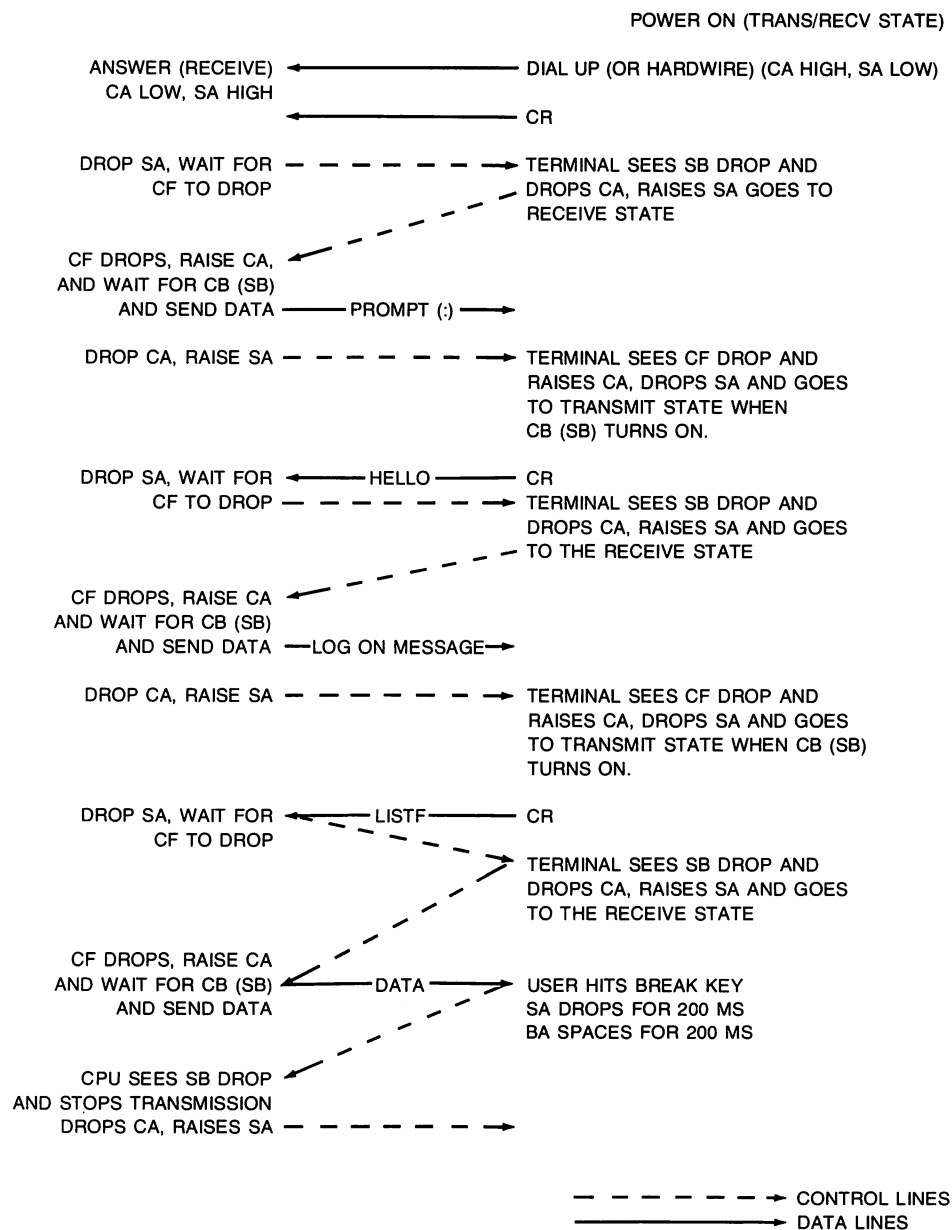


Figure 5-9. Sample Data Transfers Using Reverse Channel Protocol

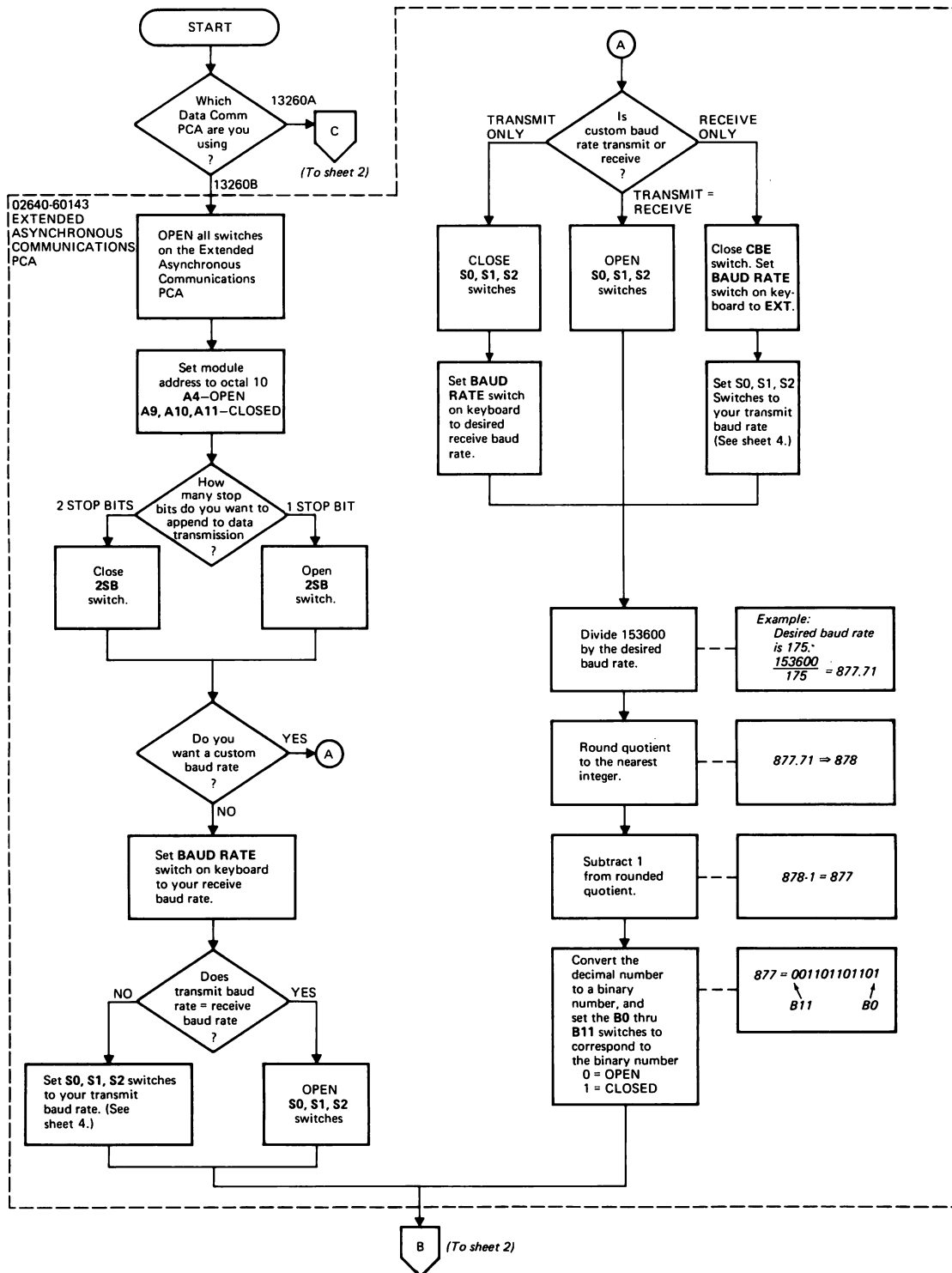


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 1 of 4)

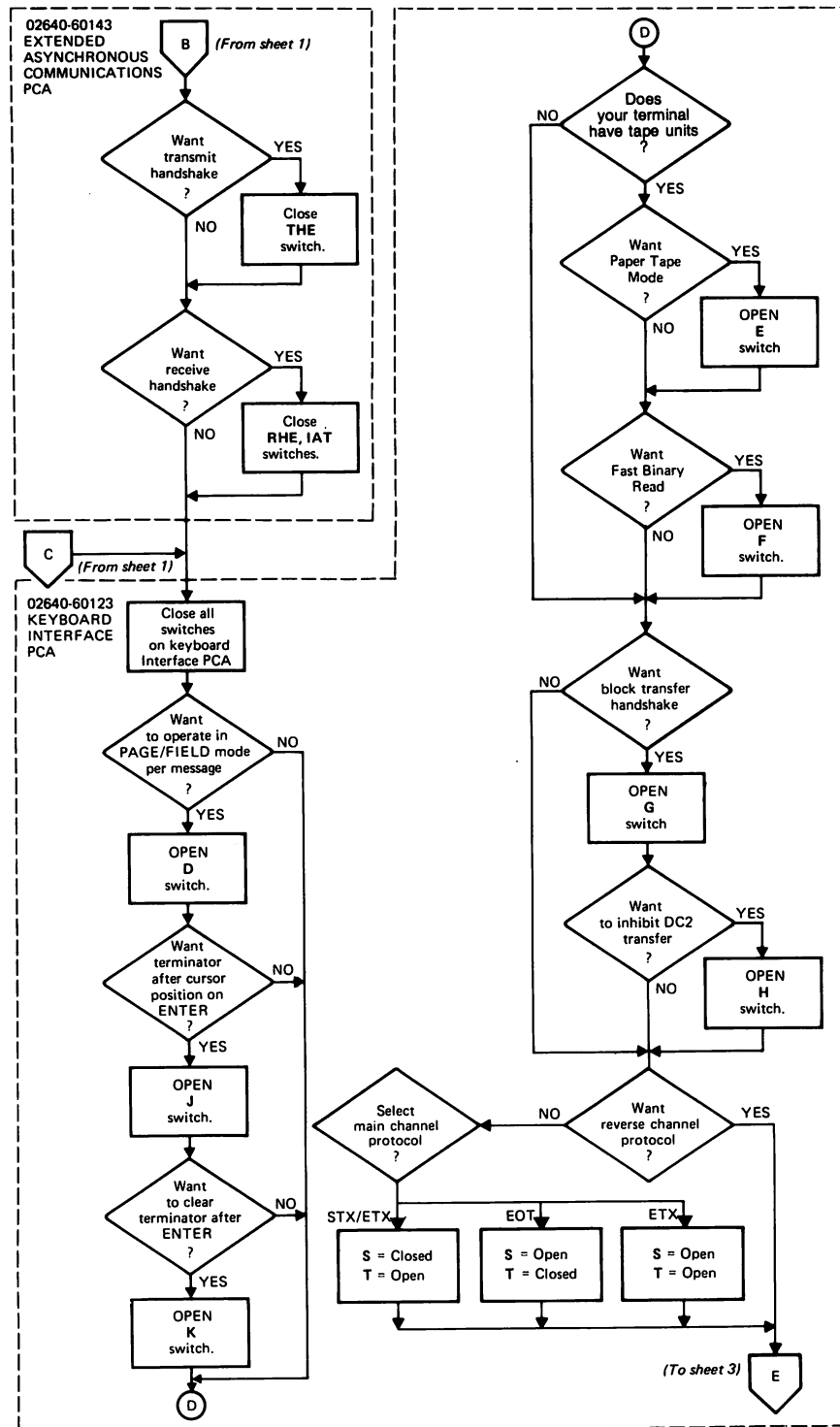


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 2 of 4)

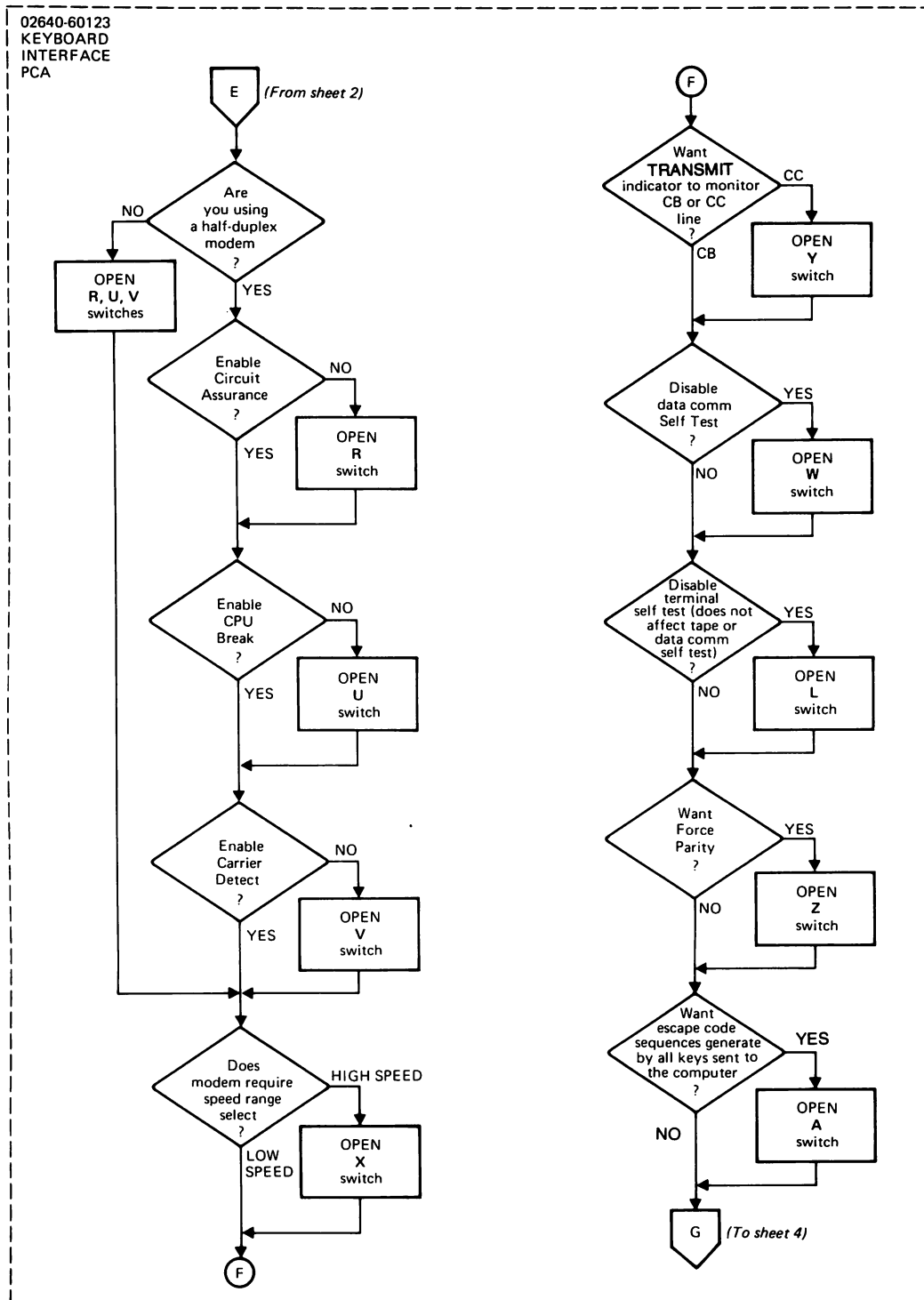


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 3 of 4)



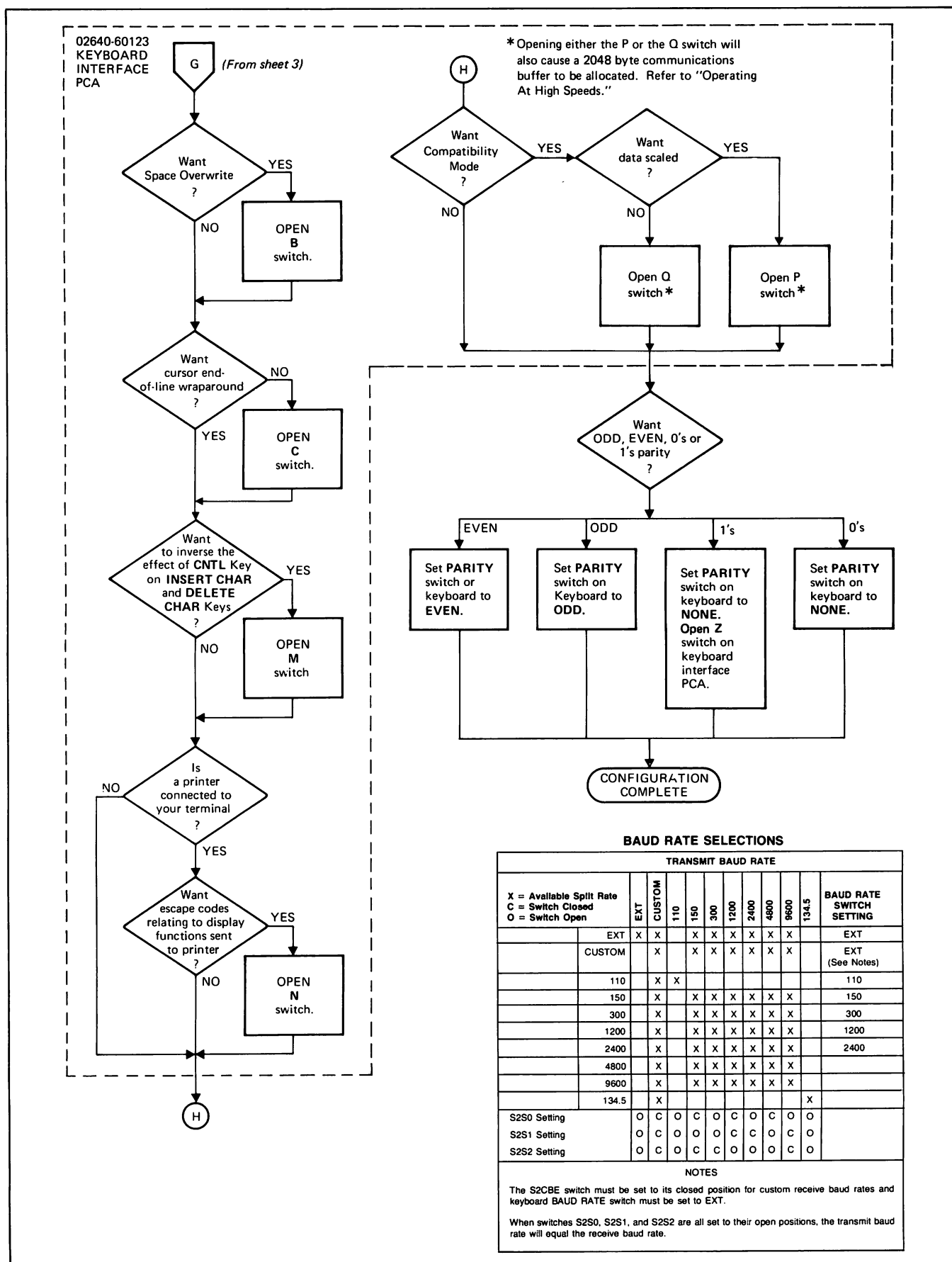


Figure 5-10. Point-to-Point Data Communications Configuration Flowchart (Sheet 4 of 4)

## 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.

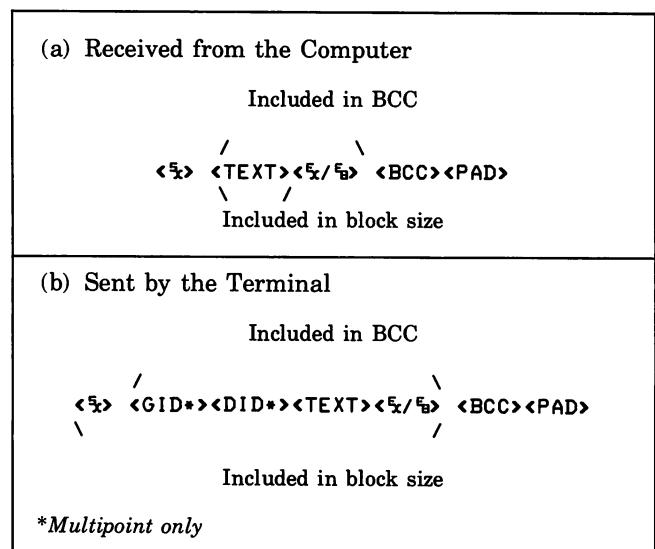


Figure 5-11. Examples of Block Transmission

**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

**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 $\backslash$ 0	020 060
ACK1 $\backslash$ 1	020 061
WACK $\backslash$ ;	020 073
RVI $\backslash$ <	020 074
EBCDIC	
graphic	octal
ACK0 $\backslash$ (no graphic equivalent)	020 160
ACK1 $\backslash$	020 141
WACK $\backslash$ ,	020 153
RVI $\backslash$ @	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:

J07	
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.

Memory is allocated from the terminal's display memory so 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

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 CorpΔΔΔΔΔΔΔ 110ΔN.Δ SeaΔ RoadΔΔΔΔ  
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.

**S**

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:

These characters are not included in the block check

/	\	\	\
␣ ♂ <data>	␣ ␣ <data>	␣ ♀ <data>	␣ ♂ BCC PAD
\			/
Begin block check		End block check	

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.

Figure 5-12 illustrates the operation of the various control characters used in the block protocol.

## 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.

**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>BREAK</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.

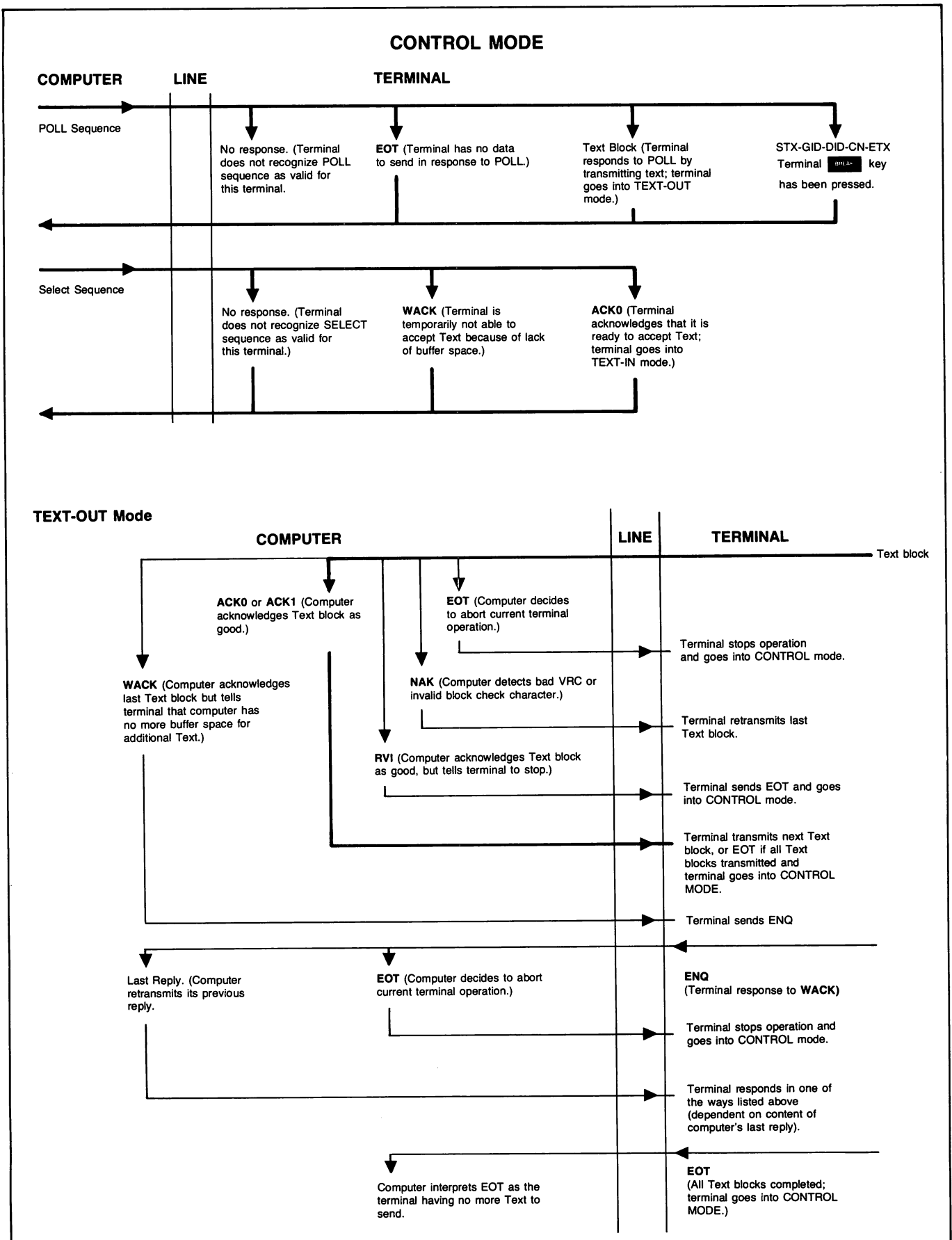


Figure 5-12. Operation of Block Protocol Control Characters (Sheet 1 of 2)



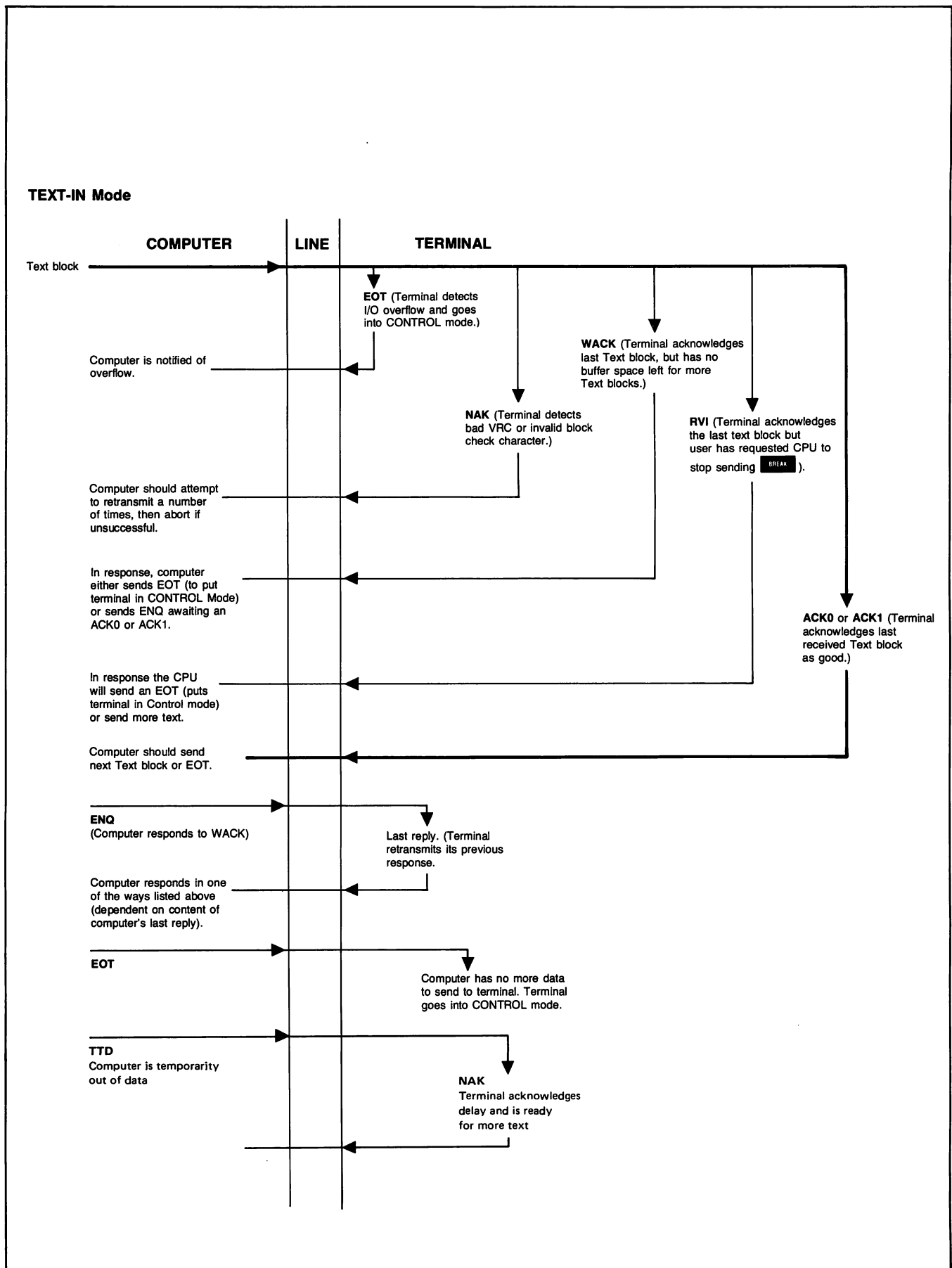


Figure 5-12. Operation of Block Protocol Control Characters (Sheet 2 of 2)

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 (BREAK).	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	(BREAK) 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	<p>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> <p>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</p>	
R	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	Space Compression	Space characters are sent normally.	Space characters are compressed.
T,U	Output Block Size	T            U	BLOCK SIZE (BYTES)
		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	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	Transparency	No effect.	Causes all data sent from the terminal to be transparent.

**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.

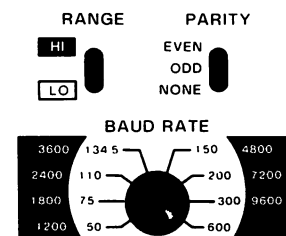


Figure 5-13. Block Keyboard Data Communication Switches

## 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:

**5 & 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:** **5 & 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:** **5 & 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:** **5 & 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** = **5 & g 110F** (H = high priority, text follows)  
**F2** = **5 & g 114A** (L = one line of text ready)  
**F3** = **5 & g 116A** (N = no data ready)  
**F4** = **5 & 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	056

////// 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) =  $2^2 = 4$

J11 (open) =  $2^1 = 2$

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).

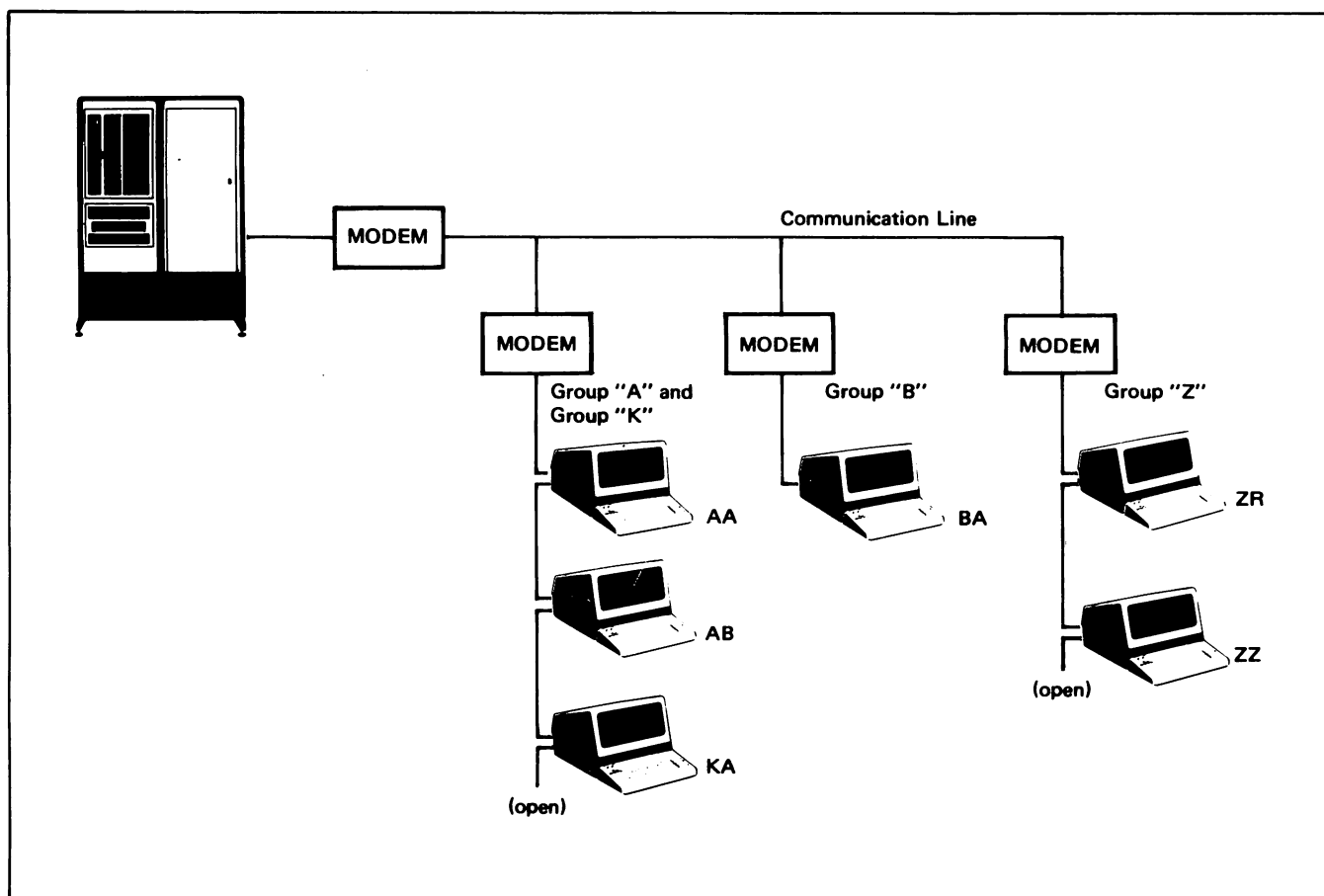


Figure 5-14.. Terminal Addressing

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)	= $2^4 = 16$
J03 (closed)	
J02 (open)	= $2^2 = 4$
J01 (closed)	
J00 (closed)	
	<hr/> 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

$\langle \text{E} \rangle \langle \text{PAD} \rangle \dagger \langle \text{GROUP ID} \rangle \langle \text{GROUP ID} \rangle \langle \text{DEV ID} \rangle \langle \text{DEV ID} \rangle \langle \text{E} \rangle \langle \text{PAD} \rangle$

#### Synchronous

$\langle \text{E} \rangle \langle \text{E} \rangle \langle \text{PAD} \rangle \dagger \langle \text{E} \rangle \langle \text{GROUP ID} \rangle \langle \text{GROUP ID} \rangle \langle \text{DEV ID} \rangle \langle \text{DEV ID} \rangle \langle \text{E} \rangle \langle \text{PAD} \rangle$

\*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:

### ASYNCHRONOUS

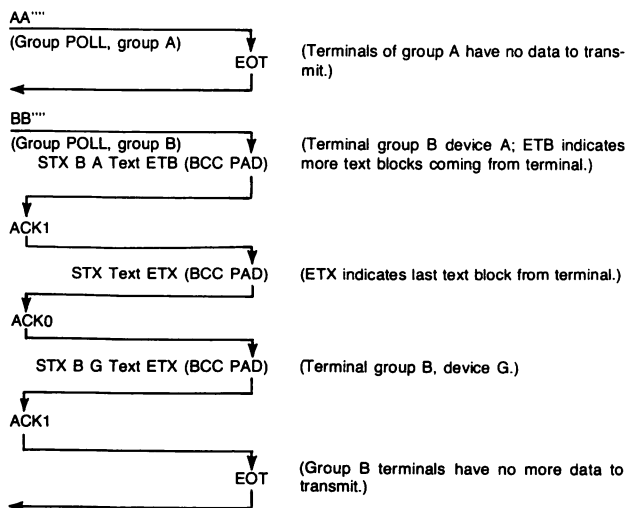
Group Poll Characters

␣ PAD A A " " ␣ PAD

### SYNCHRONOUS

␣␣␣␣ ␣ PAD\* ␣␣␣␣ A A " " ␣ PAD

\* Optional PAD character



**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:

### ASYNCHRONOUS

␣ PAD a a D D ␣ PAD

### SYNCHRONOUS

␣␣␣␣ ␣ PAD ␣␣␣␣ a a D D ␣ PAD

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:

### ASYNCHRONOUS

Lower case for select      Group select characters

␣ PAD c c ~ ~ ␣ TEXT ␣ BCC PAD

### SYNCHRONOUS

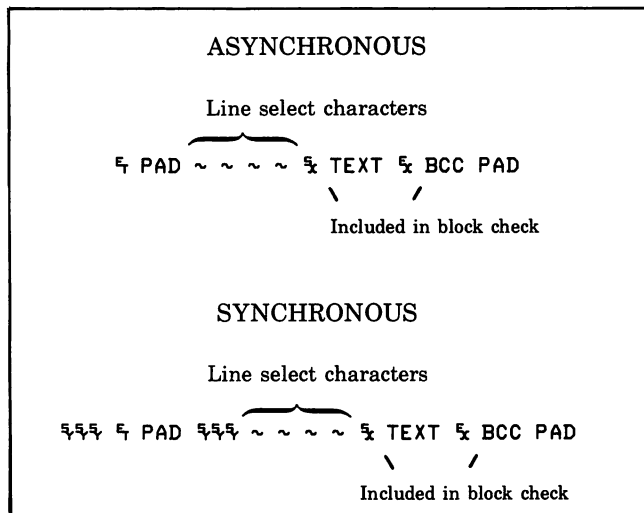
Lower case for select      Group select characters

␣␣␣␣ ␣ PAD ␣␣␣␣ c c ~ ~ ␣ TEXT ␣ BCC PAD

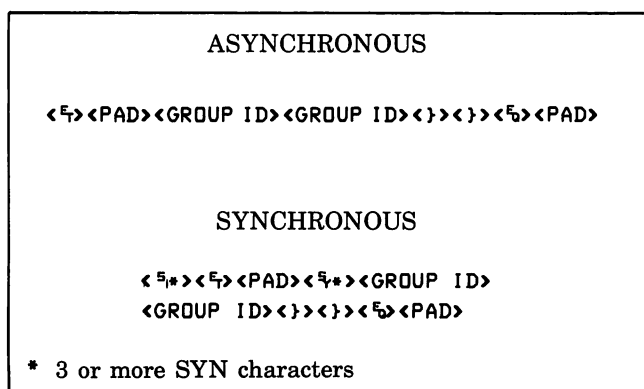
Included in block check



**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.

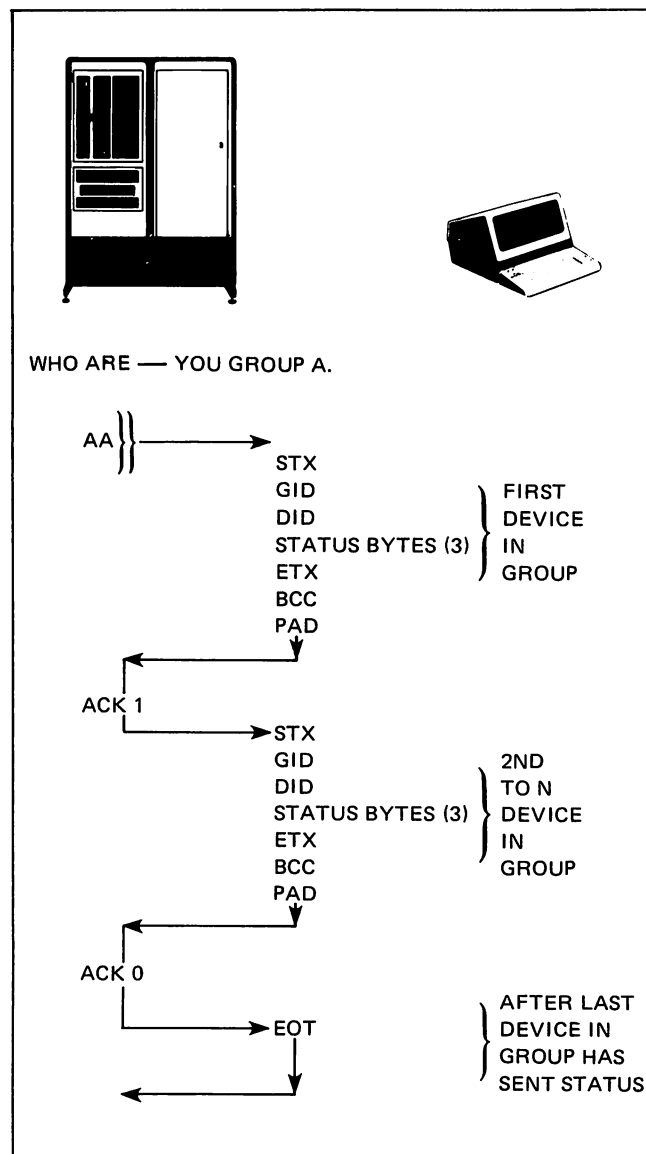
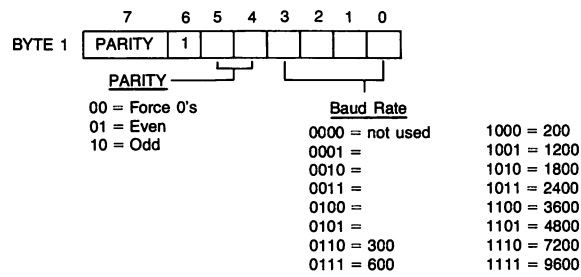


Figure 5-15. Typical Configuration Status Request and Response Sequence

## Configuration Procedure

After you have determined the required multipoint settings for your application follow the flowchart given in figure 5-17.



⌘ GID DID i@P ⌘ BCC PAD

Example: Status bytes "i@P" would be interpreted as follows:

"i" = 1101001 = odd parity, 1200 baud  
 "@" = 1000000 = ASCII, LRC, 500 byte buffers  
 "P" = 1010000 = no data, terminal in remote

#### MULTIPOINT STATUS BYTES

@	100	0000	`	110	0000
A	100	0001	a	110	0001
B	100	0010	b	110	0010
C	100	0011	c	110	0011
D	100	0100	d	110	0100
E	100	0101	e	110	0101
F	100	0110	f	110	0110
G	100	0111	g	110	0111
H	100	1000	h	110	1000
I	100	1001	i	110	1001
J	100	1010	j	110	1010
K	100	1011	k	110	1011
L	100	110	l	110	1100
M	100	1101	m	110	1101
N	100	1110	n	110	1110
O	100	1111	o	110	1111
P	101	0000	p	111	0000
Q	101	0001	q	111	0001
R	101	0010	r	111	0010
S	101	0011	s	111	0011
T	101	0100	t	111	0100
U	101	0101	u	111	0101
V	101	0110	v	111	0110
W	101	0111	w	111	0111
X	101	1000	x	111	1000
Y	101	1001	y	111	1001
Z	101	1010	z	111	1010
[	101	1011	{	111	1011
\	101	1100		111	1100
]	101	1101	}	111	1101
^	101	1110	~	111	1110
_	101	1111	■	111	1111

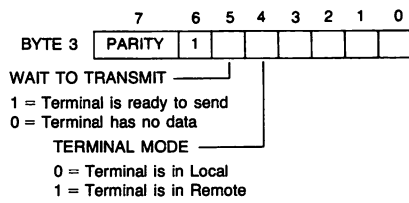
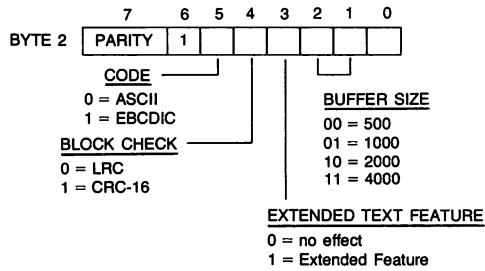


Figure 5-16. Configuration Status Byte Contents

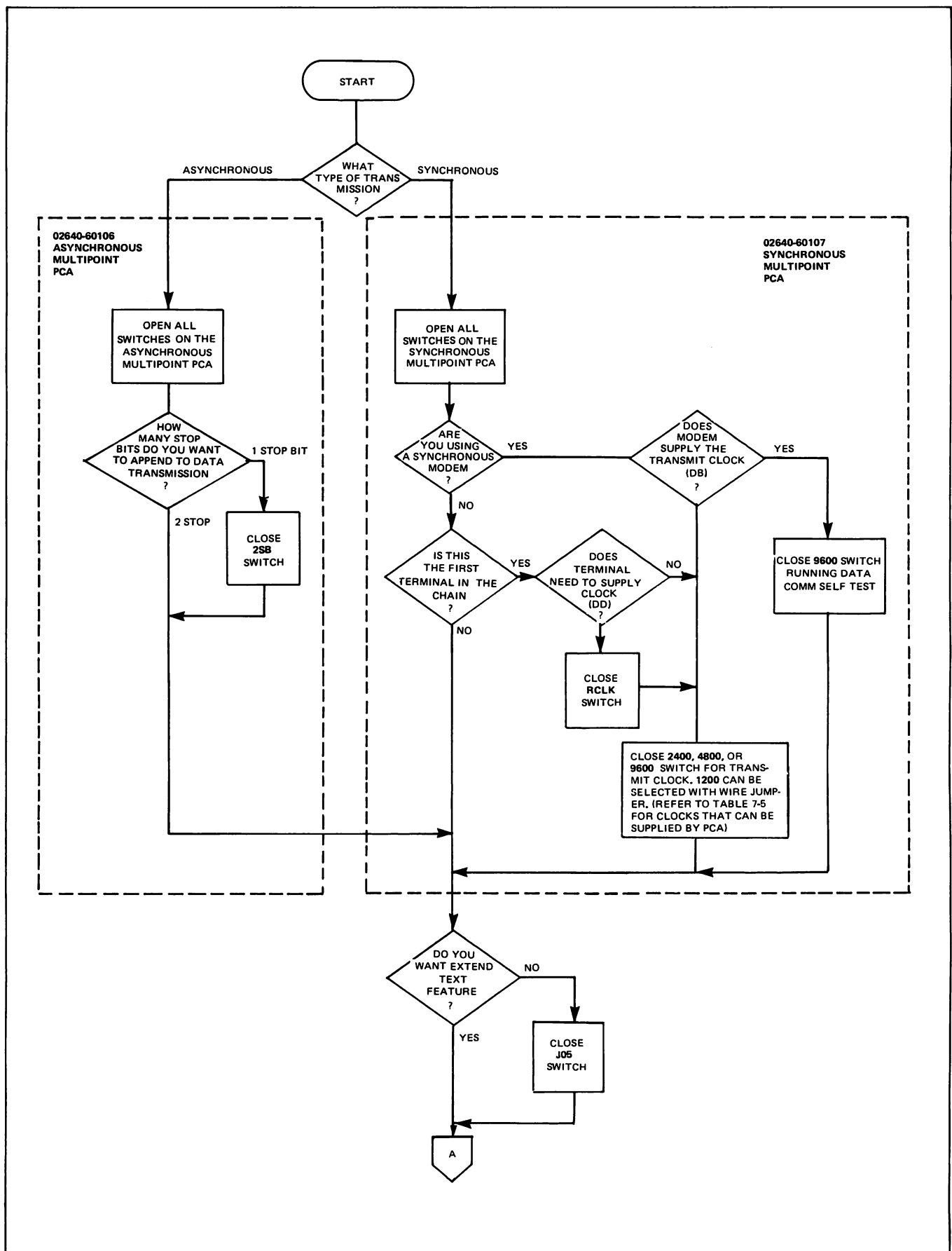


Figure 5-17. Multipoint Data Communications Configuration (Sheet 1 of 4)

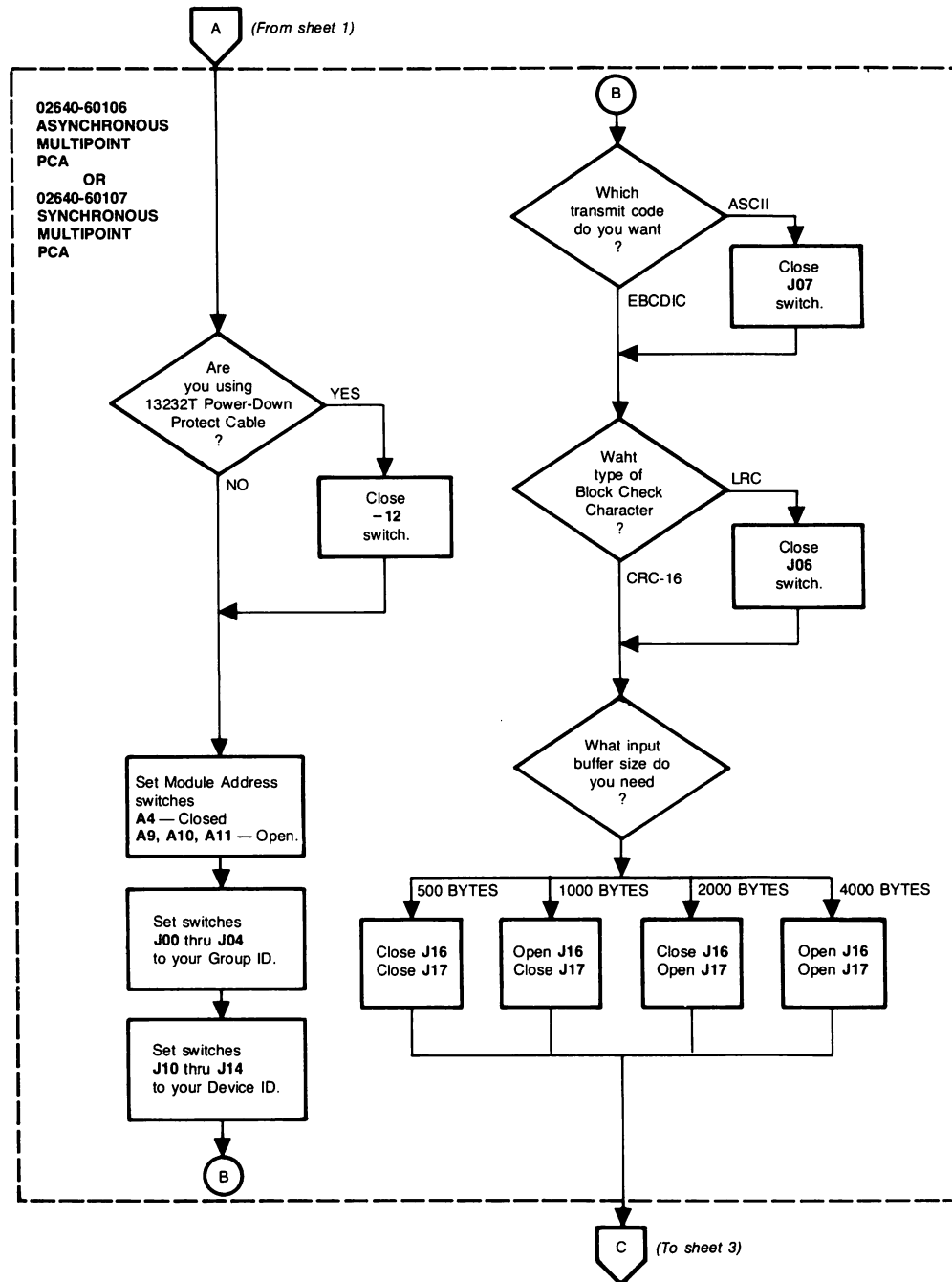


Figure 5-17. Multipoint Data Communications Configuration (Sheet 2 of 4)

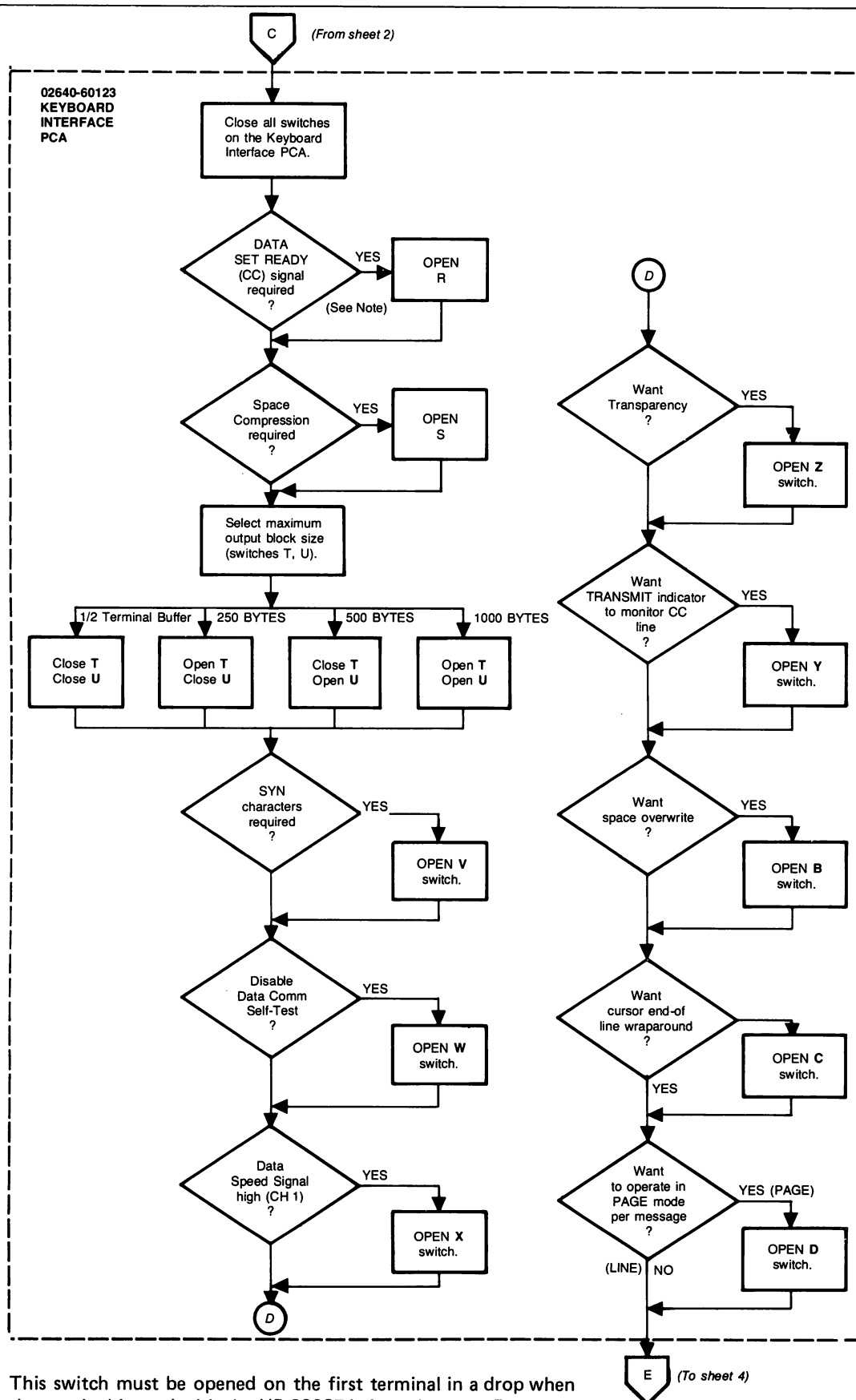


Figure 5-17. Multipoint Data Communications Configuration (Sheet 3 of 4)

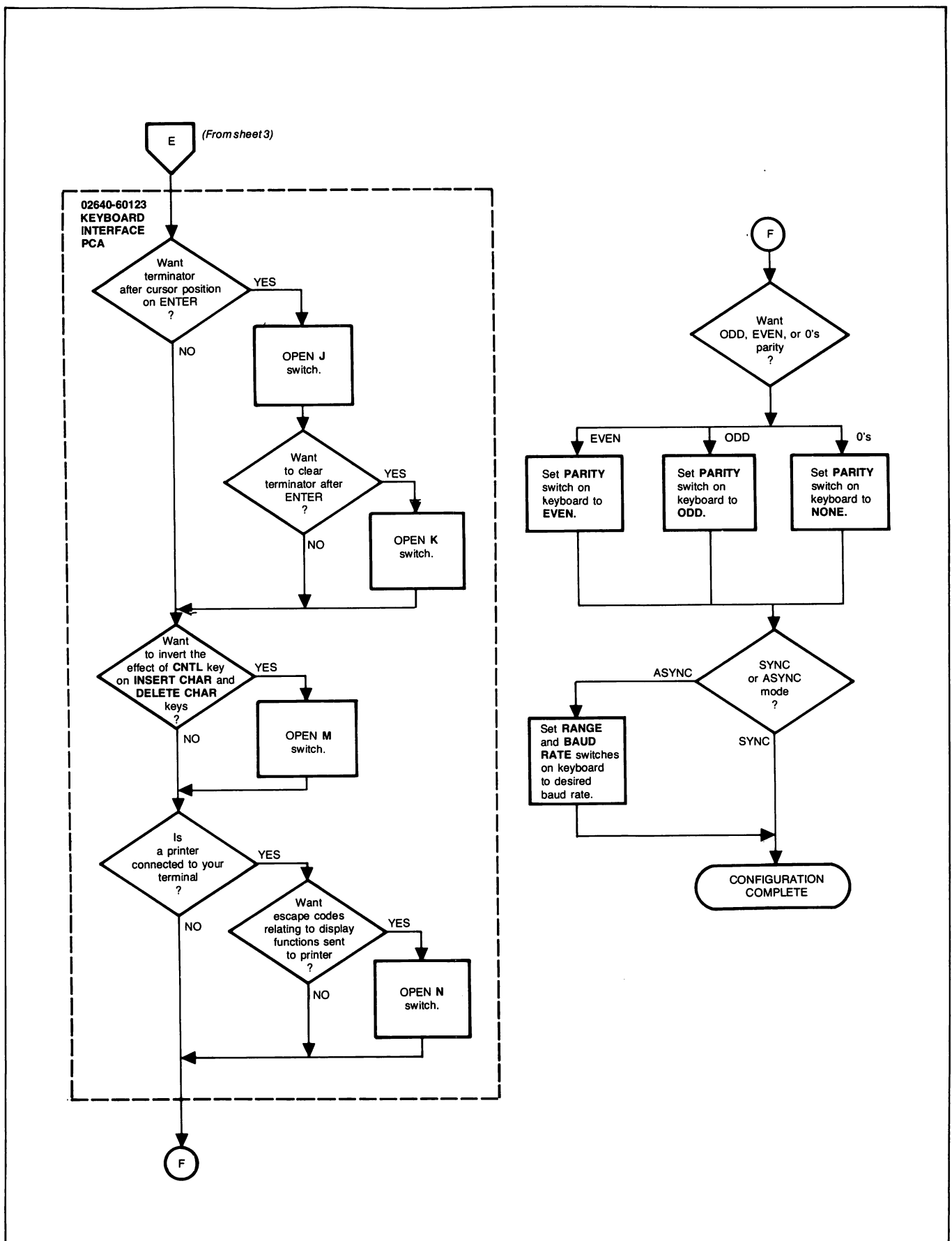


Figure 5-17. Multipoint Data Communications Configuration (Sheet 4 of 4)

## 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** ( **REMOTE** key down).

**Step 2.** **CTRL** **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.)

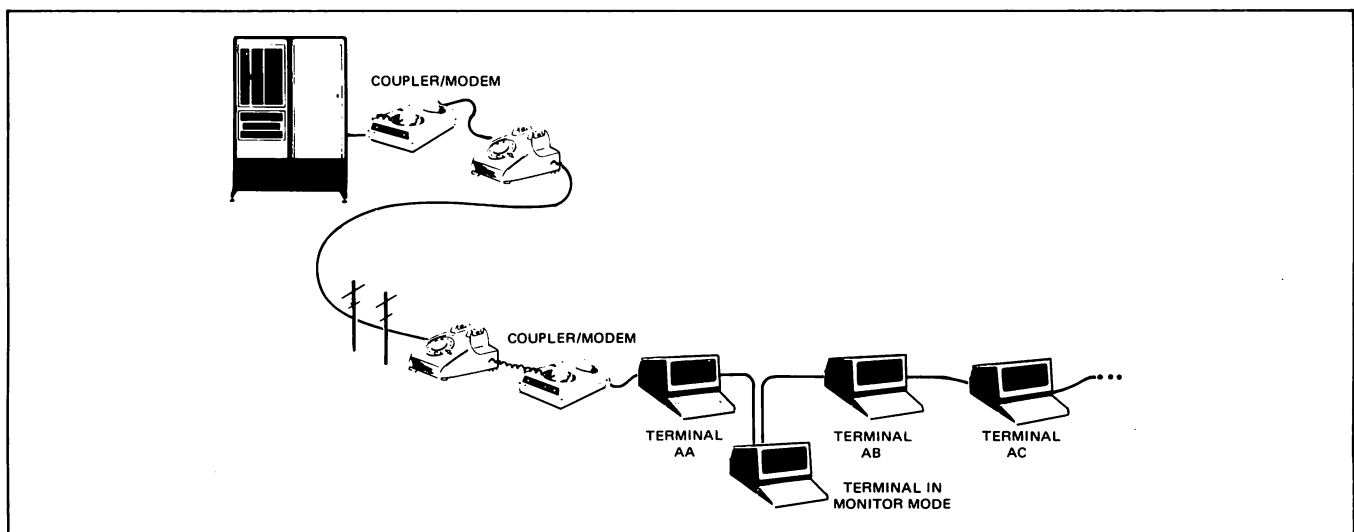
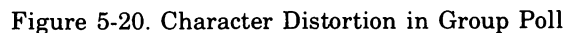


Figure 5-18. Communication Line Using a Monitor



**Figure 5-21. Data Overflow Indication**



## 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.**  ,  ,  , 

**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.** 

**Step 4.**   (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 test 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.

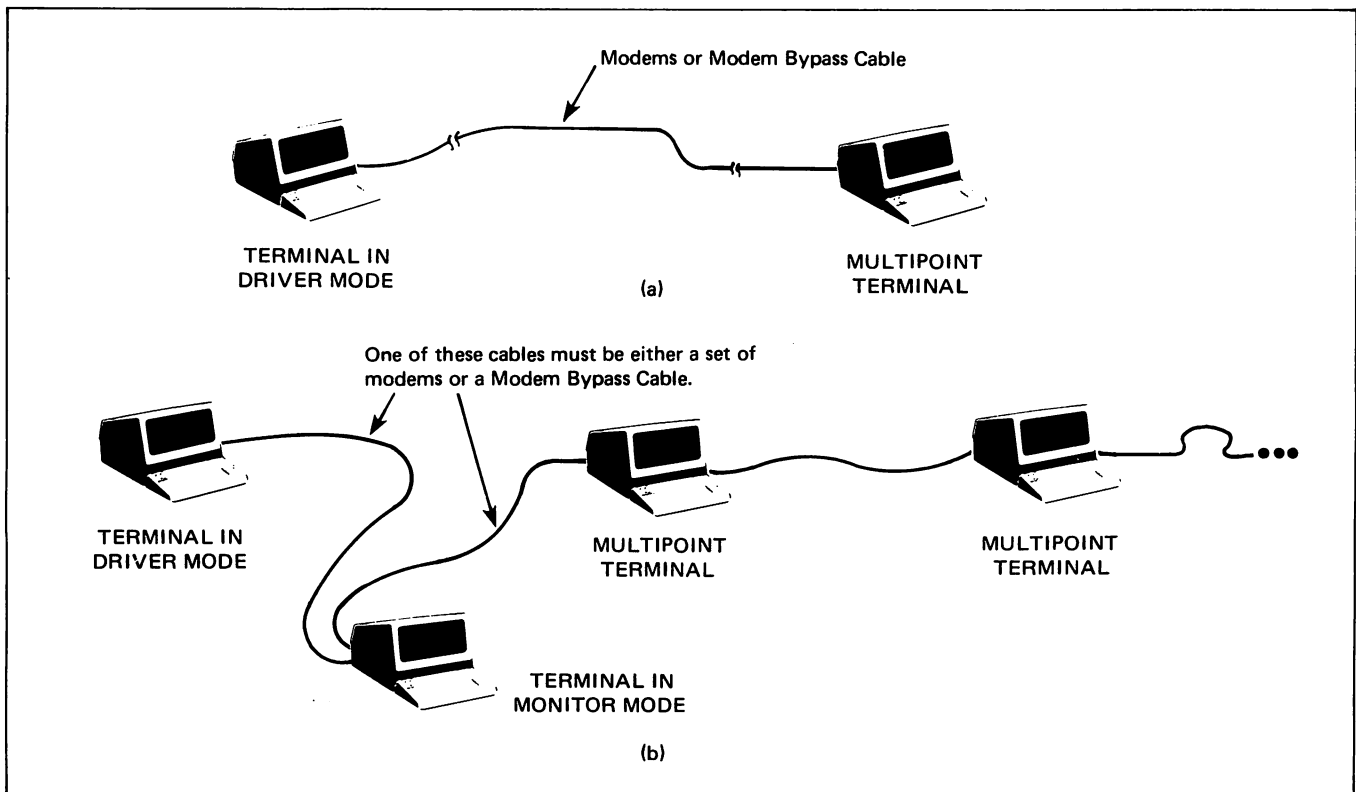


Figure 5-22. Driver Mode Configurations

```

Driver:  >40ttFF%<
Terminal: 00
Driver:  >5This is a message to be sent to a terminal%55X%<
Terminal: 01
Driver:  >4040TTFF%<
Terminal: 40

```

## a.) Conversation

```

>40TTFF%<40>40TTFF%<040>40ttFF%<00>5This is a message
to be sent to a terminal%
55X%<01>4040TTFF%<040>... 40TTFF%<40>40TTFF%
0<40

```

## b.) Display

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).

A full reset returns the driver terminal to normal operation.

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).

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.

```

TFThe complete text is normally sent with a short summary.%
4This allows the editor to scan the material inorder to%
4assign it to a field of interest.%
4%

```

## a) Display Functions On

```

TFThe complete text is normally sent with a short summary.
This allows the editor to scan the material inorder to
assign it to a field of interest.
%

```

## b) Display Functions Off

Figure 5-24. Control Character Display On Driver Terminal

```

Driver:  >40TTFF%<
Terminal: 40
Driver:  >40TTFF%<
Terminal: 05TFThis is a block of text to be sent to a computer%55Xz%
Driver:  >01%<
Terminal: 40

```

## a.) Conversation

```

>40TTFF%<40>40TTFF%<5TFThis is a block of text to be sent to a computer%
55Xz%>01%<40>40TTFF%<40

```

## b.) Display

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 **SELF TEST** 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.

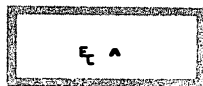
Table 6-1. ASCII Status Characters

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.

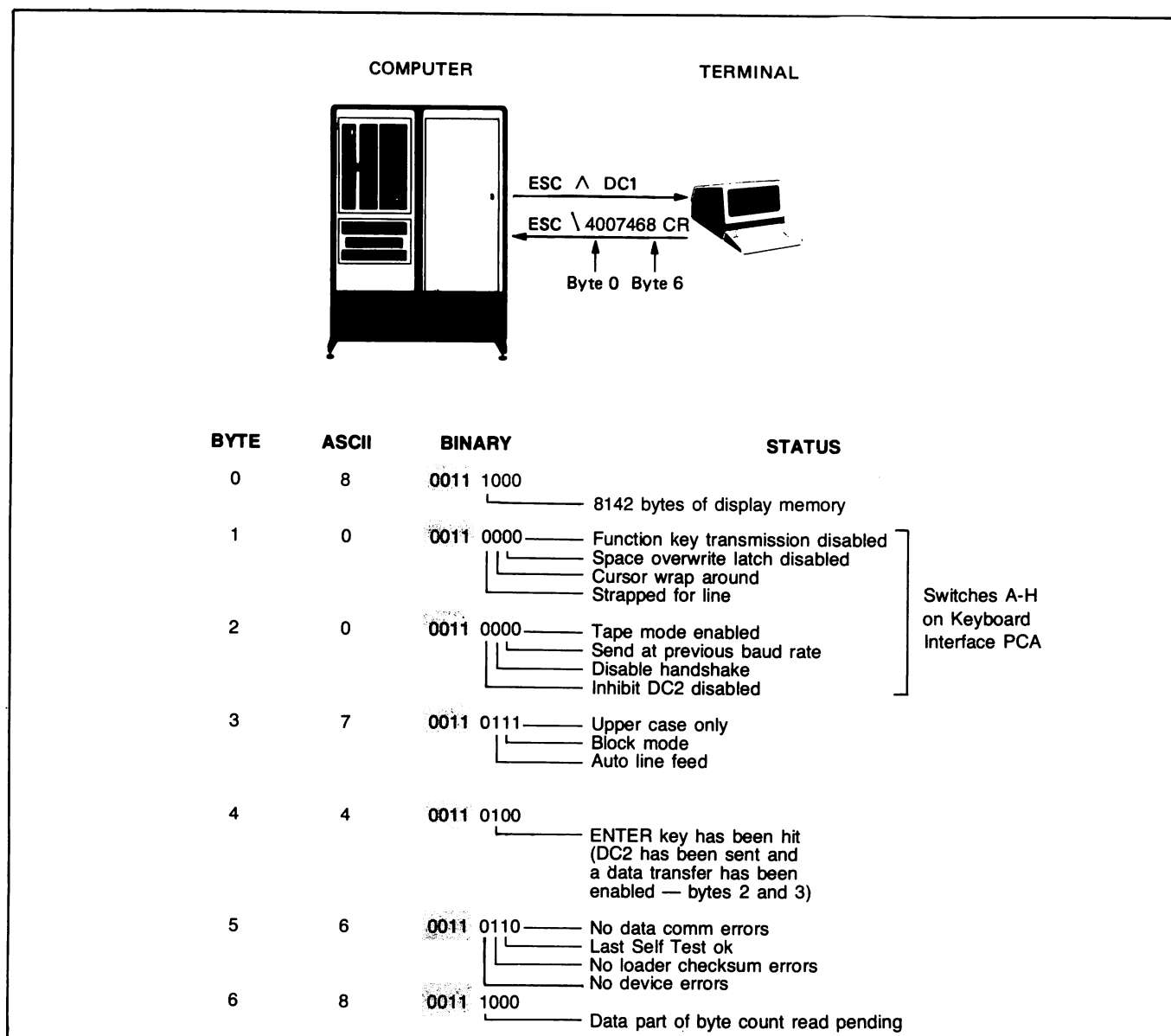
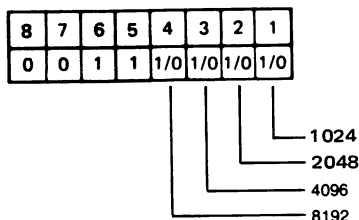


Figure 6-1. Primary Terminal Status Example

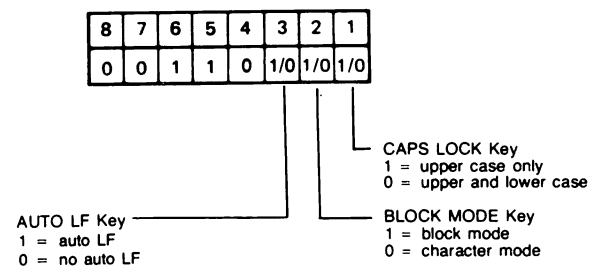
## PRIMARY TERMINAL STATUS

## BYTE 0 DISPLAY MEMORY SIZE

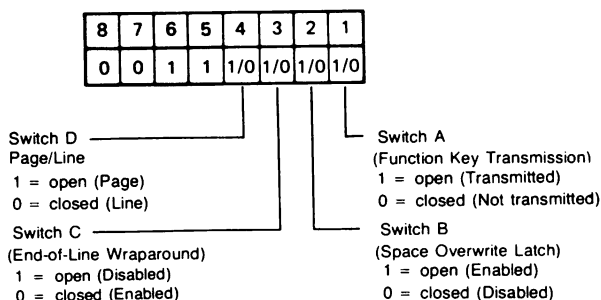


The amount of display memory (blocks of 1K) available in the terminal is returned. The amount can range from 1K to 9K bytes. The actual number of displayable characters is less than the returned figure by at least 12% minus another 500 bytes for system use.

## BYTE 3 LATCHING KEYS

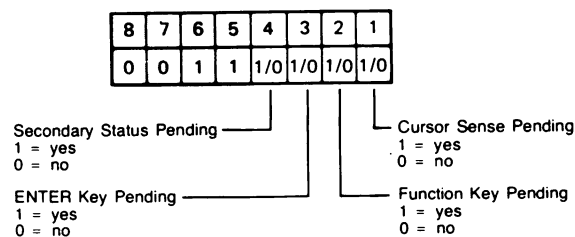


## BYTE 1 KEYBOARD INTERFACE SWITCHES (A-D)

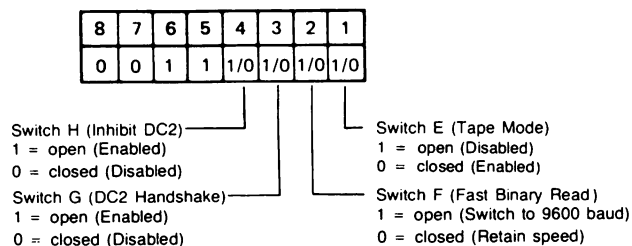


Refer to Section V for a detailed description of Keyboard Interface switches.

## BYTE 4 TRANSFER PENDING FLAGS

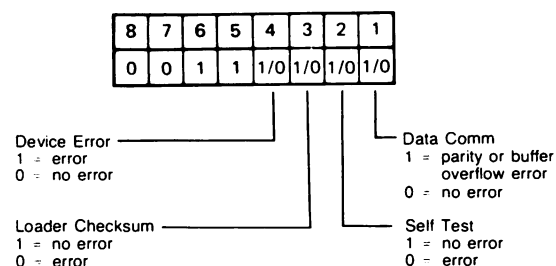


## BYTE 2 KEYBOARD INTERFACE SWITCHES (E-H)

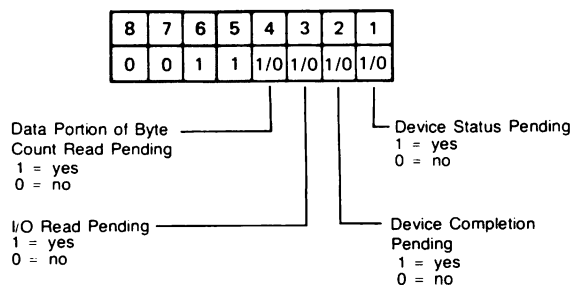


Refer to Section V for a detailed description of Keyboard Interface switches.

## 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 I and 7 status bytes followed by a terminator. A typical secondary terminal status request and response are shown in figure 6-2.

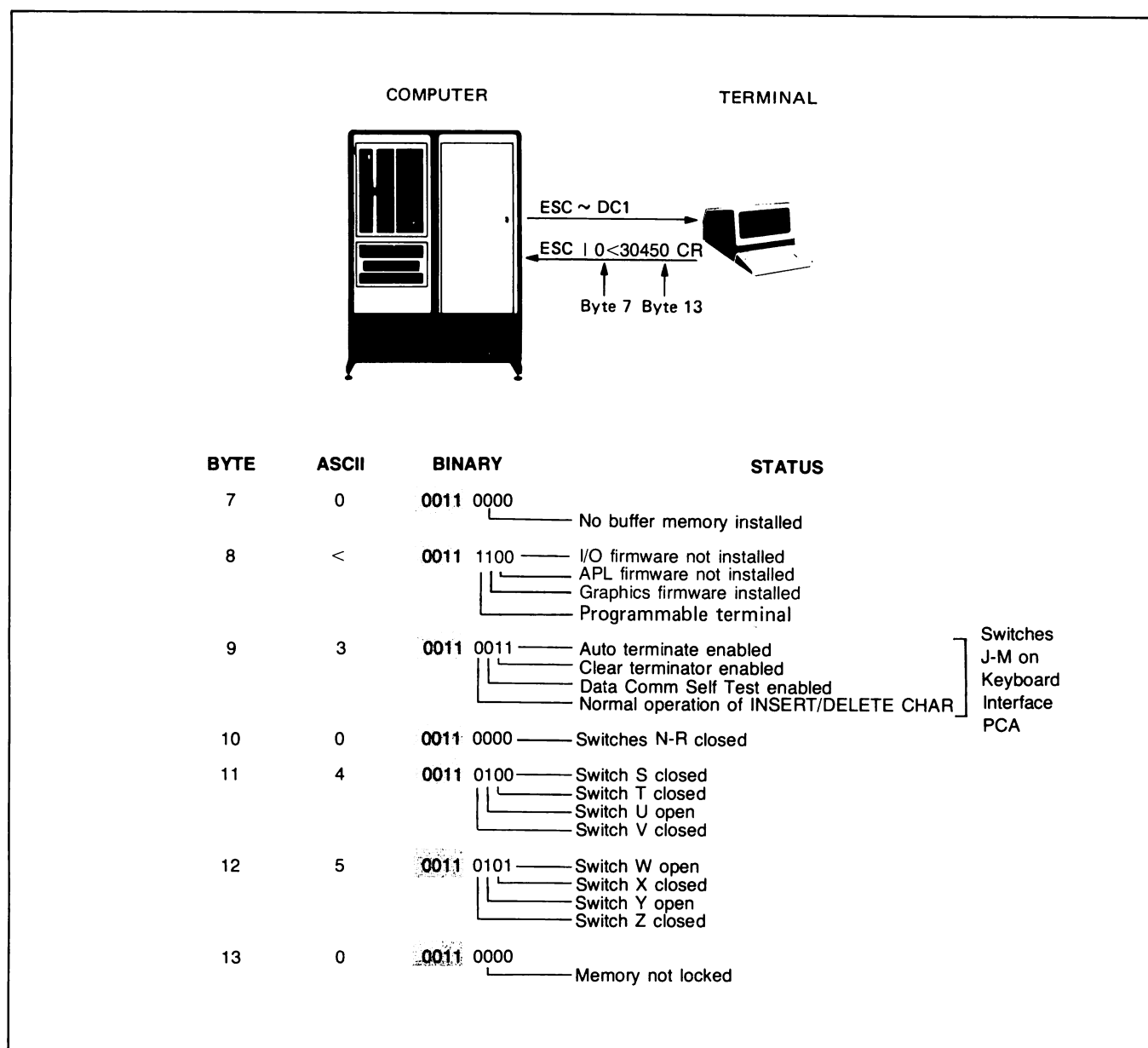


Figure 6-2. Secondary Terminal Status Example

## 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	1	1	0	1/0

1 = Programmable terminal  
0 = Not Programmable terminal  
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	1/0	1/0

1 = APL Mode  
0 = ASCII Mode  
1 = locked in row # 0  
0 = locked in row 0  
1 = memory lock on  
0 = memory lock off  
1 = memory full  
0 = memory not full



## 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 = external printer

Note: The printer designated by device code 4 is an 8-bit duplex or serial printer (NOT an HP-IB printer connected by way of an HP 13296A Shared Peripheral Interface).

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.

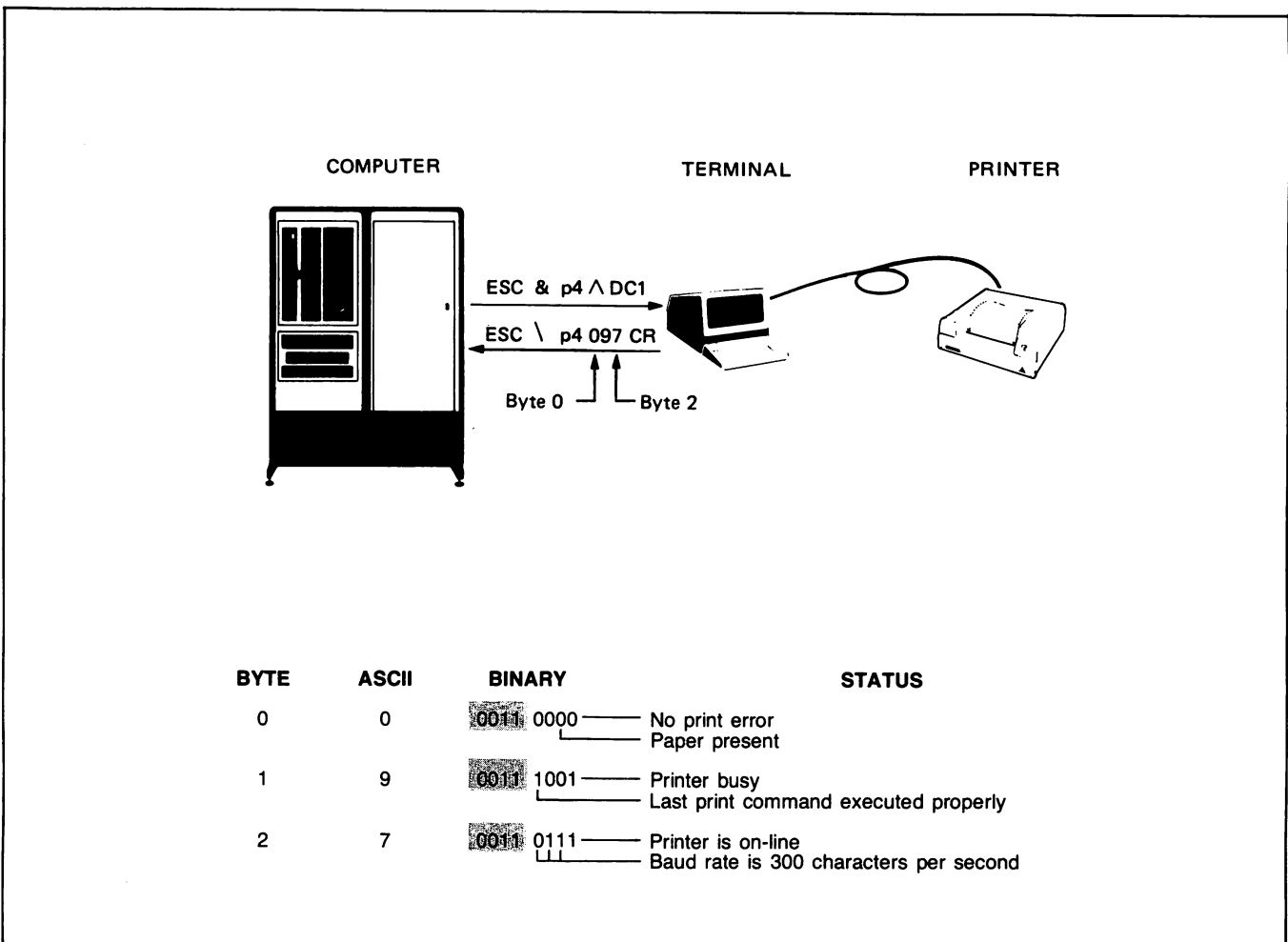


Figure 6-3. Device Status Example

## TAPE UNITS

### BYTE 0

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

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

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

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

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

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

8	7	6	5	4	3	2	1
0	0	1	1	0	0	1/0	1/0

Paper Out (varies with printer)  
1 = yes  
0 = no

Print Error (varies with printer)  
1 = yes  
0 = no

### BYTE 1

8	7	6	5	4	3	2	1
0	0	1	1	1/0	0	0	1/0

Command Execution  
1 = last command performed  
0 = last command aborted

Printer Busy  
1 = yes  
0 = no

### BYTE 2

8	7	6	5	4	3	2	1
0	0	1	1	1/0	1/0	1/0	1/0

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

## 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 13) 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 (`␣`, `␣-f`, `␣`, 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:

### Read Device I.D. (Parameter=1)

When you request a device I.D. the terminal responds with its Hewlett-Packard model number, 2647A.

Device I.D.

Request:

`␣ * s 1 ^`

The terminal responds: 2647A `<terminator>`

### Read Current Pen Position (Parameter=2)

The pen position and status are returned as a string of ASCII characters.

Pen Position

Request:

`␣ * s 2 ^`

The terminal responds: `<X>`, `<Y>`, `<Pen>`, `<terminator>`

Table 6-2. Graphics Status Requests

Parameter	Request	Response
1	Read device I.D.	2647A
2	Read current pen position	<code>&lt;X&gt;</code> , <code>&lt;Y&gt;</code> , <code>&lt;PEN&gt;</code>
3	Read graphics cursor position	<code>&lt;X&gt;</code> , <code>&lt;Y&gt;</code>
4	Read graphics cursor position with wait	<code>&lt;X&gt;</code> , <code>&lt;Y&gt;</code> , <code>&lt;KEY&gt;</code>
5	Read display size	<code>&lt;LLX&gt;</code> , <code>&lt;LLY&gt;</code> , <code>&lt;URX&gt;</code> , <code>&lt;URY&gt;</code> , <code>&lt;MMX&gt;</code> , <code>&lt;MMY&gt;</code>
6	Read device capabilities	<code>&lt;b1&gt;</code> , <code>&lt;b2&gt;</code> , . . . <code>&lt;b15&gt;</code> , <code>&lt;b16&gt;</code>
7	Read graphics text status	<code>&lt;X size&gt;</code> , <code>&lt;Y size&gt;</code> , <code>&lt;origin&gt;</code> , <code>&lt;angle&gt;</code> , <code>&lt;slant&gt;</code>
8	Read zoom status	<code>&lt;size&gt;</code> , <code>&lt;ON/OFF&gt;</code>
9	Read relocatable origin	<code>&lt;X&gt;</code> , <code>&lt;Y&gt;</code>
10	Read Reset status	<code>&lt;RESET&gt;</code> , <code>&lt;b1&gt;</code> , . . . <code>&lt;b6&gt;</code> , <code>&lt;b7&gt;</code>
11	Read graphics memory	<code>&lt;b1&gt;</code> , <code>&lt;b2&gt;</code> , . . . <code>&lt;bn&gt;</code>
12	Read area shading capability	1, 8, 8
13	Read dynamics capability	1, 1

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:  $\text{E} \text{t} * \text{s} 2 \wedge \text{<terminator> DC1}$

X coordinate Pen state

The terminal responds: +00360,+00080,0%

Y coordinate

## Read Graphics Cursor Position (Parameter=3)

The graphics cursor position is returned as a string of ASCII characters.

Read Graphics  
Cursor Request:  $\text{E} \text{t} * \text{s} 3 \wedge$

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:  $\text{E} \text{t} * \text{s} 4 \wedge$

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:  $\text{E} \text{t} * \text{s} 5 \wedge$

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:  $\text{E} \text{t} * \text{s} 6 \wedge$

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:

␣ \* ␣ 7 ^

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:

␣ \* ␣ 8 ^

The terminal responds:

<zoom size>,<zoom on/off><terminator>

where: <zoom size> = zoom setting, 1-16 (three di-  
gits 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:

␣ \* ␣ 9 ^

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:

␣ \* ␣ 10 ^

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:

␣ \* ␣ 11 ␣

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 Graphics 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:

␣ \* ␣ 12 ␣

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.

2647A <terminator>

Byte:	<byte 1>	<byte 2>	<byte 3>	.....	<byte 142>	<byte 144>
Bit:	5 4 3 2 1	5 4 3 2 1	5 4 3 2 1	.....	5 4 3 2 1	5 4 3 2 1
Data:	1 1 0 0 0	1 1 1 0 0	0 0 0 0 0		0 0 1 1 1	0 0 0 1 1
Display:	* *	* * *			* * *	* *

## COMMAND STATUS

A program executing in a host computer can obtain BASIC, AGL, and command execution error messages from the terminal by sending the following escape sequence:

Command Status Request:

**ESC, c REPORT STATUS OF COMMAND R**

In response the terminal returns a five-digit decimal value. A value of 00000 indicates that no error occurred. To extract the error code from a non-zero value, you convert the value to a 16-bit binary number, as follows:

Decimal Value: 00309

Hexadecimal Value: 0135

16-bit Binary Representation:

Bit:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	0	0	0	0	0	0	1	0	0	1	1	0	1	0	1
Value:						1	5	2		1	6	3	1	8	4	2
						0	1	5		2	4	2	6			
						2	2	6		8						
						4										

## FileSystem/User Interface Errors

Errors detected by the terminal's file system or user interface have decimal error codes within the range 257-1023 (inclusive). Below is a list of all the file system/user interface error codes. For each possible error, two error codes are shown. The first is the error's five-digit decimal code and the second (in parentheses) is the hexadecimal code for the 16-bit binary representation of that code.

### FILE SYSTEM INTRINSIC ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00257 (0101)	INVALID DEVICE SPECIFIED	Incorrectly spelling a device name; correct spelling.
00258 (0102)	INVALID FILE-ID	File-id passed to F/S intrinsics is incorrect; program error.
00259 (0103)	INVALID FILE ACCESS	Reading a write only file or writing to read only file.
00260 (0104)	ACTIVE FILE TABLE FULL	No more files can be opened until another file is closed first.
00261 (0105)	DUPLICATE DEVICES SPECIFIED	More than one of the same device name in multiple device string specification; delete one.
00262 (0106)	"SOURCE" = "DESTINATION"	Trying to read and write to the same device; change one of them.
00263 (0107)	TOO MANY DEVICES SPECIFIED	Can't have more than one source device; can't have more than one destination device for a compare operation.
00264 (0108)	CONFLICTING I/O	A write to display from I/O buffer is occurring during a GET operation.

### COMPARE COMMAND ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00265 (0109)	DIFF. LENGTH RECORDS	Record lengths on source and destination devices are different on a compare operation.
00266 (010A)	DIFF. IN BYTE x RECORD y FILE z	A mismatch on a byte on a compare operation.
00267 (010B)	DIFFERENCE IN RECORD TYPE	Record types did not agree from source and destination devices.

### COMMAND HANDLER ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00268 (010C)	ILLEGAL PARAMETER IN COMMAND	Unrecognized keyword in command parameter list.
00269 (010D)	EXTRANEOUS PARAMETER IN	Too many keywords in command parameter list.
00270 (010E)	MISSING PARAMETER IN COMMAND	Expected keyword in command parameter list not found.
00271 (010F)	NON-NUMERIC PARAMETER IN	Illegal numeric digit found in command parameter list.
00272 (0110)	EXCESSIVE NUMERIC PARAMETER IN COMMAND	Numeric digit too large in command parameter list.

### ASSIGN INTRINSIC ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00273 (0111)	ASSIGN TABLE FULL	Can't fit new assignment into assign table; delete another entry to make room.
00274 (0112)	NAME NOT IN ASSIGN TABLE	Name requested wasn't found in assign table.
00275 (0113)	RE-ASSIGNMENT NOT ALLOWED	Devices cannot be reassigned in Edit Mode or Data Logging Mode. To reassign devices, disable Edit or Data Logging Mode, reassign, then enable the mode.
00276 (0114)	ILLEGAL ASSIGN NAME	Incorrect character used in assign name. Names are truncated to 11 characters.

### COMMAND INTERPRETER ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00277 (0115)	PROGRAM NOT FOUND	Unable to find program with name specified in command channel.
00278 (0116)	UNRECOGNIZED COMMAND	Unable to find command to match command specified in command channel.
00279 (0117)	APPLICATION NOT FOUND	Unable to find a running application program.
00280 (0118)	EXECUTE FILE NOT FOUND	Unable to find an active execute file.
00281 (0119)	VOLUME TABLE FULL	Too many devices currently active. De-activate one of the devices.



# CARTRIDGE TAPE ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00288 (0120)	RUNOFF _ ON LEFT DRIVE	Tape ran off; remove tape and rethread it.
00304 (0130)	RUNOFF _ ON RIGHT DRIVE	Tape ran off; remove tape and rethread it.
00289 (0121)	ABORTED ON LEFT DRIVE	Tape operation aborted; remove tape.
00305 (0131)	ABORTED ON RIGHT DRIVE	Tape operation aborted; remove tape.
00290 (0122)	END OF TAPE ON LEFT DRIVE	No more room on tape.
00306 (0132)	END OF TAPE ON RIGHT DRIVE	No more room on tape.
00291 (0123)	END OF DATA ON LEFT DRIVE	End of valid data found on tape.
00307 (0133)	END OF DATA ON RIGHT DRIVE	End of valid data found on tape.
00292 (0124)	PROTECTED ON LEFT DRIVE	Tape write protected; replace write enable tab.
00308 (0134)	PROTECTED ON RIGHT DRIVE	Tape write protected; replace write enable tab.
00293 (0125)	NO TAPE ON LEFT DRIVE	Tape not inserted.
00309 (0135)	NO TAPE ON RIGHT DRIVE	Tape not inserted.
00294 (0126)	STALL ON LEFT DRIVE	Tape stalled; remove tape.
00310 (0136)	STALL ON RIGHT DRIVE	Tape stalled; remove tape.
00295 (0127)	READ FAIL ON LEFT DRIVE	Unable to read record from tape after nine retries.
00311 (0137)	READ FAIL ON RIGHT DRIVE	Unable to read record from tape after nine retries.
00296 (0128)	WRITE FAIL ON LEFT DRIVE	Unable to write to tape while in verify mode.
00312 (0138)	WRITE FAIL ON RIGHT DRIVE	Unable to write to tape while in verify mode.
00297 (0129)	FAIL ON LEFT DRIVE	Tape failed during a CTU test.
00313 (0139)	FAIL ON RIGHT DRIVE	Tape failed during a CTU test.
00298 (012A)	END OF FILE ON LEFT DRIVE	End of file reached on tape during file compare operation.
00314 (013A)	END OF FILE ON RIGHT DRIVE	End of file reached on tape during file compare operation.
00299 (012B)	FILE MISSING ON LEFT DRIVE	Missing file mark found during a SKIP or FIND file command.
00315 (013B)	FILE MISSING ON RIGHT DRIVE	Missing file mark found during SKIP or FIND file command.

### EXTERNAL PRINTER ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00336 (0150)	PRINT FAIL	General external printer fail; unspecific.
00337 (0151)	NO PAPER ON EXTERNAL PRINTER	Paper out; resupply paper.
00338 (0152)	NO EXTERNAL PRINTER	No printer connected, or no PCA board.

### DATACOMM ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00352 (0160)	TERMINAL NOT IN REMOTE	Terminal remote switch not down.
00353 (0161)	DATACOMM ERROR	Unspecific datacomm transmit or receive error.

### SHARED PRINTER ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00368 (0170)	NO SHARED PRINTER	No shared printer connected to HP-IB.
00369 (0171)	PRINTER IS BUSY, RETRY	Shared printer is currently being used; wait until free.
00370 (0172)	NO READ FROM PRINTER	Can't read from shared printer.
00371 (0173)	NO PP	Printer did not respond in time.

### HP-IB ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00384 (0180)	HP-IB TIME-OUT	Data transfer didn't complete before allotted time.
00385 (0181)	NO HP-IB PCA	No HP-IB PCA in terminal.
00386 (0182)	ILLEGAL HP-IB ADDR	HP-IB address specified did not exist.
00397 (0183)	HP-IB DEV BUSY, RETRY COMMAND WHEN FREE	Shared HP-IB device currently busy; wait until free.
00388 (0184)	NO HP-IB CONTROLLER	No terminal on the HP-IB has responded to a request for control.
00389 (0185)	HP-IB TEST FAIL xx	Self test of HP-IB failed.
00390 (0186)	HP-IB TEST NOT ATTEMPTED, ADDR=x, SYSCTL=YES or NO, CIC=YES or NO	Self-test of HP-IB cannot be performed under current conditions or configuration.
00391 (0187)	NOT HP-IB TALKER in BASIC.	Terminal not addressed to talk during a SENDBUS command
00392 (0188)	NOT HP-IB SYSTEM CONTROLLER	HP-IB protocol violated; REN and IFC control lines may only be accessed by system controller.

### TERMINAL ERROR MESSAGES

CODE	MESSAGE	EXPLANATION
00400 (0190)	HP-IB TE#x TO TE#y TEST FAIL	Transfer from terminal to terminal failed over HP-IB.

## BASIC Interpreter Errors

Errors detected by the Terminal BASIC Interpreter have decimal error codes within the range 1025-1279 (inclusive). Below is a list of all the BASIC Interpreter error codes. For each possible error, two error codes are shown. The first is the error's five-digit decimal code and the second (in parentheses) is the hexadecimal code for the 16-bit binary representation of that code.

**BASIC INTERPRETER ERROR MESSAGES**

CODE	MESSAGE	EXPLANATION
01026 (0402)	CHECKSUM ERROR IN BASIC	Something has caused the BASIC interpreter code to be altered. Reload the BASIC interpreter.
01027 (0403)	NEXT WITHOUT FOR	A NEXT statement is encountered and the corresponding FOR statement is missing or some intervening loop used the same loop variable. Add the appropriate FOR statement or change the loop variable in the intervening loop.
01028 (0404)	LOOP VARIABLE CAN'T BE LONG	A FOR statement must specify a simple variable of type INTEGER or SHORT. Declare the loop variable to be of type INTEGER or SHORT.
01029 (0405)	RETURN WITHOUT GOSUB	A RETURN statement is encountered and no GOSUB is currently active. Change the program to avoid accidentally falling into a subroutine or remove the offending RETURN.
01030 (0406)	NO RETURN	The end of program text is encountered while executing a subroutine. Add a RETURN statement.
01031 (0407)	RESUME WITHOUT ERROR	A RESUME statement is encountered while no error handling routine is being executed. Remove the RESUME statement.
01032 (0408)	NO RESUME	The end of program text is encountered while executing the error handling routine (see ON ERROR). Add a RESUME statement.
01033 (0409)	SUB WITHOUT SUBEND	Two SUB statements exist with no intervening SUBEND statement; or a SUB statement exists with no SUBEND statement prior to the end of program text. Add an appropriate SUBEND statement.
01034 (040A)	SUBEND WITHOUT SUB	A SUBEND statement occurs prior to any SUB statement. Add the appropriate SUB statement.
01035 (040B)	SUBPROGRAM NOT FOUND	A subprogram is referenced in a CALL statement but no corresponding SUB statement is found. Change the CALL or supply the subprogram text.
01036 (040C)	MISMATCHED QUOTES	A statement or input response or data record for READ # contains a quoted string which is not enclosed in matched quotes. Re-enter the statement or input response; or correct the data record.
01037 (040D)	MISMATCHED ELSE	More ELSE's occur in a statement than IF...THEN's. Correct the program logic.
01038 (040E)	SYNTAX ERROR	The statement syntax does not match the program text. Typically, the error occurs because a keyword is misspelled, a comma (or some other punctuation) is inappropriately used or omitted, parentheses are mismatched in an expression, a function call contains extra parameters, etc. Check the manual for the specific syntax for the statement.

# **BASIC INTERPRETER ERROR MESSAGES (Continued)**

CODE	MESSAGE	EXPLANATION
01039 (040F)	MISSING OPERAND	A function or subprogram call requires more parameters; or an expression terminates with an operator; or a READ # statement has no semicolon and/or variable list; or a PRINT # statement has a semicolon and no print list. Correct the syntax in the statement with the error.
01040 (0410)	NUMERIC OVERFLOW	A hex or octal constant exceeds the integer value range; or the results of an arithmetic calculation exceed the allowed value range. Correct the hex or octal constant; or change the program logic to prevent exceeding the appropriate value range.
01041 (0411)	DIVISION BY ZERO	A division operation has a zero divisor. Change program logic to prevent zero divisors.
01042 (0412)	UNDEFINED LINE NUMBER	A RESTORE, GOTO, GOSUB, or RESUME statement references a line number which does not exist in the current program unit. The GOTO or GOSUB may be invoked by the ON END, ON ERROR or ON KEY statements. Supply the missing line number or correct an erroneous line number specification.
01043 (0413)	LINE ALREADY EXISTS	The AUTO command is attempting to overwrite a program line which already exists. Specify the AUTO command so that it does not overwrite already existent program text, or delete the program lines which would be overwritten prior to initiating the AUTO command.
01044 (0414)	STATEMENT MUST HAVE LINE	A statement which can be executed only as part of a program has been attempted as an unnumbered command. Execute that statement only as part of a program.
01045 (0415)	STATEMENT MUST BE DIRECT	Execution of a command which is disallowed in a program has been attempted. Remove the command from the program text.
01046 (0416)	SUBSCRIPT OUT OF RANGE	A subscript value exceeds the limits of an array dimension (which were established upon first reference to that array). Explicitly declare the array with appropriate dimensions, or change the array declaration, or change program logic to avoid exceeding the dimension limits.
01047 (0417)	DIMENSION MISMATCH	The number of dimensions in an array reference does not match the number of dimensions specified in the first reference to that array; or the number of dimensions in an array passed to a subprogram as an actual parameter does not match the number of dimensions for the corresponding formal parameter. Change any array references which specify an incorrect number of dimensions.
01048 (0418)	REDECLARED VARIABLE	A declaration statement (DIM, INTEGER, SHORT or LONG) is executed more than once; or a variable occurs in more than one declaration statement. Change program logic to avoid executing declaration statements more than once; or remove extraneous declarations or a variable.
01049 (0419)	TYPE MISMATCH	An attempt is made to assign a string value to a numeric variable or a numeric value to a string variable; or the value or variable passed to a function or subprogram is not of the required type. Change the variable specification; or change the program to pass the correct type of value or variable to a function or subprogram.
01050 (041A)	STRING TOO LONG	A string expression results in a string longer than 255 characters. Change program logic.

# BASIC INTERPRETER ERROR MESSAGES (Continued)

CODE	MESSAGE	EXPLANATION
01051 (041B)	STRING FORMULA TOO COMPLEX	The evaluation of a string expression requires more intermediate results than the BASIC Interpreter allows. Break the string expression evaluation down so that you provide your own intermediate results.
01052 (041C)	NONCONTIGUOUS STRING	The character position preceding a substring (to which a value is being assigned by a LET statement or a function) is undefined. Prevent assignments to substrings which are not immediately preceded by defined characters.
01053 (041D)	SUBSTRING DESIGNATOR ERROR	The last character position in a substring specification precedes the first character position; or the last character position in a substring specification exceeds 255. Change the substring specification.
01054 (041E)	VALUE OUT OF RANGE	A value specified for a command or function parameter lies outside the permitted range, or an octal constant digit is greater than 7. The value may be a line number, a specific type of character (e.g., letter as versus punctuation mark), an integer (frequently positive and less than 256), have a required relationship to another value, etc. For example, the NUM function returns this error if the string argument is null; and the DELETE command returns this error if the last line in the line number range precedes the first line number. Correct the command or function parameter.
01055 (041F)	VARIABLE/NAME REQUIRED	The syntax requires a variable name (as versus a constant) in a function call; or a statement must start with a variable name if it does not start with some recognized statement initiator (ie., an implied LET statement is the default statement type); or a subprogram name must follow CALL or SUB. Change the statement to provide variable or subprogram names as required.
01056 (0420)	FUNCTION USAGE ERROR	A function invocation occurs in a disallowed context; eg., as the first action in a statement, or a cursor movement function in a PRINT # statement, or a GETKBD function call prior to executing GETKBD ON. See the manual for the proper context for the function usage and correct the context.
01057 (0421)	BUFFER NOT AVAILABLE	Two different files are being accessed by READ # or ENTER statements which have not used all the contents of the last record read for each file and an attempt is made to access a third file through a READ # or ENTER statement. Restructure the file contents of the input files or change the program so that no more than two input files are actively using buffers at the same time.
01058 (0422)	BUFFER OVERFLOW	The maximum record size of 255 characters in a PRINT # statement was exceeded. Change program logic to prevent records containing more than 255 characters.
01059 (0423)	LOADER FORMAT ERROR	A record which is supposed to be in the format for the terminal's binary loader fails to match that format. Recreate the file in the binary loader format.
01060 (0424)	UNASSIGNED/DISALLOWED FILE NUMBER	A READ #, LINPUT #, or PRINT # statement is attempted for a file which has not been specified in an ASSIGN statement; or the file number in an ASSIGN statement is less than 1 or greater than 255. Add an ASSIGN statement for the file number; or change the file number.

# **BASIC INTERPRETER ERROR MESSAGES (Continued)**

CODE	MESSAGE	EXPLANATION
01061 (0425)	DATA INPUT MISSING	A READ, READ # or ENTER statement encounters consecutive commas in the DATA statement or the input data or the DATA statement or input data record ends with a comma. Correct the DATA statement or the input data record so that they do not end in commas and that consecutive commas do not occur.
01062 (0426)	DATA ERROR	The data in a DATA statement or in an input buffer does not match the requirements of an item in the read list of a READ, READ #, or ENTER statement. Change the data or change the read list so that the data requirements match.
01063 (0427)	OUT OF DATA	A READ is executed and all DATA statements have already been used; or a READ # statement is executed and end of file is encountered for a file for which no ON END statement was executed in the current program unit. Supply the appropriate DATA or ON END statement.
01064 (0428)	OUT OF MEMORY	Insufficient memory exists to store the program text and all variables. Use SET SIZE to obtain more BASIC workspace or shorten the program or change the program structure to require less variable storage.
01065 (0429)	FORMAT ERROR	A format specification is non-existent, or an edit symbol is undefined, or a simple replicator is used with an invalid edit symbol, or a simple replicator exceeds 255, or parentheses are mismatched, or parentheses contain no edit symbols, or a disallowed combination of edit symbols is used for one print list item. Check the edit symbol specifications in the manual and correct any misuses.
01066 (042A)	NOT IN PROGRAM BREAK	GO is attempted when program execution was not halted by a STOP statement or a CONTROL break (default CONTROL-A) or when the program text was altered after being halted by a STOP statement or a CONTROL break. Restart the program.
01067 (042B)	MULTIPLE STATEMENTS DISALLOWED	The initial entry (ie., not through GET or MERGE) of a line containing multiple statements is attempted after the SET SINGLE command has disallowed multiple statements per line. Enter the line as separate lines or specify SET MULTIPLE.
01068 (042C)	FEATURE NOT PRESENT	A statement, command, or function is referenced and the current version of BASIC does not support that operation. This may occur when trying to run a program which was saved by one version of BASIC (eg., a version of BASIC for which REMOVE STDX has not been specified) with a different version of BASIC (eg., a version of BASIC for which REMOVE STDX has been specified) which does not support the statement, command, or function referenced. Load the appropriate version of BASIC prior to loading and/or executing any BASIC operations.
01069 (042D)	NO MESSAGE FOR ERROR CODE	The error message routine is invoked (probably by the ERROR statement) and no error message text exists for the specified error code. Issue the direct statement PRINT ERRN, ERRL to determine which error code and program line caused this error message to be displayed. Add custom error message text to the ERROR statement or change the error code which is specified.

**BASIC INTERPRETER ERROR MESSAGES (Continued)**

CODE	MESSAGE	EXPLANATION
1070 (042E)	NESTING EXCEEDS LIMIT	More than 255 levels of subprograms are active.
1071 (042F)	NOT ALLOWED W/SECURE PGM	Secured programs cannot be listed without the SECURE option.

**AGL Errors**

Errors detected by AGL have decimal error codes within the range 1281-1535 (inclusive). Below is a list of all the AGL error codes. For each possible error, two codes are shown. The first is the error's five-digit decimal error code and the second (in parentheses) is the hexadecimal code for the 16-bit binary representation of that code.

**AGL ERROR MESSAGES**

CODE	MESSAGE	EXPLANATION
01281 (0501)	AGL ERROR 1281	Syntax error.
01282 (0502)	AGL ERROR 1282	Too many parameters.
01283 (0503)	AGL ERROR 1283	Cannot default parameter.
01284 (0504)	AGL ERROR 1284	Driver not present.
01285 (0505)	AGL ERROR 1285	Parameter out of range.
01286 (0506)	AGL ERROR 1286	Illegal action request.
01287 (0507)	AGL ERROR 1287	Cannot open new device without closing old device.
01288 (0508)	AGL ERROR 1288	No devices on.
01289 (0509)	AGL ERROR 1289	Wrong number of parameters.
01290 (050A)	AGL ERROR 1290	Null area specified.
01291 (050B)	AGL ERROR 1291	Scaling values are equal.

**2647 DRIVER ERROR**

CODE	MESSAGE	EXPLANATION
01292 (050C)	2647 DRIVER ERROR 1292	Attempt to plot beyond limits of logical address space.



### PLOTTER DRIVER ERRORS

CODE	MESSAGE	EXPLANATION
01293 (050D)	PLOTTER DRIVER ERROR 1293	Instruction not recognized.
01294 (050E)	PLOTTER DRIVER ERROR 1294	Wrong number of parameters.
01295 (050F)	PLOTTER DRIVER ERROR 1295	Bad parameter.
01296 (0510)	PLOTTER DRIVER ERROR 1296	Illegal character.
01297 (0511)	PLOTTER DRIVER ERROR 1297	Unknown character set.
01298 (0512)	PLOTTER DRIVER ERROR 1298	Position overflow.

# INSTALLATION

SECTION

VII

## 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かきています。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って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).

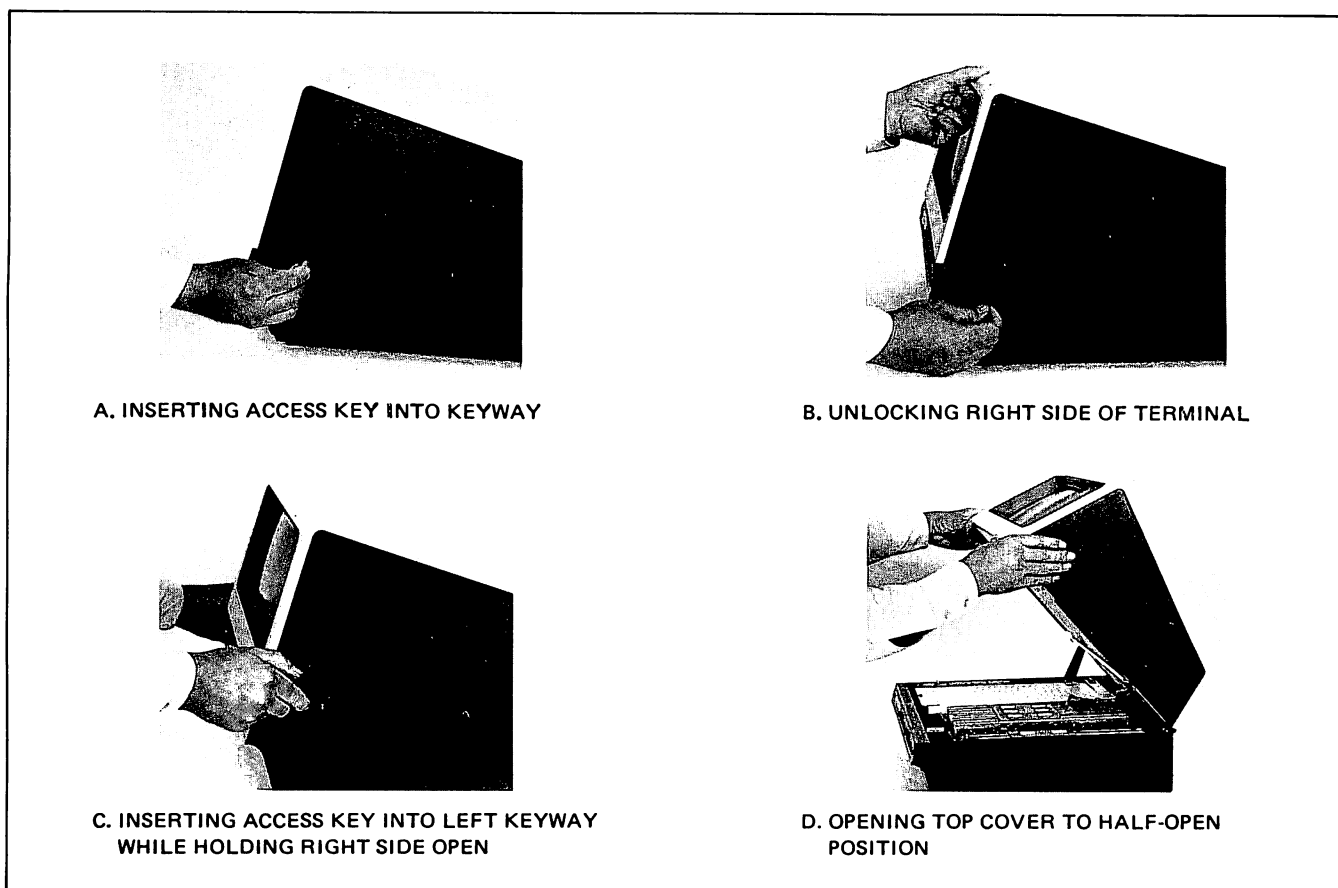
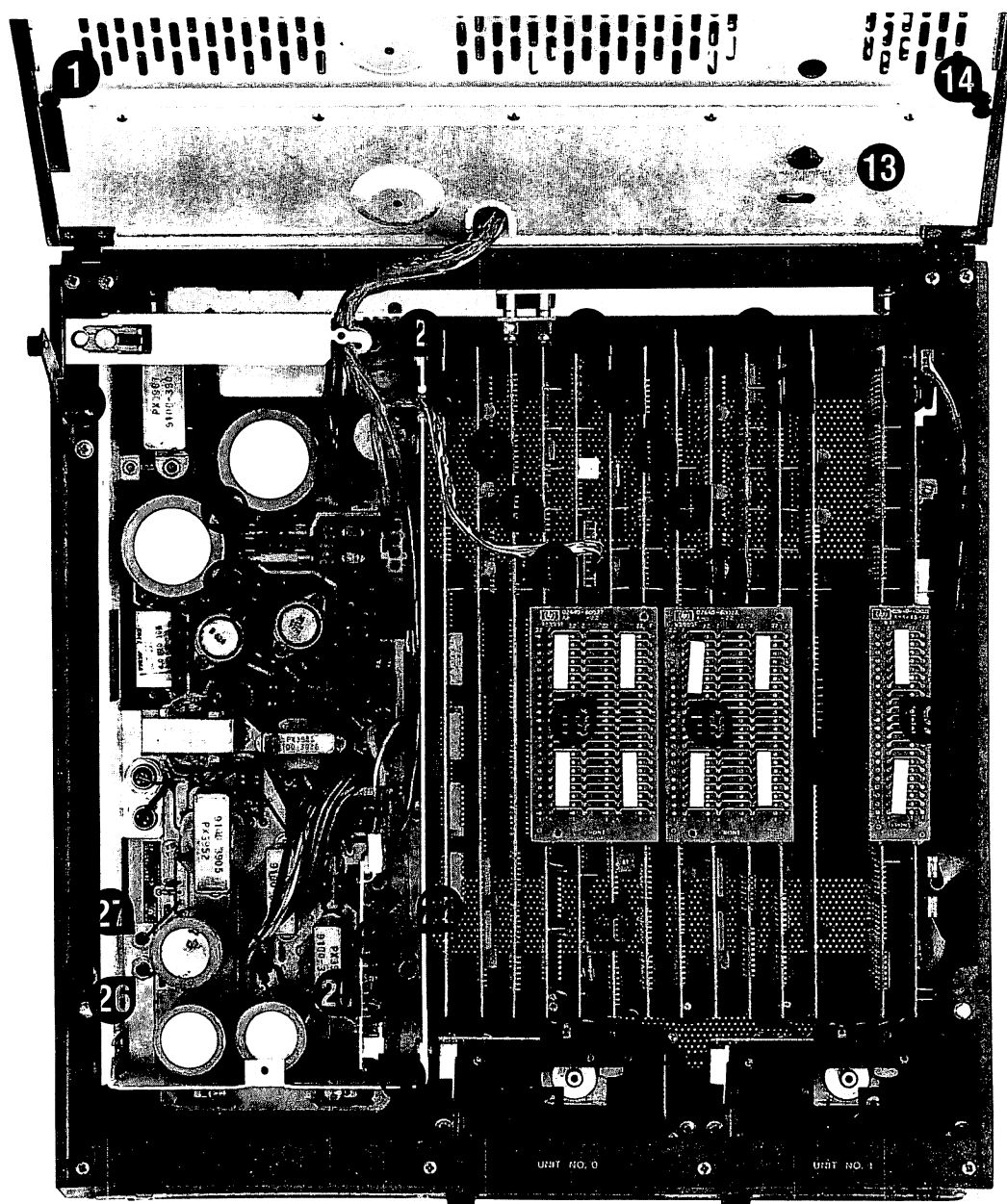


Figure 7-1. Opening the Terminal



1 SHIELD SNAP FASTENER  
 2 KEYBOARD INTERFACE PCA  
 3 32K UNIVERSAL MEMORY PCA  
 4 GRAPHICS MICROCONTROLLER PCA  
 5 GRAPHICS DISPLAY MEMORY PCA  
 6 GP DISPLAY TIMING PCA  
 7 DISPLAY MEMORY ACCESS PCA  
 8 DISPLAY CONTROL PCA  
 9 CONTROL MEMORY PCAs  
 10 PROCESSOR PCA

11 32K UNIVERSAL MEMORY PCA (CON-  
 NECTED TO TOP PLANE CONNECTOR)  
 12 ASYNCHRONOUS DATA COMM PCA  
 13 CRT SHIELD  
 14 SHIELD SNAP FASTENER  
 15 CTU INTERFACE PCA  
 16 READ/WRITE PCA  
 17 SPEED ADJUSTMENT  
 18 OPTION SLOT  
 19 TOP PLANE CONNECTORS  
 20 CRYSTAL Y1

21 CTU TRANSPORT ASSYs  
 22 -42V TEST POINT  
 23 +12V TEST POINT  
 24 +5V ADJUSTMENT  
 25 POWER SUPPLY CONTROL PCA  
 26 -12V TEST POINT  
 27 +5V TEST POINT  
 28 POWER SUPPLY FUSES  
 29 SAFETY LATCH

Figure 7-2. Mainframe Bottom Part Locations

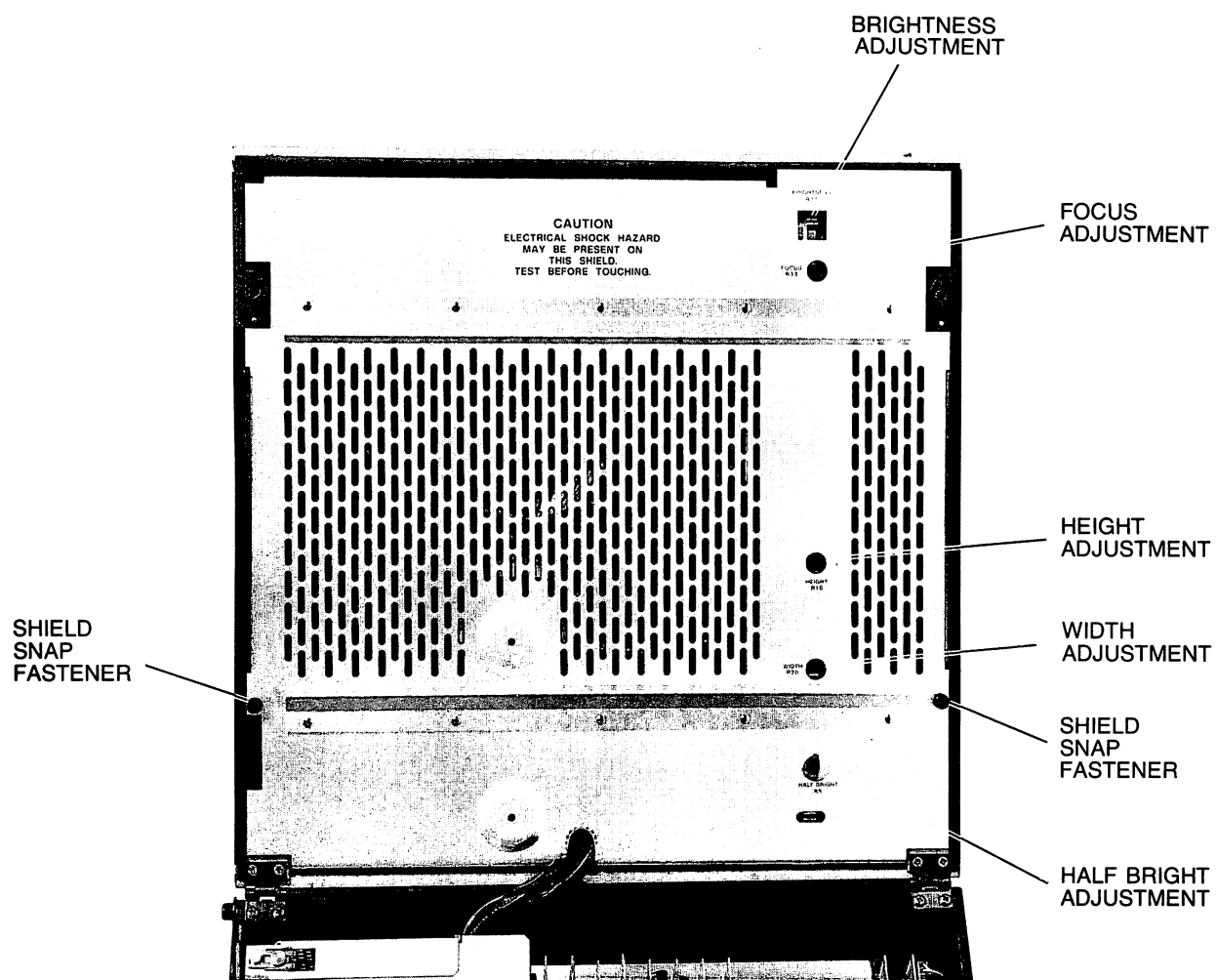


Figure 7-3. Mainframe Top Part Locations

## 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").

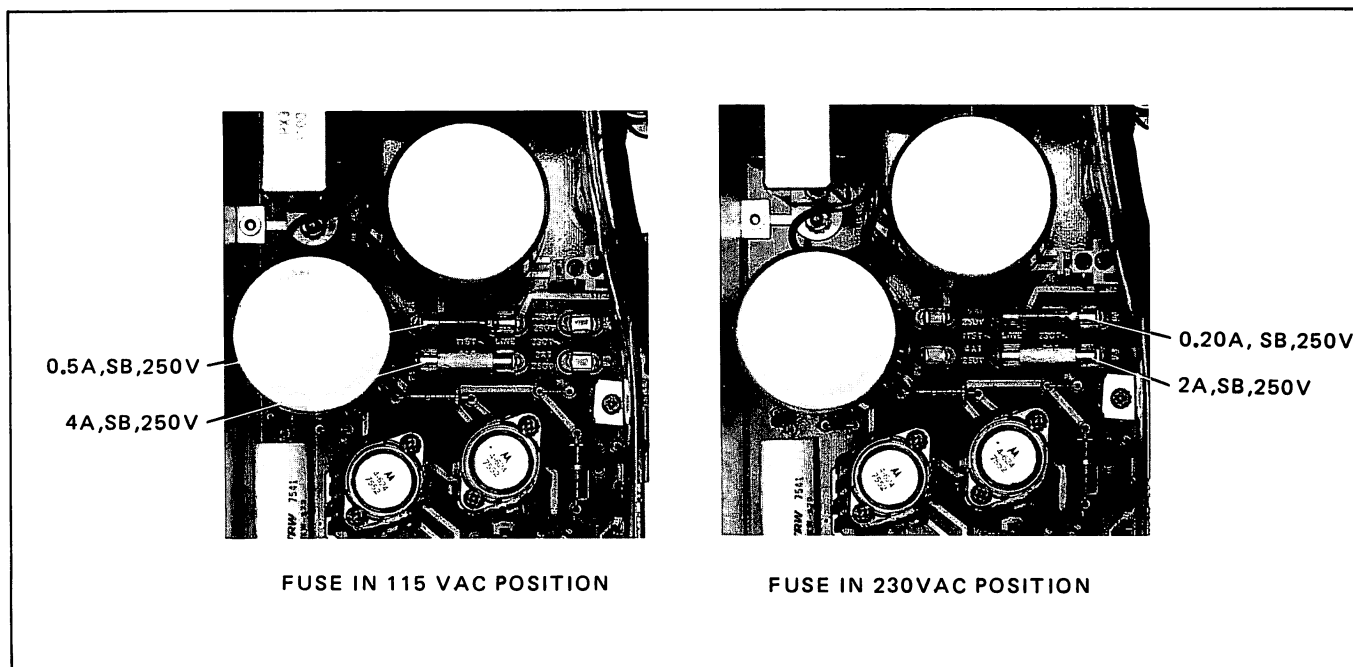


Figure 7-4. Fuse Positions for 115 VAC and 230 VAC Line Voltage

## 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, 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 Expansion 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.

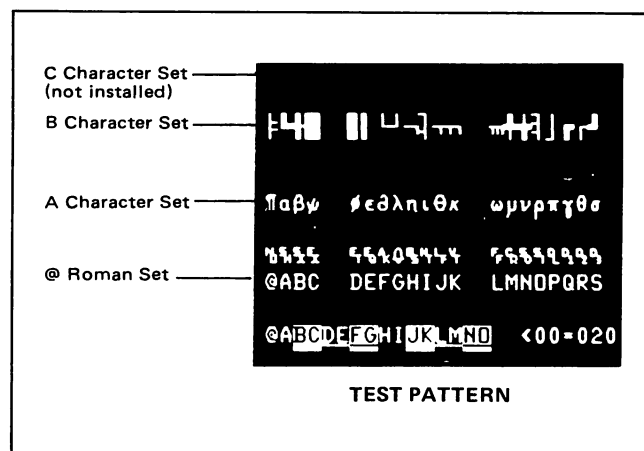


Figure 7-5. Sample Character Set Configuration

### 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 Expansion 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 Expansion 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.

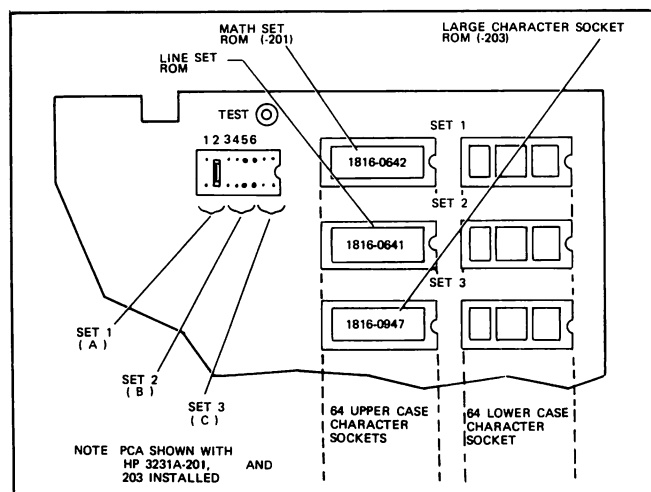


Figure 7-6. Display Enhancement PCA Jumper and ROM Socket Locations

- Open terminal to its half open position (refer to "Opening the Terminal").
- Insert connector removal tool under the Top Plane Assembly which connects the display PCA's together (see figure 7-2) 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

The 13231A Display Enhancement accessory is not normally present in graphics terminals because the two available PCA slots are used for data communications and peripheral interfacing. However, either the data communications PCA or the peripheral interface PCA may be removed to accommodate the display enhancement accessory.

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.

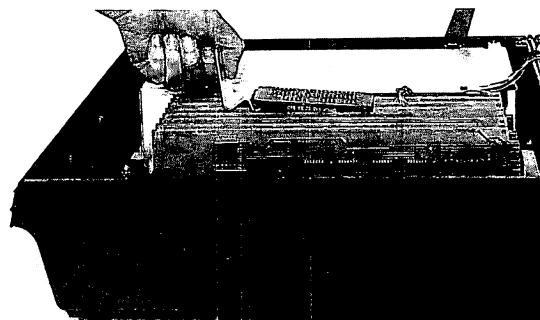


Figure 7-7. Top Plane Assembly Removal



## Installation

6. ( ) Install five-wide Top Plane Connector Assembly, part no. 02640-60016 on Display Enhancement, DMA, Display Control, Display Timing, and Graphics Display PCA connectors.
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 enhancement (inverse video, half-bright, underline, etc.). If enhancements are correct skip to step 10. If adjustment is necessary, perform step 9.
9. ( ) Perform brightness, half bright, focus, and field adjustments in accordance with the *HP Terminal Service Manual*.
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

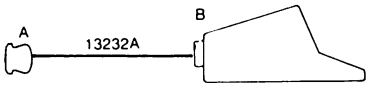
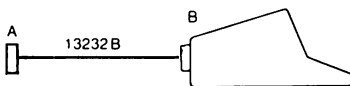
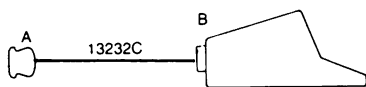
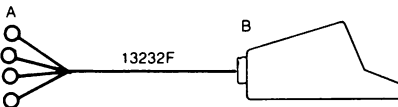
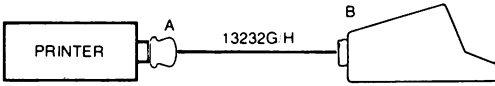
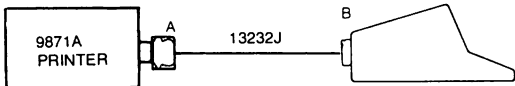
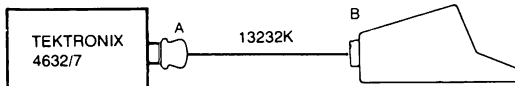
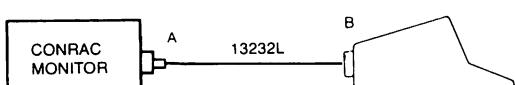
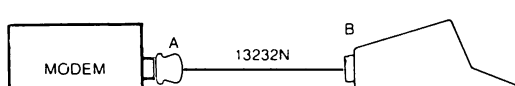
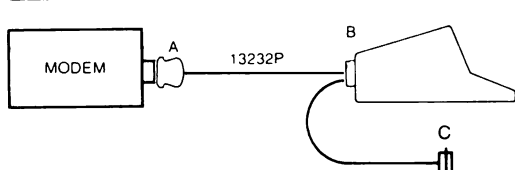
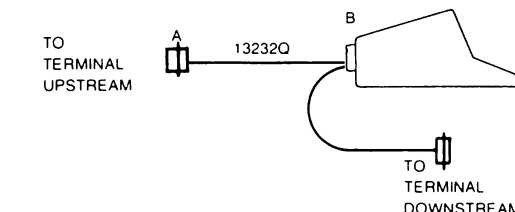
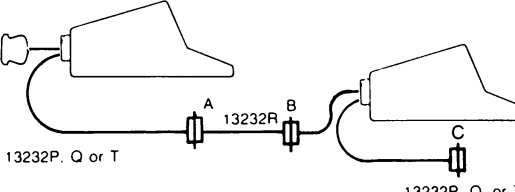
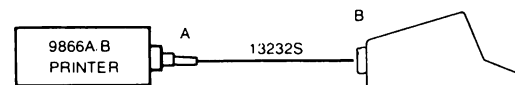
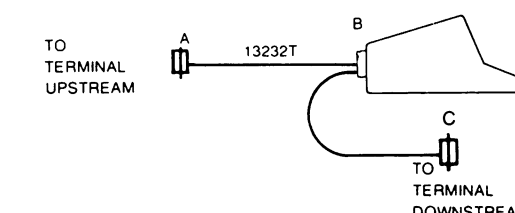
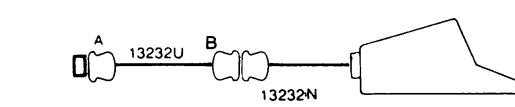
CABLE	FUNCTION	CONNECTORS			LENGTH	HOOKUP
		A	B	C		
13232A	Connects data communications interface PCA to modem 103/202. (Cable part no. 02640-60043.)	RS232C (male)	Hood	—	4.57 Metres 15 feet	
13232B	Connects 12531/12880 teleprinter interface PCA to terminal. (Cable part no. 02640-60058.)	Hood	Hood	—	15.25 Metres 50 feet	
13232C	Connects data communications interface PCA to RS232C 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 RS232C 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 Contrac Monitor. (Cable part no. 02640-60121.)	BNC	Hood	—	7.61 Metres 25 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 (female)	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	

## 32K RAM Memory PCAs

The HP 2647A in its standard configuration includes two 32K RAM memory PCAs, one (used for BASIC program storage) is accessed via the top plane and the other (used for display memory) via the bottom plane. Figure 7-8 illustrates how these two PCAs should be strapped.

## 32K Control Memory PCAs

Figure 7-9 illustrates how the two Control Memory PCAs should be strapped and figure 7-10 illustrates a typical memory map for the HP 2647A terminal.

### NOTE

All straps on the Processor PCA must be closed for the terminal to function properly.

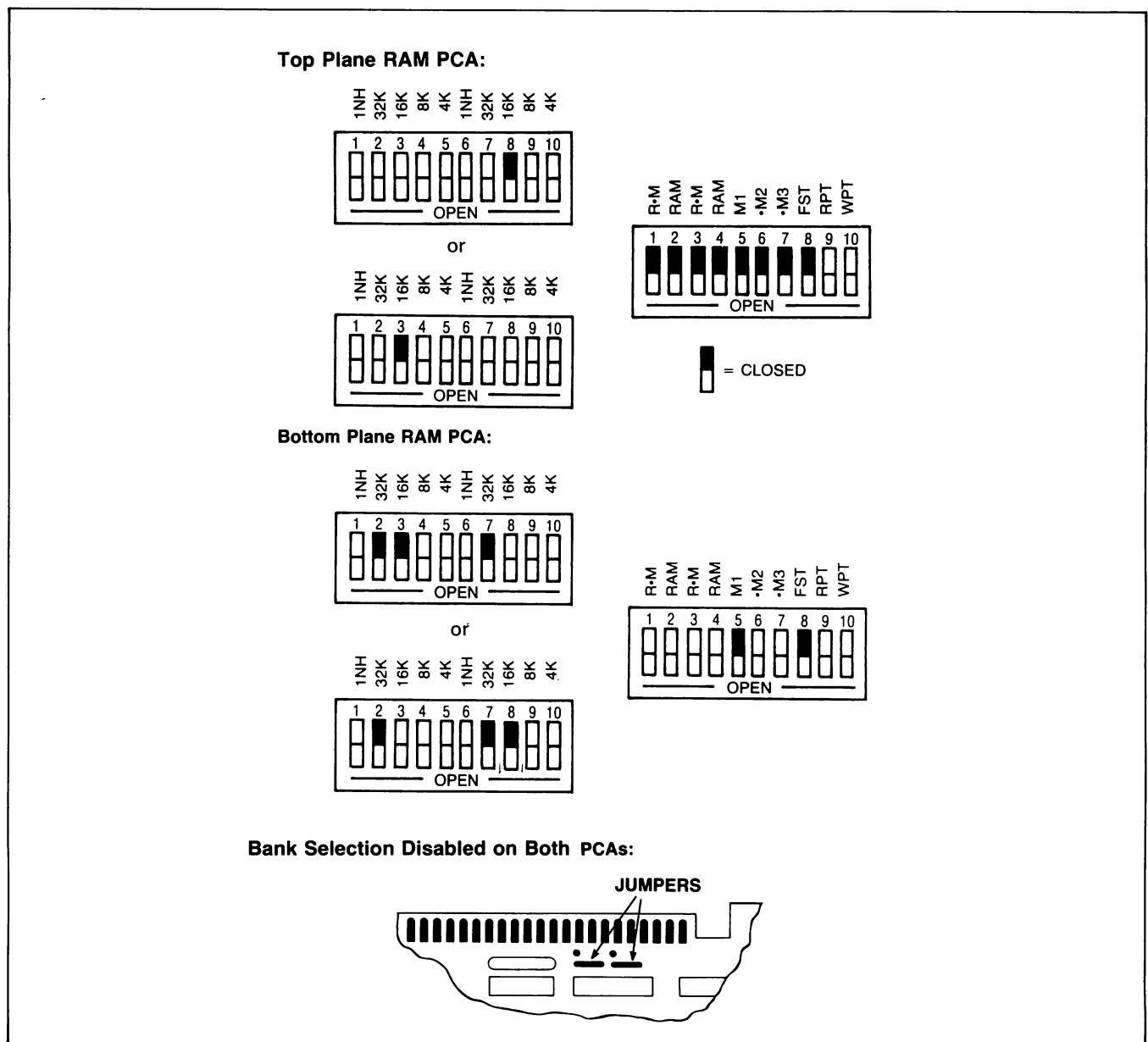
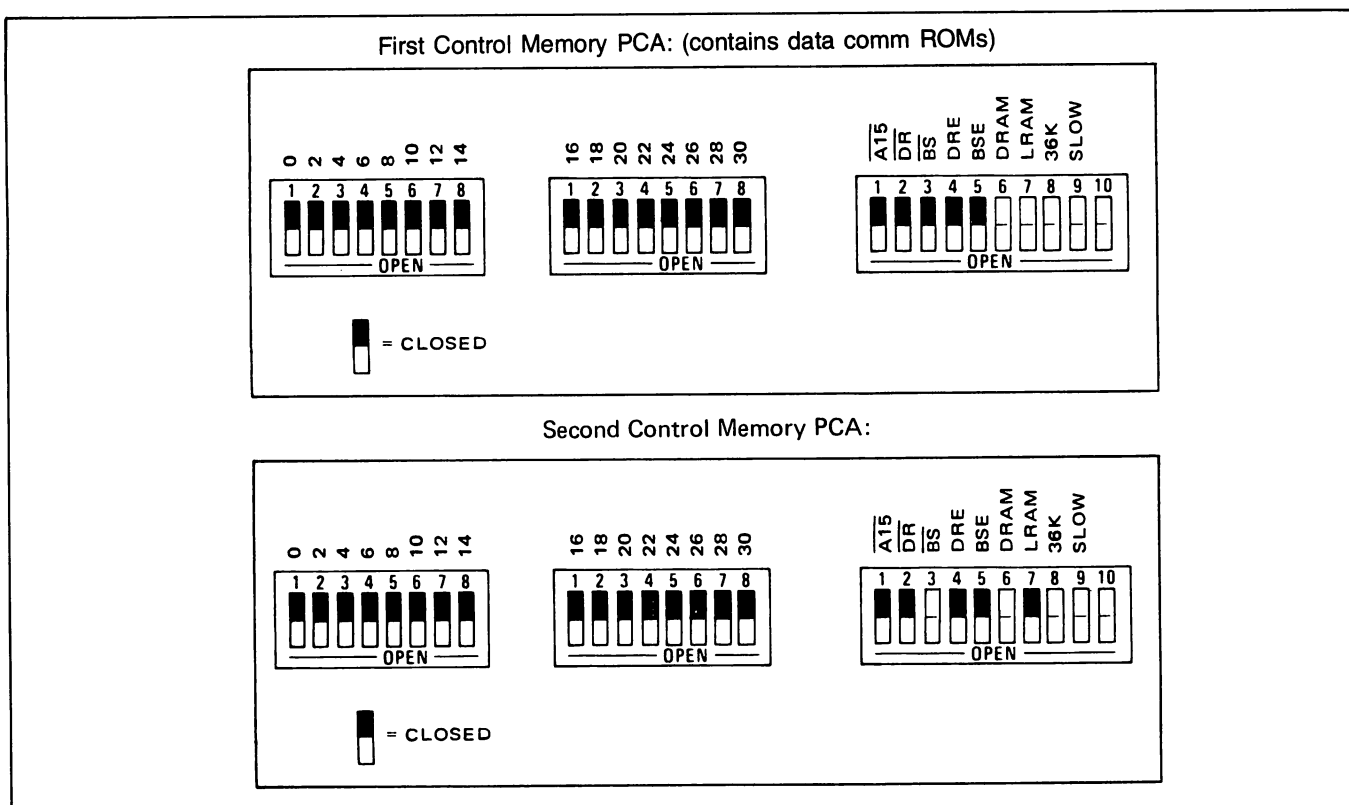
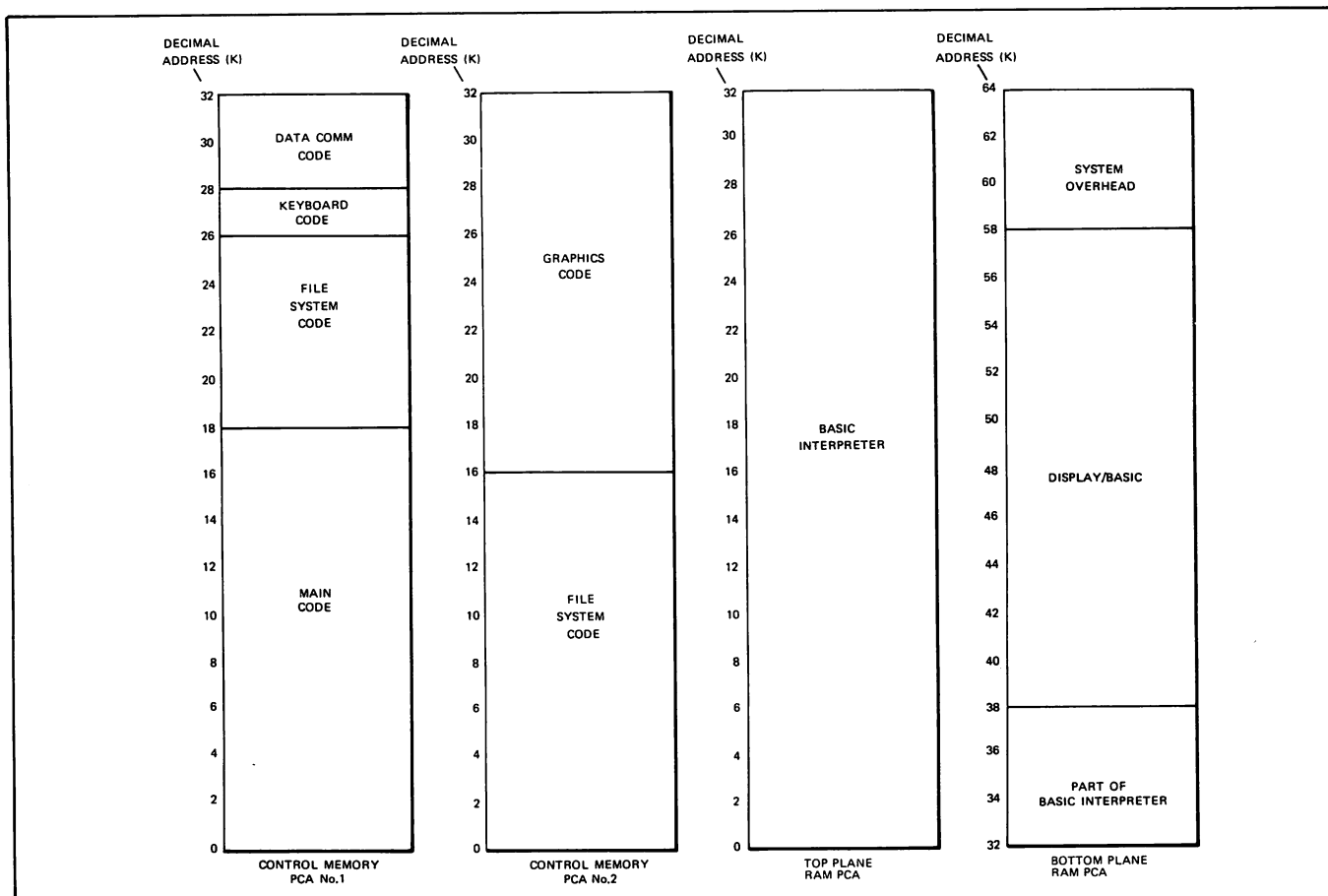


Figure 7-8. 32K RAM PCA Jumper Settings



**Figure 7-9. Control Memory PCA Jumper Settings**



**Figure 7-10. Typical Memory Map**

## HP 13238A Terminal Duplex Register

The HP 13238A Terminal Duplex Register accessory provides 8-bit parallel interface to external printers (such as the HP 9866A/B Printer and the HP 9871A Printer). (For peripheral sharing networks, refer to the "HP 13296A Shared Peripheral Interface".)

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 7-11.
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 (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 13246A/B Printer Subsystem (9866)

This accessory provides alphanumeric output and consists of a Terminal Duplex Register PCA, part no. 02640-60031; a 13232S Cable Assembly, part no. 02640-60135; and an HP 9866A or B Printer. To install this accessory, first perform the HP 13238A Terminal Duplex Register installation instructions steps 1 through 7. After the PCA and cable assembly have been installed, install the printer in accordance with the instructions contained in the *HP 9866A/B Printer Operator's Manual*, part no. 09866-90901.

## 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. 02640-90042.

## 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.

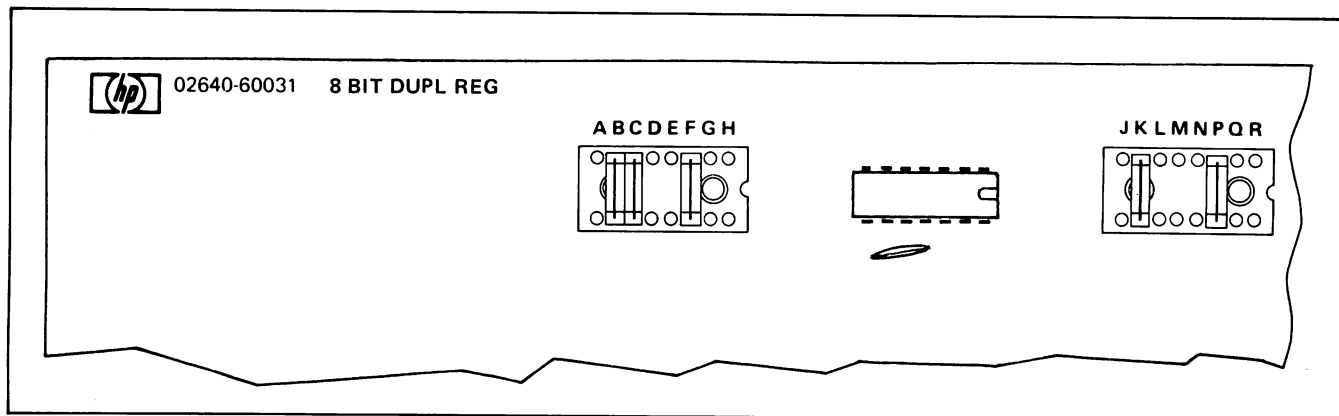


Figure 7-11. Terminal Duplex Register PCA Jumper Configuration

## 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

ITEM	PART NUMBERS			
	13260A-006	13260B-006	13260C-006	13260D-006
Interface Printed Circuit Assembly	02640-60086	02640-60143	-02640-60106	02640-60107
ROM IC's	1818-0600	1818-0600	1818-0628 (std only) 1818-0614 (opt 007 only) 1818-0629	1818-0628 (std only) 1818-0614 (opt 007 only) 1818-0629
Baudrate Label	7120-6388	7120-6388	7120-6386	7120-6386
Cable		02640-60083	02640-60083	02640-60083
Switch Cover			4040-1356	4040-1356

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. ( ) The two Control Memory PCAs (part no. 02640-60221) look identical except for their jumper settings. Using figure 7-9 as a guide, locate 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 A28 and A30 of Control Memory PCA (see figure 7-12).

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 7-12). Also, be sure that the switches on the PCA are set to the positions shown in figure 7-9.

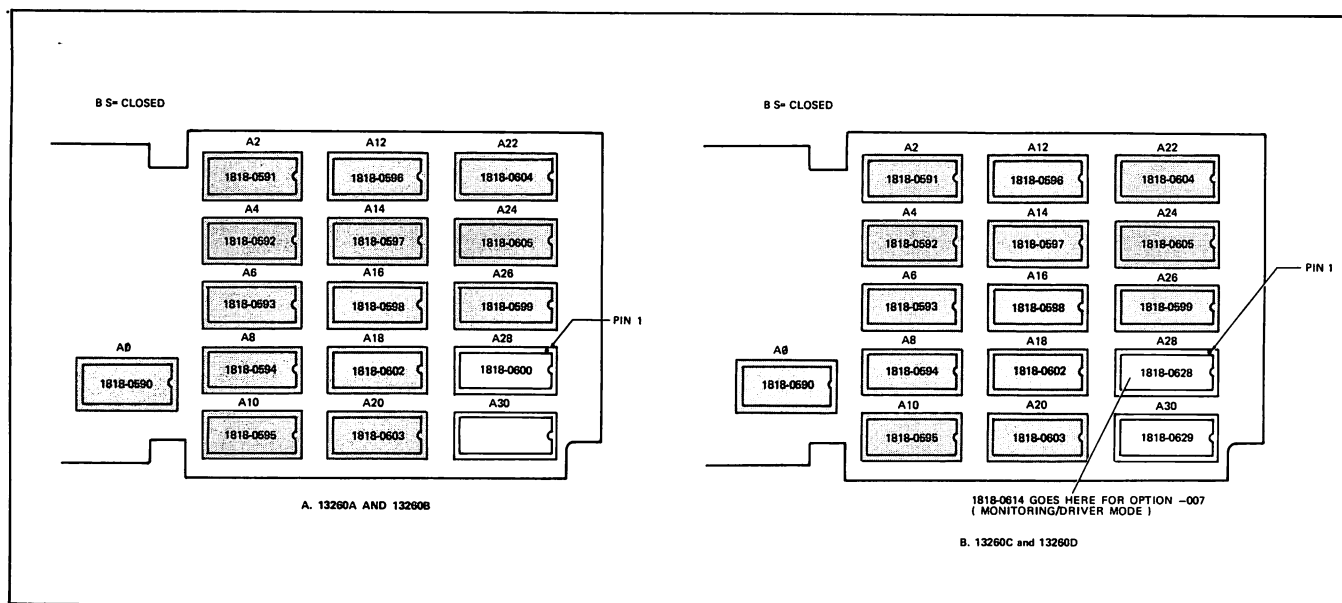


Figure 7-12. Control Memory PCA Data Comm Firmware IC Locations

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 should never be vacant connectors between PCA's except for the two CTU PCA's (Read/Write PCA and CTU Interface PCA) 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 7-13.
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 13296A Shared Peripheral Interface

This accessory provides the means of 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).

To configure and install this accessory, proceed as follows:

1. ( ) Using table 7-4 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").
3. ( ) Install the PCA in the third slot from the power supply to allow room for installing the HP-IB Interface Adapter. This will require you to move all existing PCA's (in the third through twelfth slots) one slot position to the right to make room for the HP-IB PCA.
4. ( ) 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

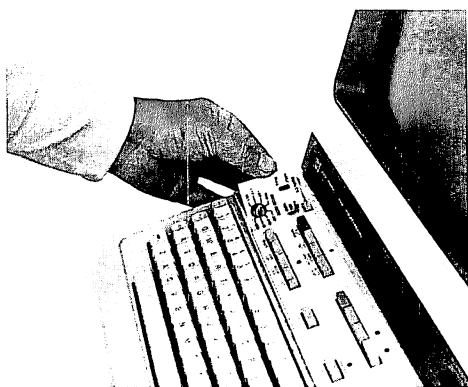
The hood connector and PCA connector P2 are identically keyed to prevent inadvertent erroneous connections. Connecting the two together requires minimal hand pressure. If excessive resistance is encountered, an incorrect connection is being attempted.

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.

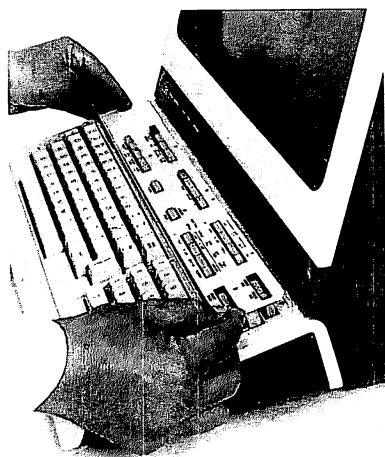
To configure the HP-IB compatible devices (such as printers, plotters, etc.) in the network, refer to section VIII, "HP-IB Configurations".



A. REMOVING A KEYBOARD OVERLAY



B. INSTALLING BAUDRATE OVERLAY



C. INSTALLING KEYBOARD OVERLAY

Figure 7-13. Installing Keyboard Overlays

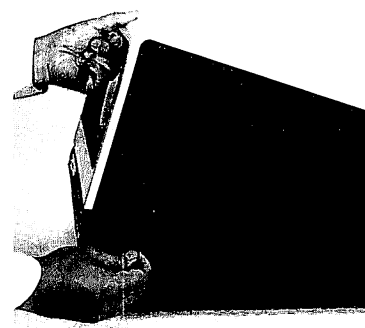


Table 7-4. HP-IB Interface Switch Settings

SWITCH(ES)	SETTING(S)	SWITCH(ES)	SETTING(S)
A4, A11, A10, A9	These four switches specify the PCA module address and must be set as follows:  <div> <div>A4</div> <div>A11</div> <div>A10</div> <div>A9</div> </div> Closed    Open    Closed    Closed		<div> <div>Decimal Address</div> <div>B4</div> <div>B3</div> <div>B2</div> <div>B1</div> <div>B0</div> </div> <div> <div>8</div> <div>C</div> <div>O</div> <div>C</div> <div>C</div> <div>C</div> </div> <div> <div>9</div> <div>C</div> <div>O</div> <div>C</div> <div>C</div> <div>O</div> </div> <div> <div>10</div> <div>C</div> <div>O</div> <div>C</div> <div>O</div> <div>C</div> </div> <div> <div>11</div> <div>C</div> <div>O</div> <div>C</div> <div>O</div> <div>O</div> </div> <div> <div>12</div> <div>C</div> <div>O</div> <div>O</div> <div>C</div> <div>C</div> </div> <div> <div>13</div> <div>C</div> <div>O</div> <div>O</div> <div>C</div> <div>O</div> </div> <div> <div>14</div> <div>C</div> <div>O</div> <div>O</div> <div>O</div> <div>C</div> </div> <div> <div>15</div> <div>C</div> <div>O</div> <div>O</div> <div>O</div> <div>O</div> </div> <div> <div>16</div> <div>O</div> <div>C</div> <div>C</div> <div>C</div> <div>C</div> </div> <div> <div>17</div> <div>O</div> <div>C</div> <div>C</div> <div>C</div> <div>O</div> </div> <div> <div>18</div> <div>O</div> <div>C</div> <div>C</div> <div>O</div> <div>C</div> </div> <div> <div>19</div> <div>O</div> <div>C</div> <div>C</div> <div>O</div> <div>O</div> </div> <div> <div>20</div> <div>O</div> <div>C</div> <div>O</div> <div>C</div> <div>C</div> </div> <div> <div>21</div> <div>O</div> <div>C</div> <div>O</div> <div>C</div> <div>O</div> </div> <div> <div>22</div> <div>O</div> <div>C</div> <div>O</div> <div>O</div> <div>C</div> </div> <div> <div>23</div> <div>O</div> <div>C</div> <div>O</div> <div>O</div> <div>O</div> </div> <div> <div>24</div> <div>O</div> <div>O</div> <div>C</div> <div>C</div> <div>C</div> </div> <div> <div>25</div> <div>O</div> <div>O</div> <div>C</div> <div>C</div> <div>O</div> </div> <div> <div>26</div> <div>O</div> <div>O</div> <div>C</div> <div>O</div> <div>C</div> </div> <div> <div>27</div> <div>O</div> <div>O</div> <div>C</div> <div>O</div> <div>O</div> </div> <div> <div>28</div> <div>O</div> <div>O</div> <div>O</div> <div>C</div> <div>C</div> </div> <div> <div>29</div> <div>O</div> <div>O</div> <div>O</div> <div>C</div> <div>O</div> </div>

## HP 13349A Printer Subsystem (9871)

This accessory provides alphanumeric output and consists of a Terminal Duplex Register PCA, part no. 02640-60031; an Interface Cable Assembly, part no. 02640-60116; and an HP 9871A Printer. To install this accessory, first perform the HP 13238A Terminal Duplex Register installation instructions steps 1 through 7. After the PCA and cable assembly have been installed, install the printer in accordance with the instructions contained in the *HP 13349A Printer Subsystem Operating Manual*, part no. 13349-90901.



### 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.)

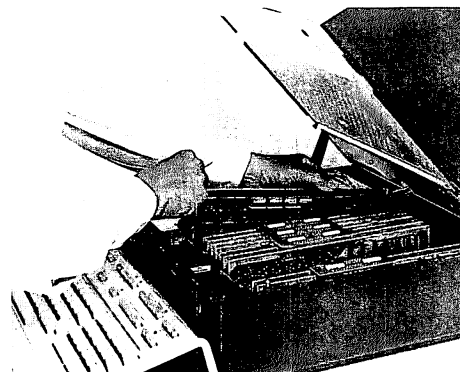
TEST POINT	VOLTAGE TOLERANCE
+ 5V diode	+4.85V to +5.25V
-42V diode	-40V to -46V
+12V diode	+11.8V to +12.6V
-12V diode	-11.8V to -12.6V

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.

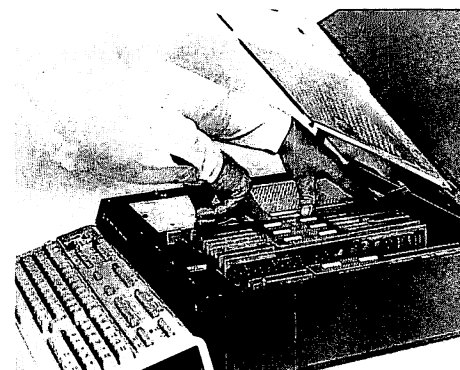
### SELECTING OPTIONAL OPERATING FUNCTIONS

The terminal is equipped with jumper and switch selectable options that can be used to alter some of its operating functions (see figure 7-14). These options and their effects on terminal operation are discussed in tables 7-5 through 7-9. To select an operating function, proceed as follows:

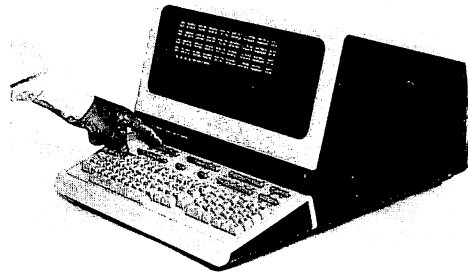
1. ( ) Open the terminal to its half open position (refer to "Opening the Terminal").



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 7-15, 7-16, 7-17, or 7-18 (as applicable) and table 7-5, 7-6, 7-7, 7-8, or 7-9 (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").

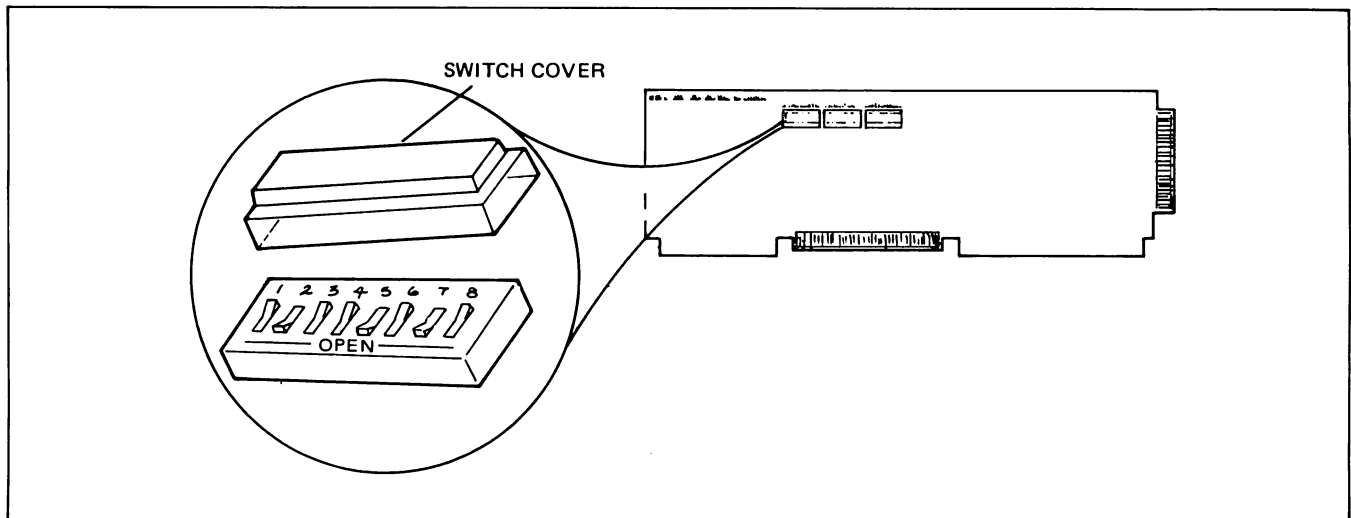


Figure 7-14. Typical Strapping Option Switch Assembly

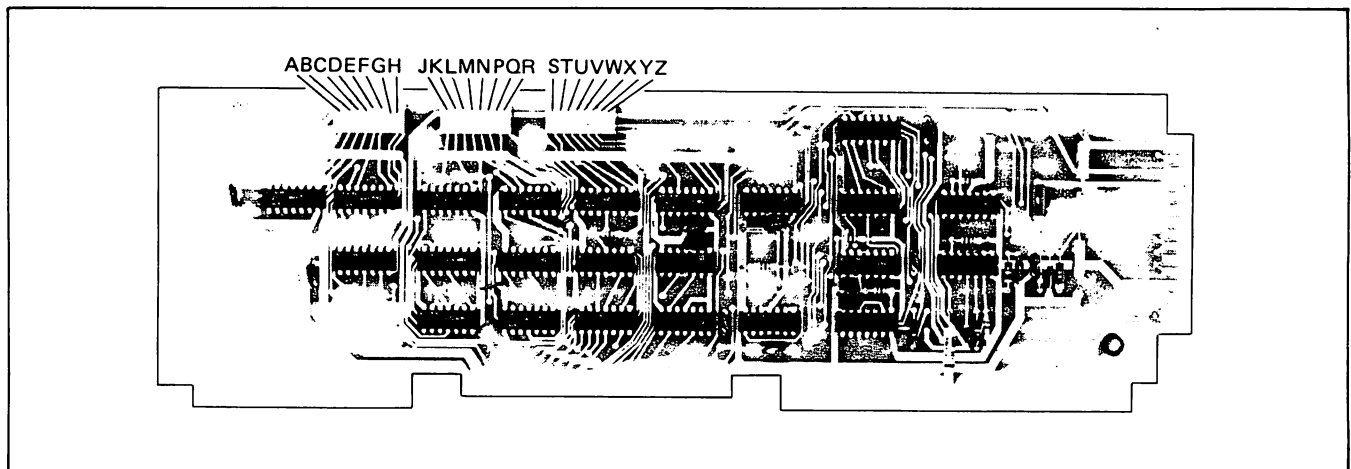


Figure 7-15. Keyboard Interface PCA Strapping Options

Table 7-5. 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  key is pressed with  key latched down, each tape record begins with an $\text{LF}$ and is terminated by a $\text{NF}$ .	Each tape record is terminated by $\text{NF}$ .
F	Fast Binary Read	The transmission rate is determined by the BAUD RATE switch on the keyboard.	When an $\text{FR}$ (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  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 $\text{FR}$ .
L	Self Test Inhibit	No effect.	Self Test function is inhibited. Pressing TEST key or issuing $\text{FZ}$ 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  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 7-5. Keyboard Interface PCA Strapping Operations 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 (122) (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(122) (Secondary Receive Data) line. The terminal responds by turning off CA (106) (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.

Table 7-6. Keyboard Interface PCA Strapping Options for Multipoint

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	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 7-6. 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	Space Compression	Space characters are sent normally.	Space characters are compressed.
T,U	Output Block Size	<p>T      U</p> <p>C      C</p> <p>O      C</p> <p>C      O</p> <p>O      O</p> <p>C = closed, O = open</p>	<p><b>BLOCK SIZE (BYTES)</b></p> <p>1/2 Data Comm Buffer (refer to switches J16, J17 on multipoint PCA).</p> <p>250 max</p> <p>500 max</p> <p>1000 max</p>
V	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	Transparency	No effect.	Causes all data sent from the terminal to be transparent.

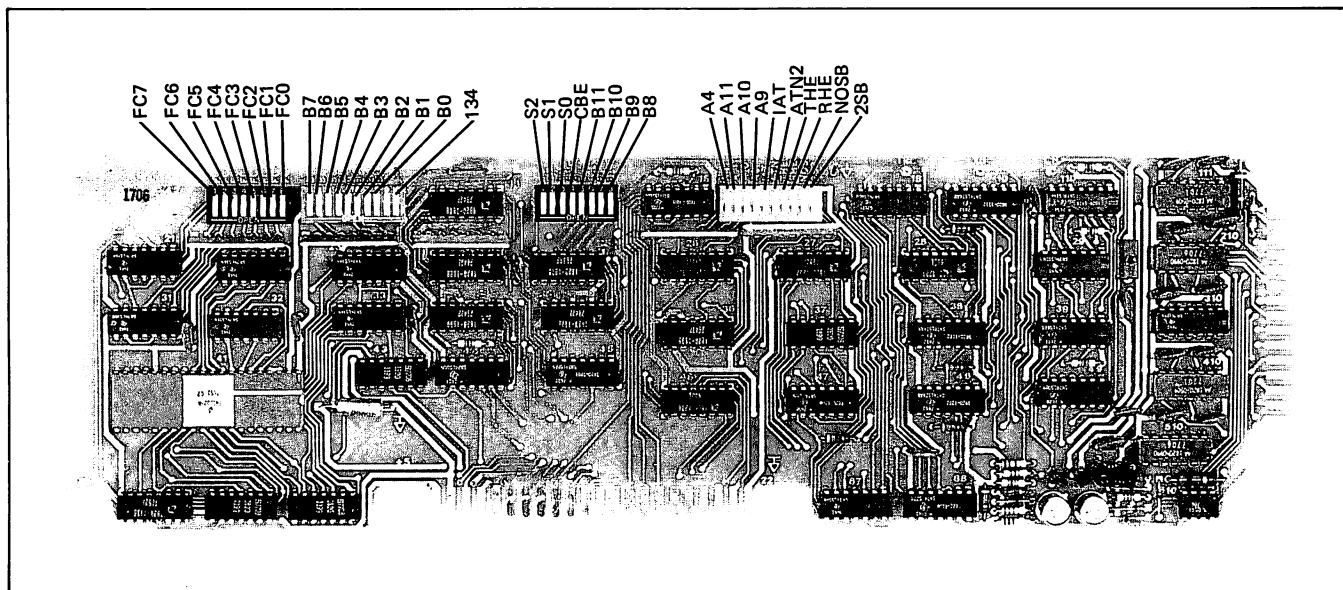


Figure 7-16. Extended Asynchronous Communications PCA Strapping Options

Table 7-7. 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	<b>SWITCH SETTING</b>	<b>TRANSMIT BAUD RATE</b>
		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	(not used)			
J16, J17	Input Buffer Size	J17	J16	BUFFER SIZE
		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	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)		

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	(not used)			
J16, J17	Input Buffer Size	J17	J16	BUFFER SIZE
		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	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.		

CAUTION

Close only one switch, otherwise damage to the PCA may result.

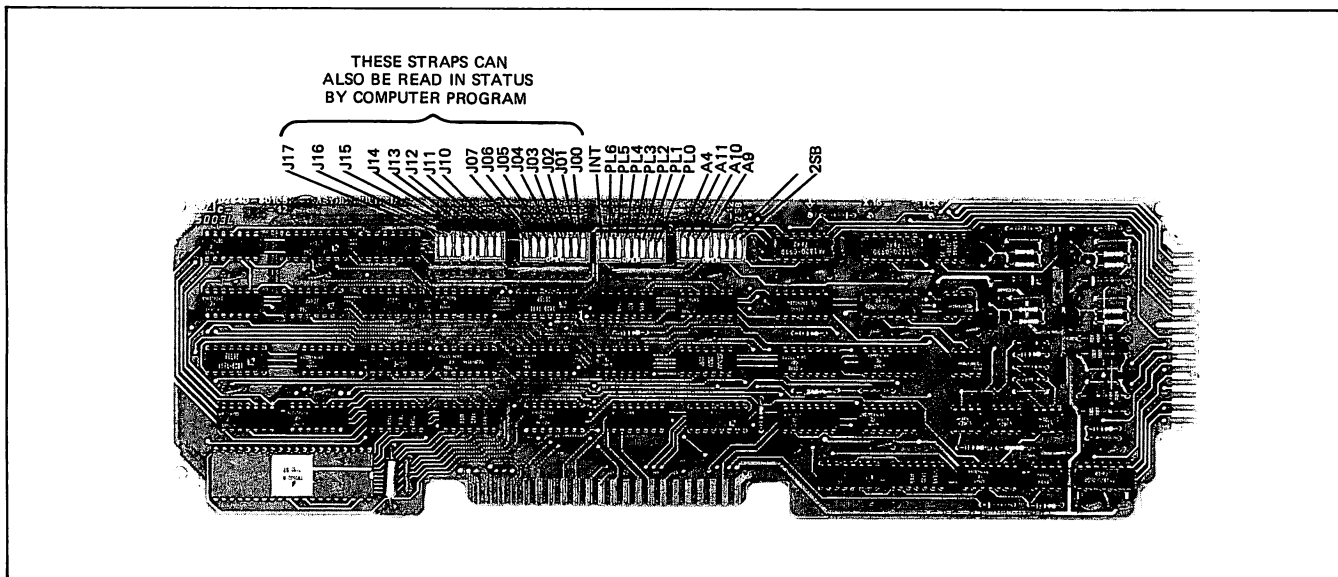


Figure 7-17. Asynchronous Multipoint Communications PCA Strapping Options

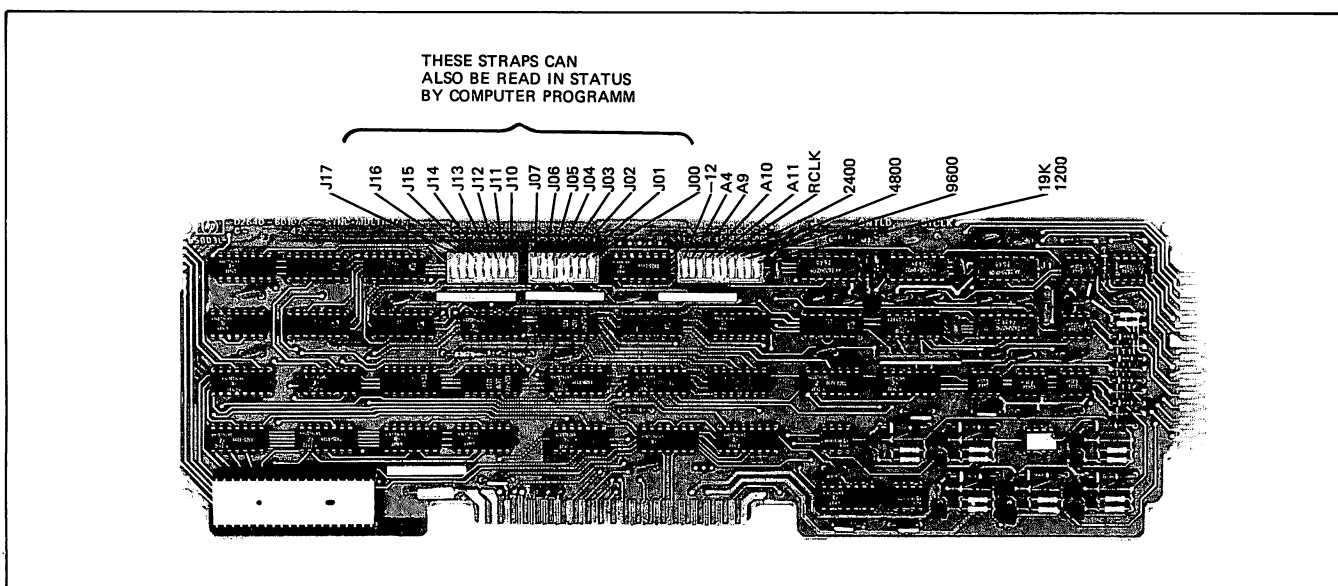


Figure 7-18. Synchronous Multipoint Communications PCA Strapping Options

## DATA COMMUNICATIONS CABLING

### Interface Signals

The RS232 signals available on each of the communication interfaces are listed in table 7-10. 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 7-11 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 7-12.

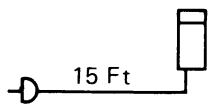
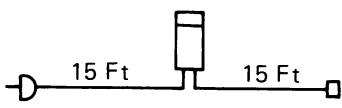
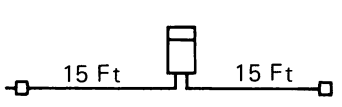
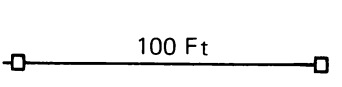

Table 7-10. EIA RS232C and CCITT V24 Interface Data and Control Signals

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
—	—	—	—	—	SBB	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 7-11. 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 7-12. Multipoint Data Communication Cables

ACCESSORY NO.	DESCRIPTION	SYMBOL
13232N	Male RS232C Modem to terminal cable	
13232P	RS232C Modem to terminal plus multipoint	
13232Q	Multipoint terminal to terminal	
13232R	Multipoint extender	
13232T	Power Down protect (same symbol as Q)	
13232U	Modem Bypass	

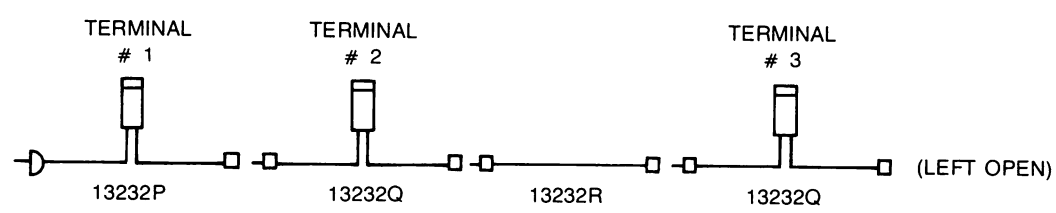
  

where:

-  = RS232 (Modem) connector (female)
-  = Terminal hood connector
-  = Multipoint connector (female)
- = Male connector i.e., —D or —□

For example, to connect 3 terminals you could use the following configuration:



### Point-To-Point Communications Cabling

Figures 7-19 through 7-21 show the cable connections and signals used by the 13260A/B/C/D and 13250A/B accessories.

### Multipoint Communications Cabling

Figures 7-22 and 7-23 show the cable connections and signals used by the 13260C/D accessories in the multipoint configuration.

### Power Down Protect Cabling

Figure 7-24 and 7-25 show the cable configuration and effects of signal switching during power-down.

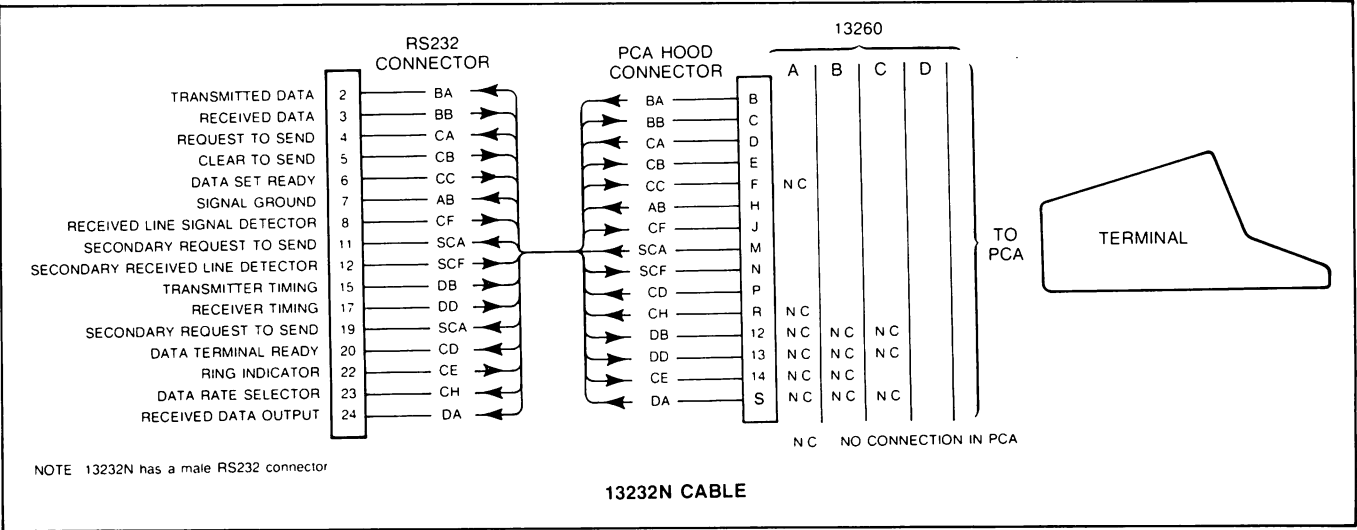


Figure 7-19. Point-to-Point Communications Cabling

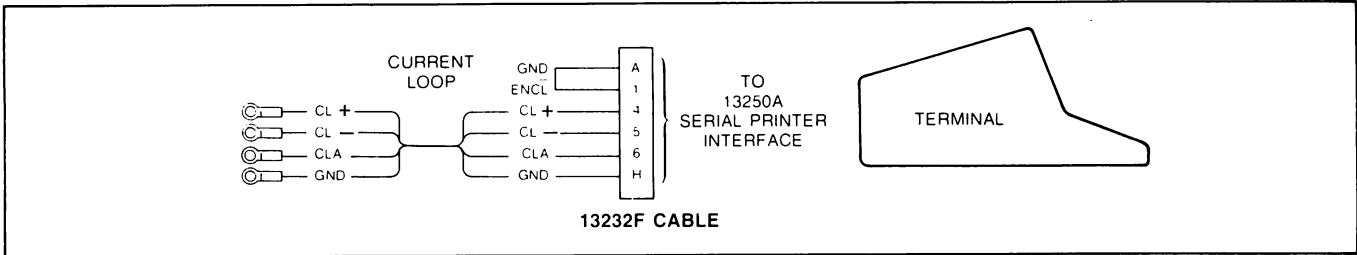


Figure 7-20. Current Loop Cabling

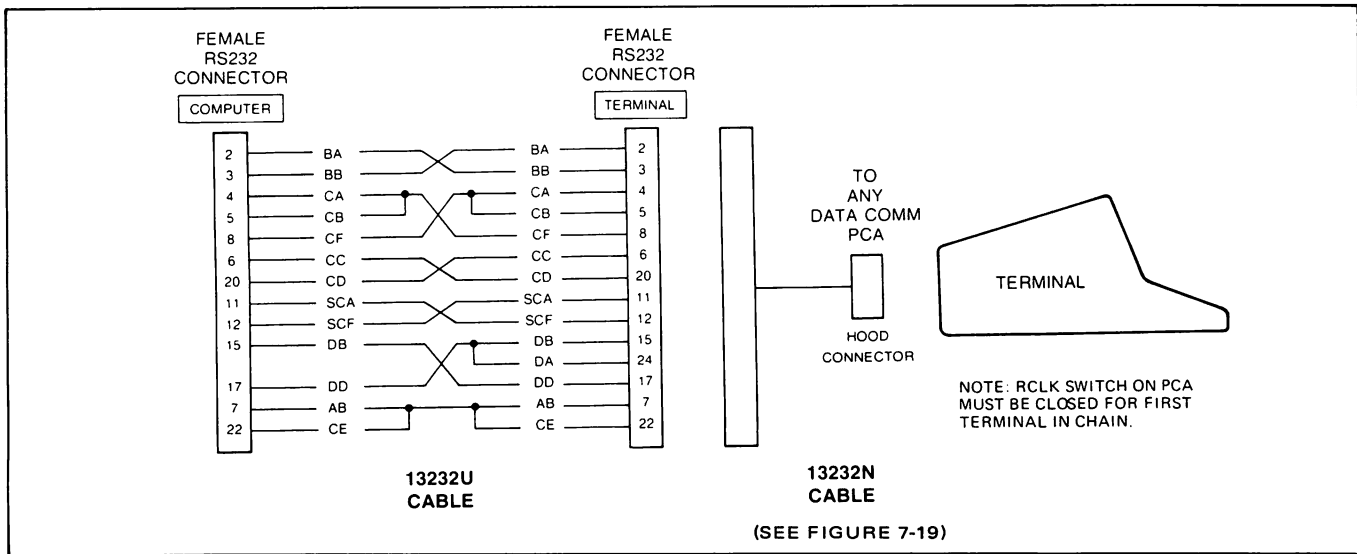


Figure 7-21. Modem By-Pass Cabling

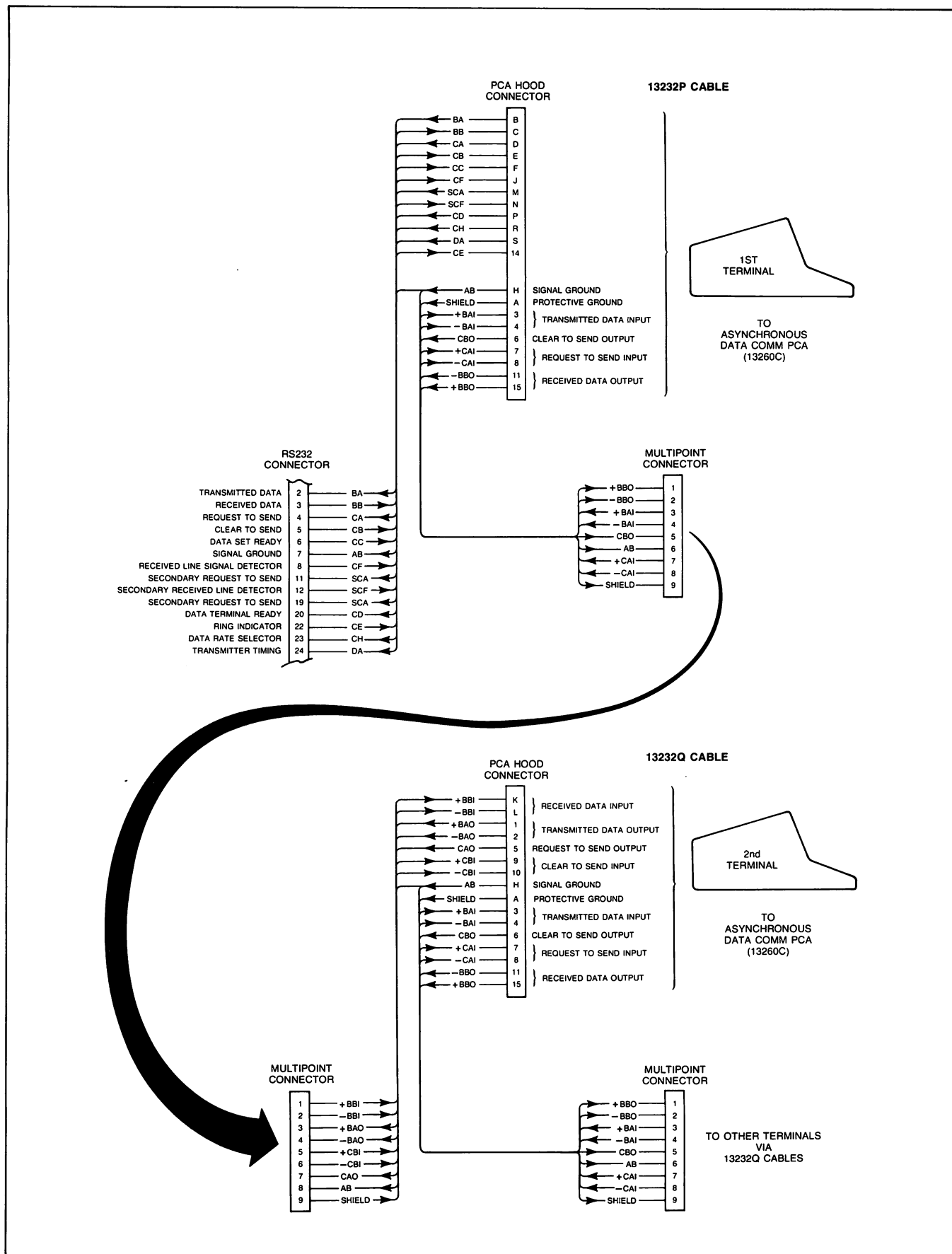


Figure 7-22. Asynchronous Multipoint Cabling

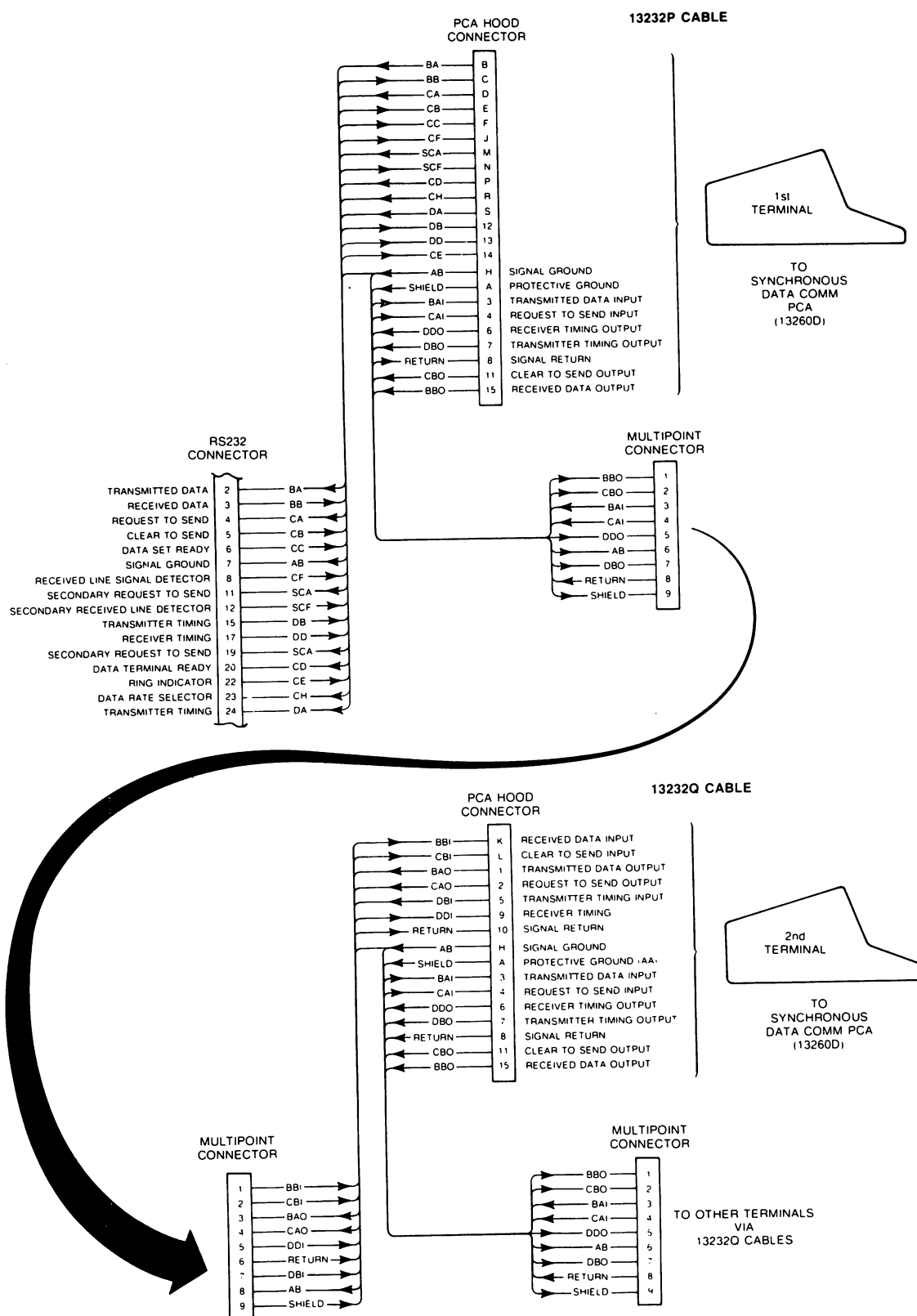
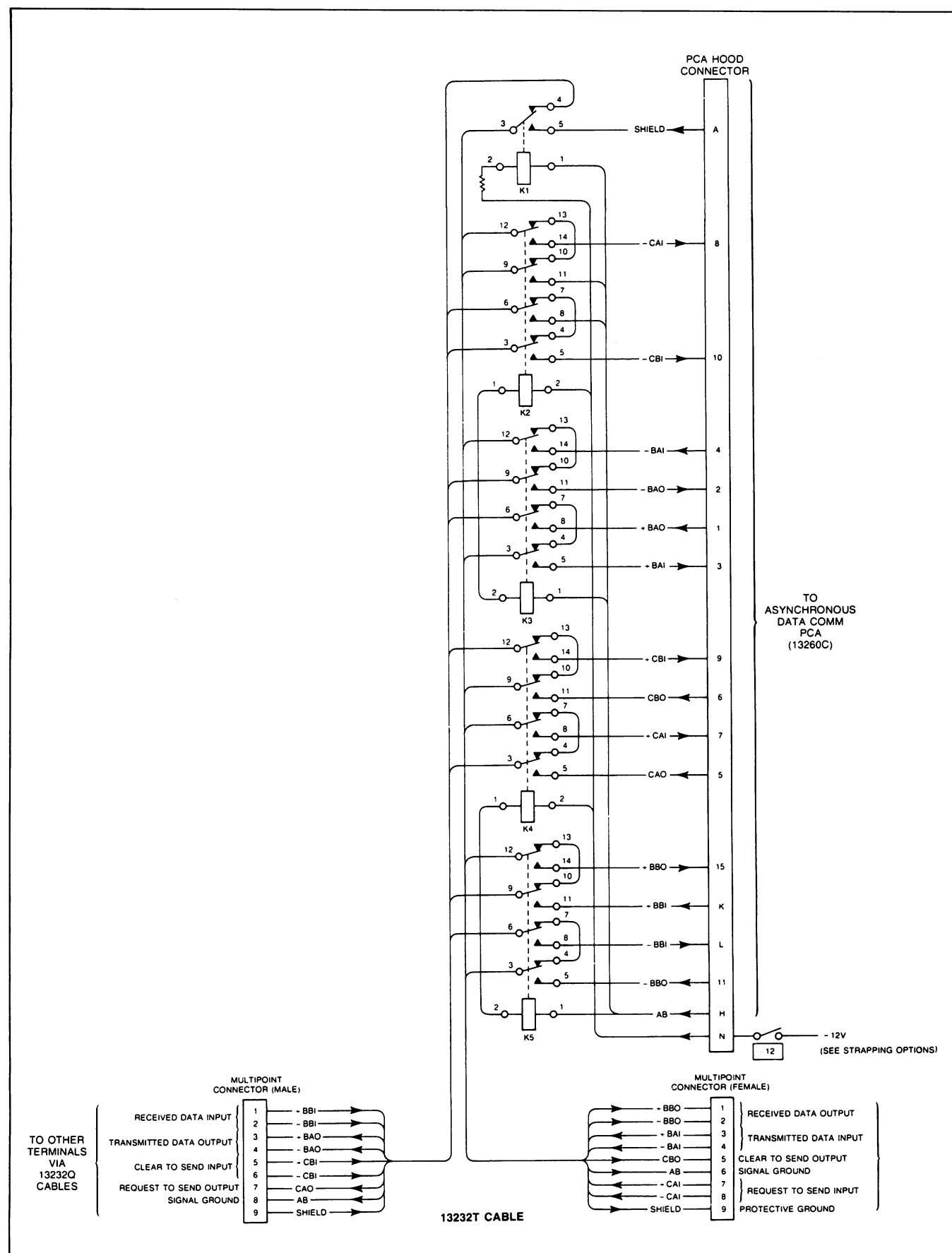


Figure 7-23. Synchronous Multipoint Cabling





**Figure 7-24. Power-Down-Protect Cabling for Asynchronous Multipoint Configuration**

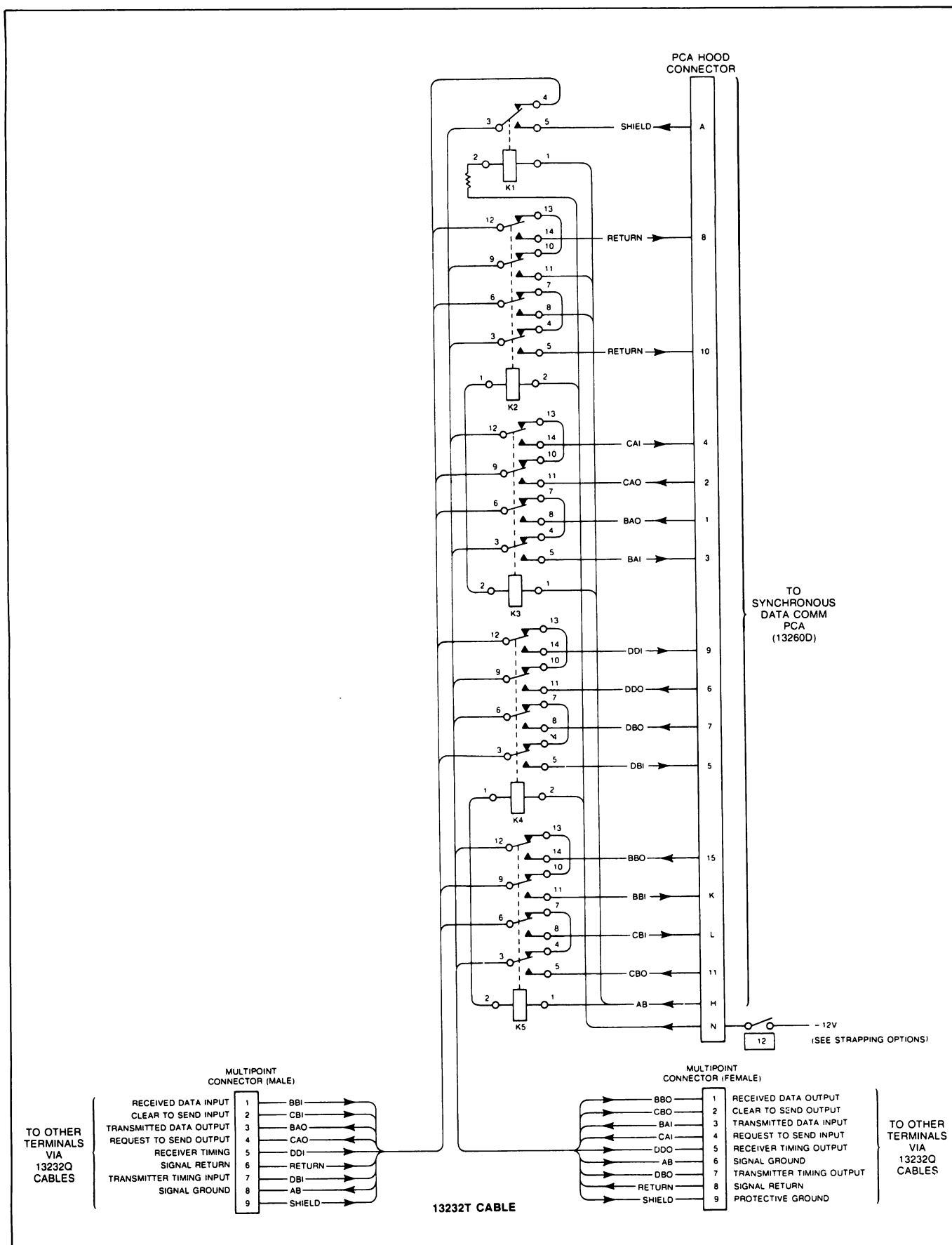


Figure 7-25. Power-Down-Protect Cabling for Synchronous Multipoint Configuration

## 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 7-13.

Figures 7-26 through 7-28 show the details of assembling each type of connector. Table 7-14 lists the interface signals on each of the data communications PCA's. Also, the illustrations of the HP cables (figures 7-19 through 7-25) 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

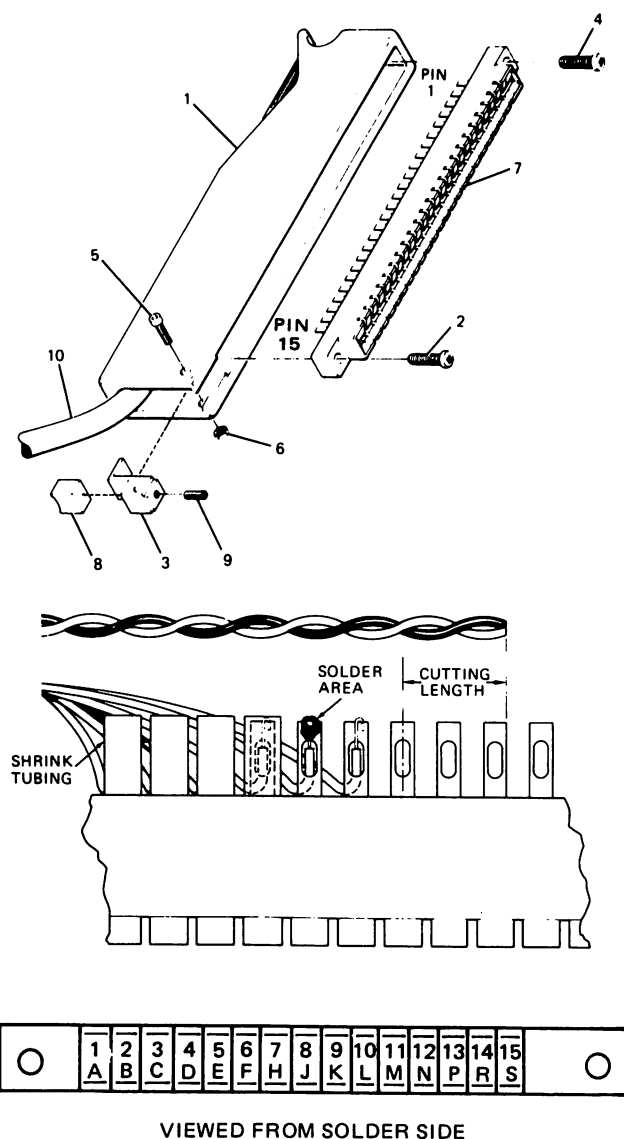
Table 7-13. Parts for Fabricating Your Custom Data Communications Cable

ITEM	HP PART NO.	ALTERNATE SOURCE	DESCRIPTION
RS232 Connector	5061-2405	Brand Rex POSS4P22	(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		22 AWG, 4 twisted pairs, overall shield, 75 ohm differential mode characteristic impedance.
Note: All connectors include contacts.			

Table 7-14. 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

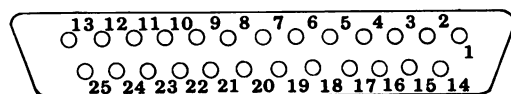
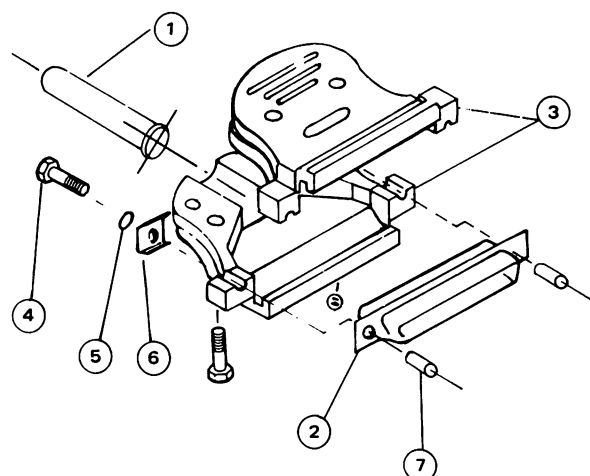
NOTE: Used in current loop mode.



Assembly Procedures:

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 7-26. Assembling the PCA Hood Connector

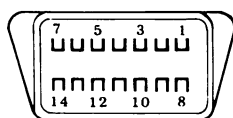
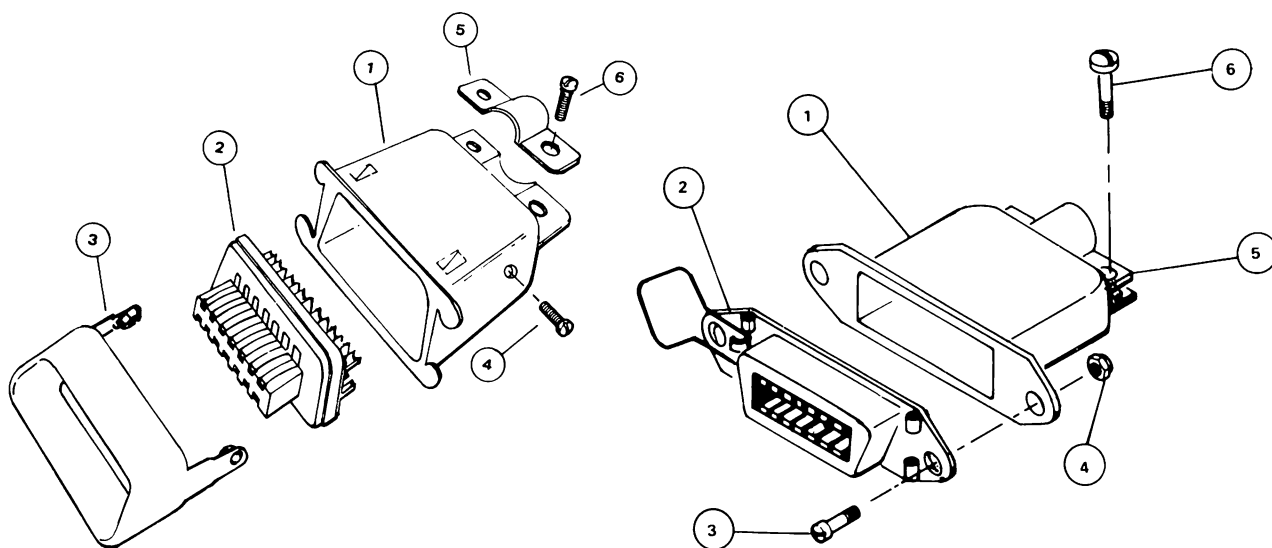


VIEWS FROM SOLDER SIDE

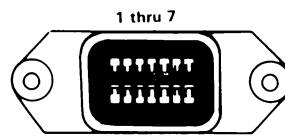
#### Assembly Procedure:

1. ( ) Slide rubber bushing (item 1) over end of cable, leaving about 6 inches of cable end exposed for wire stripping, etc.
2. ( ) Strip back the cable jacket 1-inch.
3. ( ) Clip the 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). (Select either the male or female contact assembly provided for your particular application.)
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.
7. ( ) Assemble the two halves of the connector (item 3) onto the contact assembly (item 2). (Use the screws and nuts provided.)
8. ( ) Mount the two screws, threaded spacers, and other hardware (items 4 thru 7) onto the contact assembly.

Figure 7-27. Assembling the RS232C Connector

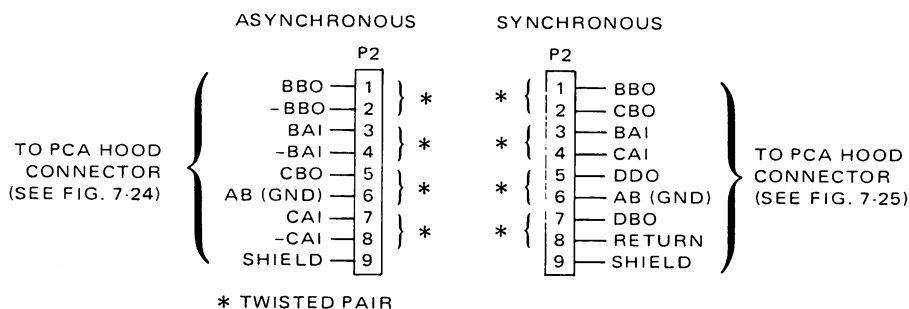


MULTIPOINT CONNECTOR P2  
(VIEWED FROM SOLDER SIDE)



1 thru 7  
8 thru 14  
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 7-28. 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 ensure that the terminal is functioning properly. There are three types of self-test, each testing a specific function of the terminal.

### Basic Self-Test

Pressing **TEST**, checks out the terminal, except for the cartridge tape units (if installed) and the data communications. The following is performed when the **TEST** key is pressed (also see the flowchart in figure 7-30):

#### 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 7-30.) Note that this checksum test is also performed as part of the power-on sequence.
- A RAM TEST appears on the display while checker-board test is performed on the random access memory. An error here causes a RAM ERROR message to be displayed. (See flowchart, figure 7-30).

- 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 vertical and horizontal lines. If the graphics memory test fails, a message indicating the failing graphics memory component is displayed (see the flowchart in figure 7-30).
- The bell is beeped indicating success up to this point. If the **TEST** key is held down to cause the self-test to be repeated, the bell beeps only for the first self-test.
- The entire character set contained in the terminal is displayed.
- If the BASIC Interpreter was active at the time the **TEST** key was pressed, BASIC performs its checksum routine.
- 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 7-29 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.

```

  12345 6789012345 6789012345 0123456789 012345 !"# $%&'()*+ ,-. /0123 456789:; <=>?
  @ABC DEFGHIJK LMNOPQRS TUVWXYZ[ \]^_`abc defghijk lmnopqrs tuvwxyz{ |}~
  @ABCDEFGHIJKLMNO 900=020 4=100000

```

A. Test Pattern for the standard terminal.

```

  <+ + '12345 6789012345 6789012345 0123456789 012345 !"# $%&'()*+ ,-. /0123 456789:; <=>?
  @ABC DEFGHIJK LMNOPQRS TUVWXYZ[ \]^_`abc defghijk lmnopqrs tuvwxyz{ |}~
  @ABCDEFGHIJKLMNO 900=020 4=100000

```

B. Test Pattern for terminals containing Display Enhancements, Math Symbol Set, and Line Drawing Set.

Figure 7-29. Basic Self-Test Patterns



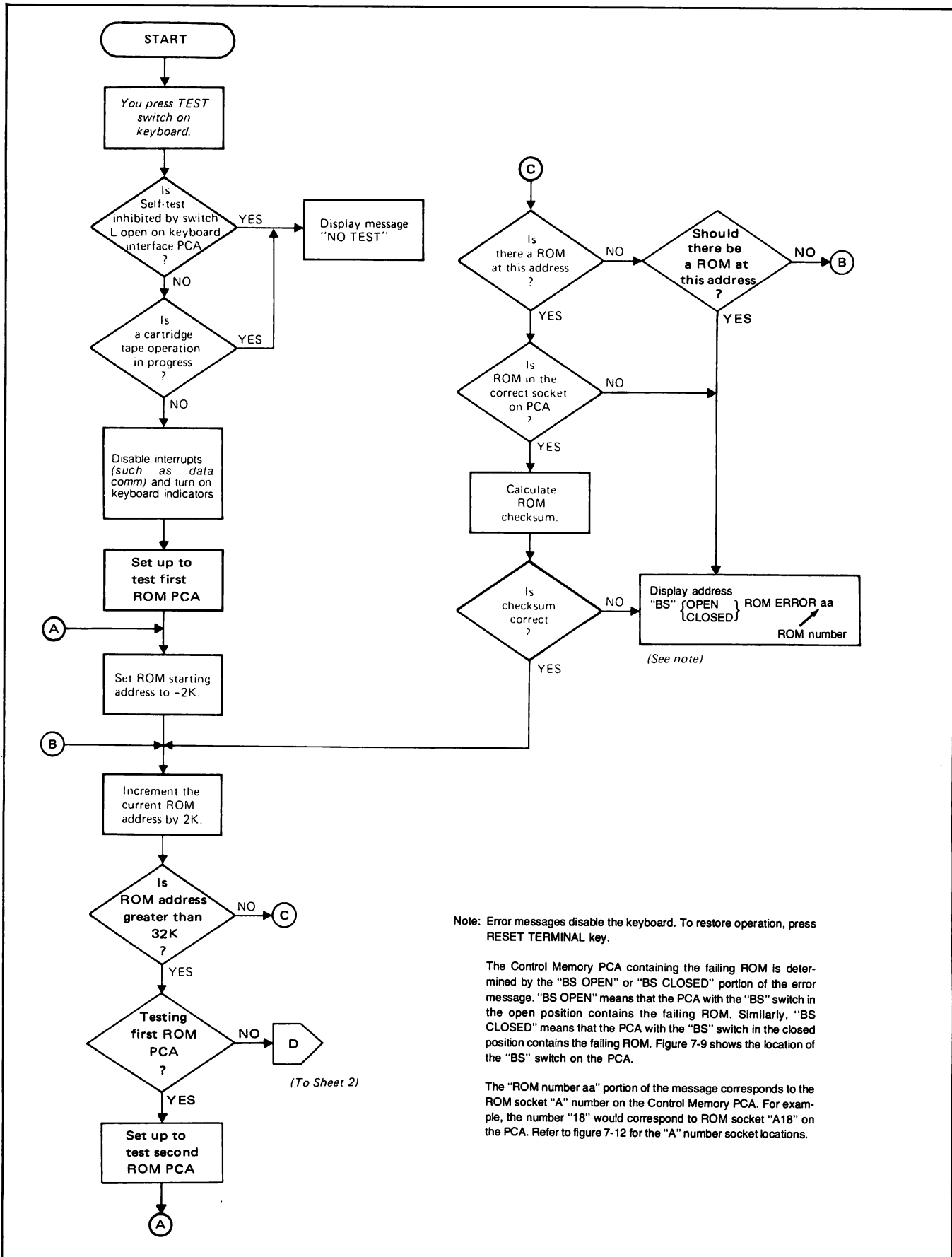
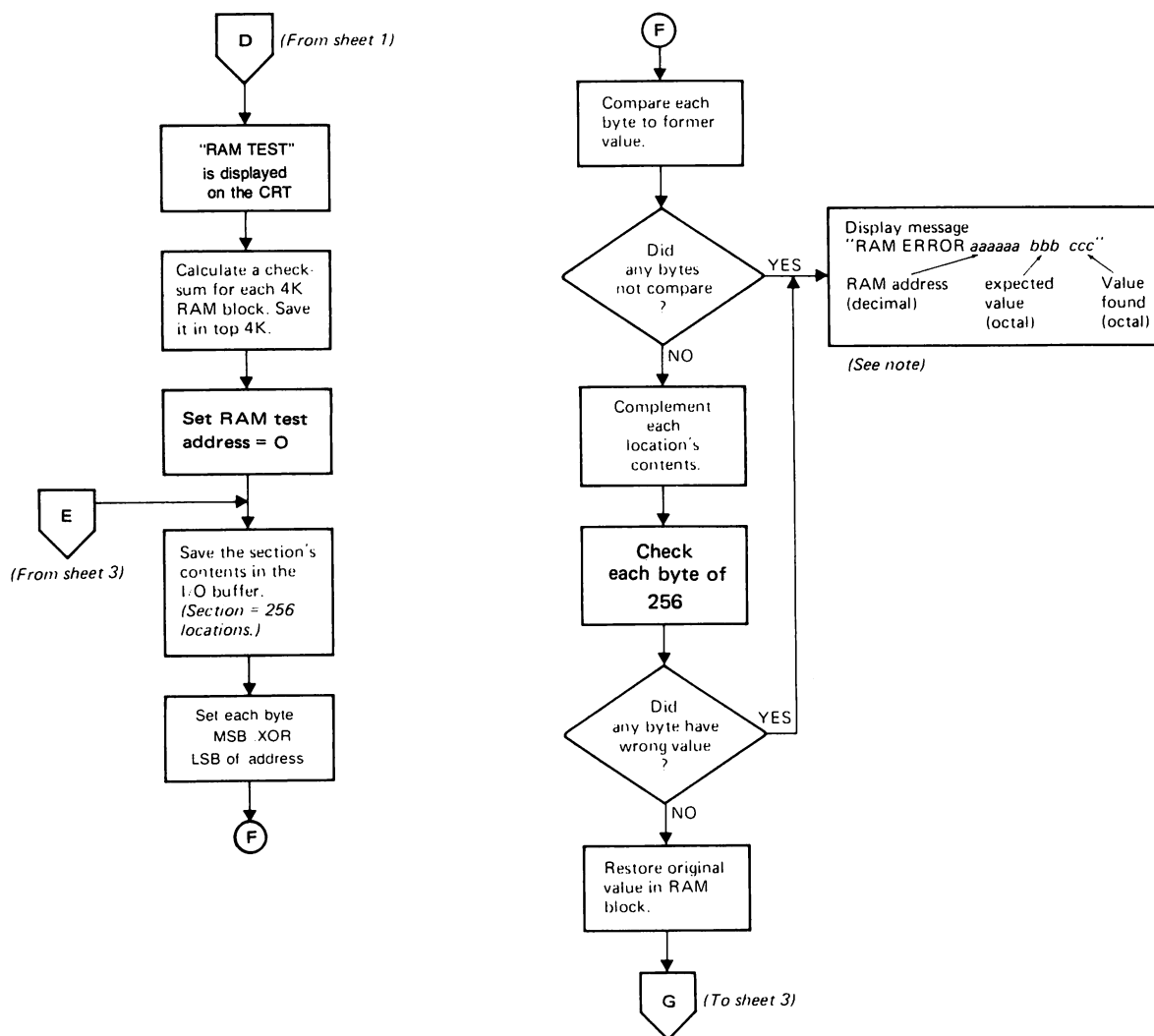


Figure 7-30. Basic Terminal Self-Test Flowchart (Sheet 1 of 3)



Notes: There must be 64K of RAM present.

Error messages disable the keyboard.  
To restore operation, press RESET  
TERMINAL key.

Figure 7-30. Basic Terminal Self-Test Flowchart (Sheet 2 of 3)

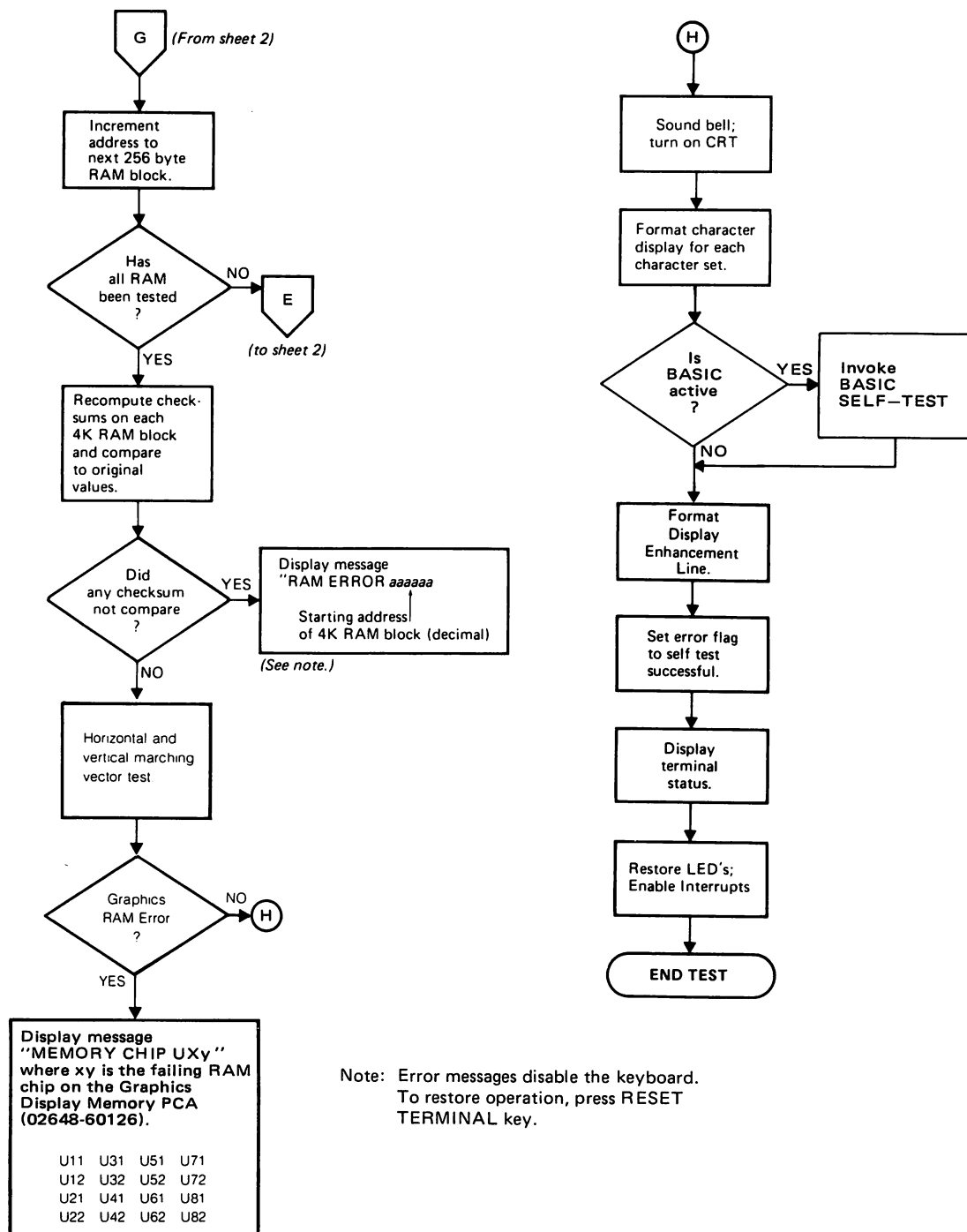


Figure 7-30. Basic Terminal Self-Test Flowchart (Sheet 3 of 3)

## 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 you type TEST TAPES in the command channel (you can also do this using the TEST and TAPES softkeys) and then press **RETURN**:

- 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 7-15. To run the self-test, follow the instructions in table 7-16 or 7-17, whichever is applicable. Flowcharts of the self-test are contained in figures 7-31 and 7-32.

Table 7-15. Data Communications Self-Test Connectors




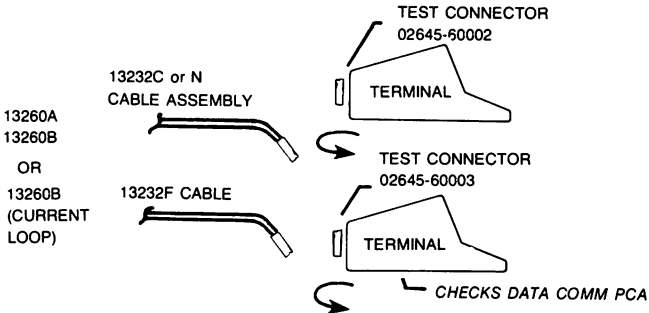
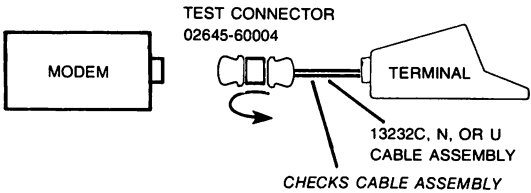
SELF-TEST CONNECTORS	HP PART NO.	USED FOR
	02645-60002	Checks RS232 circuits on 13260A, B, C, D accessory PCA's. (Does not check multipoint circuits on PCA; use 02645-60004 test connector below.) This connector is supplied with the terminal.
	02645-60003	Checks current loop circuits on 13260B accessory 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 7-16. 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, type TEST DATA COMM in the command channel (you can also do this using the TEST and DATA COMM softkeys), and then press <b>RETURN</b>.</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, type TEST DATA COMM in the command channel (you can also do this using the TEST and DATA COMM softkeys), and then press <b>RETURN</b>.</li> <li>Refer to data comm self-test flowcharts for diagnosing possible error messages.</li> </ol>	

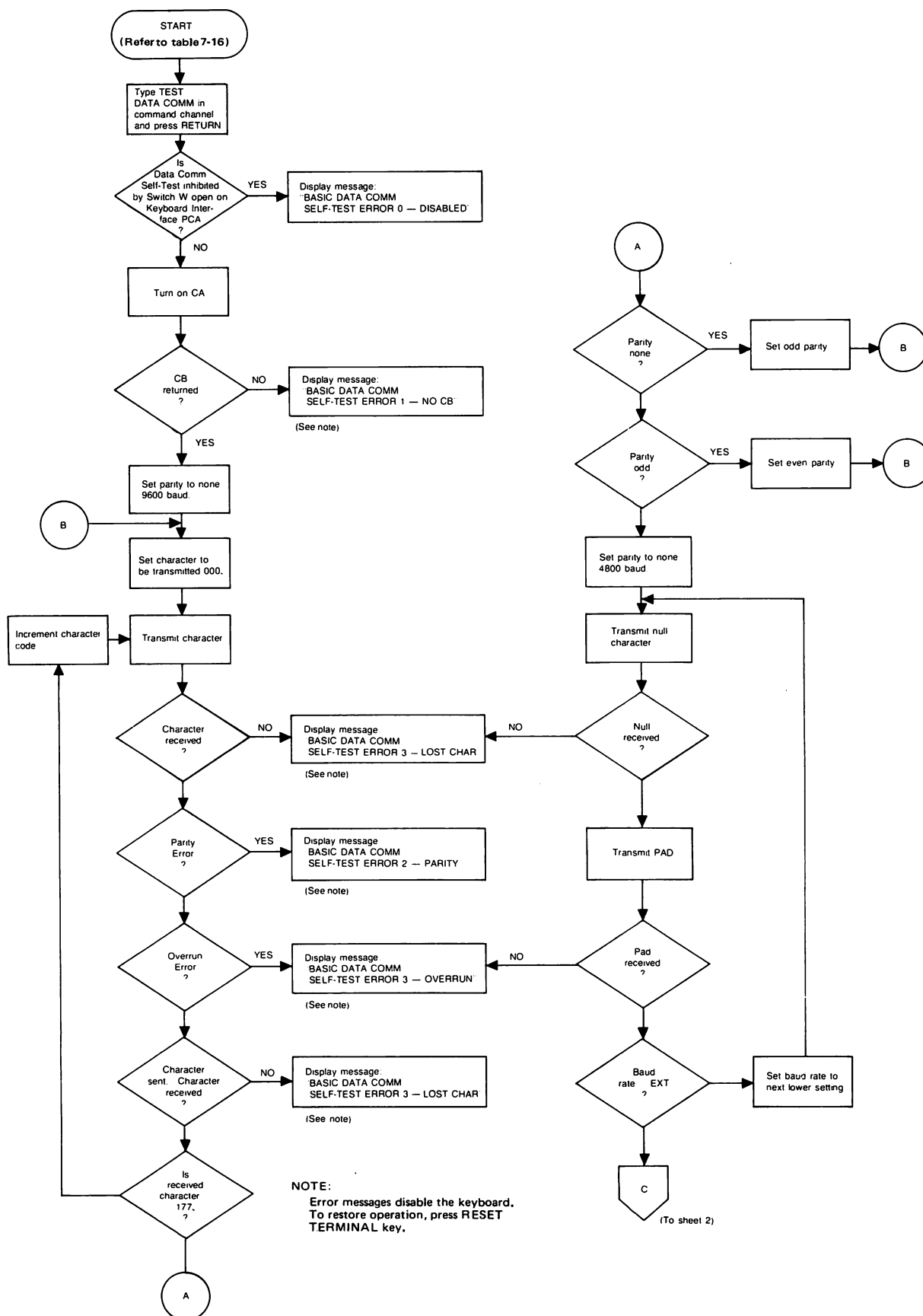


Figure 7-31. Basic Data Comm Self-Test Flowchart (Sheet 1 of 2)

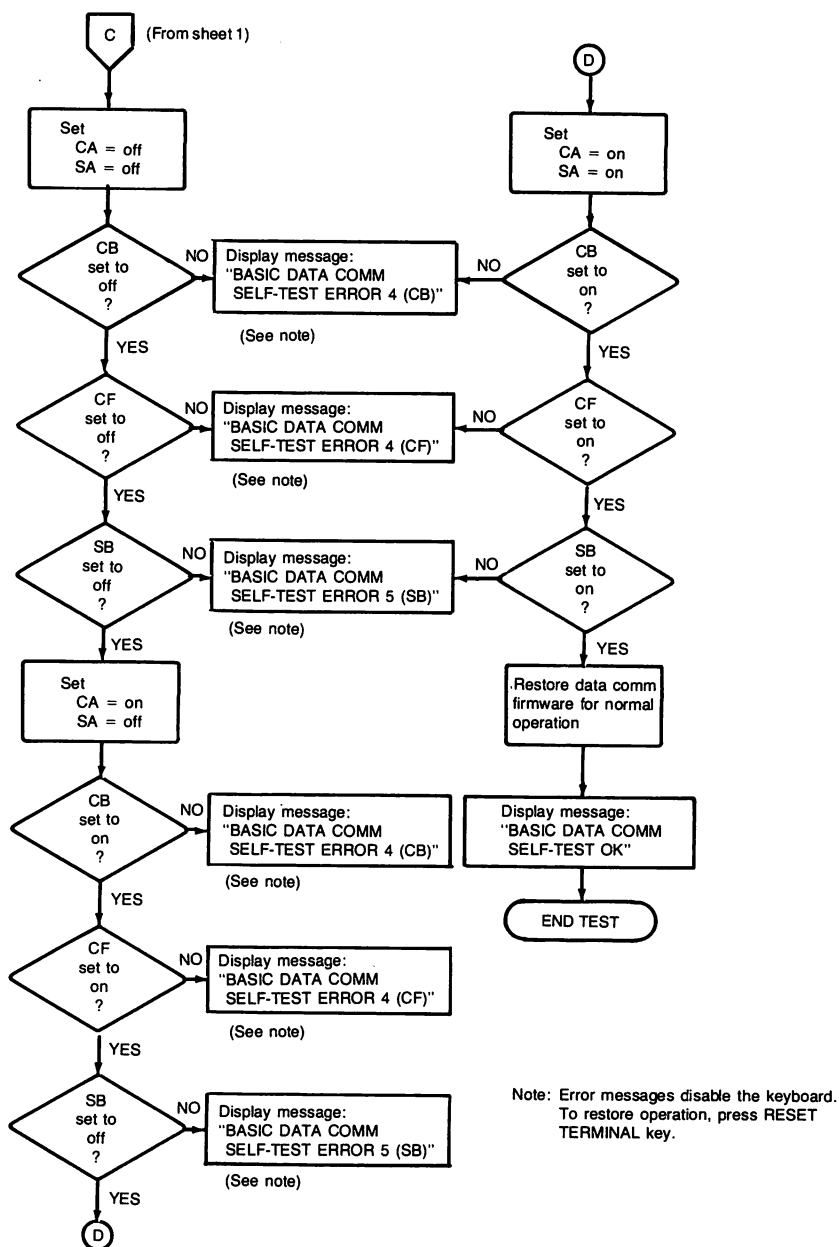
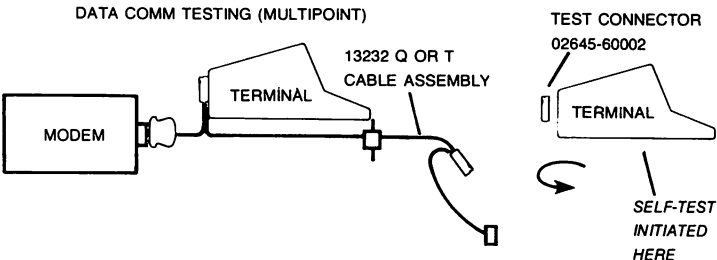
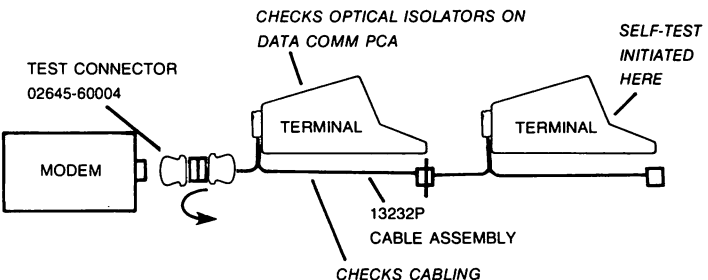
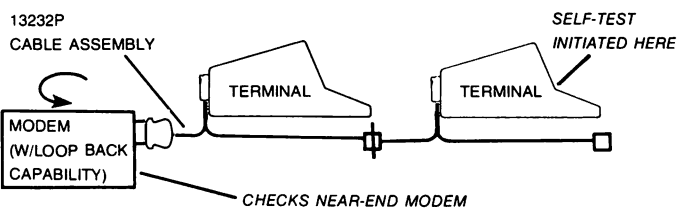
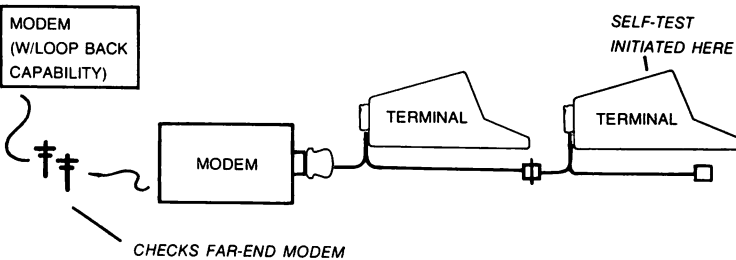


Figure 7-31. Basic Data Comm Self-Test Flowchart (Sheet 2 of 2)

Table 7-17. 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, type TEST DATACOMM in the command channel (you can also do this using the TEST and DATACOMM softkeys), and then press <b>RETURN</b>.</li> <li>Refer to multipoint data comm self-test flowchart for diagnosing possible error messages.</li> </ol>	<p><b>DATA COMM TESTING (MULTIPOINT)</b></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, type TEST DATACOMM in the command channel (you can also do this using the TEST and DATACOMM softkeys), and then press <b>RETURN</b>.</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, type TEST DATACOMM in the command channel (you can also do this using the TEST and DATACOMM softkeys), and then press <b>RETURN</b>.</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, type TEST DATACOMM in the command channel (you can also do this using the TEST and DATACOMM softkeys), and then press <b>RETURN</b>.</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>NOTE:</b> For terminals without cartridge tapes, use ESC x to perform data comm self-test.</p>	



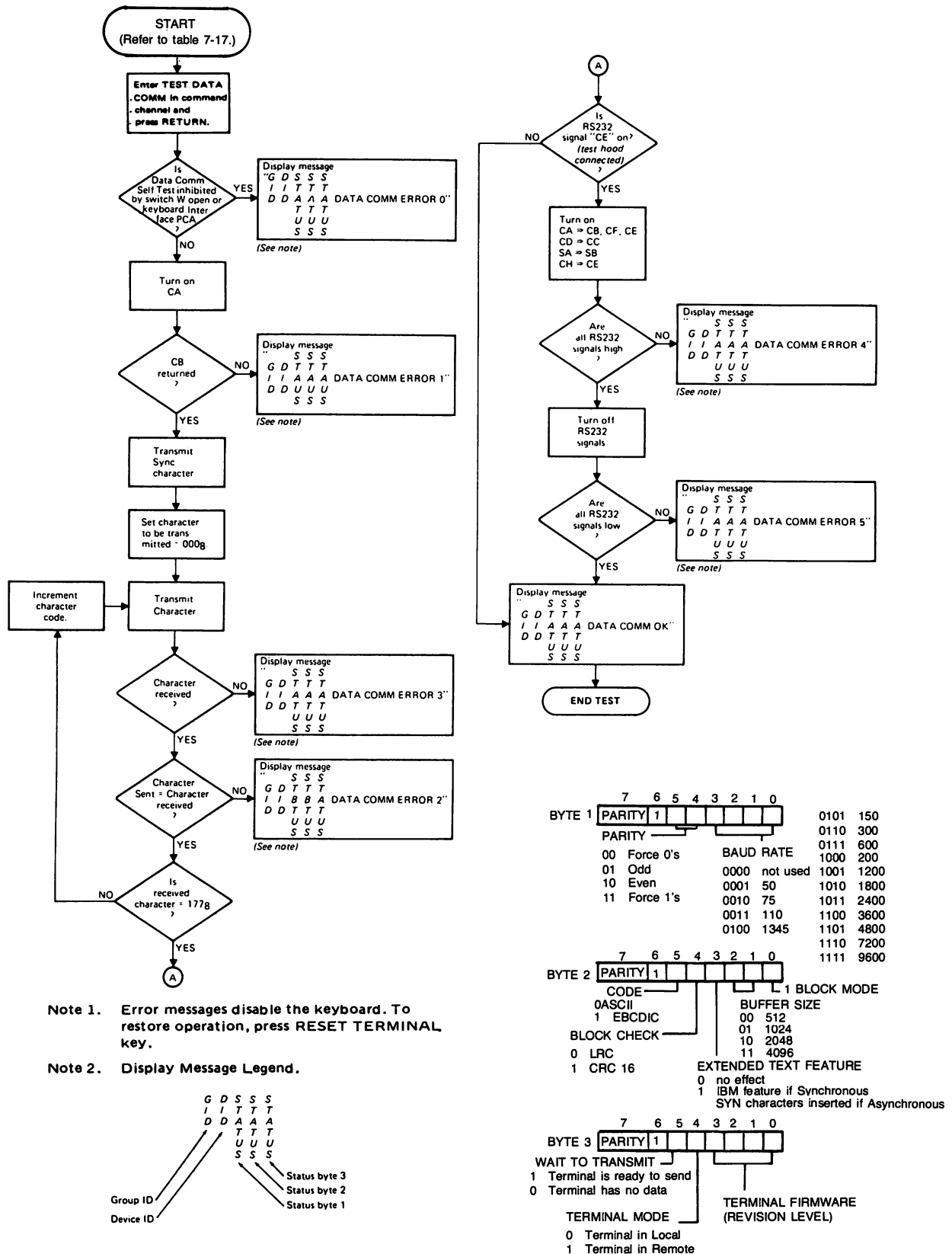


Figure 7-32. Multipoint Data Comm Self-Test Flowchart

## HP-IB Self-Test

The HP-IB self-test tests the PHI IC chip. To initiate the test, type **TEST HP-IB** in the command channel (you can also do this using the **TEST** and **HP-IB** softkeys) and then press **RETURN**.

If the HP-IB test fails, the following message is displayed:

```
HP-IB TEST FAIL, ERROR=x
```

where **x** is a number from 0 to 2 whose meaning is as follows:

0 indicates no HP-IB interface PCA is installed (or the PCA, if present, is strapped incorrectly).

1 or 2 indicates a PHI IC chip error.

If the test is successful, the following message is displayed:

```
HP-IB TEST OK, ADDR=xx, SYSCTL= YES, CIC= YES
                          NO      NO
```

The above message indicates the HP-IB PCA's current strapping and functional configuration. **xx** is an address, from 8 to 29, which uniquely identifies your terminal's HP-IB PCA and differentiates it from all other HP-IB devices connected to the HP-IB bus. The **SYSCTL** parameter indicates whether or not your terminal is configured as the System Controller of the HP-IB. The **CIC** parameter indicates whether or not your terminal is currently the Controller in Charge (CIC) of the HP-IB bus. **CIC=YES** merely means that your terminal was the terminal which most recently issued a command over the HP-IB.

In order for the HP-IB self-test to be performed, your terminal must be:

1. Both the System Controller and the CIC; or
2. Neither the System Controller nor the CIC.

If neither of the above is true (that is, if **SYSCTL=NO, CIC=YES** or **SYSCTL=YES, CIC=NO**) then the test cannot be performed and the following message is displayed:

```
HP-IB TEST NOT ATTEMPTED, ADDR=xx, SYSCTL= YES, CIC= YES
                                      NO      NO
```

In such a case, it is most convenient to manipulate the **CIC** function (manipulating whether or not your terminal is the System Controller requires reconfiguring the entire HP-IB configuration). To make your terminal the **CIC**, type **SHOW TIME** in the command channel and then press **RETURN**. To make it so that your terminal is not the **CIC**, type **SHOW TIME** in the command channel (followed by **RETURN**) of another terminal in the configuration. Then retry the HP-IB self-test.

## Terminal-to-Terminal Loop-Back Test

To test the communication capabilities between two terminals in an HP 13296A Shared Peripheral Interface configuration, enter **TEST TERMINAL#x** in the command channel, where **x** is the HP-IB device address of the target terminal, and then press **RETURN**. This initiates a loop-back test in which a 256-byte data block is sent from your terminal to the target terminal and is then sent back to your terminal for comparison.

If the test fails, then the following message is displayed:

```
HP-IB TE#x TO TE#y TEST FAILED, ERROR=z
```

where: **x** is the HP-IB device address of your terminal.

**y** is the HP-IB device address of the target terminal.

**z** is the error number, as follows:

- 1 indicates no HP-IB interface PCA
- 2 indicates no control of HP-IB
- 3 indicates target terminal did not receive entire data block
- 4 indicates target terminal did not send back entire data block
- 5 indicates returned data block did not match the one originally transmitted



## 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*.

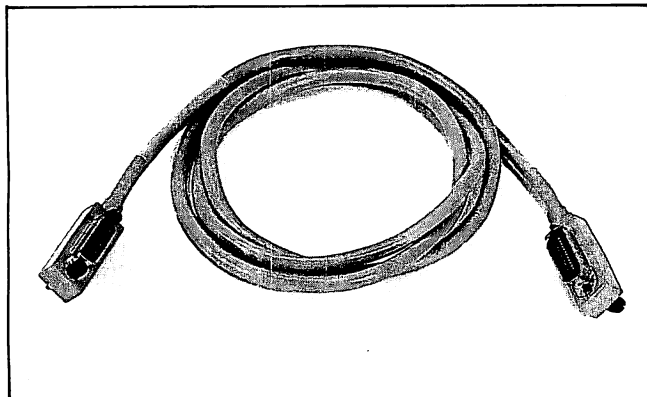


Figure 8-1. HP-IB Interconnecting Cable

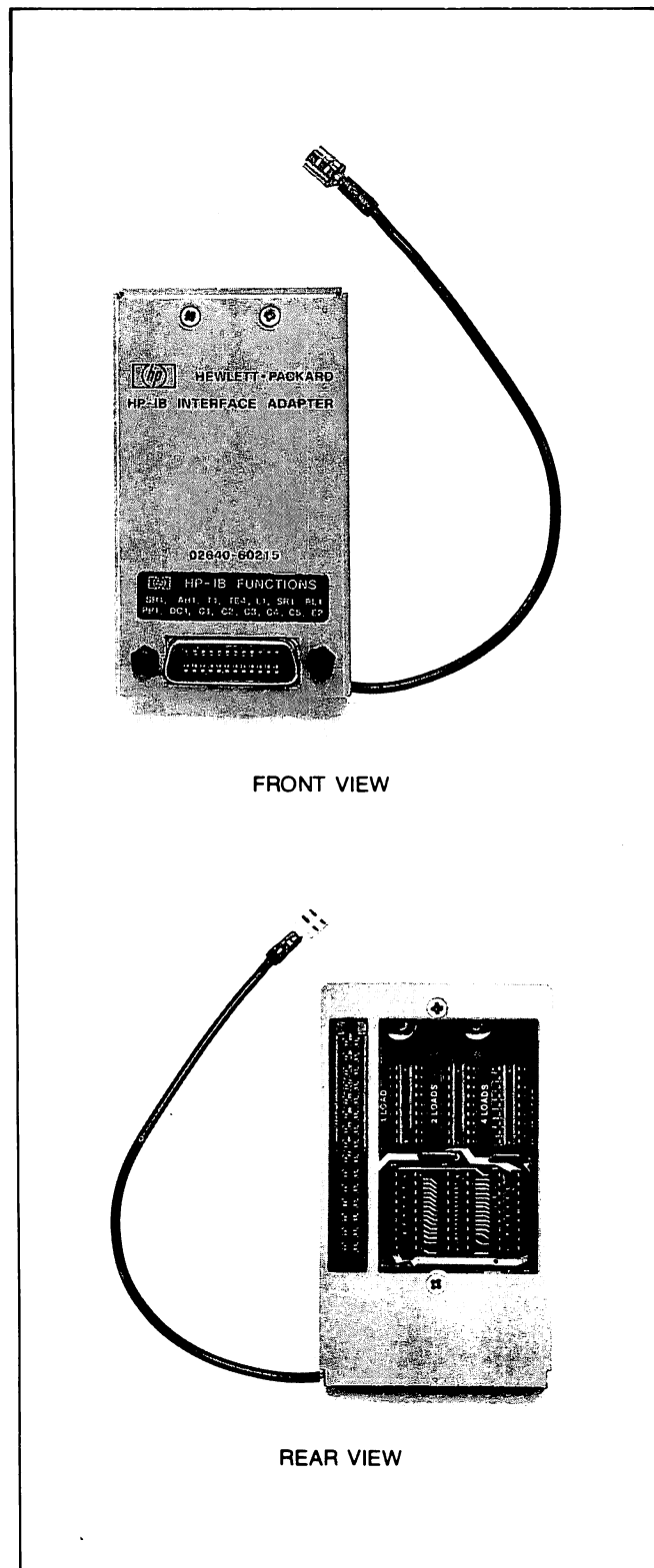


Figure 8-2. HP-IB Interface Adapter

## CABLING CONSIDERATIONS

Your HP-IB configuration is limited to a maximum combined amount of interconnecting cable that averages two meters of cable per device. 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. This adapter should be used for selecting additional device loads only when connected to the System Controller terminal; the adapters for all other terminals should be set to "0".

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.

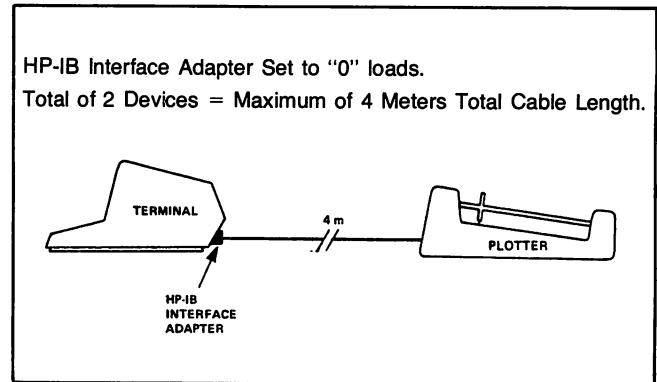
### CAUTION

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:

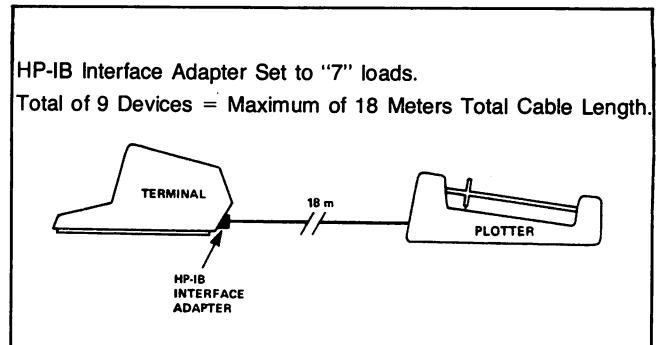
1810-0408 (1 LOAD)  
1810-0410 (2 LOADS)  
1810-0409 (4 LOADS)

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 only for the System Controller terminal. On all other terminals in the configuration it should be left unconnected. 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 9872A Plotter 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.

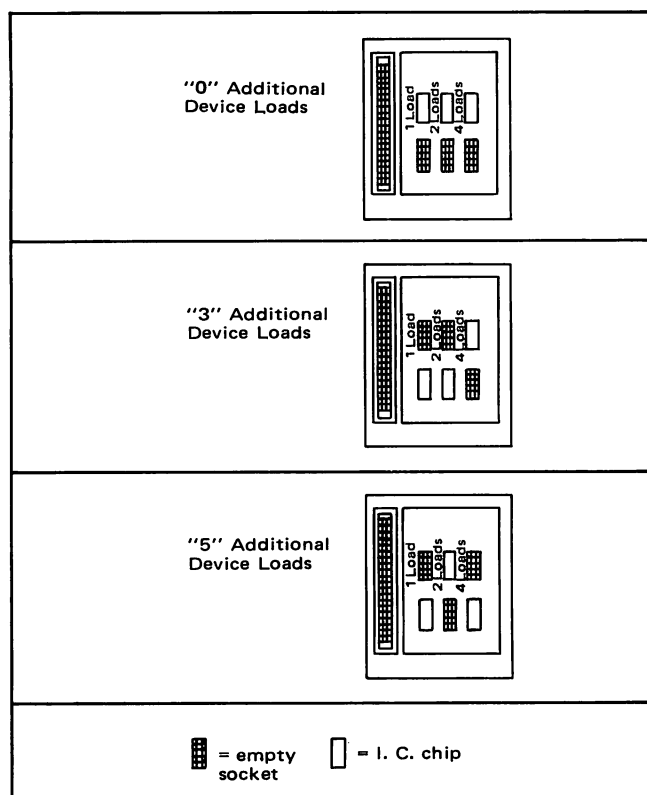


Figure 8-3. Selecting Additional Device Loads

## 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 Shared Peripheral 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

Any terminal in the configuration can be designated as the System Controller. This is accomplished by setting switch SC on the terminal's HP-IB Interface PCA to the "open" position. Only one terminal, however, should be assigned that role. An HP-IB configuration cannot function if it includes two or more System Controllers.

At any given time one of the terminals in the configuration is designated as the "Controller in Charge" (or CIC). Note that this is completely independent of the System Controller function. It merely means that the particular

terminal was the one which most recently issued a command over the HP-IB. Whenever a terminal is currently the CIC its yellow LED (above the **f4** key) is lit.

If power is removed from the terminal which is currently designated as the CIC, the entire configuration becomes inoperative. To recover from such an event, you must do a hard reset on the System Controller terminal (note that the terminal which was powered off need not be turned back on unless it also happens to be the System Controller terminal).

If power is removed from any terminal which is not currently designated as the CIC, the configuration is unaffected.

### CAUTION

Doing a hard reset on the CIC halts any operation currently in progress. To resume operation you must then do a hard reset on the System Controller terminal.

Doing a hard reset on the System Controller terminal reconfigures the HP-IB configuration.

## Local HP-IB Testing

You can test the HP-IB Interface of any terminal in the configuration locally at any time by entering the following in the command channel of the particular terminal:

**\*TEST HP-IB %**

where \* is the command channel prompt.

This test and its resultant messages are described fully at the end of Section VII, *Installation*, of this manual.

Of particular interest at this point is that you can use this test to find out if your terminal is the System Controller and/or currently the CIC (the message displayed if this test is successful tells you this information as well as your terminal's HP-IB device address).

## Remote HP-IB Testing

When a configuration includes multiple terminals, you can test the cable connections by performing terminal loop-back tests. With this test, data is sent from one terminal to another and the data is then sent back for comparison. You invoke this test by issuing the following command:

**\*TEST TERMINAL #n**

where \* is the command channel prompt and n is the address of the remote terminal with which the cable connection is to be tested.

This test and its resultant message are described fully at the end of Section VII, *Installation*, of this manual.

## DEVICE CONSIDERATIONS

When using the HP 13296A Shared Peripheral Interface, your overall configuration may include up to ten devices. The HP 13296A supports the following five peripheral devices:

- HP 2631A/G-046 Printer
- HP 9871A-001 Printer
- HP 7245A-001 Plotter/Printer
- HP 9872A Plotter
- HP 7225A Plotter

Note that 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 (see Cabling Limitations, above).

## SAMPLE CONFIGURATIONS

Figure 8-4 illustrates an HP 13296A Shared Peripheral Interface configuration containing only two devices: an HP 2647A Intelligent 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	These switch settings specify the terminal's HP-IB device address as 29.
	B1:	Closed	
	B2:	Open	
	B3:	Open	
MSB	B4:	Open	This switch setting indicates that the terminal is the System Controller.
	SC:	Open	

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 specify the plotter as the output device for plots

produced by an AGL program by including the following statement at the start of your AGL program:

PLOTTR (5,1)

where the "5" specifies the plotter's HP-IB device address.

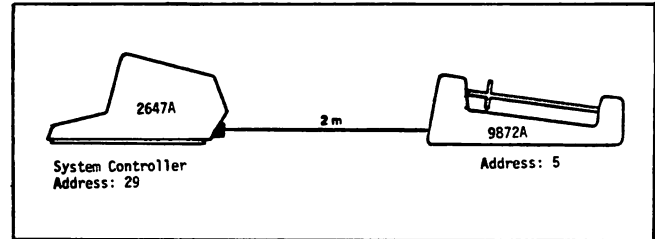


Figure 8-4. Terminal-to-Plotter Configuration

Figure 8-5 illustrates an HP 13296A Shared Peripheral Interface configuration containing four devices: two HP 2647A Intelligent Graphics Terminals, one HP 7245A Plotter/Printer, and one HP 2631A/G Printer.

The terminals are assigned HP-IB device addresses 29 and 28, the latter also being designated as the System Controller. To configure the terminals in that manner, switches B4 through B0 and SC on their HP-IB Interface PCAs are set as follows:

LSB	B0:	Open	These switch settings specify the terminal's HP-IB device address as 29.
	B1:	Closed	
	B2:	Open	
	B3:	Open	
MSB	B4:	Open	This switch setting indicates that the terminal is not the System Controller.
	SC:	Closed	
LSB	B0:	Closed	These switch settings specify the terminal's HP-IB device address as 28.
	B1:	Closed	
	B2:	Open	
	B3:	Open	
MSB	B4:	Open	This switch setting indicates that the terminal is the System Controller.
	SC:	Open	

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 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 four 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 11 meters of HP-IB interconnecting cable in our sample configuration.

Once the configuration is established as described above, you may specify the HP 7245A Plotter/Printer as the output device for plots produced by an AGL program by including the following statement at the start of your AGL program:

**PLOTR (5,1)**

where the "5" specifies the HP 7245A plotter function HP-IB device address.

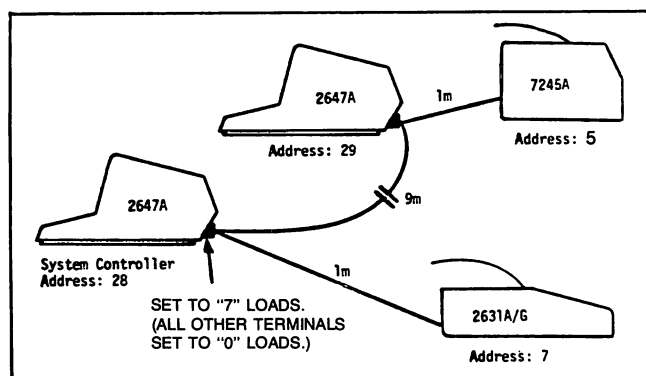


Figure 8-5. Shared Plotter/Line Printer Configuration

You may also copy the content of the terminal's graphics display memory to the HP 7245A printer function by entering the following in the terminal's command channel and then pressing **RETURN**:

**TRANSFER ALL FROM GRAPHICS TO HP-IB#6**

Note that if the HP 2631 Printer is the G version, you may do the same with that device by specifying HP-IB#7 in the above command.

You may also copy the content of the terminal's alphanumeric display memory to either of the printers by entering the following in the terminal's command channel and then pressing **RETURN**:

**COPY ALL FROM DISPLAY TO HP-IB#n**

where n is the HP-IB device address of the target printer (6 or 7 in our sample configuration).

You may also copy the content of one terminal's alphanumeric display memory to that of the other terminal by entering the following in the command channel and then pressing **RETURN**:

**COPY ALL FROM DISPLAY TO HP-IB#n**

where n is the HP-IB device address of the target terminal.

Figures 8-6 and 8-7 illustrate larger HP 13296A Shared Peripheral Interface configurations. The commands for interacting between your HP 2647A Intelligent Graphics Terminal and the other devices in the configuration are as described in the previous two examples.

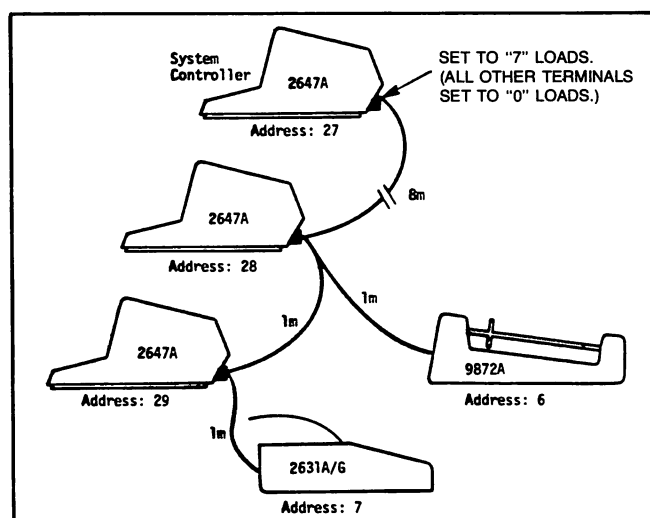


Figure 8-6. 3-Terminal, Shared Plotter/Printer Configuration

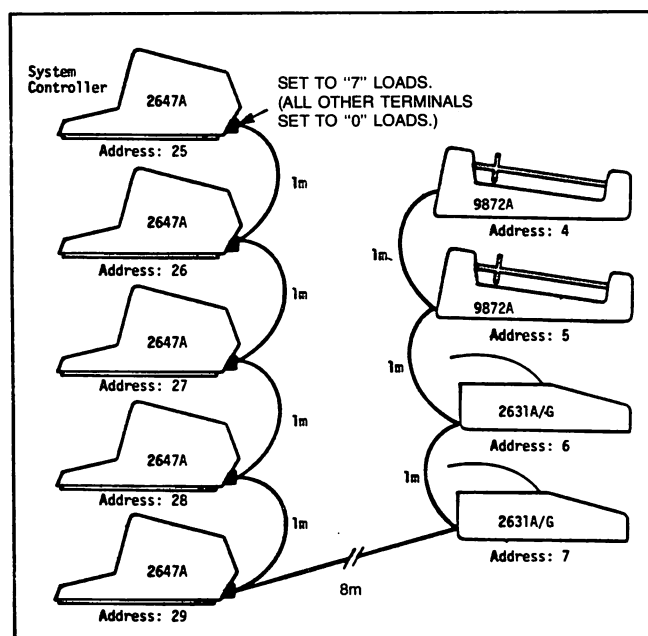


Figure 8-7. 5-Terminal, Multiple Plotter/Printer Configuration





## INTRODUCTION

This appendix contains sample applications using the terminal's unique features. They can be used as a guide in developing your own applications.

## 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					<u>d</u>	<u>d</u>	<u>A</u>	<u>A</u>		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															
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	1	^										T	C	A
N	N	N	X	C														X	C	D

The terminal returns an ACK1 after the next turnaround

S	S	S	S	D															
Y	Y	Y	Y	L															
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	B
Y	Y	Y	Y	T														S	C
N	N	N	N	X	DA	C/P	021	R	F	S	X	C							

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															
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.

## Applications

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.

## DIGITIZER PROGRAM EXAMPLE

Table A-1 gives a sample program to drive an HP 9874A Digitizer.

Table A-1. HP 9874A Digitizer Driver Using Rubber Band Line

```

1000 REM *****
1010 REM *
1020 REM * 9874 DIGITIZER DRIVER USING 2647A RUBBER BAND LINE
1030 REM *
1040 REM * SETUP: 9874 MUST USE HP-IB ADDRESS 15
1050 REM *
1060 REM * OPERATION: MOVING DIGITIZER CURSOR WILL MOVE 2647 CURSOR
1070 REM * PRESSING DIGITIZE SWITCH WILL:
1080 REM * MOVE 2647 PEN TO CURSOR POSITION
1090 REM * PRESSING DIGITIZE SWITCH IN DRAW MODE WILL
1100 REM * DRAW A LINE FROM PEN POSITION TO CURSOR
1110 REM * POSITION AND THEN CHANGE PEN TO CURSOR
1120 REM * POSITION
1130 REM * SPECIAL FUNCTION KEYS:
1140 REM * fa = MOVE MODE
1150 REM * fb = DRAW MODE
1160 REM * fc = STOP PROGRAM
1170 REM * fd,fe = NOT USED
1180 REM * WORKS IN BOTH SINGLE AND CONTINUOUS MODES
1190 REM *
1200 REM *****
1210 REM
1220 INTEGER X,Y,Z,A,St,Pn
1230 ASSIGN "H#15" TO #1
1240 REM * INITIALIZE 9874, SET SINGLE MODE, SET AIR TOGGLE MODE,
1250 REM * BEEP AND TURN ON fa LED
1260 PRINT #1;"IN;SG;AT;BP;SK 1"
1270 REM * SET PEN CONTROL FOR MOVE MODE
1280 Pn=-2
1290 REM * INITIALIZE PLOTTING DEVICE (2647)
1300 PLOTR
1310 REM * SET ASPECT RATIO AND SCALE TO ACCOMMODATE 9874
1320 SETAR (1.29)
1330 SCALE (0,17400,0,13500)
1340 REM * CLEAR GRAPHICS MEMORY, TURN ON GRAPHICS CURSOR AND
1350 REM * RUBBER BAND LINE
1360 PRINT CHR$(27)&"*dfakM"
1370 REM * REQUEST DIGITIZER STATUS AND READ IT
1380 PRINT #1;"OS"
1390 READ #1;St
1400 REM * HAS A KEY BEEN PRESSED?
1410 IF St AND 128 THEN 1540
1420 REM * IS A DATA POINT READY?
1430 IF St AND 4 THEN 1650
1440 REM * REQUEST 9874 CURSOR POSITION AND READ IT
1450 PRINT #1;"OC"
1460 READ #1;X,Y,Z,A
1470 REM * PRINT X AND Y VALUES ON 9874 DISPLAY
1480 PRINT #1 USING 1710;X,Y
1490 REM * MOVE 2647 CURSOR TO CORRESPOND TO 9874 CURSOR
1500 POINT (X,Y)
1510 REM * LOOK AT STATUS AGAIN
1520 GOTO 1380
1530 REM * REQUEST KEY CODE AND READ IT
1540 PRINT #1;"OK"
1550 READ #1;Ky
1560 REM * IF fa KEY CHANGE TO MOVE MODE
1570 IF Ky AND 1 THEN Pn=-2
1580 REM * IF fb KEY CHANGE TO DRAW MODE
1590 IF Ky AND 2 THEN Pn=-1
1600 REM * IF fc KEY STOP PROGRAM
1610 IF Ky AND 4 THEN STOP
1620 REM * GO UPDATE CURSOR
1630 GOTO 1450
1640 REM * REQUEST DATA POINT AND READ IT
1650 PRINT #1;"OD"
1660 READ #1;X,Y,Z,A
1670 REM * MOVE PEN OR DRAW LINE
1680 PLOT (X,Y,Pn)
1690 REM * GO UPDATE CURSOR
1700 GOTO 1450
1710 IMAGE "LB",XX,DDDDD,X,DDDDD,XX
1720 END

```



# REFERENCE TABLES

APPENDIX

B

## INTRODUCTION

This appendix contains the following reference information for each model terminal covered by this manual:

- List of Specifications
- Programmer's Reference Table
- Character Code Chart
- List of Options and Accessories

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 dots X 360 rows (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 upper and lower case (alphanumeric)

Cursor: Blinking-underline (alphanumeric) Crosshair (graphics)

Display Modes: White on Black; Black on White (Inverse video).  
Optional half-bright, underline and blinking.

Refresh Rate: 60 Hz or 50 Hz (2647-015 for 50 Hz)

Tube Phosphor: P4

Implosion Protection: Bonded implosion panel

Memory:

- Alphanumeric: 75 lines of 80 characters maximum (less enhancements)
- Graphics: 720 dots by 360 rows of displayable points
- Basic Workspace: approximately 500 BASIC statements in a maximum of 16K bytes BASIC workspace, actual number of lines depends on program complexity.

Option Slots: 1 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; 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 1200 baud may require nulls or handshake protocol to insure data integrity. Basic language control in the 2647A requires handshaking protocol to a host CPU. External clocking requires a TTL signal 16X bps.

Shared peripheral support limited to a maximum of 4 devices at a maximum of one meter between each device. Compatible with the following devices: 2631A-046, 9872A, 7245-001. Maximum of three printers.

Maximum baud rate of on-line BASIC programs utilizing multipoint will typically be lower than point to point protocol.

Vector Drawing Time (9600 baud, typical):  
7ms — half screen  
10ms — 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 turnaround 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

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: -10 to 60°C (-15 to +140°F)  
Operating: 5 to 40°C (+41 to +104°F)

Humidity: 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 testing 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, 11 ms, 1/2 sine

**PHYSICAL SPECIFICATIONS**

Display Monitor Weight: 21.9 kg (48 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) (648 mmD (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 50/60 Hz (±0.2%)  
230 (+10% -23%) at 50 Hz (±0.2%)

Power Consumption: 115W to 150W max

**PRODUCT SAFETY**

Product meets:

- UL Requirements for: EDP equipment, office appliances, teaching equipment
- CSA Requirements for: EDP equipment
- SEV Switzerland
- FTZ (RFI) Germany
- FTZ (DATA COMM) Germany
- GPO (DATA COMM) U.K.

U.L. and CSA labels are applied to equipment shipped to the U.S. and Canada.

SEV, FTZ (RFI), FTZ (DATACOMM). Labels are applied to equipment shipped to Europe. (Units with Option 015)

**PRODUCT SUPPORT****HP SYSTEMS SUPPORT**

Refer to appropriate HP system data sheet for use and support of 2647A in these systems. When this product is used in a customer-assembled system, the overall operational responsibility of the system rests with the customer.

**INSTALLATION**

All product preparation can be performed by the owner/user. Refer to reference manual supplied with unit for detailed instructions. HP assistance is provided for installation upon request and at prevailing rates.

**HARDWARE SUPPLIED**

2647A Intelligent Graphics Terminal  
2 Blank Tape Cartridges  
1 System Tape Cartridge

**DOCUMENTATION SUPPLIED**

2647A User's Manual (02647-90001)  
2647A Reference Manual (02647-90002)  
2647A Terminal BASIC Manual (02647-90005)

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>c</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)	Insert succeeding inputs at cursor
	CR (M <sup>c</sup> )	Move cursor to left margin		ESC R (off)	
	LF (J <sup>c</sup> )	Move cursor down one line		ESC N (on)	Character Wraparound Mode. Insert succeeding inputs at cursor with wraparound to next line.
	HT (I <sup>c</sup> )	Forward cursor to next tab position	<b>TERMINAL CONTROL GROUP</b>		
	ESC i	Back tab	ESC	[ <sup>c</sup>	Leads off an ASCII escape sequence
	ESC 1	Set tab at the current cursor column		---	Used to generate ASCII control codes and alternate key functions
	ESC 2	Clear the tab at the current cursor column		ESC&k0C(off)	Upper-case alphabetical lock
	ESC 3	Clear all tabs		ESC&k1C(on)	
	ESC 4	Set left margin		ESC l (on)	Memory overflow protect; display lock
	ESC 5	Set right margin		ESC m (off)	
	ESC J	Clear memory from cursor position to end of memory		ESC&k0A(off)	Line Feed with each terminal carriage return
	ESC K	Clear line from the cursor to end of line		ESC&k1A(on)	
	ESC S	Scroll the display up one line		ESC&k0B(off)	Block Mode: data displayed but not transmitted until requested; otherwise, terminal is in Character Mode and data transmitted as typed
	ESC T	Scroll the display down one line		ESC&k1B(on)	
	ESC U	Display the next 24 lines of memory		---	Enables block transfers
	ESC V	Display the previous 24 lines of memory		---	Transmits BREAK signal to interrupt computer
	---	Display the next display-workspace		---	Data link exists
	ESC & d	Turn on display enhance		ESC Y (on)	Control functions disabled and displayed
	ESC [	Start an unprotected field		ESC Z (off)	
	ESC ]	End an unprotected field or transmit-only field		ESC y (on)	Monitor Mode: display all codes received from data comm lines
	ESC W (on)	Turn format mode on. Only unprotected fields can be modified.		ESC Z (off)	
	ESC X (off)			ESC g	(First press): frees the keyboard and clears I/O operations
	ESC {	Start transmit-only field		ESC E	(Second press): sets the terminal to power-on state
				ESC z	Terminal Self-Test (no tape test)



**Table B-2. Programmer's Reference Table (Continued)**

KEY	CODE	FUNCTION
----	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 (C <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

Example: Cursor to 12th row 35th column (+, - for relative addressing)  
 E & a 12r 35c

ESC & b HP diagnostics ONLY

or

ESC & c  
<parameters>

ESC & d      Turn on display enhancement

where: enhancements = @ through O

Example: Select half-bright, blinking, and underline.  
E & d M

[illegible]

KEY	CODE	FUNCTION
---	ESC & f <parameters>	Define soft keys
where:		
	0 (normal)	
<parameters> = {0-8}	k 1 (local only)    a {0-8}d {1-80}L	<label string> <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.	
	♣ & f 1 k 2 a 6d 13 L LOG-ON HELLO-MYCANT ♣	
---	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 <i>Reference Manual</i> )
---	ESC & k <parameters>	Define latching keys

where:

<b>&lt;parameter&gt; =</b>	<b>0 (up)</b>	<b>a (Auto LF)</b>
	<b>1 (down)</b>	<b>b (Block Mode)</b>
		<b>c (Caps Lock)</b>
		<b>r (Remote)</b>

Example: Block Mode up Remote up Auto LF down Caps Lock down

E t & k 1 a o b 1 c o R

READ	— — — —	In REMOTE, transfers data from source device to computer. In LOCAL, transfers one file from source device to DISPLAY.
RECORD	— — — —	In REMOTE, transfers data from computer to destination device. In LOCAL, transfers one file from DISPLAY to destination device.
GRAPH	— — — —	In REMOTE, enables block transfer. In LOCAL, operates same as RECORD.
— — — —	ESC & s <parameters>	Define strap settings.

Table B-2. Programmer's Reference Table (Continued)

KEY	CODE	FUNCTION	KEY	CODE	FUNCTION
GRAPHICS CONTROL SEQUENCES			VECTOR DRAWING MODE		
ESC * <control sequence>			ESC * m <parameters>		
b = Raster dump				<mode> a	Select drawing mode (0-4)*
d = Display control				<line type> b	Select line type (1-11)**
l = Graphics text label				<pattern> c	Define line pattern (1 byte)
m = Mode control				<pattern> d	Define area shading pattern (8 bytes)
p = Plot control				<x1,y1,x2,y2> e	Fill area, absolute
r = Raster dump				<x1,y1,x2,y2> f	Fill area, relocatable
s = Status				<x,y> j	Set relocatable origin
t = Compatibility mode				k	Set relocatable origin to current pen position
				l	Set relocatable origin to graphics cursor position
DISPLAY CONTROL				<size> m	Set graphics text size (1-8)
ESC * d <parameters>				<rotation> n *	Set graphics text orientation (1-4)
SHIFT CLEAR	a	Clear graphics memory	SHIFT F SIZE	o	Turn on text slant
	b	Set graphics memory	SHIFT F ANG	p	Turn off text slant
SHIFT G DSP	c	Turn on graphics display	SHIFT F ANG	<0-9> q	Set graphics text origin
SHIFT G DSP	d	Turn off graphics display	SHIFT F ANG	r	Set graphics defaults
SHIFT A DSP	e	Turn on alphanumeric display		z	NOP
SHIFT A DSP	f	Turn off alphanumeric display		* 0 (no effect), 1 (clear), 2 (set), 3 (complement), 4 (jam)	
ZOOM	g	Turn on zoom		** 1 (solid line)      5 (line #2)      9 (line #6)	
ZOOM	h	Turn off zoom		2 (user line pattern)      6 (line #3)      10 (line #7)	
ZOOM IN or ZOOM OUT	<size> i	Set zoom size (1-16)		3 (user area pattern)      7 (line #4)      11 (point plot)	
	<x,y> j	Set zoom position		4 (line #1)      8 (line #5)	
G CURSOR	k	Turn on graphics cursor	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		
G CURSOR	l	Turn off graphics cursor	PLOTTING COMMANDS		
SHIFT RBLN	m	Turn on rubber band line	ESC * p <parameters>		
SHIFT RBLN	n	Turn off rubber band line			
	<x,y> o	Move graphics cursor absolute	a Lift the pen		
←, ↑, →, ↓	<x,y> p	Move graphics cursor incremental	b Lower the pen		
	q	Turn on alphanumeric cursor	c Use graphics cursor as new point		
	r	Turn off alphanumeric cursor	d Draw a point at the current pen position and lift the pen		
SHIFT TEXT	s	Turn on graphics text mode	e Set relocatable origin to the current pen position		
STOP	t	Turn off graphics text mode	f Data is ASCII absolute		
	z	NOP	g Data is ASCII incremental		
GRAPHICS LABEL			h Data is ASCII relocatable		
ESC * l <text label> <%r, %t, %f, or %s>			i Data is absolute		
Example: Send the text "X=TIME, Y=TEMP"			j Data is short incremental		
ESC * l X=TIME, Y=TEMP %t			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		

Table B-2. Programmer's Reference Table (Continued)

KEY	CODE	FUNCTION	KEY	CODE	FUNCTION
<b>RASTER DUMP</b>			<b>COMPATIBILITY MODE</b>		
	ESC * r <parameter>			ESC * t <parameter>	
---	a	Start transfer	---	<0/1/2> a	Set graphics input terminator (0=CR, 1=CR EOT, 2=none)
---	b	End transfer	---	<0/1> b	Set Page Full Break strap (0=out, 1=in)
---	c	Erase screen	---	<0/1> c	Set Page Full Busy strap (0=out, 1=in)
---	d	Turn on video		z	NOP
---	e	Return raster status	Keyboard Interface switches:		
---	i	Set all parameters to default values.	P open = Scaled compatibility mode.		
---	j	Return raster size status ("720,360")	Q open = Unscaled compatibility mode.		
---	k	Return model number ("2647A")	Example: Select a CR input terminator and set the Page Full Busy strap.		
---	<no. of dots> m	Window horizontal address	ESC * t 0a 1C		
---	<no. of dots> m	Window vertical address			
---	<no. of dots> p	Window horizontal dimension			
---	<no. of dots> q	Window vertical dimension			
---	<no. of dots> s	Horizontal size of picture			
---	<no. of dots> t	Vertical size of picture			
---	<no. of dots> x	X offset			
---	<no. of dots> y	Y offset			
---	z	NOP			
<b>GRAPHICS STATUS</b>					
	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					

Table B-2. Programmer's Reference Table (Continued)

## DEVICE CONTROL

NOTE 1: Uppercase characters in the command syntax are acceptable abbreviations (see "Abbreviated Command Sequence"). Characters in braces "{}" are optional.

NOTE 2: Shaded escape sequences are compatible with HP 2641A, HP 2645A and HP 2648A terminals; however, it is recommended that the ESC,c command sequences be used.

NOTE 3: When using the ESC & p format, the <x> and <y> values refer to source device and destination device numbers:

- 1 = Left tape
- 2 = Right tape
- 3 = Display
- 4 = Printer

## LEGEND


<device> defined as:

Left tape  
Right tape  
EXternal printer  
SHared printer#<n> (where n>4 i.e., SH#5)  
Display  
TERminal#<p>  
Hp-ib#<p>[#<s>[#<m>]]  
where p = primary address  
s = secondary address  
m = module address  
Datacomm  
Null  
Graphics

<name> defined as:

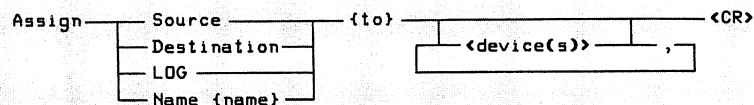
Source (only one may be selected)  
Destination  
LOG  
user specified name

COMMAND SYNTAX  
<command sequence>

ESC,c<abbreviated command sequence>   
or  
ESC & p <parameters>

## DESCRIPTION

## DEVICE CONTROL



Example: Assign "Plotter" to Hp-ib device number 4.

Assign Name PLOTTER to Hp-ib#4 <CR>


















Assign Source {to} Left tape 	ESC,cA S L  or ESC & p 1S	Assigns left tape as source device.
Assign Source {to} Right tape 	ESC,cA S R  or ESC & p 2S	Assigns right tape as Source device.
Assign Source {to} Display 	ESC,cA S DI  or ESC & p 3S	Assigns display as source device.
Assign Source {to} Hp-ib#<p> 	ESC,cA S H#<p> 	Assigns an HP-IB device as source device.
Assign Source {to} Graphics 	ESC,cA S G 	Assigns graphics memory as source device.
Assign Destination {to} Left tape 	ESC,cA D L  or ESC & p 1D	Assigns left tape as Destination device.
Assign Destination {to} Right tape 	ESC,cA D R  or ESC & p 2D	Assigns right tape as Destination device.
Assign Destination {to} Display 	ESC,cA D DI  or ESC & p 3D	Assigns display as Destination device.
Assign Destination {to} EXternal printer 	ESC,cA D EX or ESC & p 4D	Assigns local printer as Destination device.

Table B-2. Programmer's Reference Table (Continued)



































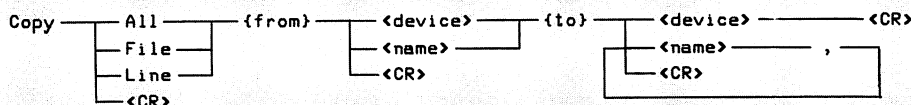
COMMAND SYNTAX <command sequence>	ESC,c<abbreviated command sequence>  or ESC &p <parameters>	DESCRIPTION
<b>DEVICE CONTROL (Continued)</b>		
Assign Destination {to} SHared printer#5 	ESC,cA D SH#5 	Assigns shared printer as Destination device.
Assign Destination {to} Hp-ib#<n>#<n>#<n> 	ESC,cA D H#<n> 	Assigns an HP-IB device(s) as Destination device(s).
Assign Destination {to} Graphics 	ESC,cA D G 	Assigns graphics memory as Destination device.
Assign LOG {to} <device> 	ESC,cA LOG <device> 	Assigns any device as LOG device.
Assign Name <user-defined name> {to} <device> 	ESC,cA N <name>. <device> 	Assigns a user-specified name to any device.
BYE ———— <CR>		
BYE 	ESC,cBYE 	Defaults user.group name to "user.group" on shared printer listings.
CLOse ———— Window#<n> ———— <CR>		
CLOse Window#5 	ESC,cCLO W#5 	Removes the Message Line on the display.
CLOse Window#6 	ESC,cCLO W#6 	Removes the Command Line on the display.
CLOse Window#7 	ESC,cCLO W#7 	Removes the Soft Key Label Line on the display.
CCompare ———— All ———— {of} ———— <device> ———— {to} ———— <device> ———— <CR>  ——— File ————  ——— <name> ————  ——— <name> ————    ——— Line ————  ——— <CR> ————  ——— <CR> ————    ——— <CR> ————		
Example: Compare a file on the display with a file on a device with a user name of "SALES". CCompare File of Display to SALES <CR>		
CCompare All {of} <device> {to} <device> 	ESC,cCO A <device><device>  or ESC &p xs yd 1M	Compare data between devices. (All)
CCompare File {of} <device> {to} <device> 	ESC,cCO F <device> <device>  or ESC &p xs yd 1F	Compare data between devices. (File)
CCompare Line {of} <device> {to} <device> 	ESC,cCO L <device> <device>  or ESC &p xs yd 1B	Compare data between devices. (Line)
CCompare All 	ESC,cCO A  or ESC &p 1M	Compare data between previously-assigned Source and Destination devices. (All)
CCompare File 	ESC,cCO F  or ESC &p 1F	Compare data between previously-assigned Source and Destination devices. (File)
CCompare Line 	ESC,cCO L  or ESC &p 1B	Compare data between previously-assigned Source and Destination devices. (Line)
CCompare All or File or Line {of}<name> {to}<name> 	ESC,cCO <A or F or L> <name> <name> 	Compares data between user-specified device names.
CCondition ———— <device> ———— <CR>  ——— <name> ————  ——— , ————		
Example: Assume the left tape is assigned the user name: "DATA". Condition left tape with a user name of "DATA" and the right tape. CCondition DATA, Right tape <CR>		
CCondition Left tape or Right tape 	ESC,cCON L or R or ESC &p 1u or 2u 4C	Conditions left tape or right tape.

Table B-2. Programmer's Reference Table (Continued)

COMMAND SYNTAX <command sequence>	ESC,c<abbreviated command sequence> $\frac{r}{h}$ or ESC &p <parameters>	DESCRIPTION
<b>DEVICE CONTROL (Continued)</b>		



Example: Copy the entire contents of the left tape to the shared printer.

Copy All from Left tape to SHared printer#5 <CR>

NOTE: If you want to copy graphics data (i.e. binary), the TRANSFER command sequence must be used.

Copy File {from} Source {to} Display $\frac{r}{h}$	ESC,c C F S DI	Transfers one file from Source device to display.
Copy File {from} Display {to} Destination $\frac{r}{h}$	ESC,c C F DI D	Transfers all data from display workspace to Destination device.
Copy All $\frac{r}{h}$	ESC,c C A $\frac{r}{h}$ or ESC &p M	All files (current position) from previously-assigned Source device are transferred to previously-assigned Destination device.
Copy $\frac{r}{h}$	ESC,c C $\frac{r}{h}$ or ESC &p xs yd F	One file (current position) from previously-assigned Source device is transferred to previously-assigned Destination device.
Copy Line $\frac{r}{h}$	ESC,c C L $\frac{r}{h}$ or ESC &p xs yd B	One line (current position) from previously-assigned Source device is transferred to previously-assigned Destination device.
Copy All {from} <device> {to} <device(s)> $\frac{r}{h}$	ESC,c C A <device> <device> $\frac{r}{h}$ or ESC &p xs yd M	All files (current position) from a specified device are transferred to one or more specified device(s).
Copy File {from} <device> {to} <device(s)> $\frac{r}{h}$	ESC,c C F <device> <device> $\frac{r}{h}$ or ESC &p xs yd F	One file (current position) from a specified device is transferred to one or more specified device(s).
Copy Line {from} <device> {to} <device(s)> $\frac{r}{h}$	ESC,c C L <device> <device> $\frac{r}{h}$ or ESC &p xs yd B	One line (current position) from a specified device is transferred to one or more specified device(s).
Copy All {from} <name> {to} <name> $\frac{r}{h}$	ESC,c C A <name> <name> $\frac{r}{h}$	All files (current position) from user-assigned device name are transferred to user-assigned device name(s).
Copy File {from} <name> {to} <name> $\frac{r}{h}$	ESC,c C F <name> <name> $\frac{r}{h}$	One file (current position) from user-assigned device name is transferred to user-assigned device name(s).
Copy Line {from} <name> {to} <name> $\frac{r}{h}$	ESC,c C L <name> <name> $\frac{r}{h}$	One line (current position) from user-assigned device name is transferred to user-assigned device name(s).

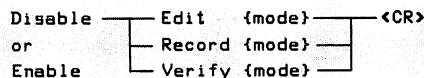
Display \_\_\_\_\_ Window#<n> \_\_\_\_\_ <CR>

Example: Display the message "Insert the tape into the left slot."

Display Window#5<CR>

Insert the tape into the left slot <CR>

Display Window#<n> $\frac{r}{h}$	ESC,c DI W#<n> $\frac{r}{h}$	Display workspace 1, 2, 3, or 4, Message Line (5), Command Line (6), or Soft Key Label Line (7).
----------------------------------	------------------------------	--



Enable/Disable Edit $\frac{r}{h}$	ESC,c E or D E $\frac{r}{h}$	Turns Edit Mode on/off.
Enable/Disable Verify $\frac{r}{h}$	ESC,c E or D V $\frac{r}{h}$ ESC &p 10C (on) ESC &p 9C (off)	Toggles Verify Mode. (Reads each record transferred and compares it with the original.
Enable/Disable Record $\frac{r}{h}$	ESC,c E or D R $\frac{r}{h}$	In REMOTE, transfers data from computer to destination device. In LOCAL, transfers one file from DISPLAY to destination device.

Table B-2. Programmer's Reference Table (Continued)

[illegible]







Table B-2. Programmer's Reference Table (Continued)

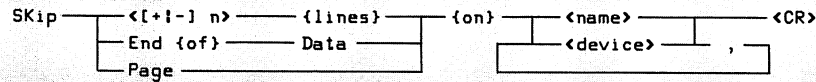
**COMMAND SYNTAX**  
**<command sequence>**

ESC,c<abbreviated command sequence>  $\text{\textasciitilde}$   
 or  
 ESC &p <parameters>

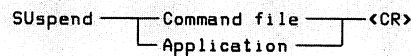
**DESCRIPTION**

**DEVICE CONTROL (Continued)**

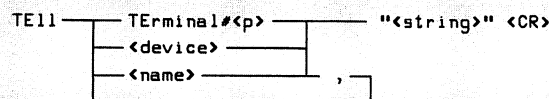
Show Time  $\text{\textasciitilde}$  ESC,cSH TI  $\text{\textasciitilde}$  Lists current time on log device.  
 Show Date  $\text{\textasciitilde}$  ESC,cSH D  $\text{\textasciitilde}$  Lists current date on log device.



SKip <+, -n> {lines} {on} Right tape  $\text{\textasciitilde}$  ESC,cSK <+, -n> R  $\text{\textasciitilde}$  Positions right tape to a relative (+, -n) line.  
 or  
 ESC &p (+, -n)p 2u 1C  
 SKip <+, -n> {lines} {on} EXternal printer  $\text{\textasciitilde}$  ESC,cSK <+, -n> EX  $\text{\textasciitilde}$  Positions external printer to a relative (+, -n) line.  
 or  
 ESC &p (+, -n)p 4u 1C  
 SKip <+, -n> {lines} {on} SHared printer#5  $\text{\textasciitilde}$  ESC,cSK <+, -n> SH#5  $\text{\textasciitilde}$  Positions SHared printer#5 to a relative (+, -n) line.  
 SKip <+, -n> {lines} {on} <name>  $\text{\textasciitilde}$  ESC,cSK <+, -n> <name>  $\text{\textasciitilde}$  Positions device that has user-assigned device name to a relative (+, -n) line.  
 SKip Page {on} EXternal printer  $\text{\textasciitilde}$  ESC,cSK P EX  $\text{\textasciitilde}$  Positions external printer to a top-of-form.  
 or  
 ESC &p (n)p 4u 2C  
 SKip Page {on} SHared printer#5  $\text{\textasciitilde}$  ESC,cSK P SH#5 Positions shared printer to a top-of-form.  
 SKip End {of} Data {on} Left tape  $\text{\textasciitilde}$  ESC,cSK E D L  $\text{\textasciitilde}$  Positions left tape beyond end-of-data mark.  
 SKip End {of} Data {on} Right tape  $\text{\textasciitilde}$  ESC,cSK E D R  $\text{\textasciitilde}$  Positions right tape beyond end-of-data mark.



SUSPEND Command file  $\text{\textasciitilde}$  ESC,cSU C  $\text{\textasciitilde}$  Suspends execution of a command file.  
 SUSPEND Application  $\text{\textasciitilde}$  ESC,cSU A  $\text{\textasciitilde}$  Suspends operation of the subsystem.

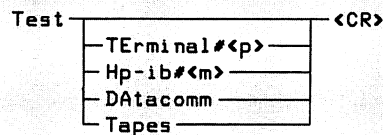


Note: Single quotes or double quotes may be used.

Example: Tell the right tape: "End of program."

TELL Right tape "End of Program" <CR>

TEll <device> "<string>"  $\text{\textasciitilde}$  ESC,cTE <device> "<string>"  $\text{\textasciitilde}$  Transfers data string to the specified device.  
 TEll TErминал#<p> "<string>"  $\text{\textasciitilde}$  ESC,cTE TE#<p> "<string>"  $\text{\textasciitilde}$  Transfers data string to the specified terminal in network.



Test  $\text{\textasciitilde}$  ESC,cT  $\text{\textasciitilde}$  Tests terminal.  
 or  
 ESC z  
 Test DATacomm  $\text{\textasciitilde}$  ESC,cT DA  $\text{\textasciitilde}$  Tests terminal datacomm. (Hood connector must be used.)  
 or  
 ESC x  
 Test Tapes  $\text{\textasciitilde}$  ESC,cT T  $\text{\textasciitilde}$  Tests terminal cartridge tape units.  
 or  
 ESC &p (1 or 2)u 7C

**COMMAND SYNTAX**  
**<command sequence>**

### DESCRIPTION

Test TErминаl#<n> 9

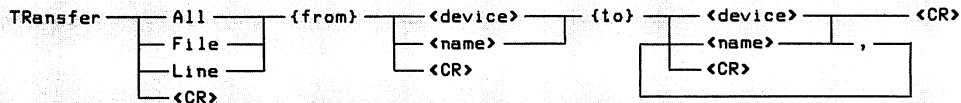
ESC,cT TE#<p> 9

Tests data path to other terminal in network.

Test Hp-ib#<n> **CR**

ESC,cT H#<m> 6

**Tests a specific HP-IB interface PCA in terminal.**



TRansfer File from Graphics to Left tape <CR>

Transfer File {from} Source {to} Display 

ESC,CTR F S DI 9

**Transfers one file from Source device to display.**

Transfer File {from}Display {to} Destination

ESC,cTR F DI D  $\epsilon_R$

**Transfers all data from display workspace to Destination device.**

Transfer All CR

ESC,CTR A  $\mathbb{R}$

All files (current position) from previously-assigned Source device are transferred to previously-assigned Destination device.

TRansfer File Cn

ESC,cTR F  $G_R$

One file (current position) from previously-assigned Source device is transferred to previously-assigned Destination device.

Transfer Line 9

ESC,CTR L  $G_R$ 

One line (current position) from previously-assigned Source device is transferred to previously-assigned Destination device.

TRansfer All {from}<device>{to}<device(s)> Cn

ESC,cTR A <device> <device> 

All files (current position) from a specified device are transferred to one or more specified device(s).

TRansfer File {from}<device>{to}<device(s)> 

ESC,cTR F <device> <device>  $\mathcal{R}$

One file (current position) from a specified device is transferred to one or more specified device(s).

Transfer Line {from} <device> {to} <device(s)> <sup>CR</sup>

ESC,cTR L <device> <device>  $\mathcal{R}$

One line (current position) from a specified device is transferred to one or more specified device(s).

TRansfer All {from}<name>{to}<name>  $\mathbb{C}_n$

ESC,cTR A <name> <name> 

All files (current position) from user-assigned device name are transferred to user-assigned device name(s).

TRansfer File {from}<name>{to}<name> C

ESC,cTR F <name> <name> 

One file (current position) from user-assigned device name is transferred to user-assigned device name(s).

Transfer Line {from}<name>{to}<name> C

ESC,cTR L <name> <name> C

One line (current position) from user-assigned device name is transferred to user-assigned device name(s).

Table B-3. Character Code Chart Reference Tables

BIT 7 6 5 4321	CONTROL (CNTL) CHARACTERS		DISPLAYABLE CHARACTERS		ESCAPE SENT FIRST					
	0 0 0 0	0 0 0 1	0 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1
0000	@ 0 NUL NUL	P 16 DLE DLE	SP 0 SP	@ 0 @	P 0 P	\ 0 \	p 0 p	SP 0 SP	0 0 0	DELAY 1 SEC @ 0 DELETE CHAR P 0 CURSOR RELATIVE SENSE f 0 p 0
0001	A 1 S0H S0H	Q 17 DC1 DC1	! 1 !	A 1 A	Q 1 Q	a 1 a	q 1 q	SET TAB 1 CURSOR UP A 1 INSERT CHAR ON Q 1 CURSOR ABSOLUTE SENSE f 1 q 1	1 1 1	1 1 1
0010	B 2 S1X STX	R 18 DC2 DC2	" 2 "	B 2 B	R 2 R	b 2 b	r 2 r	CLEAR TAB 2 CURSOR DOWN B 2 INSERT CHAR OFF R 2 KEYBOARD ENABLE f 2 r 2	2 2 2	2 2 2
0011	C 3 S2X ETX	S 19 DC3 DC3	# 3 #	C 3 C	S 3 S	c 3 c	s 3 s	CLEAR ALL TABS 3 CURSOR RIGHT C 3 ROLL UP S 3 KEYBOARD DISABLE f 3 s 3	3 3 3	3 3 3
0100	D 4 S3T EOT	T 20 DC4 DC4	\$ 4 \$	D 4 D	T 4 T	d 4 d	t 4 t	SET LEFT MARGIN 4 CURSOR LEFT D 4 ROLL DOWN T 4 ENTER f 4 t 4	4 4 4	4 4 4
0101	E 5 S4Q ENO	U 21 NAK NAK	% 5 %	E 5 E	U 5 U	e 5 e	u 5 u	SET RIGHT MARGIN 5 RESET TERMINAL E 5 NEXT PAGE U 5 BINARY READ f 5 u 5	5 5 5	5 5 5
0110	F 6 S5K ACK	V 22 SYN SYN	& 6 &	F 6 F	V 6 V	f 6 f	v 6 v	START ALPHA FIELD 6 CURSOR HOME DOWN F 6 PREV PAGE V 6 MODEM DISCONNECT f 6 v 6	6 6 6	6 6 6
0111	G 7 S6W BEL	W 23 ETB ETB	' 7 '	G 7 G	W 7 W	g 7 g	w 7 w	START NUMERIC FIELD 7 CURSOR RETURN G 7 FORMAT MODE ON W 7 SOFT RESET f 7 w 7	7 7 7	7 7 7
1000	H 8 S7S BS	X 24 CAN CAN	( 8 (	H 8 H	X 8 X	h 8 h	x 8 x	START ALPHNUM FIELD 8 HOME CURSOR (SEE NOTE 3) H 8 FORMAT MODE OFF X 8 HOME CURSOR (SEE NOTE 3) DATA COM SELF TEST f 8 x 8	8 8 8	8 8 8
1001	I 9 S8T HT	Y 25 EM EM	) 9 )	I 9 I	Y 9 Y	i 9 i	y 9 y	DEFINE CHAR SET 9 HORIZONTAL TAB I 9 DISPLAY FUNCTIONS ON Y 9 BACK TAB f 9 y 9	9 9 9	9 9 9
1010	J 10 S9F LF	Z 26 SUB SUB	* 10 *	J 10 J	Z 10 Z	j 10 j	z 10 z	GRAPHICS SEQUENCE 10 CLEAR DPLY J 10 DISPLAY FUNCTIONS OFF Z 10 SOFT KEY DISPLAY ON f 10 z 10	10 10 10	10 10 10
1011	K 11 S10V VT	[ 27 EC ESC	+ 11 +	K 11 K	[ 11 [	k 11 k	{ 11 {	ERASE TO END OF LINE 11 START UNPROTECT FIELD K 11 KEY DISPLAY OFF f 11 { 11	11 11 11	11 11 11
1100	L 12 S11F FF	\ 28 FS FS	, 12 ,	L 12 L	\ 12 \	l 12 l	12 	COMMAND SEQUENCE 12 INSERT LINE L 12 END UNPROTECT FIELD f 12   12	12 12 12	12 12 12
1101	M 13 S12R CR	] 29 GS GS	- 13 -	M 13 M	] 13 ]	m 13 m	} 13 }	DELETE LINE 13 DELETE LINE M 13 END UNPROTECT FIELD f 13 } 13	13 13 13	13 13 13
1110	N 14 S13O SO	^ 30 RS RS	. 14 .	N 14 N	^ 14 ^	n 14 n	~ 14 ~	INSERT CHAR W/WRAP ON 14 TERM PRIMARY STATUS N 14 INSERT NON-DISP TERMINATR f 14 ~ 14	14 14 14	14 14 14
1111	O 15 S14I SI	_ 31 US US	/ 15 /	O 15 O	_ 15 _	o 15 o	15 	DELETE CHAR W/WRAP 15 INSERT NON-DISP TERMINATR O 15 INSERT NON-DISP TERMINATR f 15 15	15 15 15	15 15 15

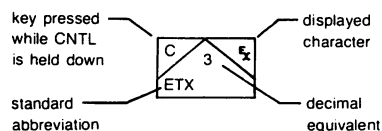
Example: J is bits 1001010; Control J is LF line feed; Escape (ESC) followed by J is CLEAR DISPLAY

## 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.

### Control Character Legend:



- |  |  |
|--|--|
| <ul style="list-style-type: none"> <li>^ — ACKNOWLEDGE</li> <li>0 — BELL</li> <li>␣ — BACKSPACE</li> <li>␣ — CANCEL LINE</li> <li>␣ — CARRIAGE RETURN</li> <li>␣ — DATA LINK ESCAPE</li> <li>D<sub>1</sub> — DEVICE CONTROL 1</li> <li>D<sub>2</sub> — DEVICE CONTROL 2</li> <li>D<sub>3</sub> — DEVICE CONTROL 3</li> <li>D<sub>4</sub> — DEVICE CONTROL 4</li> <li>■ — DELETE</li> <li>␣ — END OF MEDIUM</li> <li>␣ — ENQUIRY</li> <li>␣ — END OF TRANSMISSION</li> <li>␣ — ESCAPE</li> <li>␣ — END OF TRANSMISSION BLOCK</li> </ul> | <ul style="list-style-type: none"> <li>␣ — END OF TEXT</li> <li>␣ — FORM FEED</li> <li>␣ — FILE SEPARATOR</li> <li>␣ — GROUP SEPARATOR</li> <li>␣ — HORIZONTAL TABULATION</li> <li>␣ — LINE FEED</li> <li>␣ — NEGATIVE ACKNOWLEDGE</li> <li>␣ — RECORD SEPARATOR</li> <li>␣ — SHIFT IN</li> <li>␣ — SHIFT OUT</li> <li>SP — SPACE</li> <li>␣ — START OF HEADING</li> <li>␣ — START OF TEXT</li> <li>␣ — SUBSTITUTE</li> <li>␣ — SYNCHRONOUS IDLE</li> <li>␣ — UNIT SEPARATOR</li> <li>␣ — VERTICAL TABULATION</li> </ul> |
|--|--|

## ORDERING EXAMPLE

This is an example of ordering four 2647A Intelligent Graphics Terminals, a shared 7245 Printer/Plotter and used remotely over a modem.

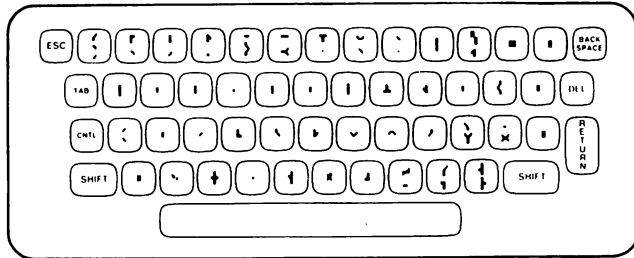
4 – 2647A	Intelligent Graphics Terminal
4 – 13296A	Shared Peripheral Interface (includes cable)
4 – 13232N	RS232C cable to modem
1 – 7245A	Printer Plotter
–001	Graphics Option

Table B-4. Options and Accessories

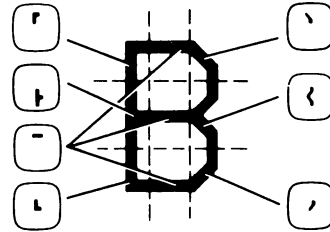
PRODUCT NUMBER	DESCRIPTION	PRICE
2647A	INTELLIGENT GRAPHICS TERMINAL 720 x 360 Dot Graphics Image Memory and random access alphanumeric memory, Basic interpreter with graphic commands, multiple automatic plotting, 128 character Roman, inverse video, 8 user defined soft keys, 75 lines of display memory, RS232C communications, integrated dual cartridge tapes, 1 option slot.	\$8,300
–013	5 mini cartridges	90
–015	50 Hz, 230V	00
–016	50 Hz, 115V	0
–030	Delete Standard Asynchronous Communications <i>Note: One of the 13260 data communications accessories must be ordered when option 030 is ordered, also deletes keyboard overlay and data communications firmware.</i>	–160
13231A	DISPLAY ENHANCEMENTS Adds blinking, half-bright and underline; and provides for addition of three 128 character sets. (Requires 1 option slot).	250
–201	64 Character Math Symbol Set: Adds display of integral signs, Greek letters, etc.	100
–203	Large Character Set	
<b>ACCESSORIES</b>		
13250B	SERIAL PRINTER INTERFACE Adds interface for connecting RS232C printing devices (requires 1 option slot). No interface cable included (local only, not shared).	275
13254A	VIDEO OUTPUT INTERFACE Provides video signal for a compatible monitor or hardcopy unit. (Requires 1 option slot.)	150
13296A	SHARED PERIPHERAL INTERFACE Provides shared interface for compatible devices, includes cable. (Requires 1 option slot.)	500
13260A-006	STANDARD ASYNCHRONOUS COMMUNICATIONS Provides standard RS232C communications interface for the 2647A. <i>Note: This is identical to the capability deleted by 2647A-030.</i>	160
13260B-006	EXTENDED ASYNCHRONOUS COMMUNICATIONS Upgrade which provides either RS232C or 20mA current loop communications for the 2647A. Has split speed and custom baud rates.	325
13260C-006	ASYNCHRONOUS MULTIPOINT COMMUNICATIONS Provides asynchronous multipoint communications interface for the 2647A allowing daisy chained line sharing. <i>Note: 2647A-030 must be ordered to delete the Standard Asynchronous Interface.</i>	435
–001	Add Monitor Mode capability.	50
13260D-006	SYNCHRONOUS MULTIPOINT COMMUNICATIONS Provides synchronous multipoint communications interface for the 2647A allowing daisy chained line sharing. <i>Note: 2647A-030 must be ordered to delete the Standard Asynchronous Interface.</i>	450
–001	Add Monitor Mode capability.	50
9162-0061	Mini-Cartridge (available in boxes of fives).	
13232A	103/202 Modem (male, 15 ft)	50
13232B	To 12880A/12531D (50 ft)	100
13232C	RS232 Connector (female, 5 ft)	50
13232F	Current Loop Connector Kit (5 ft)	60
13232G	13250 to RS232C Printer (male, 15 ft)	65
13232H	13250 to RS232C Printer (female, 25 ft)	65
13232J	13238 to 9871 Printer (6 ft)	55
13232K	13254 to video hardcopy printer (15 ft)	75
13232L	13254 to compatible TV monitor (15 ft)	75
13232M	13250B to European modems	75
13232N	103/202/201/208/209 modem (male 15 ft)	75
13232P	Modem/Multipoint Cable, male RS232C, male multipoint connector, 30 ft. total	115
13232Q	Multipoint Cable, male multipoint, female multipoint connector, 30 ft. total	90
13232R	Multipoint Extension Cable, male multipoint/female multipoint connector, 100 ft.	75
13232S	13238 to 9866 line printer (6 ft)	50
13232T	Power Protect Multipoint Cable, male multipoint connector with relays /female multipoint connector, 30 ft. total	185
13232U	Modem bypass (5 ft)	50

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	]	%. 0 %M	i	· E	U	" FM
"	"	.	Z	:	Z Z	F	"&, /& E	R	"&+ /& E E	^	9 K	j	· GL	V	" 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	50 EE
%	P P 3<D Y Y	1	- 0 E	=	%&, %&,	I	· 0 I	U	0 0 G&L	a	!. GM	m	\$- EE	y	" G?
&	!+ SIC G&L	2	!&+ !&L F&,	>	) D	J	· 0 L	V	0 0 2JD	b	· FL	n	"&+ EE	z	%. F,
'	'	3	!&+ !&+ G&L	?	!&+ >D S	K	!&+ /GA E E	W	090 HKD	c	!, G,	o	!+ GL	{	!, G,
[	!, 0 G,	4	!&+ F&C E	@	!&+ !&+ GIL	L	· 0 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	\$)0 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	!&+ 50 G&L	D	"&+ 0 0 F&L	P	"&+ /&+ E	\	2;D E	h	· EE	t	· 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 tables
- 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 Interface PCA switches.

Table C-1. ASCII Character Set

DECIMAL VALUE	GRAPHIC	COMMENTS	ALTERNATE CHARACTER	DECIMAL VALUE	GRAPHIC	COMMENTS
0		Null	@ <sup>c</sup>	64	@	Commercial at
1		Start of heading	A <sup>c</sup>	65	A	Uppercase A
2		Start of text	B <sup>c</sup>	66	B	Uppercase B
3		End of text	C <sup>c</sup>	67	C	Uppercase C
4		End of transmission	D <sup>c</sup>	68	D	Uppercase D
5		Enquiry	E <sup>c</sup>	69	E	Uppercase E
6		Acknowledge	F <sup>c</sup>	70	F	Uppercase F
7		Bell	G <sup>c</sup>	71	G	Uppercase G
8		Backspace	H <sup>c</sup>	72	H	Uppercase H
9		Horizontal tabulation	I <sup>c</sup>	73	I	Uppercase I
10		Line feed	J <sup>c</sup>	74	J	Uppercase J
11		Vertical tabulation	K <sup>c</sup>	75	K	Uppercase K
12		Form feed	L <sup>c</sup>	76	L	Uppercase L
13		Carriage return	M <sup>c</sup>	77	M	Uppercase M
14		Shift out	N <sup>c</sup>	78	N	Uppercase N
15		Shift in	O <sup>c</sup>	79	O	Uppercase O
16		Data link escape	P <sup>c</sup>	80	P	Uppercase P
17		Device control 1 (X-ON)	Q <sup>c</sup>	81	Q	Uppercase Q
18		Device control 2	R <sup>c</sup>	82	R	Uppercase R
19		Device control 3 (X-OFF)	S <sup>c</sup>	83	S	Uppercase S
20		Device control 4	T <sup>c</sup>	84	T	Uppercase T
21		Negative acknowledge	U <sup>c</sup>	85	U	Uppercase U
22		Synchronous idle	V <sup>c</sup>	86	V	Uppercase V
23		End of transmission block	W <sup>c</sup>	87	W	Uppercase W
24		Cancel	X <sup>c</sup>	88	X	Uppercase X
25		End of medium	Y <sup>c</sup>	89	Y	Uppercase Y
26		Substitute	Z <sup>c</sup>	90	Z	Uppercase Z
27		Escape	[ <sup>c</sup>	<sup>1</sup> 91	[	Opening bracket
28		File separator	\ <sup>c</sup>	<sup>2</sup> 92	\	Reverse slant
29		Group separator	] <sup>c</sup>	<sup>1</sup> 93	]	Closing bracket
30		Record separator	^ <sup>c</sup>	<sup>1</sup> 94	^	Circumflex
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-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
EDT	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-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
55	79	117	4F
&	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

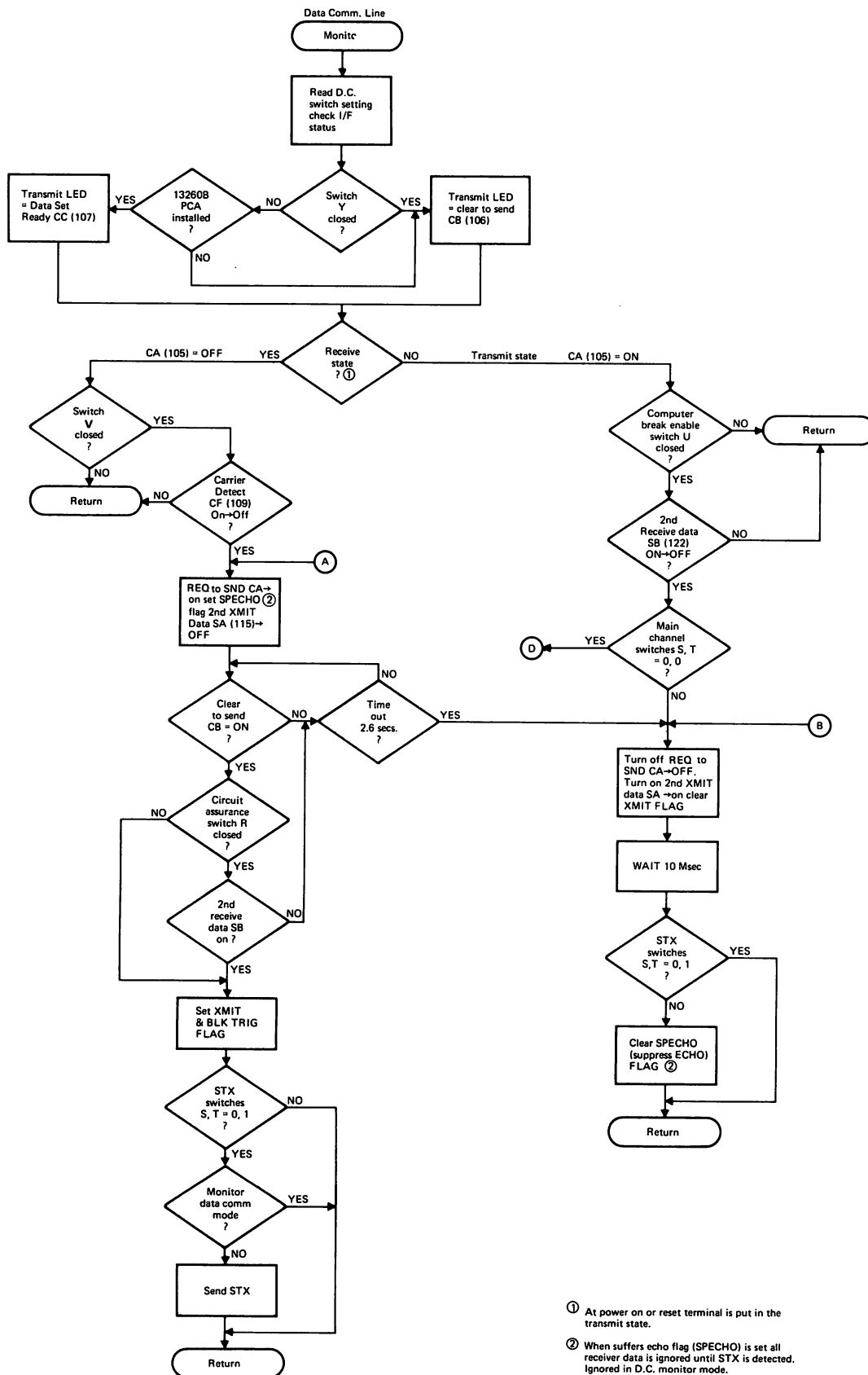


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 1 of 3)

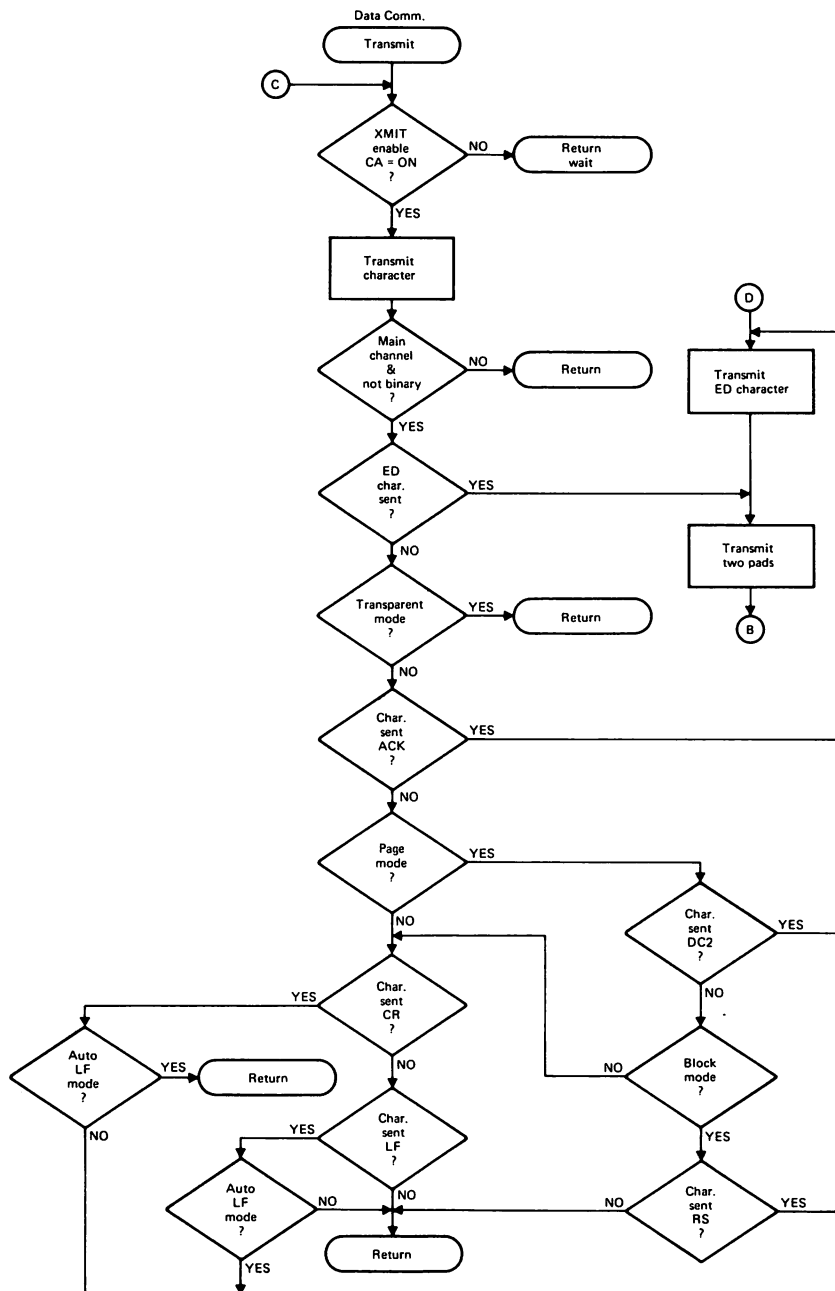


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 2 of 3)

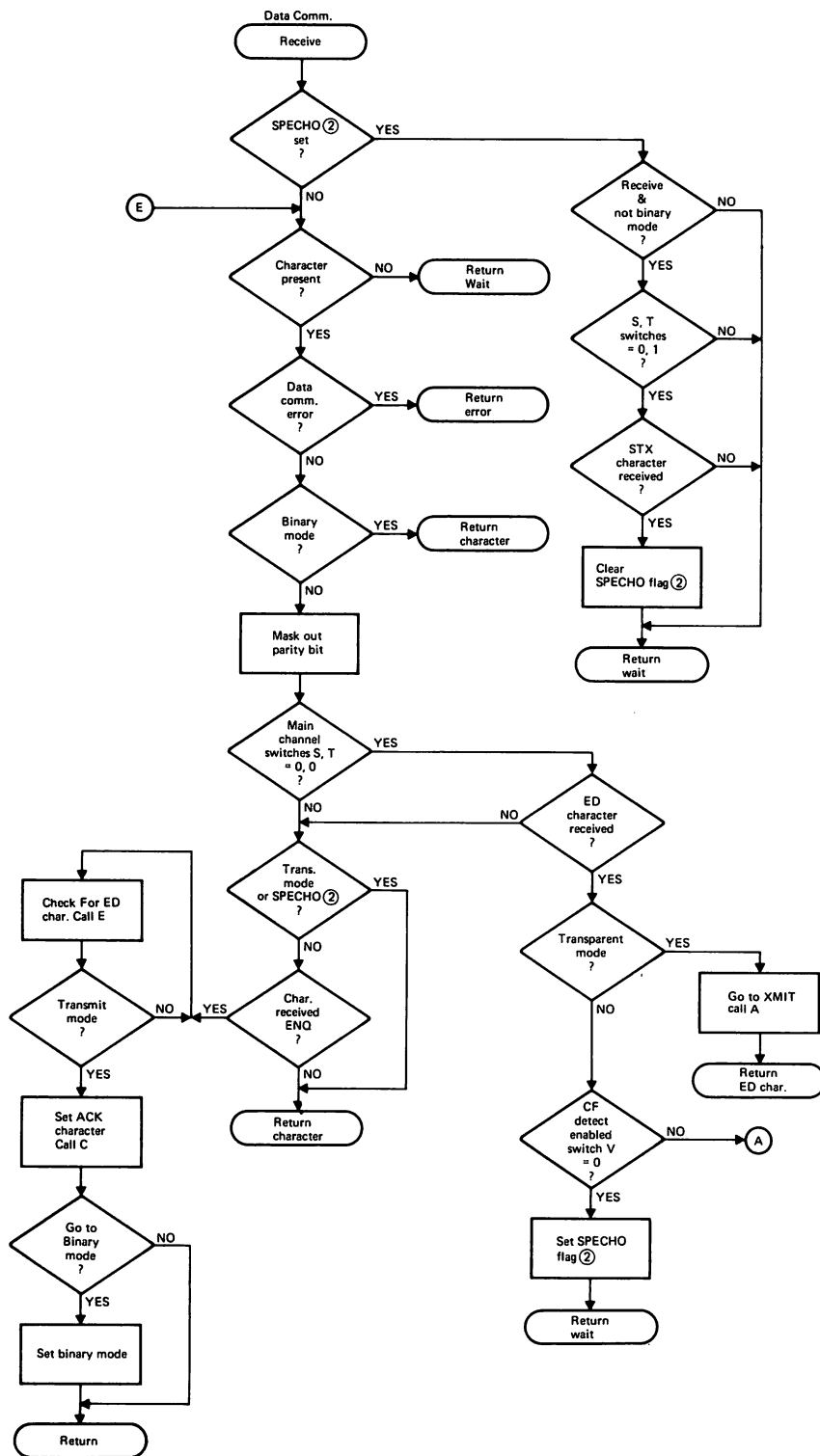


Figure C-1. Point-to-Point Communication Flowcharts (Sheet 3 of 3)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

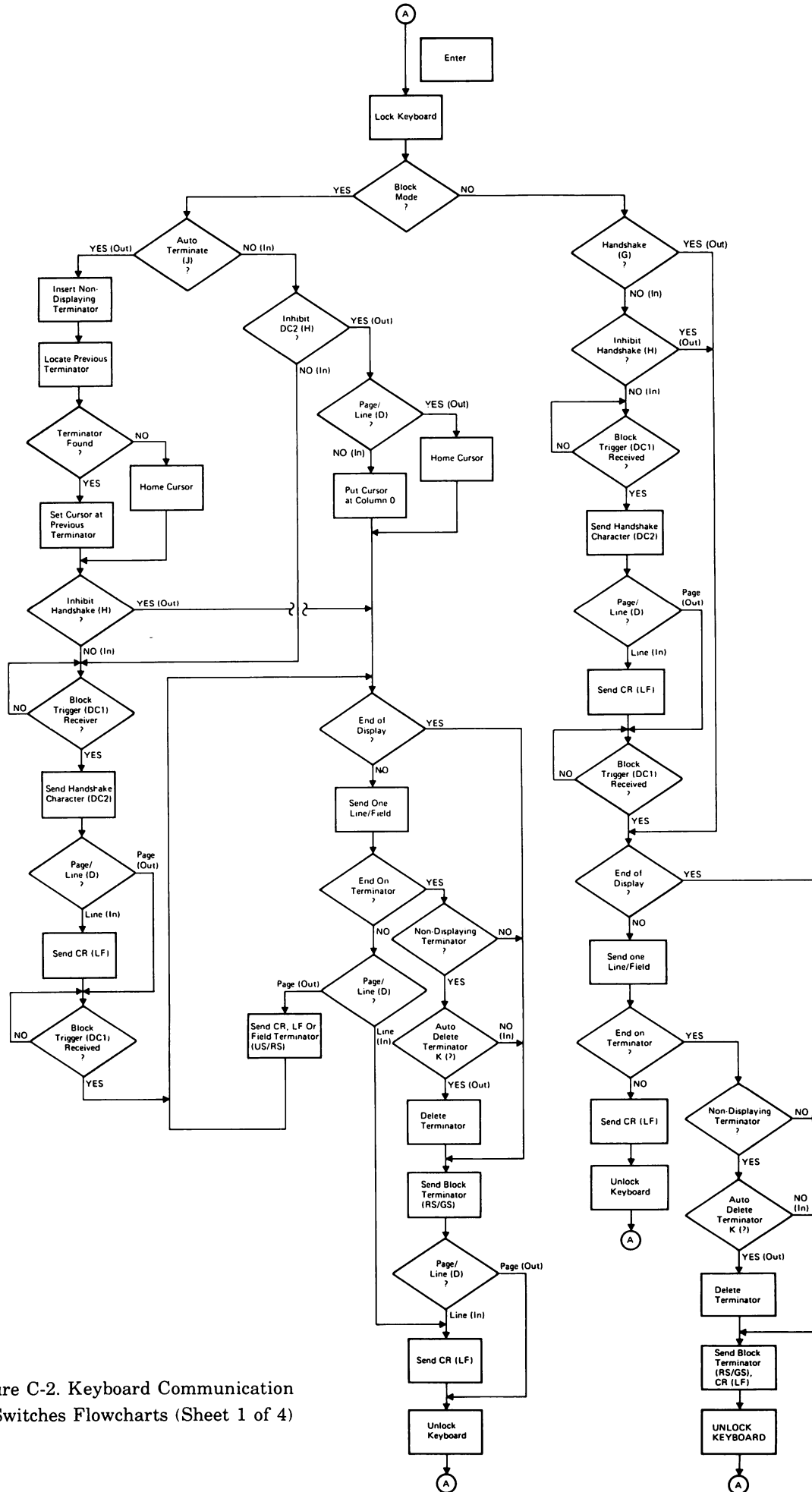


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 1 of 4)

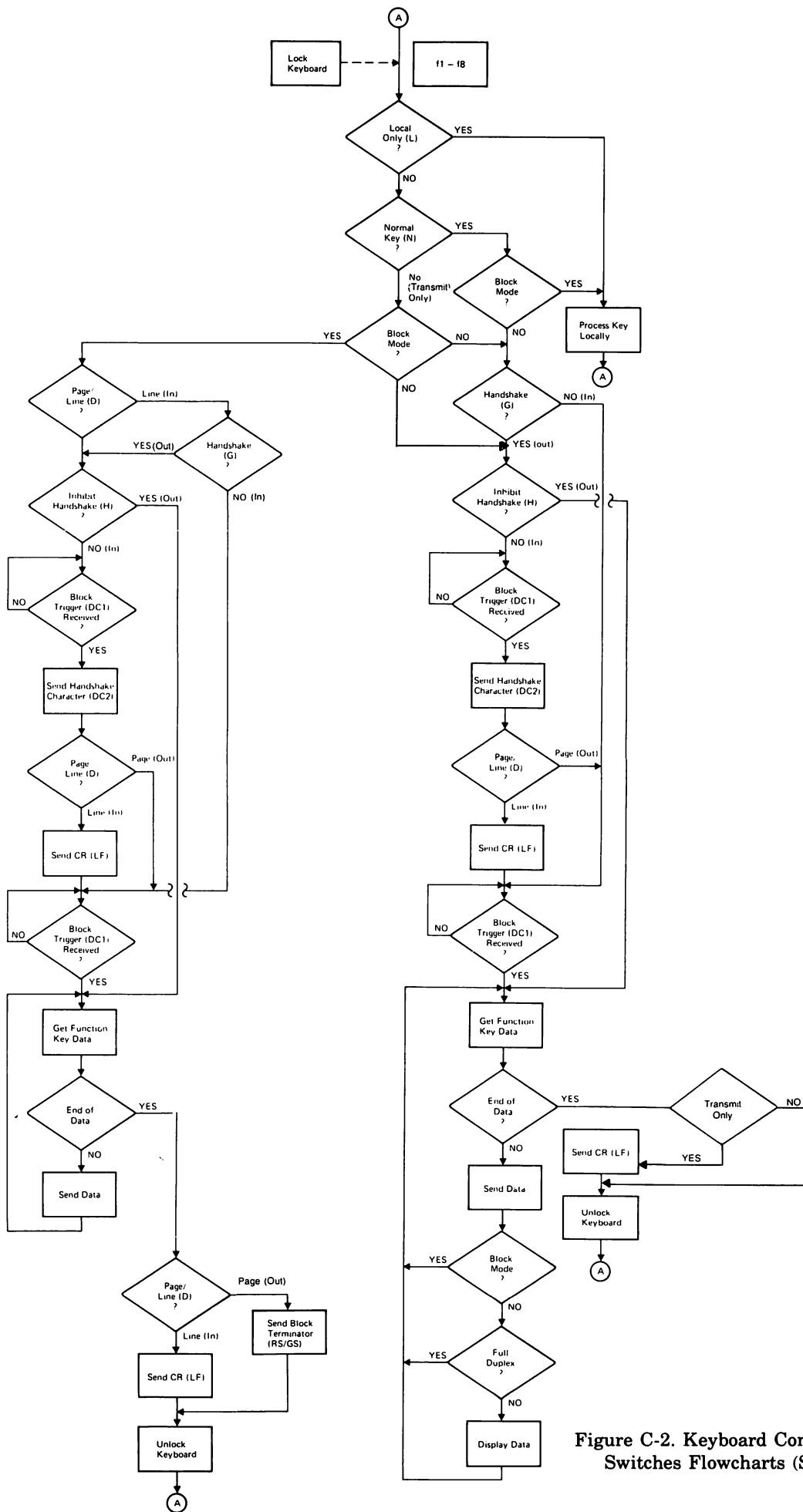


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 2 of 4)

MULTIPLE CHARACTERS TRANSFER STRAP CONTROLS

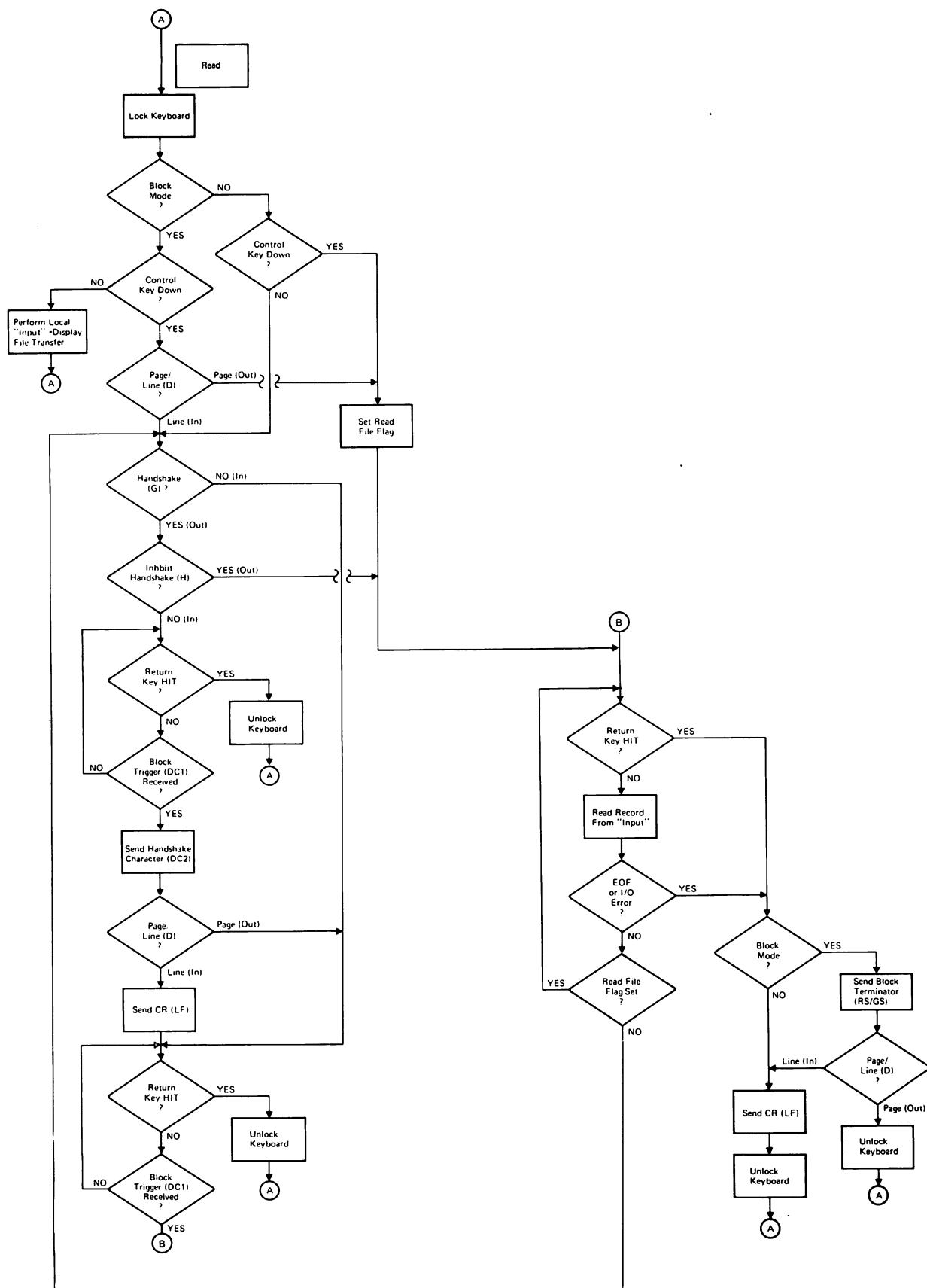


Figure C-2. Keyboard Communication Switches Flowcharts (Sheet 3 of 4)





# 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.

- a. Remove tape cartridge top cover by removing four screws from backplate with Phillips-head screwdriver.
- b. 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.
- c. 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.
- d. 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.
- e. 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.
- f. Rotate belt drive puck counterclockwise until tape is wrapped several times around left tape hub past first set of tape holes (approximately two feet).
- g. Replace tape cartridge top cover on backplate and secure in place with four screws.
- h. Condition tape (COMMAND, f1, f1, f6, f5 or f6, RETURN).

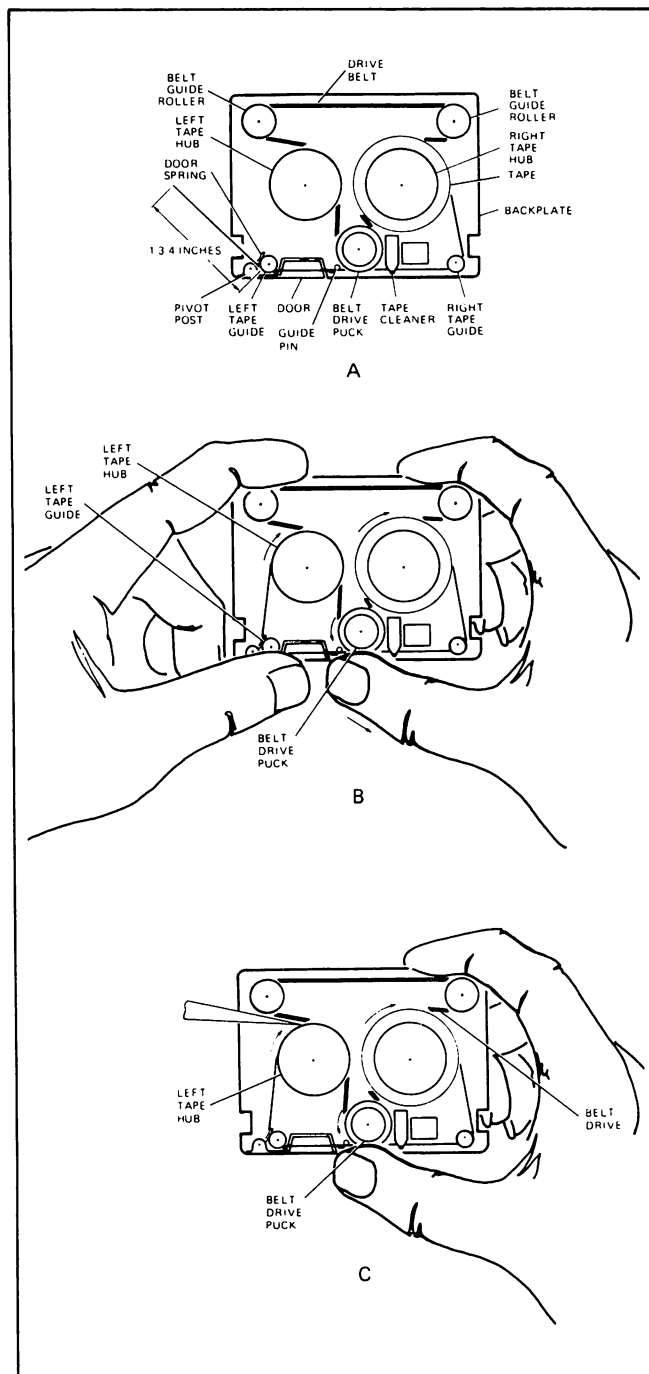


Figure D-1. Tape Cartridge Rethreading



13231 Display Enhancements .....	7-6
13232 Cable Assemblies .....	7-9
13238 Terminal Duplex Register Installation .....	7-12
13246 Printer Subsystem Installation .....	7-12
13250 Serial Printer Interface Installation .....	7-12
13254 Video Interface Installation .....	7-12
13260 Data Communications Interfaces, description .....	5-1, 5-3
13260 Data Communications Interfaces, installation .....	7-13
13296A Shared Peripheral Interface installation .....	7-14
13349 Printer Subsystem installation .....	7-17
202 Modem compatibility .....	5-9
32K RAM Memory PCAs strapping .....	7-10
9866 Printer subsystem installation .....	7-12
9871 Printer installation .....	7-17

## A

absolute, ASCII .....	3-16
absolute, binary .....	3-16
accessory installation .....	7-6
ACK .....	5-6
ACK1 .....	5-27
address, terminal .....	5-35
addressing, absolute .....	2-4
addressing, column .....	2-3
addressing, cursor relative .....	2-5
addressing, screen relative .....	2-4
AID character .....	5-39
alpha field checking .....	2-9
alphanumeric cursor operations .....	2-6
alternate character sets .....	2-10
area fill, absolute and relocatable .....	3-12
area pattern .....	3-8
area pattern, definition .....	3-10
ASCII .....	3-15
ASCII (7-bit) Character Codes, DEC, OCT, HEX .....	C-3
ASCII absolute .....	3-16
ASCII Character Set decimal values .....	C-2
ASCII code selection .....	5-24
ASCII incremental .....	3-16
ASCII relocatable .....	3-16
ASCII transfers .....	4-5, 4-12
assign devices .....	4-4
Asynchronous Multipoint Interface (13260C) straps .....	5-41, 7-24
Attention ID (AID) .....	5-39
AUTO LF key programming .....	2-13

## B

back tab .....	2-6
backspace .....	2-6, 4-11
Bar Charts .....	3-31
baud rate selection (13260B) .....	5-23
BAUD RATE switch (point-to-point) .....	5-4
baud rate switches (block) .....	5-33
BCC .....	5-24, 5-25
bell .....	2-17
binary absolute .....	3-17

binary incremental .....	3-17
binary operation (block) .....	5-29
binary read .....	4-9, 4-14
binary relocatable .....	3-18
binary short incremental .....	3-17
binary transfers .....	4-9, 4-12
blinking .....	2-10
block check character (BCC) .....	5-24, 5-25
block check mode .....	5-25
block checking .....	5-24
block input mode (block) .....	5-29
block mode .....	5-9, 5-10
Block Mode, ENTER key .....	5-9
Block Mode, G and H straps .....	5-9
BLOCK MODE key programming .....	2-13
BLOCK MODE key .....	5-4
block moves (text) .....	2-8
block operation .....	5-23
block protocols .....	5-6, 5-23
block protocol control characters operation .....	5-29
block protocol control characters summary .....	5-31
BREAK key .....	5-4
buffer, data communications .....	5-7
buffer overflow error .....	7-43
buffer size .....	5-25
bus, general description of .....	1-3
Bye command sequence .....	4-4
byte count .....	4-10
byte count, sending .....	4-12

## C

cables .....	7-8
cables, multipoint .....	7-28
cables, point-to-point .....	7-29
cables, power down protect .....	7-29, 7-32, 7-33
cable fabrication .....	7-34
cable lengths, data communications .....	7-28
cable length limitations .....	7-34
cable types .....	7-26
cabling, asynchronous multipoint .....	7-30
cabling, current loop .....	7-29
cabling, data communications .....	7-26
cabling, modem by-pass .....	7-29
cabling, synchronous multipoint .....	7-31
cabling considerations, HP-IB .....	8-2
cabling limitations, HP-IB .....	8-3
CAPS LOCK key programming .....	2-13
cartridge tape operations, Compatibility Mode .....	3-39
cartridge tape rethreading .....	D-1
CCA character .....	5-39
CCITT signals .....	7-27
centering graphics text .....	3-27
character cell, general description of .....	1-3
character checking .....	5-24
character code chart .....	B-14
character delete .....	2-7
Character Mode (block) .....	5-23
Character Mode (point-to-point) .....	5-7
character protocols .....	5-6
character set selection .....	2-10

character sets ..... 2-10  
 character sets, general description of ..... 1-4  
 checkerboard pattern ..... 3-10  
 clear display ..... 2-6  
 clear line ..... 2-7  
 clear mode, graphics ..... 3-6  
 clear tab ..... 2-5  
 CLOSe command sequence ..... 4-4  
 code, device ..... 4-10  
 code selection (ASCII/EBCDIC) ..... 5-24  
 coding the Large Character Set ..... B-16  
 command line ..... 2-2, 2-3  
 command status ..... 6-12  
 commands, graphics ..... 3-3  
 communication interface installation ..... 7-13  
 communication protocols ..... 5-6  
 communication test procedure ..... 7-43  
 communications ..... 5-1  
 communications, point-to-point ..... 5-6  
 communications flowcharts ..... C-1  
 compare all data ..... 4-5, 4-10  
 compare file ..... 4-5, 4-10  
 compare record ..... 4-5, 4-10  
 Compatibility Mode ..... 3-35  
 Compatibility Mode graphic data coding ..... 3-40  
 Compatibility Mode strapping ..... 3-36  
 complement mode, graphics ..... 3-6  
 completion, device ..... 4-7, 4-14  
 computers, connecting to ..... 5-1  
 condition tape ..... 4-5, 4-11  
 configuration (block) ..... 5-39  
 configuration (point-to-point) ..... 5-14  
 configuration status, block ..... 5-38  
 connecting terminals ..... 5-1  
 control bits, device ..... 4-10  
 control character protocol ..... 5-15  
 control codes, graphics ..... 3-3  
 control functions, terminal ..... 2-13  
 Control Memory PCAs strapping ..... 7-11  
 control parameter, device ..... 4-10  
 control sequences (block) ..... 5-27, 5-36  
 control sequences, device ..... 4-1, 4-10  
 controlling the display ..... 2-1  
 copy all data ..... 4-5, 4-10  
 copy file ..... 4-5, 4-10, 4-13  
 copy record ..... 4-5, 4-10, 4-13  
 copying to end of medium ..... 4-14  
 CRC ..... 5-24  
 Current Cursor Address (CCA) ..... 5-39  
 cursor addressing ..... 2-4  
 cursor control, graphics ..... 3-4  
 cursor movements, general description of ..... 1-5  
 cursor positioning ..... 2-4  
 cursor sensing ..... 2-4  
 cursor/display operations ..... 2-6

## D

data checking (communications) ..... 5-24  
 data checking (fields) ..... 2-9  
 data comm error ..... 7-43

data communications ..... 5-1  
 data communications, general description of ..... 1-5  
 data format, ASCII absolute ..... 3-16  
 data format, ASCII incremental ..... 3-16  
 data format, ASCII relocatable ..... 3-16  
 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 ..... 3-38  
 data format, default ..... 3-15  
 data format, packed (see binary) ..... 3-16  
 data overflow (block) ..... 5-45  
 data transfer, computer to device ..... 4-13  
 data transfer, device to computer ..... 4-12  
 data transfers (block) ..... 5-36  
 data transfers, device ..... 4-5, 4-9, 4-12  
 date, displaying of ..... 4-8  
 date, setting of ..... 4-7  
 DC1 character ..... 5-7, 5-8  
 DC1 character inhibit ..... 5-12  
 DC2 character ..... 5-7, 5-8  
 DC2 character inhibit ..... 5-12  
 default data format ..... 3-15  
 default device ..... 4-4, 4-10  
 default parameters, graphics ..... 3-14  
 defining soft keys ..... 2-15  
 delays in graphics operations ..... 3-6  
 delete character ..... 2-7  
 delete line ..... 2-7  
 device assignments, displaying of ..... 4-8  
 device code ..... 4-10  
 device completion ..... 4-7, 4-11  
 device control ..... 4-10  
 device control, cartridge tapes ..... 4-2, 4-11  
 device control, display ..... 4-12  
 device control, printer ..... 4-2, 4-12  
 device functions ..... 4-2, 4-11  
 Device ID ..... 5-35  
 device selection ..... 4-4, 4-11  
 device status ..... 6-6  
 diagonal lines ..... 3-8  
 diagonal lines, define pattern ..... 3-10  
 Digitizer (9874) example ..... A-2  
 display, alphanumeric, general description of ..... 1-1  
 display, graphics, general description of ..... 1-1  
 display, send to computer (block) ..... 2-17  
 display clear ..... 2-6  
 DIsplay command sequence ..... 4-6  
 display control ..... 2-1  
 display control, graphics ..... 3-4  
 Display Enhancements ..... 2-10  
 Display Enhancements (13231), installation ..... 7-6  
 display enhancements and graphics data ..... 3-1  
 Display Enhancements test pattern ..... 7-39  
 display memory functions ..... 2-1  
 display on/off, graphics ..... 3-5  
 display selection ..... 4-4, 4-6, 4-11  
 display window control ..... 2-1, 2-2  
 display workspaces ..... 2-1, 2-2

DLE ..... 5-27  
drawing modes, graphics ..... 3-6  
Driver Mode ..... 5-46  
Duplex Register installation ..... 7-12  
DUPLEX switch ..... 5-4

## E

EBCDIC Character Codes, DEC, OCT, HEX ..... C-4  
EBCDIC code selection ..... 5-24  
Edit Mode, enable/disable ..... 4-5  
edit operations ..... 2-7  
editing in Forms Mode ..... 2-9  
end-of-data, locating ..... 4-6, 4-11  
ENQ (block) ..... 5-27  
ENQ (point-to-point) ..... 5-6  
ENTER key ..... 5-4  
EOT ..... 5-28  
error messages (also see Users Manual) ..... 6-6  
error tests ..... 7-43  
escape sequence, device control ..... 4-1, 4-10  
ETB ..... 5-28  
ETX ..... 5-28  
EXecute command sequence ..... 4-6  
EXIT command sequence ..... 4-  
Extended Asynchronous Interface (13260B)  
    straps ..... 5-19, 7-23  
Extended Text features ..... 5-38

## F

f1-f8 keys, programming of ..... 2-14  
fail ..... 7-43  
fast binary read ..... 4-14  
field checking ..... 2-9  
fields ..... 2-9  
file copy ..... 4-5, 4-13  
file mark, recording ..... 4-7, 4-11  
file transmission ..... 4-12  
files, finding ..... 4-6, 4-11  
fill area, absolute and relocatable ..... 3-12  
find file ..... 4-6, 4-11  
firmware, general description of ..... 1-4  
form, sample ..... 2-13  
Format Mode ..... 2-8  
Forms Mode ..... 2-8  
forms, building ..... A-4  
forms, creating ..... 2-8, 2-13  
forward space ..... 4-11  
from device selection ..... 4-4, 4-11  
full duplex operation ..... 5-9  
functions, device ..... 4-11

## G

graphics commands ..... 3-3  
graphics commands, length of ..... 3-15  
graphics cursor control ..... 3-4  
graphics display, addressing points ..... 3-1  
graphics display on/off ..... 3-5  
graphics display set/clear ..... 3-5

graphics drawing modes ..... 3-6  
graphics functions, keyboard ..... 3-1  
graphics functions, recording of ..... 3-19  
graphics hardcopy operations ..... 4-9, 4-14  
graphics input terminator ..... 3-36  
graphics keys, description of ..... 3-2  
graphics keys, location of ..... 3-1  
graphics memory control ..... 3-5  
graphics memory dump to printer ..... 4-9  
graphics memory dump to tape ..... 4-9  
graphics memory dump ..... 3-19  
graphics memory restore from tape ..... 4-9  
graphics operations, inserting delays ..... 3-6  
graphics sequence termination ..... 3-15  
graphics sequence types ..... 3-3  
graphics status ..... 6-8  
graphics text centering ..... 3-27  
graphics text direction ..... 3-26  
graphics text justifying ..... 3-27  
graphics text margin ..... 3-27  
graphics text on/off ..... 3-27  
graphics text size ..... 3-26  
graphics text slant ..... 3-26  
grounding ..... 7-5  
Group ID ..... 5-35  
group polling ..... 5-37  
group select ..... 5-37

## H

half bright ..... 2-10  
half duplex operation ..... 5-9  
hardcopy operations ..... 3-24  
hardcopy operations, graphics ..... 4-9, 4-14  
HELLO command sequence ..... 4-7  
high-speed operation ..... 5-6  
home down cursor ..... 2-6  
home up cursor ..... 2-6  
horizontal lines, define pattern ..... 3-8, 3-10  
HP-IB cabling considerations ..... 8-2  
HP-IB cabling limitations ..... 8-3  
HP-IB configurations ..... 8-1  
HP-IB strapping ..... 7-16

## 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  
input/output modules, general description of ..... 1-3  
insert character ..... 2-7  
insert line ..... 2-7  
installation ..... 7-1  
installation, accessory ..... 7-6  
interface cable signals ..... 7-26  
interface signals ..... 5-3  
interfaces, data communications ..... 5-1, 7-13  
inverse video ..... 2-10

**J**

- jam mode, graphics ..... 3-6
- justifying graphics text ..... 3-27

**K**

- keyboard, general description of ..... 1-4
- Keyboard communications switches flowcharts ..... C-9
- keyboard disable ..... 2-17
- keyboard enable ..... 2-17
- 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 ..... 2-14
- keyboard overlay installation ..... 7-15

**L**

- labels, graphics text ..... 3-28
- Large Character Set coding ..... B-16
- Large Character Set installation ..... 7-6
- Large Character Set ..... 2-12, A-7
- latching keys, programming ..... 2-13
- line delete ..... 2-7
- Line Drawing character set ..... 2-12
- Line Drawing Set installation ..... 7-6
- line feed ..... 2-6
- line pattern, defining ..... 3-9
- Line Select ..... 5-38
- line strap ..... 5-10
- line type ..... 3-8
- Linear Charts ..... 3-33
- logic levels ..... 7-26
- LRC ..... 5-24

**M**

- Main Channel protocol ..... 5-14, 5-15
- margin, graphics text ..... 3-27
- margins ..... 2-5
- margins, Compatibility Mode ..... 3-35
- Math Character Set ..... 2-12
- Math Character Set installation ..... 7-6
- medium, copying to the end of ..... 4-14
- memory, general description of ..... 1-3
- memory addressing ..... 2-3
- memory configuration ..... 7-11
- memory control, graphics ..... 3-5
- memory link structure, general description of ..... 1-6
- memory lock ..... 2-6
- memory management, general description of ..... 1-5
- memory map ..... 7-11
- message blocks ..... 5-23
- message line ..... 2-2, 2-3
- messages, error ..... 6-12

- messages, transferring to device ..... 4-8
- modem disconnect command ..... 2-18
- modems ..... 5-5
- modes, graphics drawing ..... 3-6
- Monitor Mode ..... 5-14, 5-44
- moving text blocks ..... 2-8
- multicharacter transfers (block) ..... 5-23
- multicharacter transfers (point-to-point) ..... 5-7
- Multiplot ..... 3-29
- multipoint ..... 5-34
- multipoint communications ..... 5-1
- multipoint example ..... A-1
- multipoint strapping flowchart ..... 5-40

**N**

- NAK ..... 5-27
- network configurations ..... 5-2
- networks ..... 5-1
- next page ..... 2-6
- NOP, graphics ..... 3-15
- null characters ..... 5-7
- numeric field checking ..... 2-9

**O**

- opening the terminal ..... 7-2
- options and accessories ..... B-15
- origin, relocatable ..... 3-13
- output buffer (communications) ..... 5-25

**P**

- PA key ..... 5-34
- PAD ..... 5-28
- page ..... 2-6
- Page Full Break ..... 3-36
- Page Full Busy ..... 3-36
- page strap ..... 5-10
- page strapping in Format Mode ..... 5-11
- 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
- Pie Charts ..... 3-29
- plotting commands summary ..... 3-14
- point plot (line type pattern) ..... 3-8
- point-to-point ..... 5-6
- point-to-point communications flowchart ..... C-6
- point-to-point configuration ..... 5-14
- point-to-point strapping flowchart ..... 5-19
- polling ..... 5-36
- positioning the cursor ..... 2-4
- Power Down Protect Cable ..... 7-29
- power supply adjustment ..... 7-17

predefined pattern ..... 3-8  
 previous page ..... 2-6  
 primary terminal status ..... 6-2  
 printer, header listing ..... 4-4, 4-7  
 printer installation (9866) ..... 7-12  
 printer installation (9871) ..... 7-17  
 printer selection ..... 4-4, 4-11  
 printer status ..... 6-7  
 processor, general description of ..... 1-1  
 program load command ..... 2-18  
 programmable graphics functions ..... 3-1  
 programmable keys ..... 2-14  
 Programmers Reference Table ..... B-3  
 programming Keyboard Interface switches ..... 2-14  
 programming the latching keys ..... 2-13  
 programming the soft keys ..... 2-14  
 protected fields ..... 2-8  
 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, data ..... 3-16  
 raster data, CPU to terminal ..... 3-20  
 raster data, positioning on the display ..... 3-22  
 raster data, terminal to CPU ..... 3-23  
 raster dump ..... 3-19  
 raster dump, summary of commands ..... 3-24  
 raster output format ..... 3-20  
 raster scan, general description of ..... 1-3  
 read, binary ..... 4-14  
 read control byte ..... 4-10  
 record, copying ..... 4-5, 4-13  
 Record Mode, enable/disable ..... 4-5  
 record transmission ..... 4-12  
 recording file marks ..... 4-7, 4-11  
 relocatable, ASCII ..... 3-16  
 relocatable binary ..... 3-18  
 relocatable origin ..... 3-13  
 REMOTE key ..... 5-4  
 REMOTE key programming ..... 2-13  
 REPort command sequence ..... 4-7  
 reset ..... 2-17  
 reset terminal (full) ..... 2-17  
 reset terminal (soft) ..... 2-17  
 RESume command sequence ..... 4-7  
 rethreading tapes ..... D-1  
 retransmit last record ..... 4-10  
 RETURN ..... 2-6  
 RETURN key programming ..... 2-14  
 Reverse Channel Protocol ..... 5-14, 5-17  
 rewind ..... 4-7, 4-11  
 roll down ..... 2-6  
 roll up ..... 2-6  
 ROM error ..... 7-39  
 RS232C signals ..... 7-27  
 rubber band line ..... 3-14  
 RVI ..... 5-28

## S

scale, line pattern ..... 3-9  
 scale switch ..... 5-4  
 scaled Compatibility Mode ..... 3-36  
 secondary terminal status ..... 6-4  
 selection, device ..... 4-4, 4-11  
 selection, device function ..... 4-2, 4-11  
 selection (multipoint) ..... 5-37  
 selection of character sets ..... 2-10  
 self test, cartridge tapes ..... 7-43  
 self test, data communications ..... 7-43  
 self test, HP-IB ..... 7-49, 8-3  
 self test, multipoint ..... 7-47  
 self test, point-to-point ..... 7-44  
 self test, terminal-to-terminal loop back ..... 7-49  
 Self Test command ..... 2-18  
 self test connectors ..... 7-44  
 Self Test procedure ..... 7-39  
 send byte count ..... 4-10  
 send display ..... 2-17  
 sensing the cursor position ..... 2-4  
 Serial Printer Interface installation ..... 7-12  
 set mode, graphics ..... 3-6  
 set tab ..... 2-5  
 SET Time/Date command sequence ..... 4-7  
 Shared Peripheral Interface installation ..... 7-14  
 short incremental, binary ..... 3-17  
 SHow command sequence ..... 4-8  
 signal levels ..... 7-27  
 signal line control ..... 5-9  
 signals, interface ..... 5-3, 7-26  
 SKip Line, Page, End-of-Data command sequence ..... 4-8  
 soft key applications ..... 2-16  
 soft key label line ..... 2-2, 2-3  
 soft key labels ..... 2-16  
 soft key labels, displaying ..... 2-16  
 soft key programming ..... 2-14  
 soft key triggering ..... 2-16  
 soft reset ..... 2-17  
 space compression ..... 5-26  
 specifications ..... B-2  
 speed switches (block) ..... 5-33  
 status ..... 6-1  
 status, block configuration ..... 5-38  
 status, command ..... 6-12  
 status, device function ..... 4-7, 4-11  
 status, device ..... 6-6  
 status, graphics ..... 6-8  
 status, terminal primary ..... 6-2  
 status, terminal secondary ..... 6-4  
 status, terminal ..... 6-1  
 status interpretation ..... 6-1  
 strap options (block) ..... 5-24  
 strapping, 13296 Shared Peripheral Interface ..... 7-16  
 strapping, 32K RAM Memory PCAs ..... 7-10  
 strapping, Asynchronous Multipoint (13260C) ..... 7-24  
 strapping, Compatibility Mode ..... 3-36  
 strapping, Control Memory PCAs ..... 7-11



strapping, Extended Asynchronous PCA (13260B) ... 7-23  
 strapping, G and H straps ... 5-9  
 strapping, Keyboard Interface PCA (block) ... 7-21  
 strapping, Keyboard Interface PCA (point-to-point) ... 7-19  
 strapping, Line/Page ... 5-10  
 strapping, point-to-point ... 5-12, 5-19  
 strapping, Synchronous Multipoint (13260D) ... 7-25  
 strapping, Terminal Duplex Register PCA ... 7-12  
 strapping option selection ... 7-17  
 STX ... 5-27  
 Suspend command sequence ... 4-8  
 switch settings, data communications ... 5-4, 5-32  
 SYN ... 5-28  
 Synch characters ... 5-26  
 synchronization, graphics ... 3-15  
 synchronous compatibility (block) ... 5-38  
 Synchronous Multipoint Interface (13260D) straps ... 7-25

## T

tab ... 2-6  
 tabbing in Forms Mode ... 2-9  
 tabs ... 2-5  
 tabs, clearing ... 2-5  
 tabs, setting ... 2-5  
 tabs, using ... 2-5  
 tape conditioning ... 4-5, 4-11  
 tape rethreading ... D-1  
 tape runoff ... D-1  
 tape selection ... 4-4, 4-11  
 tape status ... 6-7  
 tape storage capacity ... B-2  
 tape test command ... 4-9, 4-11  
 tape test procedure ... 7-43  
 tape units, general description of ... 1-6  
 teletype compatible operation ... 5-9  
 TELL command sequence ... 4-8  
 terminal address ... 5-35  
 terminal architecture ... 1-2  
 terminal control functions ... 2-13  
 terminal display adjustments ... 7-4  
 terminal ID ... 5-35  
 terminal mainframe, general description of ... 1-1  
 terminal networks ... 5-1, 7-9  
 terminal PCA locations ... 7-3  
 terminal self-test command ... 2-18  
 terminal status ... 6-1  
 terminals, connecting ... 5-1  
 terminating graphics sequences ... 3-15  
 Test DATacomm command sequence ... 4-9  
 Test HP-IB command sequence ... 4-9  
 test patterns ... 7-39  
 test tape ... 4-2  
 Test Tapes command sequence ... 4-9  
 Test TErминаl command sequence ... 4-9  
 text, Compatibility Mode ... 3-39  
 text, graphics ... 3-25  
 text moving ... 2-8  
 time, displaying of ... 4-8  
 time, setting of ... 4-7  
 to device selection ... 4-4, 4-11

TRansfer (8-bit binary) command sequence ... 4-9  
 transferring data from computer to device ... 4-13  
 transferring data from device to computer ... 4-12  
 transferring data ... 4-9, 4-12  
 transmit byte count ... 4-12  
 transmit file ... 4-10, 4-12  
 TRANSMIT indicator ... 5-4  
 transmit last record ... 4-10  
 transmit next record ... 4-10  
 transmit only fields ... 2-9  
 transmit record ... 4-12  
 Transparency Mode ... 5-26

## U

underline ... 2-10  
 unprotected fields ... 2-8  
 unscaled Compatibility Mode ... 3-36  
 user defined area pattern ... 3-10  
 user defined line pattern ... 3-9  
 user defined pattern ... 3-8

## V

vectors ... 3-15  
 Verify Mode, enable/disable ... 4-5  
 vertical lines, define pattern ... 3-8, 3-10  
 video hardcopy ... 3-24  
 Video Interface installation ... 7-12  
 voltage selection ... 7-5  
 VRC ... 5-24

## W

WACK ... 5-27  
 wait command ... 2-17  
 window, graphics ... 3-13  
 window control ... 2-1, 2-2  
 wraparound ... 2-7  
 WRU ... 5-38

## Z

zoom ... 3-5





