

# concept AVT

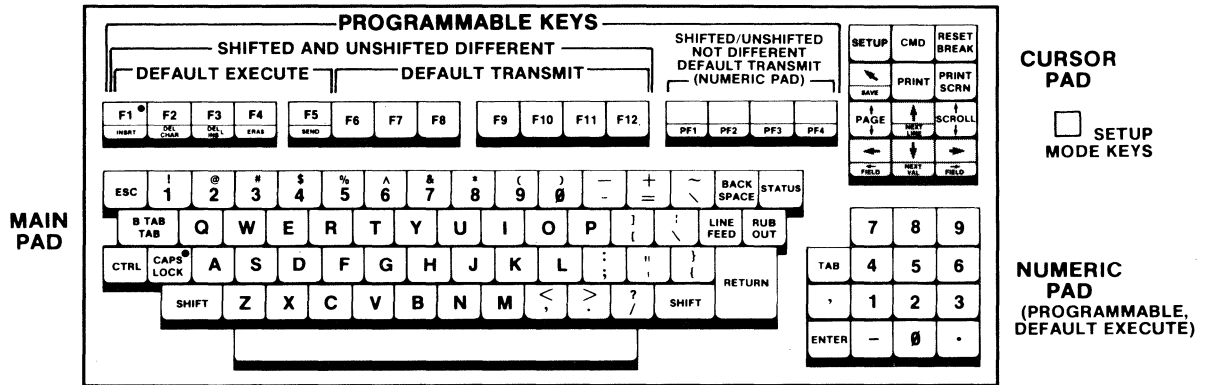
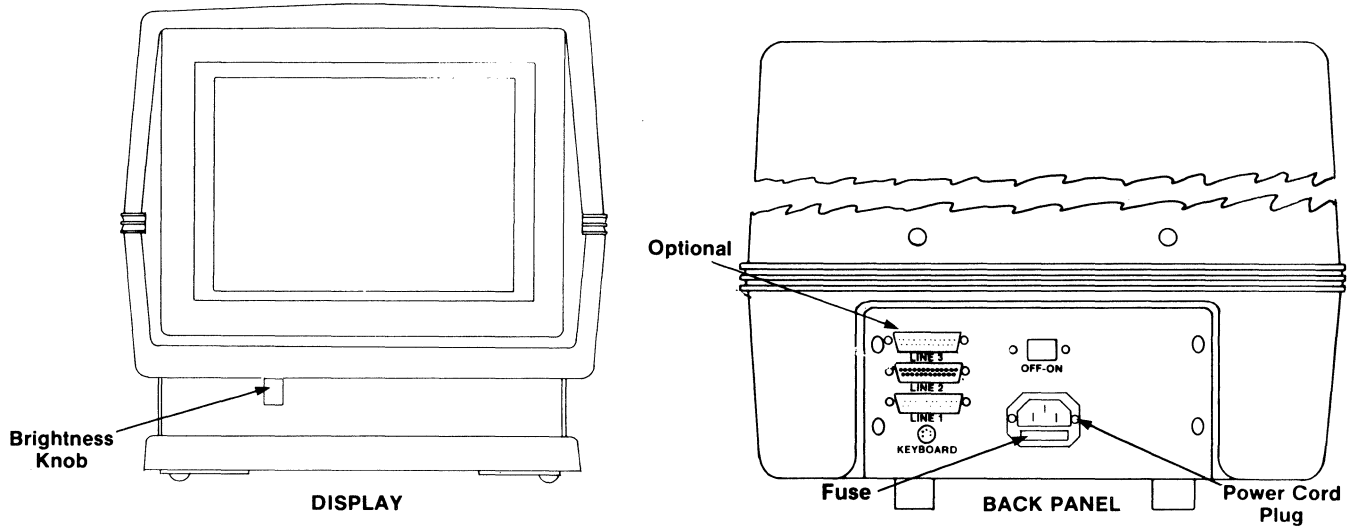
## USERS MANUAL

**HDS**

human designed systems, inc.

3440 Market Street  
Philadelphia, PA 19104  
215-382-5000

**Figure 1-1  
CONCEPT CONTROLS AND CONNECTIONS**



**USER STATUS LINE**

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
KB	50	HDX	1S	NO	REM	BLK	U/L	lin: col	ASC	top: bot: lft: rgt	ANS	S	C	V	a	CT	VPPPPP	INS/OFF	
L1	75	FDX	2S	EV	LOC	CHR	CAP	001 001	CH0	001 001 001 001	V52				A	CE		INS/ON	
L2				OD				:	CH1	:	:	:	:						
L3	9600			MK				192 132	CH2	192 192 132 132									
				SP					OSB										
									OSO										
									OS1										
									APL										

REF	DESCRIPTION	STATUS LINE START/LENGTH	EXPLANATION
A*	Status line identifier	2 2	KB = keyboard; L1-L3 = lines 1-3.
B*	Baud rate	5 4	See Command Parameter <i>baud</i> .
C*	Duplex	10 3	HDX = half duplex; FDX = full duplex.
D*	Stop bits	14 2	1S = 1 stop bits; 2S = 2 stop bits.
E*	Parity	17 2	NO = none; EV = even; OD = odd; MK = mark(1); SP = space (0).
F*	Local/remote	20 3	LOC = local; REM = remote.
G*	Block/character mode	24 3	BLK = block; CHR = character.
H*	Caps lock	28 3	U/L = upper and lower case; CAP = caps lock.
I	Cursor position	32 7	lin = line #: col = column #.
J*	Char. set (ASCII/APL)	40 3	APL and values beginning with O have character overstrike on.
K*	Window definition	44 15	top = top; bot = bottom; lft = left; rgt = right.
L*	ANSI/VT52 mode	60 3	ANS = ANSI; V52 = VT52 mode.
M*	Screen width	64 1	Place holder only (see Setup Mode).
N*	Cursor representation	66 1	Place holder only (see Setup Mode).
O*	Video/reverse video	68 1	Place holder only (see Setup Mode).
P*	Wraparound	70 1	A = both character and cursor Wraparound; a = at least one Wraparound off.
Q*	Cursor keypad operation	72 2	CT = one or more keys not in execute mode; CE = all cursor keypad keys in execute mode (see Setup Mode).
R	Version number	75 6	V = Product ID; first P = version of PROM 1; second P = version of PROM 2, etc.
S	Insert mode indicator	82 7	Only visible when screen width is 132.

\* Fields changeable through Setup Mode; see Setup Mode section for additional information.

## Introduction

The Concept AVT series of interactive display terminals offer a host of display, editing, and communications features to enhance both general interactive usage and special applications. Fundamental to the design of the Concept terminal is the principal that the availability of sophisticated features should not complicate the use of the terminal for the user not initially interested in using such features.

The Concept terminal is designed to provide a friendly, easy to use interface between the computer and users with varying degrees of expertise, including data entry or text editing operators, technical professionals, and sophisticated application system developers.

The information in this document applies to the following Concept terminals:

Concept AVT	ASCII Video Display Terminal
Concept AVT-APL	APL (and ASCII) Video Display Terminal
Concept GVT	Vector Graphics Video Display Terminal
Concept GVT-APL	APL Vector Graphics Video Display Terminal

### Note to APL Users

The ASCII character set is used throughout this manual. All definitions, explanations, and examples of terminal commands are presented in ASCII. See Section 2.1.1 for an explanation of the relationship between the APL and ASCII character sets.

Keyboards supplied with APL terminals show the APL symbols on the keytops, and the corresponding ASCII symbols on the key fronts.

This manual is organized in a manner consistent with the Concept design philosophy. The first chapter, Getting Started with the Concept AVT, provides the new user with basic information on how to unpack and set up the terminal, verify that it is working properly, attach it to his or her computer system, and use it as a basic interactive terminal.

The second chapter, Programmers Guide, describes in detail the capabilities of the terminal, and indicates ways in which they might be exploited in a variety of applications. In general, the features described in this chapter are presented in order of increasing sophistication.

Chapter 3, Command Reference, provides a complete specification of the operation of each terminal command. Commands are grouped by function.

Chapter 4, Alternate Processing Modes, lists all of the terminal operating characteristics that can be set using the Set and Reset Mode commands.

Examples are provided throughout the manual to help the reader understand how to use Concept terminal features.

## Compliance with Subpart J of Part 15 of FCC Rules

### Warning:

This equipment generates and uses radio frequency energy. If not installed and used properly, i.e., in strict accordance with the instructions manual, it may cause harmful interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment.

Shielded cables should be used with this unit to insure compliance with the class A limits.

Operation of this equipment in a residential area is likely to cause interference, in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

The following procedures may help to alleviate the Radio or Television Interference Problems.

1. Reorient the antenna of the receiver receiving the interference.
2. Relocate the equipment causing the interference with respect to the receiver (move or change relative position).
3. Reconnect the equipment causing the interference into a different outlet so the receiver and the equipment are connected to different branch circuits.
4. Remove the equipment from the power source.

### Note:

The user may find the following booklet prepared by the FCC helpful: "How to Identify and Resolve Radio-TV Interference Problems". This booklet is available from the U.S. Printing Office, Washington, D.C. 20402. Stock No. 004-000-00345-4.

## TABLE OF CONTENTS

### 1 Getting Started with the Concept AVT

	Page
1.1	Introduction . . . . . 1-1
1.2	Unpacking, Setting Up and Powering On . . . . . 1-1
1.3	Setup Mode . . . . . 1-2
1.4	Local Operation . . . . . 1-3
1.5	Keyboard - Display Operation . . . . . 1-4
1.6	Setting Up Communications . . . . . 1-6
1.6.1	Communicating with the Host Computer . . . . . 1-6
1.6.2	Connecting a Printer to the Concept AVT . . . . . 1-7
1.7	VT100 Compatibility . . . . . 1-9
1.8	The Keyboard . . . . . 1-10
1.8.1	Main Pad . . . . . 1-10
1.8.2	Numeric Pad . . . . . 1-12
1.8.3	Cursor Pad . . . . . 1-12
1.8.4	Programmable Function Keys . . . . . 1-15
1.8.4.1	Default Execute Keys . . . . . 1-15
1.8.4.2	Default Transmit Keys . . . . . 1-16
1.9	Terminal Characteristics that can be Changed in Setup Mode . . . . . 1-16
1.9.1	User Status Lines . . . . . 1-16
1.9.2	Tab Status Lines . . . . . 1-19

### 2 Programmers Guide

2.1	Some Key Concepts . . . . . 2-1
2.1.1	ASCII/APL Chart Location . . . . . 2-1
2.1.2	Terminal Commands . . . . . 2-1
	Control Codes . . . . . 2-2
	Escape Sequences . . . . . 2-3
2.1.3	Alternate Processing Modes . . . . . 2-4
2.1.4	Character Processing . . . . . 2-5
	Ordinary Characters . . . . . 2-6
	Control Codes . . . . . 2-6
	Escape Sequences . . . . . 2-7
	Communication Control Codes . . . . . 2-7
2.1.5	Lines and Devices . . . . . 2-7
	Keyboard Communication Line . . . . . 2-8
	Requesting Device . . . . . 2-8
2.2	Status and Non-Volatile Memory . . . . . 2-9
2.2.1	Status Lines . . . . . 2-9

	Display Status Information . . . . .	2-10
	Transmitting Status Information . . . . .	2-10
	Background Status Line . . . . .	2-10
2.2.2	Terminal Identifier . . . . .	2-10
2.2.3	Programmable Answerback Message . . . . .	2-11
2.2.4	Non-Volatile Memory (NVM) . . . . .	2-11
	Reset to Power-Up State . . . . .	2-11
	Reset to Factory Default State . . . . .	2-12
2.3	Trouble Shooting and Debugging . . . . .	2-13
2.3.1	Checking Out the Terminal . . . . .	2-13
	Firmware and Display Memory . . . . .	2-13
	Communications . . . . .	2-14
	Character Sets . . . . .	2-14
2.3.2	Transparent Mode . . . . .	2-14
2.3.3	Programmable Message Characters . . . . .	2-15
	Escape Message Characters (ESC) . . . . .	2-16
2.3.4	Data Communications Considerations . . . . .	2-16
	Stop Bits and Parity . . . . .	2-17
	Buffer Overflow Control . . . . .	2-17
	Response to ↑S/↑Q from the Host . . . . .	2-18
2.4	Programmable Keys and Functions . . . . .	2-20
2.4.1	Execute and Transmit Mode . . . . .	2-20
	Programmable Key Defaults . . . . .	2-21
2.4.2	Programming a Function Key . . . . .	2-22
	Programming a Key to Transmit Data . . . . .	2-22
	Terminal Commands on Programmable Keys . . . . .	2-23
2.4.3	Numeric Pad Keys . . . . .	2-24
2.4.4	Cursor Pad Keys . . . . .	2-25
2.4.5	Other Programmable Functions . . . . .	2-25
	Latent Expression . . . . .	2-25
	Programming Keys from NVM . . . . .	2-26
	Alert Line Message . . . . .	2-27
	Answerback Message . . . . .	2-27
2.5	Editing . . . . .	2-28
2.5.1	Insert Mode . . . . .	2-28
2.5.2	Delete Character . . . . .	2-29
2.5.3	Insert Line . . . . .	2-29
2.5.4	Delete Line . . . . .	2-29
2.5.5	Erase (Clear) . . . . .	2-29
2.5.6	Editing Extent . . . . .	2-29
2.5.7	Clear Characteristics . . . . .	2-30
2.6	Character Sets and Attributes . . . . .	2-31
2.6.1	Character Sets . . . . .	2-31
	Normal and Alternate Character Set . . . . .	2-31
	APL and Overstrike Mode . . . . .	2-32
2.6.2	Character Attributes . . . . .	2-33
	Changing Character Attribute . . . . .	2-35
2.6.3	Attribute Lists . . . . .	2-36
2.6.4	Block Attribute Change . . . . .	2-37
2.6.5	Character / Attribute Replacement . . . . .	2-37

2.7	Display Memory and Windows . . . . .	2-38
2.7.1	Allocating Display Memory . . . . .	2-38
2.7.2	Display Width (80 or 132 Columns) . . . . .	2-38
2.7.3	Interactive Uses . . . . .	2-39
2.7.4	Windows . . . . .	2-40
	Window Lists . . . . .	2-42
	Selecting a Window . . . . .	2-41
	Defining a Window . . . . .	2-42
	Some Applications . . . . .	2-43
2.8	Peripherals and Networking . . . . .	2-45
2.8.1	Printer Port (Line 2) . . . . .	2-45
	Printer Commands . . . . .	2-46
	The PRINT Key . . . . .	2-46
	Escape Sequences . . . . .	2-46
	The PRINT/SCRN Key . . . . .	2-47
2.8.2	Networking . . . . .	2-47
	Set Output Network . . . . .	2-49
2.8.3	Multiple Computer Applications . . . . .	2-49
	Establishing Communications . . . . .	2-49
	Talking to the Other Computer . . . . .	2-50
	Block Transfer of Data Between Hosts . . . . .	2-51
	Direct Transfer Between Hosts . . . . .	2-52
	Multiple Hosts and Multiple Windows . . . . .	2-53
2.9	Block Mode . . . . .	2-55
2.9.1	Interactive Uses . . . . .	2-55
	Host System Echo . . . . .	2-56
	Sending Data to the Host . . . . .	2-56
2.9.2	Block Mode Applications . . . . .	2-58
	Transmission Under Host Control . . . . .	2-58
	Transmit Delay . . . . .	2-59
2.10	Forms Handling . . . . .	2-60
2.10.1	Fields and Protection . . . . .	2-60
2.10.2	Protected Field Overwrite . . . . .	2-61
2.10.3	Protected Field Erasure . . . . .	2-61
2.10.4	Scrolling . . . . .	2-61
2.10.5	Tab Processing . . . . .	2-62
2.10.6	Auto Tabs . . . . .	2-62
2.10.7	Line Drawing . . . . .	2-62
2.10.8	Block Attribute Change . . . . .	2-63

### 3 Command Reference

3.1	Introduction . . . . .	3-1
3.1.1	Format of Command Entries . . . . .	3-1
	1. Command Name . . . . .	3-1
	2. Command Sequence . . . . .	3-1
	3. Special Key . . . . .	3-1
	4. Description of Command Function . . . . .	3-1

	5. Parameters . . . . .	3-1
	6. Interaction with Alternate Processing Modes . . . . .	3-2.
	7. Notes . . . . .	3-2
	8. Examples . . . . .	3-2
3.1.2	Types of Command . . . . .	3-2
3.1.3	Command Sequences . . . . .	3-2
3.1.4	Command Sequence Introducer . . . . .	3-3
	ESC [ . . . . .	3-3
3.1.5	Parameters . . . . .	3-4
3.1.6	Error Conditions . . . . .	3-4
3.2	General . . . . .	3-6
	Self Test . . . . .	3-6
	Reset Terminal . . . . .	3-6
	Save Configuration in NVM . . . . .	3-7
	Change Message Character . . . . .	3-7
	Fill Character . . . . .	3-8
	Select Attribute List . . . . .	3-8
	Copy Attribute List . . . . .	3-8
	Select Window . . . . .	3-9
3.3	APL/ASCII (Character Sets) . . . . .	3-10
	Define Normal Character Set . . . . .	3-10
	Use Normal Character Set . . . . .	3-10
	Define Alternate Character Set . . . . .	3-10
	Use Alternate Character Set . . . . .	3-10
3.4	Status Information . . . . .	3-11
	Transmit Terminal Identifier . . . . .	3-11
	Transmit Device Status . . . . .	3-11
	Display Status Line . . . . .	3-11
	Next Status Line . . . . .	3-12
	Transmit Status Line . . . . .	3-12
	Toggle/Clear Status Line . . . . .	3-13
	Set Background Status Line . . . . .	3-13
	Transmit Answerback Message . . . . .	3-14
3.5	Cursor Controls . . . . .	3-15
	Carriage Return . . . . .	3-15
	Line Feed . . . . .	3-15
	Index (Line Feed) . . . . .	3-15
	Reverse Index . . . . .	3-16
	New Line (Line Feed/CR) . . . . .	3-16
	Cursor Up . . . . .	3-16
	Cursor Down . . . . .	3-16
	Cursor Right . . . . .	3-16
	Cursor Left . . . . .	3-17
	Backspace . . . . .	3-17
	Cursor Up to Left Margin . . . . .	3-17
	Cursor Down to Left Margin . . . . .	3-17
	Position Cursor . . . . .	3-17
	Position Cursor to Column . . . . .	3-18
	Position Cursor to Line . . . . .	3-18



	End of Text . . . . .	3-18
	Transmit Cursor Position . . . . .	3-19
	Save Cursor Position and Attributes . . . . .	3-19
	Restore Cursor Position and Attributes. . . . .	3-19
	Tab . . . . .	3-19
	Forward Tab . . . . .	3-19
	Backward Tab . . . . .	3-20
	Set Tab . . . . .	3-20
	Clear Tabs . . . . .	3-20
	Tab Control . . . . .	3-20
3.6	Editing . . . . .	3-21
3.6.1	Clearing Display Memory . . . . .	3-21
	Set Clear Characteristics . . . . .	3-22
	Form Feed . . . . .	3-23
	Select Editing Extent . . . . .	3-23
	Toggle Insert Mode . . . . .	3-23
	Delete Character . . . . .	3-24
	Insert Line . . . . .	3-24
	Delete Line . . . . .	3-25
	Erase in Window . . . . .	3-25
	Erase in Line . . . . .	3-26
	Erase in Field . . . . .	3-26
	Set Margin Bell . . . . .	3-26
3.7	Display Controls . . . . .	3-28
3.7.1	Character Attributes . . . . .	3-28
	Select Alternate Attributes . . . . .	3-29
	Select Default Attributes . . . . .	3-30
	Block Attribute Change . . . . .	3-30
	Block Character Change . . . . .	3-31
	Transparent Mode On . . . . .	3-32
	Transparent Mode Off . . . . .	3-32
	Protection On . . . . .	3-32
	Protection Off . . . . .	3-33
3.8	Multiple Devices . . . . .	3-34
	Print/Line 3 Control . . . . .	3-34
	Set Output Network . . . . .	3-35
	Set Keyboard Communication Device . . . . .	3-36
3.9	Transmission Control . . . . .	3-37
	Set Parity/Parity Checking . . . . .	3-37
	Set Baud Rate . . . . .	3-37
	Set Stop Bits . . . . .	3-38
	CTS/RTS (Transmit) Protocol . . . . .	3-38
	Buffer Overflow (Receive) Protocol . . . . .	3-39
	Stop Transmission . . . . .	3-40
	Resume Transmission . . . . .	3-40
	Transmit . . . . .	3-40
	Set Transmit Extent . . . . .	3-40
	Set Transmit Delay . . . . .	3-41
	Break . . . . .	3-41
	Start of Print/Transmit . . . . .	3-42

3.10	Screen Controls . . . . .	3-43
	Scroll Down . . . . .	3-43
	Scroll Up . . . . .	3-43
	Page Down . . . . .	3-43
	Page Up . . . . .	3-43
	Set Top of Screen . . . . .	3-44
	Define Scrolling Region . . . . .	3-44
	Set Display Pages . . . . .	3-45
	Define Window . . . . .	3-45
3.11	Keyboard Controls . . . . .	3-47
	Lock Keyboard . . . . .	3-47
	Lock Keyboard . . . . .	3-47
	Cursor Pad Key Settings . . . . .	3-47
	Cursor Pad Operation . . . . .	3-48
	Ring Keyboard Bell . . . . .	3-48
3.12	Function Keys and Stored Data . . . . .	3-49
	Program Function Key . . . . .	3-49
	Program Alert Line Message . . . . .	3-51
	Program Answerback Message . . . . .	3-51
	Program Latent Expression . . . . .	3-51
	Display/Edit Function Key . . . . .	3-52
	Execute Function Key/Stored Data . . . . .	3-53
	Program Numeric Pad-Application Mode . . . . .	3-54
	Program Numeric Pad-Numeric Mode . . . . .	3-54
	Programmable Keys . . . . .	3-55
4	Alternate Processing Modes	
4.1	Introduction . . . . .	4-1
4.2	Set and Reset Mode Commands . . . . .	4-1
4.3	Defaults . . . . .	4-1
4.4	Examples . . . . .	4-2
4.5	Listing of Alternate Processing Modes . . . . .	4-2

#### Appendix A - Character Sets

A.1	ASCII . . . . .	A-1
A.2	VT100 and Concept Special Graphics . . . . .	A-2
A.3	APL . . . . .	A-3

#### Appendix B - Status Lines

B.1	User Status Line . . . . .	B-1
B.2	Programmer Status Line . . . . .	B-2
B.3	Modes Status Line . . . . .	B-3
B.4	Programmable Message Character Status Line . . . . .	B-5
B.5	Tabs Status Line . . . . .	B-5

B.6 Alert Message Status Line . . . . .	B-5
---	-----

#### Appendix C - Summary of Control Codes and Escape Sequences

C.1 Control Codes . . . . .	C-1
C.2 ESC Followed by a Single Character . . . . .	C-2
C.3 Sequences Preceded by ESC [ . . . . .	C-4
C.4 Sequences Preceded by ESC [ with ! as Intermediate . . . . .	C-6
C.5 Sequences Preceded by ESC [ with * as Intermediate . . . . .	C-7

#### Appendix D - Communications Interfaces

D.1 Basic Communications . . . . .	D-1
D.2 Current Loop (20 Milliamp) Interface Option . . . . .	D-3

#### Appendix E - VT52 Terminal Type Operation

E.1 Commands Recognized in VT52 Mode . . . . .	E-1
E.2 Numeric Pad-Numeric and Application Modes . . . . .	E-1
E.3 VT52 Cursor Keypad Sequences . . . . .	E-2

#### Appendix F - Timing Considerations

F.1 Fill Character . . . . .	F-1
F.2 Input Buffers . . . . .	F-1
F.3 Buffer Overflow Control . . . . .	F-2

#### Appendix G - Factory Default Settings

#### Appendix H - Software Version 33333

H.1 Halt Terminal . . . . .	H-1
H.2 Host Overflow Control . . . . .	H-1
H.3 Keyboard Communication Device . . . . .	H-2
H.4 GVT Terminal Identifier . . . . .	H-3
H.5 Line 3 Stop Bits and Parity . . . . .	H-3



## **Getting Started with the Concept AVT**

### **1.1 Introduction**

Refer to Figure 1-1 (inside front cover), showing the Concept AVT display, back panel, keyboard and status line, while reading the following sections.

### **1.2 Unpacking, Setting Up and Powering On**

Upon receipt of the terminal and before unpacking it, inspect it for damage to the external packaging material. Any damage should be reported immediately to HDS.

If there is no evidence of shipping damage, unpack the terminal. The box should contain the following separate items:

1. Display
2. Keyboard
3. Power Cord
4. Communication Cable
5. Packet of Reference Material

Assuming there is no physical evidence of shipping damage, set up the terminal as follows:

1. Push the keyboard connector into the labeled receptacle on the back panel.
2. Attach one end of the communication cable to the LINE 1 receptacle on the back panel, securing the connection with the connector screws provided. The other end of this cable will be connected to your computer or data communication equipment (see Section 1.6).
3. Insuring first that the ON-OFF switch on the Back Panel is in the OFF position, plug the power cord connector into the recessed power receptacle on the back panel, and the plug end into a normal 115 volt AC, three pronged (grounded) outlet. DO NOT disable the grounding by using a two pronged ungrounded adapter plug.
4. Turn the ON-OFF switch on the back panel (Figure 1-1) to the ON position. The keyboard bell will sound on power up. The screen will take a few moments to come on. A flashing block should appear in the upper left corner of a blank, dark screen. The flashing block is the cursor, which indicates where the next character typed on the keyboard or received from the computer will be displayed. If the cursor is dim or does not appear at all, or if the screen is light with lines through it, the display brightness may have become misadjusted in shipping. The brightness level may be adjusted using the knob under the left

side of the display (Figure 1-1).


5. A self test is performed by the terminal on power up. If an error is detected, an Alert Line is displayed in inverse video (dark characters on a light background) on the bottom (25th) line of the screen. Contact HDS or your terminal supplier if any self test errors are shown on the Alert Line. To remove an Alert Line, press the STATUS key (see Figure 1.1)
6. After the display has come on with the cursor in the upper left corner, check that the keyboard is operating by pressing the INSRT key (Figure 1-1). The light on the key should come on. Type INSRT again and the light should go off. If either the display or the keyboard did not operate as it should in the above procedure, the user should report the problem to HDS or the terminal supplier.
7. During use, the vents on the top on the terminal should never be covered over with papers or other material, since this might cause the terminal to overheat.

### 1.3 Setup Mode

Setup Mode provides a simple way of changing the terminal configuration in order to establish compatibility with the user's hardware and software environment. To enter Setup Mode, press the SETUP key located in the upper left corner of the Cursor Pad (see Figure 1-1). This will cause the User Status Line to appear in reverse video at the bottom of the screen. Note: Only Setup Mode functions may be performed while the terminal is in Setup Mode. Once changed, the terminal configuration may be saved in non-volatile memory (NVM) so that it will be in effect the next time the terminal is turned on (see below).

When in Setup Mode, you may change one or more of the fields displayed on the User Status Line. All of the fields that may be changed through Setup Mode are described in Section 1.9 at the end of this chapter. If you do not wish to change any of the fields, just press the SETUP key again to exit from Setup Mode.

To specify the field you wish to change, strike one of the FIELD keys (marked  $\rightarrow$  or  $\leftarrow$  on top) until the desired field appears as a dark block in the status line. Then strike the NEXT VAL key (marked  $\downarrow$  on top) until the desired value appears. Regardless of how many times this process is repeated, the features in effect will be those that appear when you exit from Setup Mode by pressing the SETUP key again.

To save the current values of all the fields in non-volatile memory, press the SAVE key (marked  on top) before exiting from Setup Mode. Any changes made after pressing the SAVE key will affect current terminal operation, but will not affect what has been saved. The combination of settings saved in NVM will be in effect whenever the terminal is turned on or reset. To change the settings saved in NVM, simply enter Setup Mode, choose the new settings, and press the SAVE key. Then exit from Setup Mode by pressing the SETUP key again.



#### 1.4 Local Operation

Before connecting the Concept AVT to your host computer, it may be helpful to become familiar with the way the terminal itself works by operating it briefly in Local Mode. Users already familiar with computers and computer terminals may want to skip the next two sections and continue at Section 1.7.

Using Setup Mode, configure the terminal as described below to demonstrate the features as they are described.

1. Make sure the terminal is in "factory default" state. This means that all options are set as they were when the terminal left the factory. The terminal should be in this condition when unpacked, but just to make sure, type the four key sequence

CMD [ 9 ~

2. Put the terminal in Setup Mode: strike the SETUP key.
3. Put the terminal in Local Mode: strike the FIELD key (marked  on top) until REM appears in inverse video (light characters on dark ground), and then strike the NEXT VAL key (marked  on top) once, or until LOC appears, indicating that the terminal is now in Local Mode.
4. Select Wraparound for both cursor and characters: strike the FIELD KEY until the FIELD marked 'a' appears in inverse video; then strike the NEXT VAL key once, or until the field appears as 'A'. This allows characters typed on the screen to "wrap around" the ends of lines without the operator having to enter a carriage return and a line feed.
5. Put the Cursor Pad in Execute Mode: strike the FIELD key until the field marked 'CT' appears in inverse video, and then strike the NEXT VAL key once, or until the field appears as 'CE'. This sets the cursor movement keys to operate locally.

6. Change the window definitions to 001; 096; 001; 080 (or 001; 192; 001; 080 for terminals with 8 pages of memory).
7. Exit from Setup Mode by striking the SETUP key again.
8. Display the User Status Line: strike the STATUS key located in the upper right corner of the Main Pad (see Figure 1-1). The Status Line is essentially independent of what is displayed on the rest of the screen, and may be removed at any time simply by striking the STATUS key again.

### 1.5 Keyboard - Display Operation

The Main Pad of the keyboard (Figure 1-1) is similar to that of an IBM Selectric ('IBM' and 'Selectric' are trademarks of IBM Corp.) typewriter, and, in Local Mode, operates in much the same way. See Section 1.8 for a detailed description of the operation of each key. Characters typed on the keyboard appear on the screen. The cursor, displayed as a flashing block, indicates the position in which the next character typed will be displayed. As each character is typed, it appears on the screen and the cursor moves one position to the right. When the cursor reaches the last column at the right, it moves automatically to the first column on the next line (assuming that Wraparound Mode has been selected as suggested in step 4 above).

The keys in the Main Pad (Figure 1-1) labeled RETURN, LINE FEED, and BACK SPACE operate analogously to those in a typewriter - moving the cursor to the left margin, next line, and the previous column of the display respectively. One important exception is that the RETURN key does not automatically move the cursor to the next line, but returns it to the left margin of the current line. (The RETURN key can be made to automatically generate a line feed: see Alternate Processing Modes.) As on a typewriter, the SHIFT key causes the shifted version of a key to be generated when held down while the key is struck. For example, shifted alphabetic keys (A - Z) cause the upper case characters to be generated. Type some characters on the screen to verify that keyboard and display are working properly.

The Concept AVT stores 96 (optionally 192) 80 column lines of text in Display Memory, as shown in Figure 1-2. Thus, the user can type 96 lines of text before losing information. The screen, however, can display only 24 of these 96 lines at a time. As the cursor moves down past line 24, the 24 line screen display area will move down to keep the cursor (that is, the area where the user is typing) on the screen. For example, assume lines 10 - 33 are displayed on the screen, and the user types a line feed on line 33. The screen will shift down so that it now displays lines 11 - 34 of Display Memory, and the cursor will move to line 34. The user should type characters and line feeds through the 96 lines of Display Memory to become familiar with the movement of the screen with respect to Display Memory. Field I of the User Status Line shows the line and column position of the cursor. Line and column numbering begin with 1. Note that the 25th line of the screen, the line used to display status information, is independent of and does not af-



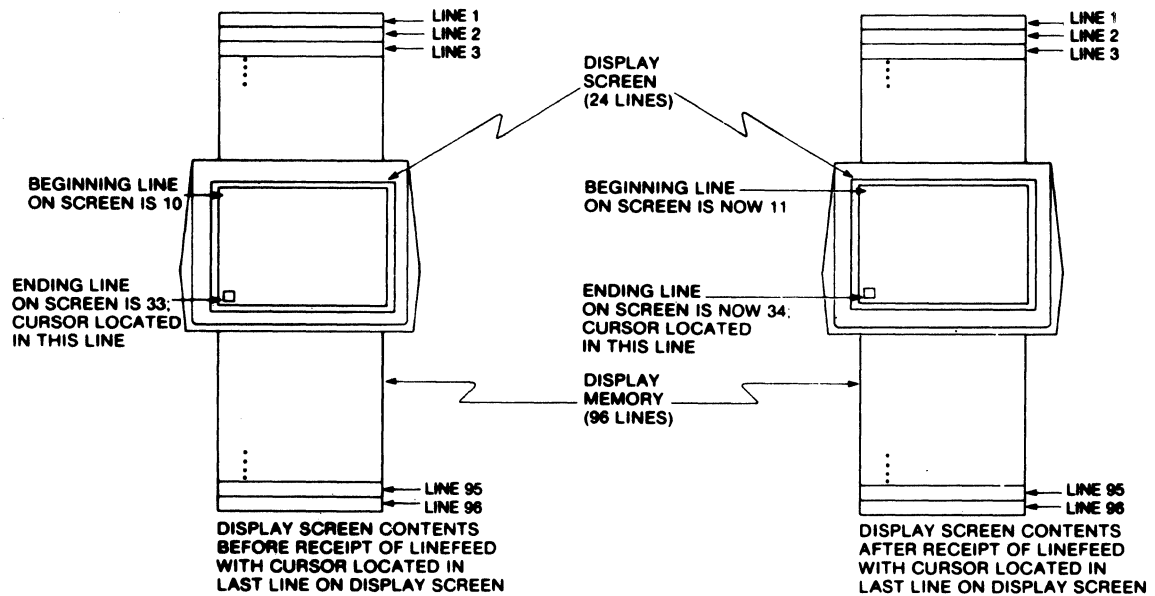


FIGURE 1-2  
Display Memory and Display Screen Relationships

fect movement of the 24 line screen through Display Memory.

As line feeds are typed on the last line of display memory, the 96 lines of display memory are "scrolled up". This means that each time a line feed is typed on the last line of display memory, every existing line of display memory is moved up one line, the top line of display memory is lost, and a new blank line is created at the bottom.

The configuration of Display Memory can be changed from 96 (or 192) 80 column lines to 56 (or 112) 132 column lines through Setup Mode (see Sections 1.3 and 1.9). As before, the screen can display 24 of these lines at a time. In this case, the first line of display memory will be scrolled up and lost after the user has entered 56 lines of data.

The cursor control keys (  $\uparrow$  ,  $\downarrow$  ,  $\rightarrow$  ,  $\leftarrow$  , and  $\nwarrow$  ) in the Cursor Pad (Figure 1-1) move the cursor up one line, down one line, left one column, right one column, and to the first column of the first line of display memory (the "home" position) respectively. As cursor controls move the cursor out of the area displayed (as with line feed above) the screen display will move to keep the cursor visible. For example, assume that lines 21 through 44 of Display Memory currently appear on the screen, and the cursor is on line 21 (at the top of the screen). If a cursor up (  $\uparrow$  ) is typed, the cursor will move up one line in the same column to line 20 and the screen will shift up one line to show lines 20 through 43 of Display Memory.

## 1.6 Setting Up Communications

### 1.6.1 Communicating with the Host Computer

To set up communications between the Concept AVT and the user's host computer system, the terminal must be: (1) physically connected to the computer or communications equipment via the Line 1 connector, and (2) configured with a compatible communications protocol/interface for the user's computer. Follow the steps below to establish communications:

1. After turning the power OFF to the terminal, the communication Line 1 connector should be plugged into the user's communication equipment or computer. The connection should be secured by the screws on either side of the connector. The terminal can then be turned on again.
2. Make sure that the terminal is in "factory default" state. This can be done by typing the four character sequence

ESC [ 9 ~


3. Enter Setup Mode by striking the SETUP key. A complete explanation of Setup Mode is given in Section 1.3. First make sure that the terminal is in Remote Mode (field F on the User Status Line), and then select the values for each of the following four communication parameters required to establish compatibility with the host computer:
  - Duplex: HDX = half duplex; FDX = Full Duplex.
  - Parity: NO = None; EV = Even; OD = Odd; MK = Mark; SP = Space.
  - Baud Rate: 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, or 9600
  - Stop Bits: 1S = One stop bit; 2S = Two stop bits.

See Section 1.9 for more information on communications settings that can be changed in Setup Mode. If you are not sure of the appropriate selections, you should check with your computer system manager.

Hints: Generally HALF duplex: computers made by Amdahl, Burroughs, Honeywell, IBM, and Univac; also Sharp and STSC timesharing services).

Generally FULL duplex: computers made by Data General, DEC, Hewlett Packard, Prime, and Texas Instruments, as well as most microcomputer systems.

Stop bits are usually set to two for baud rates of 300 or below and otherwise are set to one.

4. Having selected the appropriate communication settings, exit from Setup Mode, and check to see whether the terminal is in fact communicating properly with the host computer. If there seems to be a problem, return to Setup Mode and make sure that each of the communication parameters listed above is set correctly. Make adjustments as necessary, and exit from Setup Mode. If the problem persists, check with your computer system manager to make sure that you have chosen the proper settings for the communication parameters. If you are still unable to establish communication with the host computer, contact HDS.
5. Once you have established communication with the host computer, you can store the correct values for all of the communication parameters in non-volatile memory (NVM) so that the terminal will be properly configured whenever it is turned on or reset. This can be done either by striking the SAVE key (marked  on top) while in Setup Mode, or by using the Save Configuration command (see Section 3.2).
6. APL users should change the terminal to APL mode. This can be done easily while the terminal is in Setup Mode. Users who normally work in APL will probably want to include APL among the parameters SAVED in NVM (see previous paragraph).
7. See Section 1.2 for a complete list of terminal characteristics that can be selected (and SAVED) in Setup Mode.
8. If you encounter problems and are uncertain about current terminal settings, simply start over at step 2 above.

#### 1.6.2 Connecting a Printer to the Concept AVT

The Concept AVT is compatible with most printers (hard copy devices) that use an RS232 type serial interface. However, there is a variety of ways in which RS232 serial printers may differ from one another, and the user must know the specific data communication requirements of the printer to be connected to the Concept AVT. It is best to have the user manual provided with the printer.

Connecting a printer to the Concept AVT involves a procedure similar to that described above: first the printer must be physically connected to the Line 2 ("Printer Port") connector on the back of the terminal; then appropriate values must be selected for Line 2 communication parameters to establish compatibility with the specific printer.

No standard cabling is supplied for this connection because the required cable length is unknown and printers generally come equipped with a cable. The user may, of course, purchase special cables from HDS. Cabling requirements are described in Appendix D of this manual.

Cabling on Line 2 (the printer port) is set up so that most printers can be used without modification. Technically, this means that the transmit and receive signals are found on the opposite connector pins from those on the Line 1 (and optional Line 3) connectors. Once the correctness of the cabling is verified, proceed as follows:

1. With the terminal turned off, attach the printer cable to the Line 2 connector on the back panel (Figure 1-1), securing with screws provided on the cable end connector.
2. Turn power on to both the terminal and the printer.
3. Strike the PRINT SCRN key located in the Cursor Pad (see Figure 1.1). If the bell does NOT ring, proceed to step 3. If the bell rings, then the printer is not properly connected to the terminal. The problem is most likely caused by one or both of the following:
  - a. Incorrect cabling: the functions of the various pins in the printer cable connector are different from the corresponding pins in the Line 2 connector (see Appendix D). To correct this condition, the printer cable must be rewired, or an appropriate adapter attached.
  - b. The printer does not supply a required control signal (Clear to Send) in accordance with the RS232 communication protocol. Consult printer documentation to determine whether the printer can be made to provide the required control signal. If not, this protocol requirement can be suppressed using the CTS/RTS Transmit Protocol command. Type the following seven key sequence:


CMD [ 0 ; 2 \* x

This protocol requirement can be turned back on by typing the seven key sequence:

CMD [ 1 ; 2 \* x

See Section 3.9 for additional information on commands related to data transmission. If the problem persists, consult the manufacturer of the printer or HDS.

4. Enter Setup Mode by striking the SETUP key. A complete explanation of Setup Mode is given in Section 1.2.
5. Strike the NEXT LINE key (marked  $\uparrow$  on top) until the User Status Line for Line 2 (i.e. the one with L2 at the extreme left) appears.

6. Select values for Baud Rate, Parity, and Stop Bits as specified by the printer manufacturer. These settings may be saved in non-volatile memory by striking the SAVE key (marked  on top).
7. Exit from Setup Mode by striking the SETUP key again.
8. Test the terminal/printer interface by striking the PRINT SCRN key. The contents of the screen should be printed. If the printer does not function properly, return to Setup Mode (step 3) and make sure that the communication parameters for Line 2 are set in accordance with the requirements of the printer.

### 1.7 VT100 Compatibility.

The Concept AVT is "software compatible" with the VT100 terminal manufactured by Digital Equipment Company (DEC) ('VT100' and 'DEC' are trademarks of the Digital Equipment Corporation). This means that software designed by DEC and other vendors for use with the VT100 can be used without modification with the Concept AVT. Users who wish to have the Concept AVT operate as if it were a VT100 should make sure that all options other than communication parameters are set to factory defaults. This can be done as follows:

1. Follow the instructions in Section 1.6 for setting up communications.
2. Save the communication parameters in Non-Volatile Memory before exiting from Setup Mode.
3. Reset the terminal by striking the RESET/BREAK key while holding down the SHIFT key.

When configured in this way, the concept AVT "looks like a VT100" to the host computer in the sense that it identifies itself as a VT100 and responds appropriately to commands sent by programs designed to interact with a VT100.

Note: The Concept AVT does not physically emulate the VT100, in the sense that it does not look to the user exactly like a VT100. For example, the Concept AVT does not imitate the "smooth scroll" of the VT100. Similarly, characters that would appear in double width on a VT100 appear normally on a Concept AVT.

## 1.8 The Keyboard

Section 1.5 above described in general the interaction of the keyboard operator and the display, mentioning only those keyboard commands, such as cursor controls, that help in understanding the operation. This section describes the keyboard in detail. Throughout this section Figure 1-1 can be used to locate keys described in the text.

The keyboard has been designed to provide a comfortable and efficient operator interface to the display terminal. The familiar typewriter layout, audible key click, and specially designed key plunge pressure response all help to increase typing efficiency. Other human factors related to the keyboard include matte finished keytops to reduce glare, a calculator layout numeric pad for fast entry of numeric data, and auto-repeat on all keys (i.e., keys repeat automatically at 15 characters per second until released when held down for more than the set delay period).

The keyboard is composed of four separate areas - the Main Pad, the Numeric Pad, the Cursor Pad, and the Programmable Function Keys (as indicated below, Numeric Pad keys are also programmable).

### 1.8.1 Main Pad

The Main Pad, as mentioned above, is similar to that of an IBM Selectric typewriter. In addition to the alphabetic, numeric, and special character keys, the pad contains 9 special action keys. Six of these - SHIFT, RETURN, LINE FEED, BACK SPACE, CAP LOCKS, and TAB - are analogous to special action keys on a typewriter. The other three - RUB OUT, ESC, and CTRL are unique to communication terminals. The function of each special action key is described below.

**RETURN** - Analogous to a typewriter, this key returns the cursor to the left margin of the screen. Unlike a typewriter, no automatic line feed occurs. The cursor remains on the current line.

**LINE FEED** - Moves the cursor down one line in the current column. A line feed with the cursor on the last line of display memory will cause all lines to scroll up one line, losing the first line of information and creating a new blank last line. If the cursor is displayed in the 24th line of the screen display area, as described above, a line feed will cause the screen display area to move down one line to keep the cursor visible, but no data are lost until a line feed is performed on the last line (96th or 56th depending on the number of columns displayed) of Display Memory.

**BACK SPACE** - Moves the cursor one column to the left. If Cursor Wraparound is on (see Section 1.9), and the cursor is in the first column, it will move to the last column of the previous line, (unless it is already on the top line).

**TAB** -

**Unshifted: Forward Tab.** Moves the cursor to the next tab stop column on the current line. If there is no next tab stop, it will move the cursor to the last column of the current line. Initially, tab stops are set at every 8 columns. Tab stops can be set and cleared with keys described in the Cursor Control Pad section below.

**Shifted: Backward Tab.** Moves the cursor to the previous tab stop. If there is no previous tab, it will move the cursor to the first column of the current line. Note: this key is functionally part of the Cursor Pad, in the sense that it can be set to either Normal or Application mode by the Cursor Pad Keys Settings command. Normal mode, described here, is the default (see Section 3.11).

**SHIFT** - Analogous to a typewriter. When this key is held down while striking another key, the shifted version of that key is generated.

**CAPS LOCK** - This is an alternate action key. When pressed it lights and sets CAPS LOCK on for the keyboard. When pressed again it turns off the light and resets the keyboard to normal operation. With CAPS LOCK on, the character keys A through Z generate the shifted (upper case) version whether or not the SHIFT key is depressed. No other keys are affected.

**CTRL** - The CTRL key, like the SHIFT key, is used in conjunction with another key to generate the control version of that key. The control version of a character is indicated in this manual by a ↑. For example, a control-M would be indicated by ↑M. Control codes are used to cause special action. For example, ↑M causes a return and is identical to hitting the return key, ↑J is a line feed, ↑H is a backspace, and ↑I is a tab. The specially marked keys (e.g. RETURN, LINE FEED) are provided for the convenience of the operator, and are functionally equivalent to striking the corresponding character while holding down the CTRL key. The user may verify this by trying the above control codes.

**RUBOUT** - Causes the RUBOUT control code (ASCII 127) to be generated. This character is not displayed and causes no action by the terminal. Many computer systems use this character as a fill character, transmitting it only to take up time to allow some terminal function to complete before sending more "real" data. Other computer systems use this character to indicate that the previous character typed is to be erased.

ESC - Causes the ESCAPE control code character (↑[, ASCII 027) to be generated. Some computer systems recognize this character as a signal to abort the current running program.

Note: This character is not displayed and causes no action by the terminal when entered from the keyboard. Use the CMD key (top row center in the Cursor Pad) to key in the ESC at the beginning of terminal commands.

### 1.8.2 Numeric Pad

The Numeric Pad is a calculator layout of the numbers '0' to '9', period ('.'), dash (or minus, '-'), ENTER, comma (','), TAB and four unassigned function keys (PF1 - PF4). Keys in the Numeric Pad are not functionally equivalent to similar keys in the Main Pad, in the sense that, for example, striking the 6 key in the Numeric Pad does not always have the same result as striking the 6 key in the Main Pad. This is because all keys in the Numeric Pad are programmable (see Section 3.12), and can be set to produce something other than what is indicated on the key tops. The SHIFT key has no effect on keys in the Numeric Pad.

### 1.8.3 Cursor Pad

The Cursor Pad includes cursor movement keys (also used in Setup Mode), as well as special keys that generate frequently used terminal commands. Note that cursor keys move the cursor through the display memory and may cause a shift of the screen display area to keep the cursor visible as described above in Keyboard Display Operation. However, these keys will not cause the permanent loss of data from display memory.

SETUP - Puts the terminal in Setup Mode if it is in normal operating mode, and puts it in normal operating mode if it is in Setup Mode.

CMD -

Unshifted: Enters the ESC at the beginning of terminal commands.


Shifted: Enters the sequence ESC [ used at the beginning of certain terminal commands (see Section 3.1).

RESET/BREAK -

Unshifted: Causes a "break" condition on the communication line. BREAK is recognized by some computer systems as a signal to stop output, and is typically used to interrupt listings or other output that has started.



Shifted: Resets the terminal to "power-up" defaults, which may be different from factory defaults (see Setup Mode). See Reset Terminal command in Section 3.2.

 Moves the cursor to home position (line 1, column 1). In Setup Mode, causes all current terminal settings (except programmable keys) to be saved in non-volatile memory (NVM).

PRINT -

Unshifted: Attaches the printer (or other device) connected to Line 2 (the Printer Port) so that it will receive all data going to the screen. If the printer (or other device) is unavailable, the bell will sound. (See Section 1.6.2 for details on connecting a printer to the Concept AVT.)

Shifted: Detaches the printer (or other device) connected to Line 2, so that it will no longer receive data going to the screen.

PRINT/SCRN -

Unshifted: Transmits the contents of the window (see Section 2.1), up to but not including the current cursor. If the device connected to Line 2 is unavailable, the bell rings.

Shifted: Transmits the entire contents of the window (see Section 2.1) to the printer (or other device) connected to Line 2. If the device connected to Line 2 is unavailable, the bell rings.

PAGE -

Unshifted: Moves the screen display area down 24 lines (1 page) with respect to display memory. For example, if lines 5 through 28 are being displayed, PAGE will cause lines 29 through 52 to be displayed. The cursor is also moved down 24 lines maintaining its relative position on the screen. If there are not 24 additional lines left to display when PAGE is struck, the last 24 lines of display memory will be displayed.

Shifted: Moves the screen display area up 24 lines (1 page) with respect to display memory. If lines 29 through 52 are being displayed, a shifted PAGE will cause lines 5 through 28 to be displayed. The cursor is also moved up 24 lines maintaining its same relative position on the screen. If there are not 24 lines previous to those displayed when PAGE is typed, the first 24 lines will be displayed.

↑ Moves the cursor up one line in the same column. If Cursor Wraparound is on (see Setup Mode), and the cursor is on the top line, it will move to the same column on the last line of the window (see Section 2.1). In Setup Mode, this key causes the next Status Line to be displayed.

#### SCROL -

Unshifted: Moves the screen display area down one line. For example, if lines 27 through 50 are being displayed, SCROL will cause lines 28 through 51 to be displayed. The cursor is also moved down one line, making it appear in the same position relative to the screen. See the before and after example in Figure 1-2. If the last line of display memory is already displayed on the bottom of the screen, SCROL will have no effect.

Shifted: Moves the screen display area up one line. For example if lines 27 through 50 are being displayed, shifted SCROL will cause lines 26 through 49 to be displayed. The cursor is also moved up one line, making it appear in the same position relative to the screen. See the before and after example in Figure 1-2. If the beginning of display memory is already displayed at the top of the screen, SCROL will have no effect.

← Moves the cursor one position to the left. If Cursor Wraparound is on (see Setup Mode), and the cursor is in the first column of a line, the cursor will move to the last column of the previous line; if it is in the first column of the first line, the cursor will move to the last column of the last line of the window (see Section 2.1). In Setup Mode, this key selects the next field to the left on the User Status Line.

↓ Moves the cursor down one line in the same column. If Cursor Wraparound is on (see Setup Mode), and the cursor is on the last line, it will move to the same column on the first line of the window (see Section 2.1). In Setup Mode, this key selects the next value for the field currently selected.

→ Moves the cursor one position to the right. If Cursor Wraparound is on (see Setup Mode), and the cursor is in the last column of a line, the cursor will move to the first column of the next line; if it is in the last column of the last line, the cursor will move to the first column of the first line of the window (see Section 2.1). In Setup Mode, this key selects the next field to the right on the User Status Line.

#### 1.8.4 Programmable Function Keys

The Programmable Function Keys consist of both the Default Execute and the Default Transmit Keys. These keys, in addition to performing the functions described below, can be programmed to contain a user specified sequence of characters and/or terminal commands. This means that a single key can be used to transmit frequently used character sequences. This capability is described in detail in Section 3.12.

##### 1.8.4.1 Default Execute Keys

The Default Execute group is composed of five keys that provide the capability to edit locally on the screen. (See Section 3.6 for a detailed explanation of terminal editing functions). These keys may also be used by special applications and may or may not be generally applicable for interactive use. The term "Default Execute" indicates that in their default ("power-up") state, these keys perform the terminal function indicated. As programmable keys, they can also be programmed to execute or transmit any sequence the user chooses. The labels listed below appear on the key fronts.

INSRT (F1 Shifted or Unshifted) - Switches the keyboard between Insert and Replacement (normal) mode, changing the indicator light accordingly. In Insert mode, text from the cursor position to the first clear position is shifted to the right (wrapping around lines if necessary) one location before new characters typed are inserted. This allows new text to be inserted within the existing text on the screen rather than replacing it.

DEL CHAR (F2 Shifted or Unshifted) - Deletes the character at the current cursor position by left shifting all text to the right of the cursor up to the first clear position or the end of the line, and clearing the last position shifted.

INS L (F3 Unshifted) - Inserts a blank line at the current cursor position by moving the text on the current and subsequent lines down one line (the last line of text is lost), and clearing the line on which the cursor is positioned.

DEL L (F3 Shifted) - Deletes the line on which the cursor is positioned by moving the text on all subsequent lines up one line and creating a blank last line.

ERAS (F4) -

Unshifted: Clears the text from the current cursor position to the end of the line.

Shifted: Clears the text from the current cursor position to the end of the window (see Section 2.1).

SEND (F5) - Transmits contents of Display Memory to the keyboard communication line (normally Line 1) attached to the host computer. See the Transmit command in Section 3 for a detailed discussion of data transmission.

#### 1.8.4.2 Default Transmit Keys

Keys in the Default Transmit group do not perform any terminal function in their "power-up" state. Instead, each one transmits a unique character sequence to the keyboard communication line (normally Line 1, attached to the host computer). See the Programmable Keys table in Section 3.6 for a listing of the Default Transmit sequences. Specific applications may either use these default sequences as commands, or program these keys to transmit other sequences, and/or execute terminal commands.

### 1.9 Terminal Characteristics that can be Changed in Setup Mode

Settings that can be changed in Setup Mode include most of those on the User Status Lines for the keyboard and Lines 1 - 3, as well as Tab stops.

#### 1.9.1 User Status Lines

Terminal characteristics that can be changed in Setup Mode are listed below in the order in which they appear on the User Status Line. The letter in parentheses following the name of the field corresponds to the reference letter printed above the User Status Line in Figure 1-1.

Line (A) Identifies the communication line (device) to which the status line applies:

KB = Keyboard communication line (line 1 by default).  
L1 = Line 1; L2 = Line 2 (Printer Port); L3 = Line 3.

Baud Rate (B) Determines the rate at which data is transmitted between the terminal and the host computer (or other device). 15 baud rates ranging from 50 to 9600 are available. See Section 1.6 on setting up communications. See also Section 3.9.

Duplex (C) Determines whether keyboard input is sent directly to the screen:

HDX = half duplex: text entered from the keyboard is displayed on the screen as it is typed in. This mode is normally chosen when the host computer does not "echo" characters back to the terminal.

FDX = full duplex: text entered from the keyboard is not displayed on the screen as it is typed in. This mode is nor-

mally chosen when the host computer does "echo" characters back to the terminal.

**Stop Bits (D)** Determines the number of Stop Bits the terminal uses in transmitting data: 1S = one stop bit; 2S = two stop bits. See Section 1.6 and Appendix D on communications.

**Parity (E)** Determines how (if at all) the terminal generates parity bits in transmitting data: NO = no parity bit; EV = even; OD = odd; MK = mark(1); SP = space(0). See Section 1.6 and Appendix D on communications.

**Local/Remote (F)** Determines whether data entered from the keyboard is transmitted to the host computer.

LOC = Local Mode: data is displayed on the screen, but never transmitted.

REM = Remote Mode: data is transmitted but not displayed unless the terminal is in Half Duplex.

**Character/Block(G)** Determines when data entered from the keyboard is transmitted to the host computer, assuming that the terminal is in Remote Mode.

CHR = Each character is transmitted as it is entered, but displayed only when in Half Duplex.

BLK = Each character is displayed on the screen as it is entered, but not transmitted until a Transmit command is entered either from the keyboard (SEND key) or from a program running on the host computer (see Section 3.9).

**Caps Lock (H)** Determines how alphabetic characters entered from the keyboard are stored in Display Memory.

U/L = Upper or lower case (as entered).

CAP = Lower case converted to upper case.

**ASCII/APL (J)** Determines the character set to use. \*

ASC = ASCII characters without character overstrike

APL = APL characters with character overstrike.\*

**Window (K)** Coordinates of the display window (top;bot;lft;rgt). In Setup Mode, bot may be either 24 or the maximum of display memory; rgt may be either 80 or 132; top and lft may be set to 1. See Section 2.1 for a discussion of windows. \*

ANSI/VT52 (L) Determines general terminal function: ANS = ANSI mode (VT100 compatible); V52 = VT52 mode.

Screen Width(M) Determines whether display width is 80 or 132 columns. \*\*

Cursor (N) Determines whether cursor is represented as an underline or as a block. \*\*

Screen Video(O) Determines whether overall screen video is light characters on a dark background or dark characters on a light background. \*\*

Wraparound (P) Determines whether characters and cursor wraparound window boundaries (such as the end of a line).\*

a = Neither characters nor cursor wrap around.

A = Both cursor and characters wrap around.

Cursor Pad Operation (Q) Determines mode of operation of cursor pad keys. \*

CE = All keys in execute mode (i.e. pressing a cursor Pad key results in the terminal function associated with the key).

CT = The 4 cursor movement keys ( ↑ , ↓ , ← , and → ) transmit special sequences to the host computer instead of moving the cursor. The remaining Cursor Pad keys are in execute mode.

#### Notes on User Status Line Fields:

\* These fields may have values other than those which are selectable through Setup Mode. The only way to be certain that a field is in fact set to the value shown on the Setup Status Line is to use the NEXT VAL key (marked ↓ on top) to select the desired value. If the desired value appears upon entering Setup Mode, use the NEXT VAL key to change it, and then change it back again.

\*\* The character in parentheses appears on the status line as a place holder only in order to show which field is being modified; it does not change to reflect changes in the value selected.

### 1.9.2 Tab Status Line

Tab stops are indicated by the occurrence of a period (.) in place of the one-digit column number indicator. To clear an existing tab stop, or set a new tab stop, strike one of the FIELD keys until the desired column location appears in inverse video, then strike the NEXT VAL key until the tab stop is cleared or set (as desired).

Tab stops are saved in NVM along with other terminal settings when the SAVE key is pressed (while in Setup Mode).





## **Programmers Guide**

### **2.1 Some Key Concepts**

This chapter describes the capabilities and operation of the Concept AVT line of video display terminals. The information is organized by terminal function. Each section describes a particular feature or capability, including examples illustrating typical uses.

To provide maximum flexibility to both the programmer and the user, Concept terminal functions are implemented in such a way as to be independent of one another. Unless otherwise indicated, the features described below can be used in any combination. For example, editing functions can be used in either Block or Character Mode. Similarly, the explanation of each feature is independent of the explanations of other features, though all of the following sections in this chapter assume that the reader has read this introductory section. Application developers are encouraged to read through this entire chapter for ideas on how Concept terminal capabilities might be used to enhance the effectiveness of an application's user interface.

#### **2.1.1 ASCII/APL Chart Location**

To avoid ambiguity, references to individual characters are frequently supplemented by specification of their "ASCII chart location". Readers not already familiar with the ASCII character set might find it useful to refer to Appendix A in connection with the discussion of terminal commands. Any character recognized by the terminal as either data or terminal command can be uniquely specified by its location in the ASCII chart. In Section A.1, the chart location is given as a three digit number at the top of each cell. For example, the lower case letter 'a' has the ASCII chart location 097 (ASCII 097), while the upper case letter 'A' has the ASCII chart location 065 (ASCII 065).

In addition to providing an unambiguous way of specifying characters, the ASCII chart provides a simple means of translating between the ASCII and APL character sets. The APL user simply substitutes the APL character whose location in the APL chart (Appendix A.3) is the same as the location in the ASCII chart of the character to be translated. For example, the ASCII character 'm' (ASCII 109) becomes the APL character 'M' (APL 109), and the ASCII character 'M' (ASCII 077) becomes the APL character '|' (APL 077).

#### **2.1.2 Terminal Commands**

The Concept AVT is an ANSI-compatible "smart" video display terminal. All of the terminal functions and capabilities described below can be invoked by sending sequences of characters either from the keyboard or from the host computer. Terminal commands can be sent from the host computer by any process used to send ordinary data to the terminal, provided that it is capable of transmitting the entire range of 128 ASCII characters. In particular, the process used to send commands to the terminal must be able to send control codes (see below).

For example, programmable key definitions might be stored in a text file on the host system, so that the keys could be programmed by simply "listing" the file to the terminal. The file could be built using any text editor, as long as the text editor had some way of entering control codes into the text. Most text editors have some facility or convention for entering control codes into text, but problems can arise. For example, it might be difficult to construct a terminal command that includes a carriage return (ASCII 013) using a line-oriented text editor.

Similarly, for an application program to send terminal commands, the programming language in which it is written must have some way of sending characters to the terminal that can handle control codes. In APL, for example, the `▢ ARBOUT` function sends characters to the terminal based on their chart locations. The ways in which input/output operations are handled by some programming languages can create problems in this respect. In some FORTRAN implementations, for instance, a carriage return is automatically sent following every sequence of characters to mark the end of a record. This could create problems both in trying to include a carriage return as part of a command (as in programming a function key), and in avoiding the effects of a carriage return following the successful execution of the terminal command. Another kind of problem is created by programming languages (like COBOL in some installations) that allow transmission of data to the terminal only in fixed-length records. This restriction forces the program to "pad" each terminal command sequence with "fill characters" (NUL, ASCII 000, or RUB, ASCII 127).

Problems resulting from the way in which the transmission of data to a terminal is handled by a particular programming language can sometimes be avoided by using assembler language subroutines to send commands to the terminal. Many operating systems have general purpose subroutines that perform simple functions like sending a single character to the terminal, and that can be called from programs in higher level languages (such as FORTRAN and COBOL). Consult programming language reference manuals for details on the operation of input/output functions. Contact the system programmer or administrator of your computer installation for information on public subroutine libraries.

There are two kinds of terminal commands: control codes and escape sequences.

### Control Codes

A control code is a single character whose chart location is in the range 000 - 031 or 127. Control codes are standardly represented by an up-arrow (`↑`) followed by the character located 4 rows below the control code in the chart. Thus, ASCII 001 is represented as `'↑A'` ('control-A'). Some control codes also have functional names, such as 'line feed' (ASCII 010), which could also be referred to as `↑J`. ASCII 127 is referred to as either RUB or DEL, and is not associated with any alphabetic character.

To enter a control code from the keyboard, strike the associated alphabetic key while holding down the CTRL key located in the lower left corner of the main pad (see Figure 1-1). For example, to enter ↑C, strike the C key while holding down the CTRL key. Some control codes can also be entered using specialized keys, such as RETURN (↑M), LINE FEED (↑J) and, BACK SPACE (↑H).

Control codes are "unprintable" --they are not shown on the screen, and they are generally not printed by hard copy devices. The two-character symbols that appear in the first two rows of the ASCII chart are displayed on the screen only when the terminal is in Transparent Mode (Section 2.3.2). (In APL these characters are subscripts and superscripts.)

Although control codes are not printable, some of them have visible effects on the screen display. For example, the character in chart location 013 is a carriage return, a terminal command that moves the cursor to the beginning of the current line.

### Escape Sequences

An escape sequence is a series of characters beginning with the escape message character, indicated by the standard abbreviation 'ESC' throughout this manual. ESC functions as a command introducer, signalling the terminal that the following characters are to be interpreted as part of a terminal command, rather than as data. The escape message character is programmable (Section 2.3.3), but defaults to ASCII 027, known as the escape control code.

To enter the ESC at the beginning of a terminal command from the keyboard, use the red CMD key (unshifted). This key is located in the middle of the top row of the cursor pad (Figure 1-1). Do NOT use the ESC key (in the upper left corner of the main pad) to introduce commands from the keyboard. (The ESC key is sometimes used in defining programmable keys and stored data fields (see Section 2.4) but never as a command introducer.) Since many terminal commands begin with ESC [, this two character sequence can be entered from the keyboard by striking the CMD while holding down the SHIFT key (SHIFT-CMD).

Escape sequences beginning with ESC [ (SHIFT-CMD from the keyboard) may contain one or more parameters, each of which contains one or more numbers separated by semicolons (ASCII 059) or colons (ASCII 058). See Section 3.1 for a detailed explanation of parameters and escape sequences. Each parameter contributes to determining exactly what the command does. For example, the Position Cursor command is given as

ESC [ lin ; col H

where lin and col are parameters that specify the line number and the column number respectively to which the cursor is to be positioned. 'H' is the command identifier. To use this command, replace lin and col with the appropriate numbers. Thus, to position the cursor at line 7 column 20, send the command

```
ESC [ 7 ; 20 H
```

from the computer, or enter

```
SHIFT-CMD 7 ; 20 H
```

from the keyboard. In subsequent examples, the keyboard version of the command will be omitted, on the assumption that the reader can easily infer it from the sequence given.

Note: Terminal commands may contain no embedded spaces. Blank space in the representation of terminal commands is for purposes of clarity only. Where an explicit space (ASCII 032) is required, it is indicated by 'Sp'.

Almost all parameters have default values; that is, values that are assumed by the terminal if the parameter is omitted. Default values have been chosen in the interests of minimizing the length of commands. For example, the default value for both lin and col is 1. This means that either parameter (or both) can be omitted if the desired value is 1. Thus, to position the cursor at the beginning (column 1) of line 7, the command would be `ESC [ 7 H`. Since the second parameter is omitted, there is no need for the semicolon to separate the parameters. However, if the first parameter were omitted, the semicolon would be required to show that it had been omitted. Therefore, to position the cursor at the top (line 1) of column 20, the command would be `ESC [ ; 20 H`. Finally, both parameters can be omitted when using this command to "home" the cursor; that is, position it at line 1 column 1: `ESC [ H`.

### 2.1.3 Alternate Processing Modes

Many aspects of the way in which data is processed by the terminal are determined by the various Alternate Processing Modes. Each mode is either in its Set or its Reset State. The settings of these modes also affect the operation of many of the terminal commands in ways that are explained in the individual command entries in Chapter 3. A complete listing of Alternate Processing Modes is provided in Chapter 4.

Note: "Reset" should not be confused with "default". "Set" and "Reset" are simply names for the two options available for each of the operating characteristics. The listing in Section 4.5 indicates the default state for each of the Alternate Processing Modes.

As explained in Chapter 4, there are three groups of Alternate Processing Modes: ANSI, DEC, and HDS. Associated with each group there is a single Set Mode command that can be used to select the Set state of as many as fifteen members of the group, and a single Reset Mode command that can be used to select the Reset state for as many as fifteen members of that group. Since the HDS group includes equivalents for all of the ANSI and DEC numbers, only the Set and Reset commands for HDS modes are shown here. See Section 4.2 for details on ANSI and DEC modes. To select the Set state for up to 15 HDS Modes, the command would be:

ESC [ = hds h

where hds is a list of up to 15 HDS Mode numbers. Thus to set HDS Modes 16, 104, 105, 106, 110, and 118, the command would be:

ESC [ = 16 ; 104 ; 105 ; 106 ; 110 ; 118 h

This command might be used to initialize the terminal for a typical Block Mode data entry application (Sections 2.9 and 2.10). This single Set Mode command: sets transmit termination to be the current cursor position (16); prevents scrolling in response to line feeds (104); causes tabs to be processed so that they move the cursor to the next unprotected area (105); enables autotabs (106); puts the terminal in Block Mode (110); and causes protected fields to be displayed in bold (118).

To select the Reset state for up to 15 HDS Modes, the command would be:

ESC [ = hds l

where the final character is the lower-case letter 'l' (ASCII 108). For example, upon exiting from the data entry application imagined in the previous example, the following command could be used to Reset the same modes that had been Set during initialization:

ESC [ = 16 ; 104 ; 105 ; 106 ; 110 ; 118 l

The above example also serves to illustrate the interaction between Alternate Processing Modes and terminal commands. HDS Modes 16 and 110 affect the operation of the Transmit and Print commands; HDS Mode 104 affects the operation of Line Feed; HDS Mode 105 affects the operation of the Tab, Forward Tab, and Backward Tab commands; and HDS Modes 106 and 118 affect the processing of data. See Chapter 3 for details.

#### 2.1.4 Character Processing

The following outline of the way the terminal processes each character it receives should aid in understanding the operation of terminal commands. When a character is received by the terminal from a communication line, it is first checked to determine whether it is one of the following:

DLY	Programmable Delay Character (default: not used)	•
↑@	Fill Character (not programmable)	
RUB	Fill Character (not programmable)	
↑S	Stop Transmission (XOFF)	
↑Q	Resume Transmission (XON)	

The processing for these communication control codes is described below.

## Ordinary Characters

All other characters (including other control codes) are "networked" according to the current "output network" for the device (keyboard or communication line ) from which it was received. As explained in Section 2.8.2, the output network is a list of devices to which data is to be sent. Normally, characters received from all of the communication lines are sent to the display screen, and characters received from the keyboard are sent to Line 1 (the Keyboard Communication Line). Characters received from the keyboard are also sent to the display screen if the terminal is set to Half Duplex. All characters not on the above list are networked in the same way, even if they are control codes.

Control codes are only processed differently from other characters when they are sent to the display screen. When an ordinary character is sent to the screen, it produces a graphic representation according to the character set (Section 2.6.1) and the Attribute List currently in use by the device (communication line or keyboard) from which the character was received (Section 2.6.2). The character is placed in display memory at the current cursor position in the Window currently in use by the device from which the character was received (Section 2.7.4). By default, all devices use the same Attribute List (1), which is defined so that all characters appear normally on the screen; and all devices use the same Window (1), which is defined as a single page of display memory.

## Control Codes

When a control code is sent to the screen, it is "executed" rather than being displayed, unless the device is in Transparent Mode (Section 2.3.2). In Transparent Mode, control codes are displayed using the same process as was described above for ordinary characters. All control codes including ↑@, RUB, ↑S, and ↑Q, are represented, but not executed. If the device is not in Transparent Mode, the control code is executed, but not stored in display memory. Many control codes (such as ASCII 025) are not terminal commands, so the terminal executes them by "doing nothing".

For example, when a carriage return is sent to the screen, the terminal executes it by moving the cursor to the beginning of the current line. (Carriage returns and other appropriate control codes are reinserted by commands such as Transmit and Print/Line 3 Control that are used to transmit the contents of display memory. See Sections 2.9 and 2.8.1).

## Escape Sequences

Unless the terminal is in Transparent Mode, receipt of an ESC (CMD from the keyboard) signals the beginning of an escape sequence. Subsequent characters received from the same device (with the exception of communication control codes) are assumed to be part of the command sequence until either the sequence is determined to be complete and valid (in which case it is executed), or the sequence is determined to be invalid (in which case it is aborted). None of the characters in a valid escape sequence (including the escape message character) is sent either to the screen or to

another communication line. If an escape sequence is determined to be invalid, no action is taken, and characters received after the error was detected are treated as ordinary characters (that is, networked normally). Invalid escape sequences beginning with ESC [ cause the bell to ring when entered from the keyboard.

#### Communication Control Codes

Certain characters, used to control data flow between the terminal and the host computer, are processed somewhat differently from other characters. If the device from which they are received is in Transparent Mode, then they are processed the same as other control codes (see above). If, however, the device is not in Transparent Mode, these characters are used by the terminal as soon as they are received, and are not sent to any other line or device. The programmable character DLY (if defined), and the two fill characters, |@ (ASCII 000) and RUB (ASCII 127), are always ignored by the terminal. This means that they may be interspersed anywhere, even in the middle of an escape sequence, for timing purposes.

↑S causes the terminal to stop sending data to the line from which it was received, and ↑Q causes the terminal to resume sending data to the line from which it was received. See Section 2.3.4 for a way to prevent the terminal from responding to ↑S.

#### 2.1.5 Lines and Devices

Throughout the Users Manual, the terms "line" and "device" are generally used interchangeably in discussions of data communication. The reason for this is that in some cases it is useful to think of the keyboard as sending data to the terminal, and the screen as receiving data from the terminal, even though they function differently in many respects from external devices (such as host computers and printers) connected to the terminal via communication lines.

To support the extensive communication capabilities described in Section 2.8, many terminal commands that control the transmission of data between terminal and host computer (or other device, such as a printer), involve specification of the communication line to which the command applies. However, since most applications involve sending and receiving data over the same communication line (typically Line 1) attached to a single host computer, commands and defaults have been designed so that users working in this configuration need not worry about identifying the communication line.

### Keyboard Communication Line

Whenever a Transmit command is executed, the data is sent to the line currently defined as the Keyboard Communication Line. Although the Keyboard Communication Line can be defined to be Line 1, Line 2, or Line 3 (if available), it is defined as Line 1 by default, since most users connect the host computer to Line 1. A number of other commands related to data transmission also apply to the Keyboard Communication Line. Since changing the Keyboard Communication Line also changes the keyboard's "output network", it also changes the line to which normal keyboard input is sent. See Section 2.8.3 on Multiple Computer applications, and Section 2.9 on Block Mode for more information on Keyboard Communication Line.

### Requesting Device

The operation of many terminal commands, particularly those controlling data transfer, depends upon the device or line from which the command was received, referred to as the "requesting device". Typically, terminal commands come either from the keyboard, or from the host computer connected to Line 1. For example, the Transmit command causes the contents of display memory to be transmitted to the requesting device; that is, to the device (line) from which the Transmit command was received. If the command was issued by a program running on the host computer, the contents of display memory would be transmitted to the communication line (normally Line 1) to which the host computer was connected. If the Transmit command was entered from the keyboard (using the SEND key), the data would be sent to the Keyboard Communication Line (also normally Line 1).



## 2.2 Status and Non-Volatile Memory

All aspects of the terminal's configuration, except the contents of programmable keys, are recorded on a series of Status Lines, and can be saved in Non-Volatile Memory (NVM) so that they will be in effect whenever the terminal is powered up or reset. The contents of NVM can easily be changed or reset to factory default condition by commands described below. See Appendix G for a complete listing of factory default settings. All status line information can be either displayed on the screen or transmitted to the host computer, where it can be used by an application program to determine (for example) whether a terminal command has been executed successfully.

In addition to the information available from Status Lines, several predefined sequences indicating type of terminal and basic hardware configuration, as well as error conditions, are transmitted to the host on request.

### 2.2.1 Status Lines

When status information is displayed on the screen, it appears as a non-scrolling 25th line that is independent of the ordinary data being displayed at the same time. There is a total of 15 different status lines, of six types: User, Programmer, Modes, Programmable Message Characters, Tabs, and Alert Message. There are separate User, Programmer, and Modes Status Lines for the keyboard, Line 1, Line 2, and Line 3. Thus there are four separate User Status Lines, though the Keyboard User Status Line would show much of the same information as the User Status Line for the line currently defined as the Keyboard Communication Line. See Section 3.4 and Appendix B for detailed descriptions of all status lines.

The User and Tabs Status Lines contain information that would be of use to the terminal operator, and are accessible through Setup Mode. User Status Lines include communications parameters, character set, cursor position, window definition, and terminal firmware version. The Tabs status line shows the current tab stops.

Programmer, Modes, and Programmable Message Character Status Lines contain information that would be of use to programmers, particularly in testing and debugging programs with sophisticated terminal interfaces. The Alert Message Status Line is used to call the terminal operator's attention to special conditions. It can be programmed to display any message upon request (Section 2.4.5), but is also used to indicate when the keyboard is locked (Section 3.11), as well as the amount of memory available for programming function keys (Section 2.4.2). These status lines cannot be accessed through Setup Mode.

## Displaying Status Information

Striking the STATUS key (located in the upper right corner of the main pad) causes the keyboard's User Status Line to appear at the bottom of the screen. Striking this key again causes the status line to disappear. Striking the STATUS key while pressing the SHIFT key causes the next status line to be displayed. See Section 3.4 for the order in which the status lines are displayed. These status lines can also be displayed upon command from the host computer: ESC [ \* v has the same effect as STATUS; ESC [ \* u has the same effect as SHIFT-STATUS. See Section 3.4 for a command that causes a specific status line to be displayed.

## Transmitting Status Information

Programs running on the host computer can get current terminal status information by using the Transmit Status Line command to specify the precise piece of information needed. For example, the command

```
ESC [ 1 ; 2 ; 25 ; 42 * t
```

causes the terminal to transmit characters 25 through 42 of the Programmer Status Line (2) for Line 1 (1) to the host computer (preceded by the SOM and followed by the EOM message characters, if defined). This would enable the host to determine the number and definition of the window currently in use by Line 1. See Section 3.4 for further details.

## Background Status Line

The 25th line on the screen can be used to display one line of a specified window whenever it is not being used to display status information. What appears at the bottom of the screen is the current contents of the line on which the cursor is now located in the window specified by the Set Background Status Line command (Section 3.4). Since the Background Status Line is updated continually, it provides an easy way to warn the terminal operator of error or other special conditions. See Section 2.7.4 on windows.

### 2.2.2 Terminal Identifier

Two commands are available to enable the host to determine the type and availability of the terminal. The Transmit Terminal Identifier command, ESC [ 0 c , causes the terminal to transmit to the requesting device one of three fixed sequences identifying the terminal either as a VT100 or as a Concept AVT with four or eight pages of memory. The Transmit Terminal Status command, ESC [ 5 n , causes the terminal to transmit fixed sequences to the requesting device indicating whether any malfunctions have been detected in the terminal. See Section 3.4 for details on the format of terminal responses.

### 2.2.3 Programmable Answerback Message

A message of up to 20 characters can be defined to be transmitted by the terminal in response to ↑E, the Transmit Answerback Message command. The Answerback Message can be programmed either from the keyboard or by a program running on the host in the same way as a programmable key (see Section 2.4.5). In fact, since it is possible to invoke any programmable key from the host, any programmable key could be made to serve the same purpose as the Answerback Message in a given application (though the programmable key would not be invoked by the Transmit Answerback Message command). See Section 3.12.

### 2.2.4 Non-Volatile Memory (NVM)

NVM makes it possible for the user to determine what terminal operating characteristics will be in effect when the terminal is powered up (or reset, see below). Non-Volatile Memory stores all aspects of the terminal's configuration, including communications parameters, choice of character sets, Alternate Processing Modes, Programmable Message Characters, and the Latent Expression, regardless of whether the terminal is turned on (or even connected to a power source). The contents of programmable keys is not stored in NVM. Whenever the terminal is powered up or reset, NVM is referenced to determine the settings for all terminal operating characteristics.

The current terminal configuration can be saved in NVM either by entering the command ESC [ ~ or by entering Setup Mode, and striking the SAVE key. Since the entire terminal configuration is saved at once, the configuration previously saved is completely replaced. The setting for a particular terminal feature or option currently stored in NVM can be determined by resetting the terminal to power-up state and checking the appropriate status line fields. See Appendix B for detailed information on status lines.

#### Reset to Power-Up State (NVM)

Because so many of the terminal's operating characteristics can be modified both from the keyboard and from the host computer, it is often convenient to reset the terminal to some known configuration prior to beginning a particular application. Resetting the terminal is also a common way of recovering from errors resulting from mistyping terminal commands or from data communication problems. Striking the red RESET/BREAK key in the upper right corner of the cursor pad while holding down the SHIFT key causes all terminal operating characteristics to be set to the values currently stored in Non-Volatile Memory. (This has exactly the same effect upon terminal operating characteristics as turning the terminal off and then on, and is easier on terminal hardware.)

Resetting the terminal to power-up state clears all of display memory, clears all input buffers, reprograms all programmable keys to their power-up default state, and invokes the Latent Expression (a user-defined field, see Section 2.4.5).

### Reset to Factory Default State

The terminal can be returned to the configuration it had upon leaving the factory by entering the command `ESC [ 9 ~ .` This command returns NVM to Factory Default settings, and resets the terminal to power-up state. See Appendix G for a summary of the factory default terminal configuration.

## 2.3 Trouble Shooting and Debugging

Proper functioning of any interactive application depends upon an often complex interaction among many components: application program, operating system, data communications processor, transmission lines, modems, and, of course, the terminal. When something goes wrong, it can be difficult to pinpoint the source of the problem, especially in applications utilizing sophisticated terminal capabilities. This section describes a number of terminal features that can be helpful in tracking down and solving programming problems.

### 2.3.1 Checking Out the Terminal

If nothing appears on the screen at all, begin the trouble shooting process by making sure that the terminal is plugged in and turned on. Check the adjustment of the brightness control located under the left side of the display near the front (Figure 1-1). Make sure also that the keyboard cable is connected properly to the terminal. If it is, pressing the CAPS LOCK key should make the keytop light go on. If the light does not go on and the keyboard cable is connected, contact your Concept terminal distributor or the nearest HDS sales office.

Reset the terminal to its factory default state by entering ESC [ 9 ~ (see Section 2.2.4). This should cause the cursor to disappear momentarily. If the cursor does not reappear, turn the terminal off, wait a minute, and turn it on again. If the cursor does not reappear, contact HDS.

If the terminal resets properly, put it in Local Mode to determine whether data can be sent to the screen from the keyboard. This can be done through Setup Mode (see section 1.3). While in Local Mode, characters entered on the keyboard should be displayed on the screen, as described in Section 1.4. If keyboard input does not appear on the screen in Local Mode, contact HDS.

Several internal test routines are provided to help in identifying problems in the terminal hardware and firmware. The discussion below assumes that the terminal is in Factory Default state. To ensure that it is, enter ESC [ 9 ~ . See Section 3.2 for further information on Self Tests.

#### Firmware and Display Memory

The command ESC [ y causes a series of internal tests to be performed on the terminal firmware (ROM) and memory (RAM and NVM). The results are displayed on a status line at the bottom of the terminal screen in the form of either an appropriate error message, or the message 'NO ERROR'. See Section 3.2 for an explanation of Self Test error codes.

## Communications

A communications self-test can be performed by attaching a loop-back connector to Line 1 and issuing the command ESC [ 1 y from the keyboard. A loop-back connector is anything that connects pin two to pin three.

Note: Use the ESC key (in the upper left corner of the main pad), not the CMD key to enter this command from the keyboard.

In performing this test, the terminal sends a series of characters out Line 1 and examines them as they come back in Line 1. If every character is received as it is sent, the test is successful, and 'NO ERROR' is displayed on the status line. Otherwise the error code 8 is displayed.

If nothing is available to serve as a loop-back connector, terminal communication functions can be tested manually by connecting Line 1 to Line 2. In this configuration, data entered on the keyboard should be sent out Line 1, received in Line 2, and displayed on the screen. If the terminal is in Full Duplex, each letter typed on the keyboard should appear once on the screen (as it comes in Line 2). If the terminal is in Half Duplex, each character should appear twice: once as it is sent directly from the keyboard to the screen, and once as it comes in Line 2.

If the procedure just described was successful, it showed that data is transmitted properly from Line 1. To test whether data is received by Line 1, change the Keyboard Communication Line to Line 2 by entering the command ESC [ 2 z . Now, characters typed on the keyboard are sent out Line 2 and received in Line 1.

## Character Sets

There is also a Character Set self test that displays all of the available character sets with a combination of different attributes. This test can be invoked by the command ESC [ 2 y . 'NO ERROR' is displayed on the status line.

The self tests described above can be performed in any combination, simply by listing the parameters separated by semicolons. For example, the command ESC [ 0 ; 2 y invokes both the ROM/RAM/NVM and the character set tests.

### 2.3.2 Transparent Mode

If the terminal functions normally in the testing procedures described above, but does not seem to respond correctly to commands sent from the host computer, it is often useful to enter Transparent Mode to see exactly what data and command sequences are being received by the terminal. This can help both in debugging application programs and in detecting unexpected output from operating system software and data communications equipment.

Enter ESC Q to put the terminal in Transparent Mode. In this mode, all control codes and escape sequences are displayed on the screen exactly as they are received, rather than being acted upon (executed) as terminal commands. Character Wraparound (HDS Mode 207) should be Set when the terminal is in Transparent Mode, since otherwise only one line of characters would be visible. Character Wraparound can be selected in Setup Mode.

The only sequence recognized as a terminal command while in Transparent Mode is ESC R, which takes the terminal out of Transparent Mode. When the escape message character is received, it is not displayed immediately, but buffered until the next character is received. If the following character is 'R' (ASCII 082), the terminal exits from Transparent Mode without displaying either the escape message character or the 'R'. If the character following the escape message character is not 'R', both the escape message character and the following character are displayed.

For example, when a line feed (ASCII 010) is received in Transparent Mode, the special symbol '␣' is displayed on the screen, but no line feed is executed, so the character received after the line feed appears on the same line as the line feed. Similarly for escape sequences: the Transmit Status Line command used as an example in Section 2.2.1 would appear as follows if received from the host while the terminal was in Transparent Mode:

```
␣ [1;2;25;42*t
```

Since commands are not executed in Transparent Mode, no status line would be transmitted (and the program issuing the command would be left waiting for the terminal's response).

Because everything, including communication control and timing characters (XOFF, XON, NULL, RUB), is displayed, Transparent Mode can help the programmer to solve problems caused by the way in which a particular operating system or data communications processor sends data to terminals. Of course it can also help the programmer to find bugs in the application program routines that generate terminal commands.

### 2.3.3 Programmable Message Characters

Application development problems sometimes arise from differences between systems in the ASCII characters assigned to special data communication functions. In order to provide for maximum flexibility and compatibility with the widest range of hardware and software, Concept AVT terminals allow the user or programmer to specify which ASCII characters are to have each of these special functions. In some cases a pair of ASCII characters can be associated with a single function.

Programmable message characters include Escape (command introducer, ESC), Start of Message (SOM), End of Message (EOM), End of Line (EOL), End of Field (EOF), Delay Character (DLY), Function Key Identifier (FKD), and Cursor Pad End of Message (CPM). See the Change Message Character command in Section 3.2 for default values. The current values for all of the message characters are shown on the Programmable Message Character status line (Appendix B).

For example, it might be convenient in a COBOL block mode application to transmit a number of fields together as a single "record" with fixed-length fields. To do this it would be necessary to prevent the terminal from sending an End of Field character (EOF) at the end of each field. This would be done by using the Change Message Character command

```
ESC [ 5 ! t
```

to define EOF as no character at all (not used). If fields might contain trailing spaces, it would also be necessary to Reset HDS Mode 114 so that trailing spaces would be transmitted. Both commands might be included in the terminal initialization procedure at the beginning of the block mode application. The 5 specifies that it is the EOF character that is to be changed, and the absence of a parameter specifying the character to use means that no character is to be used. EOF could be changed back to its default value upon exiting from the application either by resetting the terminal, or by sending the command ESC [ 5 ; 23 ! t. The current definitions of all programmable message characters are stored in NVM.

#### Escape Message Character (ESC)

The programmable escape message character is interpreted by the terminal as a signal that the following characters are part of a terminal command (Section 2.1.2). Although the default value of this programmable character is ASCII 027, known as the Escape Control Code, the two should not be confused. The ESC key on the keyboard sends the Escape Control Code to the Keyboard Communication Line (usually Line 1), regardless of how the Escape Message Character is programmed. However, if the programmable Escape Message Character has been specified (by the Change Message Character command) to be something other than ASCII 027, then the terminal will not interpret ASCII 027 as a command introducer.

Since a separate programmable Escape Message Character is stored in each Attribute List (Section 2.6.3), the terminal can be configured to recognize different characters as command introducers, depending upon which line or device they come from.

#### 2.3.4 Data Communications Considerations

Naturally, proper terminal operation requires compatibility between host computer, data transmission equipment (front-end processors, telephone lines, modems, etc.) and terminal. See Appendix D for a detailed explanation of the Concept AVT communication interfaces. Contact the systems programmer or administrator of your computer installation for information on the data communications requirements of your host system. The following considerations may be useful in situations where the terminal seems to be able to communicate with the host computer, but there is evidence of intermittent loss or corruption of data.



## Stop Bits and Parity

As explained in Appendix D, character data is transmitted as a 7 bit data word, preceded by a single start bit, and followed by 0 or 1 parity bit, and 1 or 2 stop bits. While the settings for stop bits and parity may not matter in purely conversational uses of the terminal (especially at relatively low baud rates), they can be the cause of seemingly unrelated problems in other applications using the same terminal and host computer. In particular, incorrect settings for stop bits and parity can cause problems whenever data is transmitted in blocks (rather than character by character) from the terminal. This includes the Transmit command (Section 2.9) as well as programmable keys and cursor pad keys set to operate in transmit mode (Section 2.4.1).

The reason for this is that Concept terminals are generally able to correctly process incoming data at any baud rate up to 9600, regardless of what combinations of stop and parity bits have been used in generating the character. Most host systems are similarly tolerant, but only at much lower speeds. In conversational mode, data is transmitted to the host system only at the rate at which the terminal operator can type, so the host system is able to correctly process the data without regard to stop bits and parity. However, when the terminal transmits data in a block, it transmits the characters as fast as the baud rate allows. If, for example, a key is programmed to transmit the sequence 'LOGON<CR>', the six characters are transmitted as fast as the baud rate will allow. Even at 300 baud, this is approximately 30 characters per second, which is considerably faster than anyone could type them, and often faster than the host system can process them correctly if the settings for stop bits and parity are wrong.

This problem can arise when as few as two characters are transmitted together, and the symptom is usually that the first character is processed correctly by the host, but subsequent characters are misinterpreted.

To correct the problem, try each of the 10 possible combinations of stop bit and parity settings, using a transmit mode function key to generate a standard test string of about 50 characters. The best way to determine exactly how the strings are being processed is to have a text editor store them in a file, and then display the file on the terminal.

Note: It is possible for characters to be echoed correctly, and yet processed incorrectly by the host computer. This could occur if the echo were provided by a front-end processor that was more tolerant of stop bit and parity settings than the host computer.

## Buffer Overflow Control (XON/XOFF)

Commands and data sent to the terminal from a host computer can be lost or misinterpreted due to overflow of the terminal's input buffer. Since Concept AVT terminals have large input buffers (256 characters for Line 1 and 128 characters each for Lines 2 and 3), and since the terminal can put characters up on the screen faster than they can be transmitted (even at 9600), there is no need to worry about input buffer overflow when processing ordinary data.

However, some terminal commands, such as those involved in editing, require significantly more time to process than an equivalent number of ordinary characters. Since data continues to fill the input buffer while commands are being executed, it is possible (especially at high baud rates) for the input buffer to overflow, resulting in loss of data, while the terminal is executing certain commands.

To prevent this from happening, make sure that the terminal is set to use XON/XOFF protocols to control the transmission of data from the host. By default, the terminal uses this protocol for Line 1, but not for Line 2 or Line 3. The command `ESC [ 1 ; 2 * q` causes the terminal to use XON/XOFF for Line 2. When XON/XOFF is enabled for a line, the terminal sends `↑S` to signal the host to stop transmission, and `↑Q` to signal the host to resume transmission. See Section 3.9 for details on terminal buffer overflow control.

Note: The host system must be configured to respond to `↑S` by ceasing transmission, and to `↑Q` by resuming transmission. Many operating systems allow the individual user to enable or disable host response to XON/XOFF controls sent from the terminal. Contact your computer installation for information on host data communications capabilities.

If the host cannot be configured to respond appropriately to XON/XOFF protocols, terminal input buffer overflow can be prevented by adding timing or delay characters after time-consuming commands to give the terminal time to complete the command before the next significant characters are received. `↑@` (NUL, ASCII 000) and `RUB` (ASCII 127) may be interspersed anywhere in data or commands as timing characters. The number of such characters required depends upon the command and the baud rate. See Appendix F.

#### Response to `↑S/↑Q` from the Host

In factory default state, the terminal responds to `↑S` (XOFF, ASCII 019) by ceasing all transmission to the line from which the `↑S` was received until a `↑Q` (ASCII 017) is received from the same line. This applies to character data typed on the keyboard as well as to block mode transmissions and function keys in transmit mode. Though this feature facilitates communication between terminal and host in block mode applications, it can lead to problems if the host system uses `↑S` for purposes other than buffer overflow control. In particular, some systems use `↑S` as an end-of-line indicator in sending data to the terminal. If they do not follow the `↑S` with a `↑Q`, the terminal may appear to be "hung", when in fact it is just waiting for a `↑Q` to resume transmission to the host.

Beginning with version 33333 of the terminal software, a Host Overflow Control command is provided to allow the user to determine whether the terminal will respond in this way to `↑S`. See Appendix H.

In earlier versions of th terminal software, this problem can be solved by defining the programmable delay character (DLY) to be ↑S. This causes the terminal to interpret ↑S as a timing character rather than as XOFF. When interpreted in this way, the ↑S has no effect on the terminal at all (it is neither displayed nor executed as a command). The Change Message Character command

ESC [ 4 ; 19 ! t

causes the terminal to regard ↑S as a delay character regardless of what line it comes from. To make the terminal respond to ↑S again, enter ESC [ 4 ! t , which resets the programmable delay character to its default value (not used).

Note: Concept AVT terminals regard ↑@ (NUL, ASCII 000) and RUB (ASCII 127) as timing character no matter how the programmable delay character is defined.

## 2.4 Programmable Keys and Functions

This section explains the use of the programmable keys described in Section 1.8.4, as well as the cursor pad keys, which, though not fully programmable, can operate in either execute or transmit mode. Also discussed in this section are three terminal functions that involve storage of user-defined data: Latent Expression, Alert Line Message, and Answerback Message. See Section 3.12 for detailed information on commands described in this section.

A programmable key is one whose function can be defined by the user either directly from the keyboard, or by a program running on the host computer. Programmable keys can be used to reduce the number of keystrokes required to perform any task by permitting substitution of a single keystroke for a frequently used sequence of many keystrokes. Since programmable keys can also be executed ("struck") by commands sent from the host computer, they can be used to store data frequently transmitted from the host, thereby reducing the volume of data transmission.

Concept AVT terminals have 31 programmable keys: 12 gray Function Keys, labelled F1 to F12, located over the main pad, and 19 beige Numeric Pad Keys (including the four keys labelled PF1 - PF4 located over the right end of the main pad). See Figure 1-1 on the inside front cover of the manual. Since the 12 grey Function Keys can each be assigned two functions (unshifted and shifted), there are a total of 43 programmable key functions. The expressions 'programmable key', 'function key', and 'programmable function key' are generally used interchangeably in this manual.

Programmable keys can be defined to store any combination of ASCII characters, including control codes. An individual key can be programmed to include ordinary data, terminal commands or both. For example, a user who frequently runs the program EDITOR might want to define a programmable key to store the string "RUN EDITOR" followed by a carriage return. This would enable the user to initiate the program by striking a single key, rather than 11. Similarly, a user who frequently switches between 80 and 132 column display width could save keystrokes by programming one key to contain the terminal command that changes the display from 80 to 132 columns ( ESC [ = 103 h ), and another to contain the command that changes the display from 132 to 80 ( ESC [ = 103 l ).

### 2.4.1 Execute and Transmit Modes

The operation of a programmed key depends upon whether it has been defined to be in execute or transmit mode. Striking a key programmed in execute mode has exactly the same effect as typing on the keyboard the sequence of characters stored on the key. If a key is programmed to operate in transmit mode, striking it causes the characters stored on the key to be transmitted to the host computer without being displayed on the screen

In a full duplex, character mode (conversational) environment, there is no effective difference between execute and transmit with regard to ordinary data, since keyboard input is sent to the host computer character by character without being displayed on the screen. However, execute mode should be specified when the programmable key is being used to issue a

terminal command, such as changing the display width.

The difference between execute and transmit mode function keys is most important in block mode applications (see Section 2.9). When the terminal is in Block Mode, keyboard input is displayed on the screen as it is typed in, but is not sent to the host computer until a Transmit command is issued. Since characters stored on transmit mode function keys are sent to the host computer regardless of whether the terminal is in Block or Character Mode, transmit mode function keys enable the keyboard operator to communicate with the host.

### Programmable Key Defaults

A programmable key that has not been programmed is in its default state. When the terminal is powered up or reset, all programmable keys are returned to their "power up default" states. Associated with each programmable key is a pair of defaults: a default command that it can execute on the terminal, and a default sequence that it can transmit to the host computer. Which of these operations occurs when the key is struck depends on whether the key has been specified as a "default execute" or a "default transmit" key. In power-up state, Numeric Pad keys "execute"; that is, they enter individual characters. A complete list of default states for all programmable keys is given at the end of Section 3.12.

For example, if key F2 has not been specifically programmed (as described below) to have some other function, it can either execute the Delete Character command ( ESC [ P ), or transmit the sequence FKD 002 EOM to the host computer, depending upon whether it is currently a default execute or default transmit key. As indicated in the table in Section 3.12, when the terminal is powered up or reset, F2 reverts to being a default execute key; unless it is programmed to do otherwise, F2 executes a Delete Character command on the terminal, and transmits nothing to the host computer. Keys F1 through F5 all power up as default execute keys that perform the functions indicated on their front labels (see Section 2.5 on editing).

The Program Function Key command

```
ESC [ 2 ; 3 u
```

can be used to change F2 from default execute to default transmit (specified by the 3). Note that this command is similar to the one used below to program a specific sequence to execute or transmit, but in the present case, no text is specified.

By changing the second parameter to 2, the same command could be used to change F2 back to default execute: ESC [ 2 ; 2 u . By omitting the first parameter entirely, the same command can be used to set all 12 function keys (F1 - F12, shifted and unshifted) to default execute ( ESC [ ; 2 u ), or to default transmit ( ESC [ ; 3 u ).

### 2.4.2 Programming a Function Key

There are two ways to program keys from the keyboard: the string of characters to be stored on the key can be constructed on the display screen and then transferred to the key using the Display/Edit Function Key command, or the string of characters to be stored on the key can be entered directly as part of the Program Function Key command. Only the second of these options is generally used when programming keys from the host. Two examples are given below: in one case the key is programmed to transmit an ordinary character sequence to the host; in the second case, a key is programmed to invoke a terminal command. In each case the key is programmed first from the screen, and then directly.

To use these examples as a demonstration, enter Setup Mode and make sure that the cursor movement keys are in execute (CE) rather than transmit mode (CT).

#### Programming a Key to Transmit Data

For the first example, program function key F7 (unshifted) to transmit the sequence RUN EDITOR, followed by a carriage return <CR>, to the host computer. It may be helpful to substitute for RUN EDITOR<CR> a sequence that is a valid command to your host computer, since doing so may make it easier to verify that the key has been programmed correctly.

Begin by using the Display/Edit Function Key command to setup the terminal for programming function key F7:

```
ESC [ 7 ; 1 v
```

7 indicates the key to be programmed, and 1 requests the operation "Set Up for Edit". For convenience, the key to be programmed can also be specified by pressing it instead of typing in the number that corresponds to it in the table of Programmable Keys in Section 3.12. The terminal responds to this command by clearing the screen, displaying the current contents of key F7 in the upper left corner, and displaying a message on the status line at the bottom of the screen. This message shows the key being programmed, as well as the amount of memory currently available for programming keys (see Section 3.12). This command also puts the terminal in Local and Transparent Modes, as required for editing the character string to be stored on the key.

Assuming that F7 had not previously been programmed, the display in the upper left corner of the screen should show its current contents as |t| , where | is a delimiter, and t indicates that the key is in "default transmit" mode. To program F7, first use local terminal editing capabilities (Section 2.5) to change this display so that it looks like this:

```
|TRUN EDITOR 6|
```

It does not matter how this display is created, or where on the screen it is located. The character immediately following the first delimiter (|) must be either an upper case T (for transmit) or an upper case X (for execute). The beginning and ending delimiters must match (or the terminal will try to store all of display memory in the programmable key).

Once the correct message has been constructed, use the cursor movement keys to place the cursor at the first delimiter, and then enter the following command:

```
ESC [ 7 ; 2 v
```

where 7 indicates the key, and 2 specifies that it be programmed according to the message beginning at the cursor location. Here again, F7 itself could be pressed to indicate the key to be programmed. There is no visible response to this command, and the terminal remains in Local and Transparent Modes.

The entire process of programming a key described above could have been accomplished with just the Program Function Key command:

```
ESC [ 7 ; 1 u / R U N <Sp> E D I T O R <CR> /
```

where 7 indicates the key to be programmed, 1 indicates transmit mode, and / is the delimiter. In addition to involving fewer keystrokes, this method has the advantage that does not disrupt interactive use of the terminal: it does not put the terminal in Local or Transparent Mode, and it does not clear display memory. The disadvantage is that the command is not displayed on the screen as it is entered, so it is more difficult to catch errors.

To verify that the key has been programmed correctly, use the Display/Edit Function Key command to display the current contents of the key F7:

```
ESC [ 7 v
```

The string |TRUN EDITOR<sup>c</sup>| should appear on the screen at the current cursor position.

#### Terminal Commands on Programmable Keys

In the above example, the Display/Edit Function Key command was used to setup the terminal for programming the key. This is convenient, in that it shows the current contents of the key, but it is not necessary. All that is necessary is that the sequence to be stored on the key somehow be put up somewhere on the screen in the format shown above. Ordinarily this requires only that the terminal be in Local and Transparent Modes. Local Mode can be entered either through Setup Mode or by entering ESC [ = 111 h . ESC Q puts the terminal in Transparent Mode. The next example assumes only that the terminal is in Local and Transparent Modes.

In this example, numeric pad key PF1 is programmed to change the display width from 80 to 132 characters. To do this, PF1 must have the same effect as entering the appropriate terminal command from the keyboard. This means that PF1 must be programmed to execute the sequence ESC [= 103 h . To program PF1 from the screen, first create the appropriate string somewhere on the screen:

```
|X E [=103 h|
```

Then position the cursor at the first delimiter, and issue the appropriate Display/Edit Function Key command:

```
ESC [ 101 ; 2 v
```

where 101 indicates key PF1, and 2 specifies that the key be programmed from the screen.

To achieve the same result using the Program Function Key command, enter

```
ESC [ 101 ; 0 u / ESC [ = 101 h /
```

where 0 indicates execute mode. Since execute mode is the default, the semicolon and the 0 could be omitted: ESC [ 101 u / ESC [ = 103 h / .

Note: To program an execute mode function key so that the ESC stored on it will be interpreted as a character (the Escape Control Code, ASCII 027) rather than as the terminal's command introducer, strike the ESC key twice for every occurrence. For example, to program key F10 so that striking it will have the same effect as typing the sequence 'LOGIN<ESC>':

```
ESC [ 10 u / L O G I N <ESC> <ESC> /
```

The same command would be sent from the host to program F10 in this way.

### 2.4.3 Numeric Pad Keys

In addition to being individually programmable as described above, keys in the Numeric Pad can be switched as a group between Numeric and Application modes. This feature is provided for VT100 compatibility. ESC = puts the Numeric Pad in Application Mode by reprogramming all keys in this pad to default transmit operation; ESC > puts the Numeric Pad in Numeric Mode by reprogramming all keys to default execute operation, as indicated in the table of Programmable Keys in Section 3.12.



#### 2.4.4 Cursor Pad Keys

Keys in the Cursor Pad can be set, either individually or in a group, to one of four modes of operation: Execute, Transmit, Execute and Transmit, or Disabled. In Execute Mode, each key performs the terminal function indicated on its key top label: ↓ moves the cursor down one line; PRINT attaches the printer or other peripheral connected to Line 2; CMD introduces a terminal command. In Transmit Mode, each key transmits the terminal command that would perform the function indicated on its key top label. In Execute and Transmit, both functions are performed. When a key is disabled, striking it causes the bell to ring, but no function is executed, and no sequence is transmitted.

To provide for VT100 compatibility, the sequences transmitted by cursor pad keys can be modified by setting HDS Mode 201. See the table of Cursor Pad Operation in Section 3.12 for further details.

#### 2.4.5 Other Programmable Functions

There are three user defined data fields that are programmed in the same way as programmable keys, but which have different functions. Unlike programmable keys, each of these fields has a limit on the number of characters that it can store.

##### Latent Expression

The Latent Expression is an 80 character field that is stored in Non-Volatile Memory (NVM) and invoked whenever the terminal is powered up or reset. It operates in execute mode only, and can be used to perform any terminal setup procedure the user wishes to execute whenever the terminal is powered up or reset. Up to 80 characters can be stored in the Latent Expression.

Note: Do not store in the Latent Expression any command or sequence of commands that has the effect, directly or indirectly, of resetting the terminal. Since the Latent Expression is invoked whenever the terminal is reset, including such a command would put the terminal in a processing loop that could not be broken without disassembling the terminal and physically resetting NVM. Commands that reset the terminal include Reset Terminal ( ESC [ c ), Reset to Factory Default ( ESC [ 9 ~ ), and Set Display Pages ( ESC [ pages ! p ).

The terminal would also be useless if the Keyboard Lock command ( ESC ` ) were programmed into the Latent Expression.

For example, the Latent Expression could be used to enter "LOGIN<CR>" as if it had been typed on the keyboard. This would be programmed from the screen in the same way as a programmable key (above) with the exception that no X or T is used following the initial delimiter to specify operational mode. Put the terminal in Local and Transparent Modes (as described above) and construct the string |LOGIN| on the screen. Then position the cursor at the first delimiter and enter the command ESC [ 99 ; 2 v , where 99

indicates the Latent Expression, and 2 specifies that it is to be programmed from the screen. To verify that it has been programmed correctly, use the Display/Edit Function Key command to display the current contents of the field: ESC [ 99 v .

The same effect can be achieved using just the Program Function Key command:

```
ESC [ 99 u / L O G I N <CR> /
```

Once the Latent Expression has been programmed correctly, it is automatically stored in NVM, where it remains until another command is executed to program the Latent Expression, or the terminal is reset to factory default state (Section 2.2.4).

#### Programming Keys from NVM

Though the Latent Expression can also execute terminal commands, it would generally not be used in this way, since the entire terminal configuration can be saved in NVM. One common use of the Latent Expression, however, is programming function keys, since these are not saved in NVM. For example, PF1 could be programmed by the Latent Expression to contain the terminal command that changes the display width from 80 to 132 columns. The text to be stored in the Latent Expression is the Program Function Key command

```
ESC [ 101 u / ESC [ = 103 h /
```

This stores the terminal command ESC [ = 103 h on programmable key number 101 (PF1), which is to operate in execute mode (default).

To program this from the screen, put the terminal in Local and Transparent Modes, and construct the text to be stored in the Latent Expression on the screen, using the ESC key in the upper left corner of the main pad to enter  $\xi$

```
| $\xi$  [101u/ $\xi$  [=103h/|
```

Then position the cursor at the first | and enter ESC [ 99 ; 2 v . Notice that two sets of delimiters are used: | indicates the beginning and end of the text to be stored on the Latent Expression, and / indicates the beginning and end of the text to be stored on the function key by the Latent Expression. Any two characters may be used, as long as neither occurs in the delimited text.

This Latent Expression could have been programmed directly:

```
ESC [ 99 u | ESC [ 101 u / ESC [ = 103 h / |
```

either from the keyboard or from the host.

### Alert Line Message

The Alert Line Message is a 40 character field that is displayed on the 25th line of the screen in response to a Display Status Line command that specifies the Alert Message Line (for example, ESC [ 1 ; 6 \* u). (It is also displayed on the screen when it is programmed.) It can serve a variety of application functions, such as providing an area independent of main display memory for displaying warnings and error messages. Since it would normally be used only under application control, the Alert Line Message is not programmable from the screen. It can be programmed either from the keyboard or from the host computer using the Program Function Key command with the value 97 for the key parameter. To program the message 'Invalid Command', enter:

```
ESC [ 97 u / I n v a l i d <Sp> C o m m a n d /
```

See Section 2.2.1 for further information on status lines.

### Answerback Message

The Answerback Message is a 20 character field that is saved in NVM and transmitted to the host computer in response to ↑E, the Transmit Answerback Message command (see Section 2.2 on status information). It can be used to provide terminal identification or status information in a user-defined format. To program the Answerback Message either from the screen or directly (from the keyboard or from the host computer), follow the same procedure as for the Latent Expression, substituting 98 for 99 as the programmable function identifier. For example, the comand

```
ESC [ 98 u / C o n c e p t <Sp> A V T /
```

would store the text "Concept AVT" as the Answerback Message.

**Note:** The Answerback Message is transmitted exactly as it is programmed, without any Start of Message or End of Message character. If the host operating system requires a carriage return or other control code as a delimiter, that must be programmed as part of the Answerback Message.

## 2.5 Editing

The current contents of display memory, including what is visible on the screen, can be modified by a number of flexible and powerful editing commands. All editing functions, including insertion and deletion of characters or lines, and erasure of specified areas, can be invoked either from the keyboard or by a program running on the host. All of the features described below can be demonstrated by putting the terminal in Local Mode and typing in sample text to be edited.

Editing commands are usually executed from the keyboard in block mode applications (Section 2.9) or when the terminal is being used in Local Mode. By default, the first four programmable keys (F1 - F4) execute the editing functions indicated on their front labels. (see Section 2.4 on programmable keys). Like most terminal commands, though, editing commands can be programmed onto any of the programmable keys.

Many "full screen" applications, notably full screen text editors, maintain control of the terminal's display memory by having all editing functions invoked by the program. This is typically achieved by having the terminal operator strike transmit mode programmable keys or control codes to request editing functions. The application program then responds by transmitting the appropriate editing command. Full screen editors designed to support the Concept AVT generally use keys F1 - F4 (in transmit mode) for this purpose. In such cases, the terminal operator may not realize that the actual editing commands are being sent from the host computer.

The Delete Character and Insert and Delete Line commands include a parameter that specifies how many characters to delete, or how many lines to insert or delete. This significantly increases the speed of the terminal in carrying out these functions, since a large amount of display memory can be modified at the cost of processing a single terminal command. Character insertion differs somewhat from other editing functions in that it is accomplished by setting a mode. Details on the operation of editing commands can be found in Section 3.6.

### 2.5.1 Insert Mode (HDS Mode 4)

Normally when a character is sent to the screen it replaces the character currently stored at the cursor position. Setting HDS Mode 4 causes subsequent characters to be inserted just ahead of the current cursor position, rather than replacing the character at the cursor position, until HDS Mode 4 is reset. When a character is inserted, characters to the right of the cursor are shifted to the right one space, and the cursor moves to the next position, so that the second character inserted is placed to the right of the first character inserted.

Although the standard Set and Reset Mode commands could be used to control insertion, a special command, Toggle Insert Mode ( ESC l ) is provided for convenience. This command, which is executed by key F1 in its default state, Sets HDS Mode 4 if it is Reset, and Resets it if it is Set.

### 2.5.2 Delete Character

The Delete Character command causes a specified number of characters to be deleted, starting with the character at the current cursor position. Thus, ESC [ 17 P deletes the character at the current character position, and the 16 characters following it. For each character deleted, the characters to the right are shifted left one character position, as far as the editing extent permits (see below). The DEL/CHAR key (F2) executes a Delete Character command with a repeat count of 1 ( ESC [ P ) in its default state.

### 2.5.3 Insert Line

The Insert Line command causes a specified number of cleared lines to be inserted at the current cursor position by pushing existing text down the required number of lines (starting with the current line). For example, ESC [ 7 L inserts 7 lines. The DEL/INS L key (F3) unshifted executes an Insert Line command with a repeat count of 1 ( ESC [ L ).

### 2.5.4 Delete Line

The Delete Line command causes a specified number of lines to be deleted, starting at the current cursor position. All subsequent lines are scrolled up, and a cleared line created at the bottom of the window for each line deleted. For example, ESC [ 3 M deletes the line on which the cursor is currently located, and the two lines following it. The DEL/INS L key (F3) shifted executes a Delete Line command with a repeat count of 1 ( ESC [ M ).

### 2.5.5 Erase (Clear)

There are three Erase commands (Erase in Window, Erase in Line, and Erase in Field) that clear specified areas of display memory. See Sections 3.6.1 on clearing display memory. Each of the Erase commands includes a parameter that determines what portion of the window, line, or field is to be cleared. The ERAS key (F4) executes an Erase in Line command ( ESC [ K ) when unshifted, and an Erase in Window command ( ESC [ J ) when shifted.

### 2.5.6 Editing Extent

When a character is inserted or deleted, characters immediately to the right of the cursor position are shifted to the right (to make room for the new character) or to the left (to take up the space vacated). The shifting to the right or to the left proceeds until a space (ASCII 032) is encountered which has the non-display attribute on (see Section 2.6.2 on Character Attributes). If a non-displayed space is not encountered, the shifting continues up to the end of the current window, line, or field, depending on the Editing Extent.

By default, the editing extent is the current line. This means that when a character is inserted in the middle of a line, for example, every character from the point of insertion to the end of the line is moved one character to the right, and the last character in the line is lost.

The Select Editing Extent command can be used to change the editing extent to Window ( ESC [ Q ) or Field ( ESC [ 3 Q ) as required by the application. When the editing extent is Window, character insertion works the same as when the editing extent is line, except that the last character on the line is moved to the first position of the next line, which is therefore shifted to the right one position, so that the last character on that line is moved to the beginning of the next line. This "wrapping around" continues up to the end of the window, where the last character shifted to the right is lost. The same process occurs in reverse when a character is deleted, except that when the last character in the window is shifted to the left, a "cleared" character (see below) is created in its place. If the window is large, the shifting of data can be time consuming.

When the editing extent is Field, the effects of character insertion and deletion are confined to the current field (see Section 2.10.1 on fields and protection).

### 2.5.7 Clear Characteristics

Many editing functions involve "clearing" an area of display memory, whether it be a single character, a field, a line, or a window. By default, a character position is cleared by making it a space (ASCII 032) with all attributes set to default, so that it appears blank on the screen. The Set Clear Characteristics command can be used to specify a different character and/or set of attributes to use in clearing display memory. See Section 3.6.1 for a detailed explanation, including examples of changing clear characteristics.

## 2.6 Character Sets and Attributes

The way in which a character is presented on the screen, as well as some important aspects of the way it is processed, are determined by the Attribute List currently in use by the line or device from which the character was received. As explained below, each line or device can use a different Attribute List, so characters coming from different lines or devices can be processed differently.

### 2.6.1 Character Sets

Concept AVT terminals can be equipped with up to four different character sets, enabling the user to switch between character sets by issuing simple terminal commands, either from the keyboard or from the host computer. The Concept AVT is standardly equipped with both the ASCII Character Set (Appendix A.1) and the VT100 and Concept Special Graphics characters (Appendix A.2). The Concept AVT-APL is standardly equipped with these two plus the APL Character Set (Appendix A.3). Other character sets, including special mathematical/technical symbols, are available from HDS. Each character set is implemented as a separate ROM chip (two for APL), so installation of new character sets is relatively easy.

Each character set associates a particular visual representation with each of the 128 combinations of 7 data bits recognized by the terminal as valid characters. The choice of character set by itself has no effect on the way a particular character is processed by the terminal other than the way the character is represented on the screen. (However, see below regarding the interaction between the APL character set and Overstrike Mode). For example, chart location 013 is processed as a carriage return regardless of which character set is in use. Of course, if the terminal is in Transparent Mode (Section 2.3.2), the representation of the control code depends on the character set.

If an unused character set position is selected, data coming from the keyboard and the host computer is processed normally, except that all characters are displayed as uniform blanks (the cursor moves across the screen, but nothing appears).

#### Normal and Alternate Character Set

Although all four character set positions are always accessible by issuing simple terminal commands, at any time, one of the four is defined as the "normal" character set, and one (possibly the same one) as the "alternate" character set. This makes it possible to use a single control code to switch between the two most commonly used character sets. The Select Alternate Character Set command, ↑N (ASCII 015, "Shift In") selects whichever character set is currently defined as the Alternate Character Set. Similarly, ↑O (ASCII 017, "Shift Out") selects whichever character set is currently defined as the Normal Character Set. See Section 3.3 for details of commands affecting the choice of character sets.

The current choices for Normal and Alternate Character Set are shown near the left end of the Programmer Status Line. The cset values for Normal and Alternate are shown on either side of a < or > which indicates which of the two is currently in use. In factory default state, the status line shows B<B, indicating that both Normal and Alternate character sets are defined as ASCII, and that the Normal Character Set is currently in use. (This configuration was chosen to maintain VT100 compatibility.)

Note: The definitions of Normal and Alternate Character Set apply to all attribute lists, but the selection of which to use -- Normal or Alternate -- applies to the individual attribute list.

The Define Normal Character Set command, ESC ( cset , determines which of the four character sets is to be treated as Normal. For example, the command ESC ( 0 would define VT100 and Concept Special Graphics as the Normal Character Set. If the factory default character set configuration ( B<B ) had been in effect, this command would change the character set currently in use: the status line would read 0<B , and subsequent data would be displayed on the screen as VT100 and Concept Special Graphics characters instead of as ASCII characters. However the Define Normal and Define Alternate Character Set commands do not affect which of the two is currently selected. Therefore a Define Alternate Character Set command issued when the character set in use is the Normal Character set has no immediate effect on the appearance of data on the screen. Its effect would only become apparent following the next Select Alternate Character Set command (↑N).

#### APL and Overstrike Mode

Although Concept AVT-APL terminals are standardly equipped with the APL character set (occupying the fourth and overstrike character positions), APL is not defined as either the Normal or the Alternate character set when the terminal is in its factory default state. The standard APL character set configuration can be defined by simply selecting APL (rather than ASC) in Setup Mode. If APL is already indicated upon entering Setup Mode, strike the NEXT VAL key twice.

Selecting APL in Setup Mode defines ASCII as the Normal and APL as the Alternate Character Set, and selects the Alternate Character Set (APL). When the character sets are defined in this way, ↑N turns overstrike on in addition to choosing the APL character set, and ↑0 turns overstrike off in addition to choosing the ASCII character set. Overstrike is not coordinated with choice of character set if APL is defined as the Normal Character Set.

Users working primarily in APL can make the terminal power up in APL by using the SAVE key to store this configuration in NVM before exiting Setup Mode.



### 2.6.2 Character Attributes

The way in which data is displayed on the screen depends on character attributes, as well as the choice of character sets. These attributes include both graphic characteristics that serve to enhance visual differentiation of data, and logical characteristics that affect the way in which data is processed by various terminal functions. Both kinds of attributes are frequently used in text editing as well as data entry and retrieval applications.

There are six character attributes, identified by numbers as in the list below. Associated with each character location in display memory is a list that shows which of the six attributes is in its default state, and which is in its alternate state. Because the default states, taken together, represent the "normal" appearance of characters on the screen, it is convenient to refer to attributes by their alternate states. The appearance of the character on the screen, as well as certain aspects of the way it is processed, are determined by the particular combination of default and alternate attributes associated with that character. Attributes are independent of one another, and may be used in any combination (though it is possible to link Bold and Protection; see HDS Mode 118).

Note: Reverse Video is the only alternate attribute that can be selected for an overstruck character.

#### 1 Bold / Normal Brightness

Light areas of the screen, whether characters or background, appear brighter than in default state. Bold is normally used for highlighting in text editing and data entry applications. See also attribute 99 (Protection).

Note: This attribute can also be used to achieve "half-bright" display: use Bold in displaying text that is to appear normally on the screen, and Normal Brightness to display text that is to appear half-bright. Then, reduce the overall brightness of the screen by adjusting the brightness control (see Figure 1-1).

#### 4 Underline / No Underline

All characters (including spaces) appear underlined. Underline characters (ASCII 095) received as input can be stored either as attributes or as separate characters, depending on Alternate Processing Mode 121 (see Section 4.5). Also, if Mode 115 is reset, each underlined character is followed by a backspace and an underline character (ASCII 095) when transmitted.

5 Blinking / Non-Blinking

Character blinks on and off, while background and underline (if any) remain constant.

7 Inverse Video / Normal Video

Character and background are opposite to Screen Video (Mode 205). If Screen Video is light characters on dark background, then characters with inverse video appear as dark characters on light background. If Screen Video is dark characters on light background, characters with inverse video appear as light characters on dark background. This is the only attribute available for overstruck characters.

8 Non-Displayable / Displayable

Character is not displayed on the screen, but is otherwise processed normally. The area on the screen where it would have appeared remains blank, but the character is stored in display memory and can be made to appear on the screen by changing this attribute to its default state. This attribute can be used for security purposes. See Section 2.5 for the effect of Non-Displayable spaces on character insertion and deletion.

(=)99 Protection / No Protection

Protected characters cannot normally be overwritten, erased, or deleted, though this protection can be overridden by setting appropriate Alternate Processing Modes. (See HDS Modes 6 and 108.) Protection does not by itself affect the appearance of a character on the screen, but setting HDS Mode 118 links attributes 1 (Bold) and 99 (Protection) so that protected characters are automatically displayed in Bold. When HDS Mode 1 is reset, protected characters are not sent out in response to Transmit commands, and are replaced by spaces when sent out in response to Print commands (see Section 2.8.1). Protection is used in a variety of ways in block mode and data entry applications (see Sections 2.9 and 2.10).

Note: As a character attribute, protection is associated with individual characters. A "protected field" is therefore just a string of protected characters. See Section 2.10.1 on Fields and Protection.

**Character Attributes**

Attribute	Default State	Alternate State
1	Normal Brightness	Bold
4	No Underline	Underline
5	Non-blinking	Blinking
7	Normal Video	Inverse Video
8	Displayable	Non-displayable
(=) 99	No Protection	Protection

Note: Whenever attribute 99 is included in the parameter list of any command, the entire parameter list must be preceded by an equal sign (ASCII 061).

### Changing Character Attributes

To use the following discussion as a demonstration, reset the terminal to factory default state ( ESC [ 9 ~ ), and enter Local Mode (through Setup Mode, or by entering ESC [ = 111 h). Strike the STATUS key five times while holding down the SHIFT key to display the Programmer Status Line (K1) referred to below.

Looking at the Programmer Status Lines, the string 1-0m located near the left end of the line shows both the attribute list (1) in use by this line and the attributes that are in their alternate state (0). The 'm' is the command identifier of the Select Alternate Attribute command, which provides a convenient reminder of how the list can be changed.

For example, the command ESC [ 7 m selects the alternate state of attribute 7 (Inverse Video) for the attribute list currently in use by the requesting device. The Programmer Status Line should change to read 1-7m , and subsequent keyboard input should appear in inverse video. Type a few characters to verify the effect of this command. To see the relationship between screen video and character video, enter Setup Mode and switch back and forth between normal and reverse screen video ('V' near the right end of the status line). Display Programmer Status Line K1 again.

The effect of the Select Alternate Attribute command is to add alternate attributes to those already on the list. Thus, Blinking can be added to Inverse Video (already selected) by the command ESC [ 5 m . Now the status line shows 1-5;7m , and subsequent keyboard input appears in blinking inverse video. Using 0 for an attribute number resets all attributes to their default states. Since 0 is the default value for this parameter, ESC [ m changes the status line to read 1-0m , and causes subsequent keyboard input to appear normally on the screen.

A single Select Alternate Attribute command can be used to set all the attributes required at a given time. Since parameters are processed in the order in which they are listed, 0 can be used to reset the Attribute List to default condition prior to specifying alternate attributes. For example, ESC [ = 0 ; 1 ; 99 m selects default state for all attributes, and then selects the alternate state for 1 (Bold) and 99 (Protection). Notice that the equal sign is used because 99 is in the parameter list.

To see the effect of protection, type a few characters on the screen using the attributes just selected. Then back space so that the cursor is located at a protected character, and try to type over the protected character. The keyboard bell should sound to indicate that protected characters cannot be overwritten.

The set of character attributes currently in use can also be changed by changing attribute lists, provided that two different attribute lists have been defined. In the preceding example, Attribute List 1 was left with Bold

and Protection selected ( 1-1;99m on Programmer Status Line K1). Since the keyboard is currently using an attribute list that has been changed from its default state, the way in which keyboard input is displayed on the screen can be changed by selecting one of the attribute lists still in default condition. For example, ESC [ 2 ; 9 ! u selects attribute list 2 for the keyboard. Now the status line reads 2-0m , and keyboard data is displayed normally on the screen.

### 2.6.3 Attribute Lists

During normal character input, the combination of attributes associated with a given character in display memory is determined by the attribute list being used by the device from which the characters were received (generally referred to as the "current attribute list"). The association of an attribute list with a device is determined by the Select Attribute List command. The particular combination of attributes in a given list is determined by Select Alternate Attribute and Select Default Attribute commands received from the device currently associated with the attribute list in question. See Section 3.7 for details on commands that affect character attributes.

In addition to specifying which of the character attributes discussed above are in default, and which in alternate state, the attribute list contains the settings for a number of other terminal operating characteristics that affect the processing of individual characters.

Character Attributes	Overstrike (102)
Escape Message Character	Scrolling (104)
Editing Extent	Tab Processing (105)
Transmit Unprotected/All (1)	Autotabs (106)
Transparent Mode (3)	Cursor Wraparound (107)
Replace/Insert Characters (4)	Protected Field Overwrite (108)
Erase Protected Characters (6)	Character/Attribute Replacement (120)
Linefeed Processing (20)	Character Wraparound (207)

**Figure 2-1**  
**Attribute List Contents**  
 (HDS Mode Numbers in Parentheses)

In factory default state, all four attribute lists are identical, and all devices use Attribute List 1. In this configuration, the fact that the operating characteristics shown in Figure 2-1 are stored in attribute lists is of no significance. However, the selection of different attribute lists for different devices provides a way of making the terminal process characters from different devices differently.

Two commands are provided for manipulating attribute lists. The Select Attribute List command associates an attribute with a specific device. For example, ESC [ 2 ; 1 ! u selects Attribute List 2 for Line 1. Once this command has been processed, subsequent commands received from Line 1 that affect settings stored on an attribute list will affect Attribute List 2. There is also a Copy Attribute List command. For example,

ESC [ 1 ; 1 ! v

makes Attribute List 1 the same as the attribute list currently in use by Line 1.

#### 2.6.4 Block Attribute Change

As indicated earlier, the set of attributes associated with a character in display memory is initially determined by the attribute list in use by the line or device from which the character was received. However, it is also possible to modify the attributes associated with a character or characters already stored in display memory. See Section 2.10.8 on forms handling for examples.

#### 2.6.5 Character / Attribute Replacement

Ordinarily, when a character in display memory is replaced by a new character, the attributes associated with the old character are replaced by the current attributes (that is, the attributes in the list associated with the device from which the character was received). However, by setting Character/Attribute Replacement (HDS Mode 120) to Character Only (Set state), it is possible to have the attributes remain constant while the characters change. This feature might be used in a block mode application so that characters typed on the keyboard would take on different attributes depending on where they appeared on the screen.

The display attributes of a given area of display memory are also affected by commands (such as Delete Character) that "clear" display memory. (See Section 3.6.1 for a discussion of clearing display memory.) The attributes associated with a cleared area of display memory may be specified via the Set Clear Characteristics command (Section 3.6).

## 2.7 Display Memory and Windows

Concept AVT terminals have 4 (optionally 8) pages of display memory. Since each page stores 24 lines (in 80 column mode), up to 96 (optionally 192) lines of data can be stored in the terminal. Data is stored in display memory as it is received by the terminal. Only data "networked to the display" is stored in display memory. Normally all communication lines are networked to the display; the keyboard is networked to the display unless the terminal is in Full Duplex and Character Mode (see Section 2.8.2). As explained in Section 1.5, some portion of display memory is always "visible": if the display is blank, then either the "visible" portion of display memory is blank or all of the characters are set to non-display (see Section 2.6.2 on character attributes). See Section 3.10 for details on commands affecting display memory and window configuration.

### 2.7.1 Allocating Display Memory

Display memory can be used either to store data that has been sent to the screen, or to supplement the memory available for programmable keys, thereby greatly increasing the amount of data and/or commands that can be stored on them. By default, all four (or eight) pages of display memory are allocated to display. The Set Display Pages command can be used to specify the number of pages to be allocated to display, the remainder becoming available for programmable key storage. For example, ESC [ 3 ! p reserves 3 pages of memory for display, allowing the remaining page (or 5 pages) to be used for programmable keys. See Section 3.12 for more information on how this affects programmable key storage.

Note: The Set Display Pages command causes the terminal to be reset to power-up defaults. Do not include this command in the Latent Expression, since doing so will cause the terminal to go into a processing loop that can only be ended by disassembling the terminal. See Section 2.4.5.

### 2.7.2 Display Width (80 or 132 Columns)

The width of the display can be switched between 80 and 132 columns either in Setup Mode, or by issuing simple commands either from the keyboard or from the host. Since 132 columns is the Set state of HDS Mode 103, it can be selected by the command ESC [ = 103 h . The corresponding Reset Mode command, ESC [ = 103 l , switches back to 80 columns. Data currently on the screen is preserved when switching from 80 to 132 column display width. However, when the display width is changed from 132 to 80, data currently displayed in columns 81 to 132 is lost. Users who switch frequently between the two might program function keys to execute these commands (Section 2.4.2).

To maintain VT100 compatibility, display width is also implemented as DEC Mode 3, so the command ESC [ ? 3 h could be used to select 132 columns, and ESC [ ? 3 l to select 80 columns. This method is not recommended however, since both commands clear all of display memory, and all data is lost.

### 2.7.3 Interactive Uses

The ability to store 96 (or 192) lines of data in the terminal can be of great value in any interactive application, especially at low baud rates. At any point in the interaction, the terminal operator can "look back" over data that appeared on the screen earlier -- data that would have been lost completely on a terminal that stored only what was visible at the time. The following example shows how this capability can reduce the need for hardcopy, and cut down on the volume of data transmitted from the host.

Using a typical interactive data analysis package might involve creating a command file, attempting to run it, and then correcting errors detected by the program. This can be difficult to do on an ordinary display terminal, since the error messages needed to correct the command file scroll up off the screen as soon as the user begins correcting the errors. On a Concept terminal, however, the data is retained even after it scrolls off the top of the screen. Using the cursor pad keys and the User Status Line, the user can manipulate terminal memory so that the error messages remain available as long as needed to correct the command file.

To begin with, the Cursor Pad must be in execute mode, which can be specified in Setup Mode. The objective is to retain certain information (error messages) in display memory while carrying out a perhaps extensive interaction (using a text editor to correct the command file). This can be accomplished by keeping the error messages at the "top" of display memory, and taking care that the interaction with the text editor be confined to the remaining lines. As long as data generated by the interaction is not allowed to go past the last line of display memory, the data stored at the top of display memory will not be lost.

The basic strategy is to keep track of the cursor position (with respect to display memory) as the interaction progresses, and reposition it as required to keep data from the interaction from exceeding the last line of display memory. Since the User Status Line records the current cursor position (in the format lin;col), it can be used to monitor the amount of display memory left. To simplify monitoring of the cursor position, strike the  $\uparrow$  key to home the cursor, and F4 (ERAS) shifted to clear all of display memory before running the program that creates the text to be saved. This will coordinate the "top" of display memory, as indicated by the cursor position shown on the User Status Line, with the information to be saved.

Assume now that running the analysis package with the initial version of the command file causes 43 lines of data to be stored in display memory including the "run" command entered at the keyboard). As a result, the cursor is positioned at the second column (following an operating system prompt) of the 44th line (044;002 on the User Status Line). Now the command file must be corrected by running a text editor and changing the file as indicated by the error messages stored in the first 43 lines of display memory. Soon after beginning the interaction with the text editor, the error messages scroll off the top of the screen. They are still available, however, and can be easily referenced by using the SCROL key shifted (or the PAGE key shifted) to move the display screen up with respect to display memory (so that an "earlier" portion of display memory is visible). The same keys can be used unshifted to move the visible display "down" with

respect to display memory, to proceed with the text editing interaction.

Note: The SCROL and PAGE keys do not cause any change in the contents of display memory; they change only the portion of display memory currently visible on the display screen. The cursor is moved only as necessary to keep it visible.

As the text editing interaction proceeds, data sent to the terminal begins to take up more and more of the remaining display memory, as indicated by the cursor position shown in the User Status Line. At a convenient point in the interaction, SCROL or PAGE up until the error messages being retained at the top of display memory become visible, position the cursor below the end of the error messages, and clear to the end of display memory (using ERAS, F4 shifted). As the interaction continues, data coming to the terminal will be stored in display memory just after the error messages still maintained at the top of display memory.

This process could be facilitated by programming a function key to position the cursor just below the error messages ( ESC [ 44 H ) and then clear the rest of display memory ( ESC [ J ). See Section 2.4.2 on programmable keys. Alternatively, the data to be retained through the interaction could be isolated as a separate window (see below).

#### 2.7.4 Windows

A window is a rectangular portion of display memory. Memory currently allocated to display can be divided into as many as four different windows, each with different dimensions. Although different windows can overlap, or even have the same dimensions, most applications that use multiple windows take advantage of the fact that data in two non-overlapping windows are completely independent with respect to nearly all terminal functions. The use of windows is controlled by the Select and Define Window commands described below. By default, the keyboard and all communication lines use Window 1, and all windows are defined as the first 24 lines of display memory. The choice of this small window as the factory default is required by VT100 compatibility, since the VT100 has only one page of display memory. Setup Mode provides an easy way to expand the window to include all of memory currently allocated to display.

For example, the Position Cursor command ESC [ 3 ; 19 H places the cursor at line 3, column 19 of the window currently in use by the device from which the command was received. If the current window is defined as columns 41 through 80 of all of display memory, for example, then the Position Cursor command would place the cursor at line 3 column 59 of display memory. However, since all cursor movement and editing commands operate in relation to the current window, it is generally not of interest where the cursor is in relation to display memory as a whole. Thus, when the terminal transmits the current cursor position to the host in response to the Transmit Cursor Position command ( ESC [ 6 n ), it reports the window-relative cursor position. Similarly, the Save Cursor and Restore Cursor commands save and restore the cursor relative to the window.



## Window Lists

The functional independence of different windows requires that most terminal commands that modify or manipulate display memory be "window-relative". All commands that move the cursor move it within the current window. All commands that reference an actual cursor address, interpret that address as relative to the home position of the current window. This is all completely transparent to the user, and will not be noticed at all in the default configuration, since all windows have the same definition.

Associated with each window is a list of operating characteristics and settings that apply only to that window. Figure 2-2 shows the settings stored in the window list.

Cursor Position	Transmit Termination (16)
Window Definition	Transmit Initiation (116)
Start of Print/Transmit	Cursor Addressing (206)
Scrolling Region (VT100)	Saved Cursor Position
Transmit Extent	Saved Character Attributes
	Saved Cursor Addressing Mode

**Figure 2-2**  
**Window List Contents**  
(HDS Mode Numbers in Parentheses)

Since all devices use Window List 1 in factory default state, changes to items on this list would affect the processing of commands and data regardless of where they come from. But if, for example, Line 1 were to select Window 2, changes in window list items resulting from data or commands received from Line 1 would be recorded only in Window List 2, and would therefore not affect the processing of data and commands coming from the keyboard (which is still using Window List 1).

## Scrolling Region (VT100)

To provide for software compatibility with the VT100, a scrolling region can be defined as a subset of a window. By default, the scrolling region is the same size as the window. If HDS Mode 206 is set, cursor addresses are interpreted as relative to the scrolling region rather than the window. This feature is less flexible than the Concept windowing capability, and is not recommended for use in applications designed for the Concept AVT.

### Selecting a Window

The Select Window command (Section 3.2) associates a window with a communication line or the keyboard so that subsequent data received from that device is stored in that window. The association of a line or device with a window can be changed at any time without affecting data already stored in the window. Whenever data is transmitted from display memory, it comes from the window currently associated with the requesting device (the line or device from which the Transmit or Print/Line 3 Control command was received).

For example, the command `ESC [ 2 ; 1 ! w` specifies that data coming from communication line 1 is to go into Window 2. Since all windows have the same default definition, this command would have no immediate effect on the size of the window. However, since the current cursor position is maintained separately for each window, the selection of Window 2 for Line 1 may affect where data from Line 1 appears on the screen. If Window 2 had not been used previously, its cursor would be positioned at line 1, column 1 (home), and subsequent data from Line 1 would begin appearing there, rather than where it had been appearing when the Select Window command was issued. However, the Window 2 cursor would not be visible, since only the cursor in the window associated with the keyboard is represented on the screen (by a flashing block or underline).

Note that selecting Window 2 for Line 1 does not also select Window 2 for the keyboard, even though Line 1 is the Keyboard Communication Line (Section 2.1.5). Keyboard input sent directly to the screen (as in Half Duplex or Block or Local Mode) would still go into Window 1, and would therefore appear in a different part of the screen from the data coming from Line 1. In Full Duplex, keyboard input is not sent directly to the screen, but appears only after being echoed by the host system, so it would appear along with the rest of the data sent by the host. However, the visible cursor would remain where it was when the Select Window command was issued.

If the device parameter is omitted from the Select Window command, the device from which the command is received (the requesting device) is assumed. Therefore the command `ESC [ 2 ! w` entered from the keyboard selects window 2 for the keyboard. Since 1 is the default value for window, `ESC [ ! w` would select Window 1 for the requesting device.

### Defining a Window

The Define Window command (Section 3.10) specifies the dimensions of the window currently in use by the communication line or device from which the Define Window command was received. Since the keyboard is currently using Window 1, the Define Window command

```
ESC [ 1 ; 24 ; 1 ; 40 w
```

defines window 1 to be the left half (columns 1 to 40 inclusive) of the first 24 lines of display memory. Column 40 is now treated as the right-most column in the display for data coming from the keyboard. The effect of this command can be observed by putting the terminal in Local Mode

and selecting Cursor/Character Wraparound (both of which can be done through Setup Mode). Now both cursor and character input wrap around at column 40, just as they formerly did at column 80. Return to Remote Mode before proceeding.

The use of windows to keep data separate can be demonstrated by defining one window for the keyboard, and another for Line 1. The previous example defined the keyboard's window as the left half of the screen. In order to define the right half of the screen as a separate window, the keyboard must select Window 2, and then define it:

```
ESC [ 2 ! w
ESC [ 1 ; 24 ; 41 ; 80 w
```

The second of these commands should cause the cursor to move to the "home" position of the newly defined window. Having defined Window 2, the keyboard should now return to window 1: ESC [ 1 ! w . To complete the example, select Window 2 for Line 1 (data coming from the host computer):

```
ESC [ 2 ; 1 ! w
```

Put the terminal in Half Duplex, even if the host computer echoes data from the terminal, and type input of any kind (such as simple operating system commands). Keyboard input should appear on the left side of the screen (Window 1) and the response from the host computer should appear on the right side (Window 2). If the host system echoes data from the terminal, keyboard input will appear twice: once in Window 1, directly from the keyboard, and once in Window 2, echoed back from the host computer.

### Some Applications

Multiple windows can be used in a wide variety of applications, whenever it would be advantageous to store data in functionally separate groupings in the terminal. The possibilities include, but are not limited to, "split screen" approaches, in which the display is divided into functionally distinct areas, all of which are visible at the same time. Thus, the top two lines of the screen might be a "command window", the next 20 divided vertically between an interactive area on the left, and a help window on the right, and the last two lines used to display error messages. Having defined the four windows as part of the initialization procedure, the application program would direct functionally different types of data into their respective windows by preceding each transmission of data to the terminal with a Select Window command.

In other applications, only one of the windows might be visible at a time. For example, a text editor might use a full screen window (24 lines by 80 columns) for the main work area, and define three other full screen windows to store displays, such as menus, that do not change in the course of the interaction. The application program would load the appropriate text into these non-changing windows just once as part of the initialization process. Then, when the user needed to see the information, the application program would simply issue a Select Window command to make the data visible to the user, rather than transmitting it all to the terminal each time it was

needed.

Multiple windows can also be used in conjunction with the terminal's powerful networking capabilities to provide for the control of multiple tasks from a single keyboard. This combination of features makes the terminal well adapted for use in multiple computer applications, as well as in environments where more than one communication line is available to a single host system. Section 2.8.3 contains a detailed example of a multiple computer application.

## 2.8 Peripherals and Networking

The Concept AVT has two (optionally 3) bi-directional communication ports, referred to as Line 1, Line 2, and Line 3. In the typical configuration, Line 1 is connected to the host computer and Line 2 is connected to a printer. The procedures for connecting the terminal to a host computer and a printer are described in Section 1.6. Line 3 might be connected to some other peripheral, such as a tape cassette or disk drive, or to another host computer. However, since each line provides an independent, full-purpose communications interface, the terminal can be used with any combination of three peripherals or host computers. Section 2.8.3 contains a detailed example of using the terminal in a multiple computer environment.

### 2.8.1 Printer Port (Line 2)

Because it is most often connected to a printer, Line 2 is configured somewhat differently from Lines 1 and 3 to make it better suited to functioning as a printer port. The differences are relatively minor, so it is not difficult to make Line 2 operate in the same way as Line 1 and Line 3. In addition to these slight differences in the communications interface, a number of terminal commands are specifically designed for use with a printer connected to Line 2.

As explained in Appendix D, Line 2 has a different set of pin assignments from Lines 1 and 3. In particular, note that pins 2 and 3 are reversed: on Line 2, pin 2 is Received Data and pin 3 is Transmitted data, whereas on Lines 1 and 3 pin 2 is Transmitted Data and Pin 3 is Received Data. This set of pin assignments makes the terminal compatible with most printers that have a serial RS232 interface. Consult the user manual supplied with your printer for information regarding the printer's communication interface.

If Line 2 is to be connected to a host computer, pins 2 and 3 will probably have to be reversed in the cable, as will pins 4 and 5. Consult the systems programmer or administrator of your host computer installation for details concerning the communications requirements of your host system.

The terminal supports both CTS/RTS (transmit) and XON/XOFF (receive) buffer overflow control protocols for all three lines. The three lines differ in the factory defaults for these features:

Line	CTS/RTS	XON/XOFF
----	-----	-----
1	off	on
2	on	off
3	on	off

These defaults are easily changed by commands explained in Section 3.9, and the desired settings can be stored in Non-Volatile Memory (NVM).

## Printer Commands

As explained below (Section 2.8.2), there are general purpose commands that control the transmission of data between different devices connected to the terminal. To facilitate use of the terminal in the standard configuration described above (host connected to Line 1, printer connected to Line 2), two special keys, PRINT and PRINT/SCRN, have been provided to simplify control of the printer from the keyboard. Both assume that the printer is connected to Line 2. All of the printer functions described in this section can also be invoked from the host using the Printer/Line 3 Control command (Section 3.8).

### The PRINT Key

Striking the PRINT key by itself "attaches" the printer, and striking the PRINT key while holding down the SHIFT key "detaches" the printer. When the printer is attached, all data that is sent to the terminal screen is also sent to the printer. (In this configuration, the printer is said to be "slaved" to the terminal.) If the terminal is in Full Duplex, only data coming from the host is sent to the printer; if the terminal is in Half Duplex, keyboard input is also sent to the printer.

If the keyboard bell rings when the PRINT key is struck, then the terminal was unable to attach the printer. The attempt to attach could have failed for a number of different reasons. First make sure that the printer cable is connected securely both to the terminal and to the printer. Then check that the pin assignments on the terminal match those on the printer. If the printer does not use CTS/RTS, this protocol must be turned off on the terminal.

### Escape Sequences

Terminal commands are not sent to the printer unless the terminal is in Transparent Mode (Section 2.3.2). Many printers use escape sequences to signal special functions (such as bold, underline, special fonts, etc.), in which case the stream of data sent to them includes these escape sequences. If the terminal is not in Transparent Mode, it may interpret some of these escape sequences as terminal commands (causing unpredictable results), and in any case will not pass them on to the printer. The following procedure could be used when sending data containing escape sequences to a printer through the terminal:

Step	From Keyboard	From Host
-----	-----	-----
1. Attach Printer	PRINT	ESC [ 5 i
2. Enter Transparent Mode	CMD Q	ESC Q
3. <Send data from host>		
4. Exit Transparent Mode	CMD R	ESC R
5. Detach Printer	SHIFT-PRINT	ESC [ 4 i

Note: This method will not work if the data sent to the terminal contains the sequence ESC R (since that would cause the terminal to exit from Transparent Mode). An alternative method would be to change the Escape Message Character (ESC) to something other than the default (ASCII 027). See Section 2.3.3 on Programmable Message Characters. Application developers should also consider using the generalized networking capabilities described below to send data directly from Line 1 to Line 2, bypassing the screen.

### The PRINT/SCRN Key

This key is used to print data currently in display memory, rather than data coming from the host computer. Striking the PRINT/SCRN key by itself prints data in the current window up to, but not including, the current cursor position. Striking the PRINT/SCRN key while holding down the SHIFT key prints all of the data in the current window, regardless of the cursor position. In both cases, the terminal attaches the printer, sends data to it, and detaches it.

Each line of data is followed by a carriage return and a line feed. Control codes are replaced by spaces. Protected characters are also replaced by spaces if HDS Mode 1 is Reset. If HDS Mode 116 is Set, Start of Print/Transmit can be used to mark a place other than the beginning of the window at which transmission of data is to begin. By default, a form feed (ASCII 012) is sent to the printer at the beginning of each PRINT/SCRN command. This can be avoided by Resetting Mode 117. See the Print/Line 3 Control command in Section 3.8 for additional details.

### 2.8.2 Networking

The printer functions described above take advantage of the terminal's generalized ability to manage the flow of data among all of the devices to which it is connected. In this discussion, each of the three communication lines is considered both an input and an output device, even if it is connected to a peripheral (such as a printer) that sends no data. The keyboard is considered an input device only, and the display an output device only. The terminal may have as many as four input devices (keyboard, Line 1, Line 2, and Line 3), and four output devices (display, Line 1, Line 2, and Line 3).

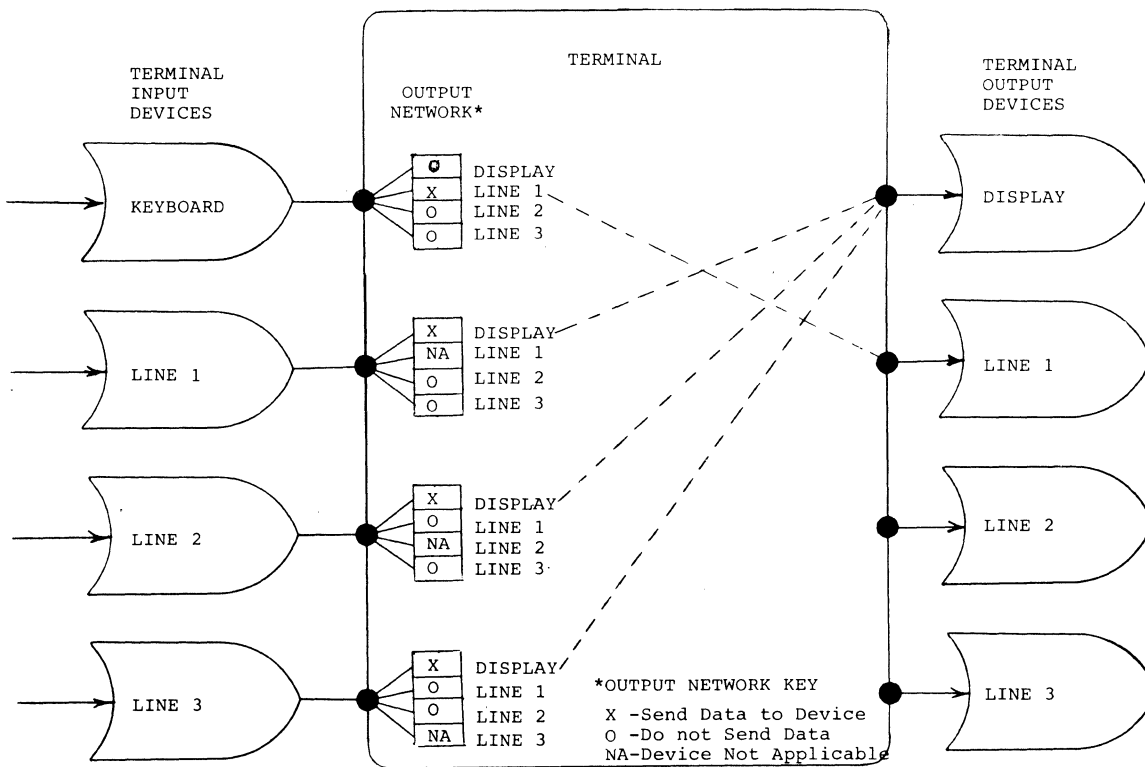
The basic mechanism for controlling the flow of data among devices is setting the "output network" of a device; that is, specifying where data coming from that device is to be sent. Each device's output network is shown near the right end of the Programmer Status Line as a series of numbers separated by semicolons, and followed by a 't'.

Note the following concerning networking:

1. Only characters are networked. Escape sequences that are terminal commands are executed by the terminal but not sent anywhere.

2. With the exception of the communication control codes NUL (ASCII 000), RUB (ASCII 127), ↑S (ASCII 019), ↑Q (ASCII 017) and the programmable delay character DLY (if defined), control codes are networked normally. If a control code is networked to the display, it is executed unless the terminal is in Transparent Mode, in which case it is displayed.
3. Invalid terminal commands (escape sequences that are not terminal commands) are networked from the point at which the error is detected.

Figure 2-3 shows the factory default networking. As the diagram indicates, data from each of the three communication lines is sent only to the screen, and characters typed on the keyboard are sent only to Line 1. The Set Output Network command, described below, can be used to establish any combination of output networks. A number of other terminal commands, provided for user convenience, also affect output networks. The default



**FIGURE 2-3**  
**Networking Example**  
**Default Terminal Configuration**

configuration depicted in Figure 2-3 represents normal Full Duplex operation. Putting the terminal in Half Duplex, either through Setup Mode or by entering the appropriate Reset Mode command, would add the display to the keyboard's output network. Entering Local Mode would remove Line 1 from the keyboard's output network. If a device had no devices in its output network, then data from that device would go nowhere: it would enter the terminal, but would not appear on the screen, and would not be sent to



any other communication line. Terminal commands coming from such a device would be processed normally.

### Set Output Network

This command specifies an input device and the list of output devices to which data from the input device is to go. The list of output devices replaces the existing list, rather than adding to it. For example,

```
ESC [ 9 : 9 t
```

would change the keyboard's output network so that it included only the screen. The Set Output Network command to produce the Half Duplex configuration from the default configuration would have to specify both Line 1 (1) and the screen (9) as the output network:

```
ESC [ 1 ; 9 : 9 t
```

(Notice that a colon is used to delimit the first parameter.)

Note: Standard terminal operating characteristics, such as Full and Half Duplex, Block and Character Mode, and Local and Remote should normally be selected either through Setup Mode or by using the appropriate Set and Reset Mode commands (Section 4.5). In particular, use of these commands should not be combined with use of the Set Output Network command, since the interaction might produce unexpected results.

See Section 3.8 for additional details regarding the Set Output Network command.

### 2.8.3 Multiple Computer Applications

The powerful networking capabilities described above enable the user to interact simultaneously with two (or three) host systems, and to transfer data between them through the terminal. In the following example, the terminal is connected to two different host systems, but the discussion can easily be applied to a three computer application. Any two host systems can be used, as long as the terminal is normally able to communicate with each of them individually.

#### Establishing Communications

Connect the communication cables between the two host systems (or communications equipment) so that Host 1 is connected to Line 1, and Host 2 is connected to Line 2. As noted above (Section 2.8.1), it will probably be necessary to modify the Line 2 interface by reversing pins 2 and 3 and pins 4 and 5. It may also be necessary to turn off CTS/RTS protocols for Line 2 ( ESC [ ; 2 \* x ). If Line 3 is used instead of Line 2, it will not be necessary to change the cabling, but it will probably still be necessary to turn off CTS/RTS for Line 3 ( ESC [ ; 3 \* x ).

Set the baud rate, stop bits, and parity for Lines 1 and 2 as required for Hosts 1 and 2 respectively. This can be done in Setup Mode: select the desired settings for Line 1; then use NEXT/LINE key (marked ↑ on top) to select the User Status Line for Line 2, and select the desired settings for Line 2. At this point, the terminal should be able to communicate with Host 1. Verify that it can by carrying out some simple interaction, such as logging in.

### Talking to the Other Computer

Normally, characters typed on the keyboard are sent to Host 1 because, by default, the keyboard's output network includes Line 1. In order to interact with Host 2, the keyboard's output network must be changed so that it includes Line 2, but not Line 1. The most convenient way to accomplish this is by changing the Keyboard Communication Line (KCL) to Line 2 (it is Line 1 by default). As explained in Section 2.1.5, the KCL is the line to which many commands related to the transmission of data apply. For example, the Set Keyboard Communication Line command

ESC [ 2 z

defines Line 2 as the KCL. In addition, it removes Line 1 from the keyboard's output network, and adds Line 2. As a result, all characters typed on the keyboard will be sent to Line 2 instead of Line 1. In addition, Transmit, Break, and commands setting Duplex, Stop Bits, Parity and Baud Rate will apply to Line 2 rather than Line 1.

The Set Keyboard Communication Line command is used throughout multiple computer applications to switch from addressing one host to addressing the other. ESC [ 1 z causes subsequent keyboard input to be sent to Line 1, and ESC [ 2 z causes subsequent keyboard input to be sent to Line 2. Programmable keys could be defined to execute these two commands (see below).

Note: Changing the Keyboard Communication Line has no effect on data coming into the terminal: data from all communication lines is normally displayed on the screen regardless of where keyboard input is currently being sent. Control of data coming from the two hosts is discussed below.

Verify that the terminal is able to communicate with Host 2 by logging in or carrying out some other simple interaction. If the terminal does not communicate correctly with Host 2, check that the pin assignments and communications parameters are compatible with the host communication interface.

### Block Transfer of Data Between Hosts

At this point the terminal is configured to communicate with two different host systems: the user can interact with one host system, then change the Keyboard Communication Line and interact with the other. This configuration enables the user to transfer information from one host to another by using the terminal's display memory as a temporary storage area for data from the sending host, which is then sent in blocks to the receiving host by means of the Transmit command (Section 2.9). Another method of transferring data between hosts, making use of the terminal's ability to network data directly from one communication line to another, is described below.

The transfer of data in blocks between two host systems can be controlled from the keyboard, or from either the sending or receiving host. The objective is to control the flow of data so that data is sent to the receiving device (whether the terminal or the receiving host) only when it is prepared to accept it. In the best case, where both host systems use XON/XOFF protocols on both input and output, this objective is relatively easy to achieve. Unfortunately it frequently turns out that one or both of the hosts does not use this protocol on input or output, so the user must take care to structure the interaction between the terminal and the two hosts so as to avoid loss of data.

In the absence of XON/XOFF protocols on the receiving host (Host 2), it is crucial to choose the size of the block of data to be transmitted on the basis of Host 2's input buffer, since this determines how much data Host 2 can accept (without loss) even if the user program for which the data is intended is not ready to use it (because it is in a pause or wait state, for example). If, for example, Host 2's input buffer is 80 characters, then data can be transferred from Host 1 to Host 2 in blocks no bigger than 80 characters. There must then be some way of sending data from Host 1 to the terminal in blocks of 80 characters or less, upon request. Normally the user would write a program that sent such a block of characters to the terminal in response to some prompt, though there may be standard software (such as a line-oriented text editor) that could be used in this way. On the receiving end, Host 2 would have a program that sent a "ready" message to the terminal to indicate that it was prepared to accept a block of data, and then waited for data from the terminal.

To control the transfer from the keyboard, the user would establish communications with both hosts, and invoke the programs to send and receive blocks of data on Hosts 1 and 2 respectively. If all goes well, the READY from Host 2 should appear on the screen. The sending program on Host 1 should be waiting for a command from the terminal. For each block of data to be transferred, the user would:

1. Wait for ready signal from Host 2
2. Home the cursor and erase the window (< ␣ > <SHIFT-ERAS>)
3. Change KCL to 1 ( ESC [ 1 z )
4. Prompt Host 1 to send a block of data to the terminal (<RETURN>)
5. Change KCL to 2 ( ESC [ 2 z )
6. Transmit data from screen (<SEND>)

This procedure assumes that the Transmit Extent is Window ( ESC [ \* y ), and that HDS Mode 16 is Set ( ESC [ = 12 h ), so that data is transmitted up to the current cursor position (rather than the end of the window). This is convenient because the cursor would be positioned correctly when the block of data from Host 1 is received by the terminal.

Note: As in any block transmission of data from the terminal, the host system receiving the transmission (Host 2 in this example) must not echo characters received from the terminal. See Section 2.9 on Block Mode. Users normally working in a Full Duplex environment generally find it best to turn off host echo (at least on the receiving host) and set the terminal to Half Duplex when transferring data between host systems.

#### Direct Transfer of Data Between Hosts

In the above example, data had to be stored in display memory before it could be transmitted to the receiving host (Host 2). The Set Output Network command can be used to establish a direct link between Line 1 and Line 2, so that data received from Line 1 is sent directly to Line 2. Normally, Line 1 has only the screen in its output network. To make data coming from Line 1 go to Line 2 as well as the screen (9), type:

```
ESC [ 2 ; 9 : 1 t
```

Similarly, Line 2 could be networked to Line 1 as well as the screen:

```
ESC [ 1 ; 9 : 2 t
```

With the output networks set in this way, each host system communicates with the other as if it were communicating with a terminal: the sending program operates as if it were sending data to a terminal, and the receiving program operates as if it were receiving data from a terminal. This simplifies the data transfer process enormously. For each block of data to be transferred, the sending program would simply wait for the ready signal from the terminal (actually Host 2), and then send a block of data to the terminal (and Host 2). When the sending program ran out of data to send, it could send some special character or sequence that would signal the receiving program to close the file and stop.

Because Line 1 and Line 2 both have the screen in their output networks, data coming from Line 1 appears on the screen as it is sent to Line 2, and data coming from Line 2 (the READY signal) appears on the screen as it is sent to Line 1. While this might be useful for monitoring the process, it is not necessary. For example, Line 2's output network could be set to include only Line 1 and not the screen:

ESC [ 1 : 2 t

with the result that the READY signal would be sent to Line 1 without appearing on the screen.

### Multiple Hosts and Multiple Windows

The use of multiple windows and programmable keys can greatly enhance the terminal interface in a multiple computer environment. As the terminal is configured at this point in the example, data coming to the terminal from the two host systems is displayed on the screen as it arrives. As a result, data from the two hosts is likely to be interspersed. Data from the two hosts can easily be kept visually separate by defining two different windows, and assigning Line 1 to one, and Line 2 to the other.

For example, Window 1 and Window 2 could be the top and bottom halves of the display, assigned to Line 1 and Line 2 respectively. The Define Window command (Section 2.7.4) applies to the window currently in use by the device from which the command is received (the requesting device). Since by default all devices use Window 1, the command

ESC [ 1 ; 12 w

defines Window 1 as the top half (lines 1 through 12) of the display. The left and right boundaries of the window default to the left and right boundaries of the display, so that the width of the window is either 80 or 132 columns, depending on the current display width (HDS Mode 103). To define Window 2, the keyboard must first select Window 2 and then issue a Define Window command:

ESC [ 2 ! w  
ESC [ 14 ; 24 w

Line 13 has not been included in either window so that it serves as a kind of visual buffer. Only the first 24 lines of display memory have been used in this example to ensure that both are completely visible at all times for demonstration purposes. Either or both could have been defined to be as large as available display memory permits.

Next, each communication line must be associated with a different window using the Select Window command:

ESC [ 1 ; 1 ! w      Select Window 1 for Line 1  
ESC [ 2 ; 2 ! w      Select Window 2 for Line 2

From now on, data coming from Host 1 will appear in the top half of the screen, and data coming from Host 2 will appear in the bottom. To ensure that keyboard input appears in the same window as the output from the host to which it is addressed, the keyboard's window must be changed whenever the KCL is changed. The terminal operator would then enter a pair of commands to redirect keyboard input from one host to the other:

```

ESC [ 1 z   ESC [ 1 ! w   Address Host 1
ESC [ 2 z   ESC [ 2 ! w   Address Host 2

```

While changing the keyboard's window is not required in Full Duplex just to keep the two interactions separate, it is still recommended, since the visible cursor (flashing block or underline) always appears in the window most recently selected by the keyboard, and the visible cursor determines what portion of the window is visible on the screen.

Finally, programmable keys can be used to facilitate redirecting keyboard input. Thus the following commands would program PF1 to address Host 1, and PF2 to address Host 2:

```

ESC [ 101 u / ESC [ 1 z ESC [ 1 ! w /
ESC [ 102 u / ESC [ 2 z ESC [ 2 ! w /

```

Programmable keys could also be used to change Line 1's output network so that data would go both to the screen and Line 2, or to the screen only, as required:

```

ESC [ 103 u / ESC [ 2 ; 9 : 1 t /   PF3: Host 1 to screen and host 2
ESC [ 104 u / ESC [ 9 : 1 t /       PF4: Host 1 to screen only

```

The entire procedure for setting up the multiple computer application just described could be stored as a file on one of the host computers and then transmitted to the terminal to program it. The file would contain the following:

```

ESC [ 1 ! w           Select Window 1
ESC [ 1 ; 12 w        Define Window 1
ESC [ 2 ! w           Select Window 2
ESC [ 14 ; 24 w       Define Window 2
ESC [ 1 ; 1 ! w       Select Window 1 for Line 1
ESC [ 2 ; 2 ! w       Select Window 2 for Line 2
ESC [ 101 u / ESC [ 1 z ESC [ 1 ! w /   PF1: Address Host 1
ESC [ 102 u / ESC [ 2 z ESC [ 2 ! w /   PF2: Address Host 2
ESC [ 103 u / ESC [ 2 ; 9 : 1 t /       PF3: Host 1 to screen, Host 2
ESC [ 104 u / ESC [ 9 : 1 t /           PF4: Host 1 to screen only

```

Note that the comments at the end of each line can be included. They do not interfere with processing of commands and they provide useful documentation.

## 2.9 Block Mode

Block Mode provides a combination of local editing capabilities and communication with the host computer that can be used in a variety of applications to enhance the quality of the user interface and reduce the volume of data transmission. When the terminal is in Block Mode, characters typed on the keyboard are displayed on the screen as they are typed. Once the data is on the screen, it can be edited and corrected using all of the local editing functions described in Section 2.5 (insert and delete character, insert and delete line, erase). However, none of the data that appears on the screen is transmitted to the host until a Transmit command is issued. Although characters typed on the main pad are not transmitted directly to the host, programmable keys set to operate in transmit mode are, thereby providing a communication link to the host.

By contrast, when the terminal is in Character Mode, each character typed on the keyboard is sent to the host as soon as it is typed. Also, keyboard input is sent directly to the screen only when in Half Duplex. In Full Duplex, characters are displayed on the screen only after being echoed back to the terminal by the host computer. Block Mode operation is not affected by the terminal's Duplex setting, though it generally requires that the host system not echo data back to the terminal (see below).

### 2.9.1 Interactive Uses

Although Block Mode is frequently used in data entry and retrieval applications in which the interaction is closely controlled by the program running on the host, it can also be used by the terminal operator to compensate for the poor quality of the user interface in software not designed to take full advantage of Concept terminal features. In APL, for example, the fact that overwriting a character causes it to be overstruck rather than replaced makes it difficult to correct typing errors, even on the same line. In Block Mode, however, the full range of terminal editing functions can be used to correct a line of input before it is transmitted to the host system. Block Mode could be used to similar advantage with line-oriented text editors.

Interactive users of Block Mode can use Setup Mode to switch between Character and Block Modes, depending on the task. Since it is one of the Alternate Processing Modes, it can also be selected by the appropriate Set Mode command: ESC [ = 110 h . Selecting Block Mode has no effect on whether the terminal is in Local or Remote Mode, but the terminal must be in Remote Mode for any data to be sent to the host under any circumstances.

## Host System Echo

It is also important that the host system not echo data received from the terminal. If the terminal is normally set to Half Duplex (when in Character Mode), then there should be no problem in switching to Block Mode, since Half Duplex is used when the host does not provide echo. If, however, the terminal is normally set to Full Duplex, the user must somehow signal the host system to stop echoing data received from the terminal. If host echo is not disabled, data being echoed back from the host may interfere with the transmission of data from the screen.

Note: Host echo cannot be disabled by a terminal command. Putting the terminal in Half Duplex does not disable host echo: it simply configures the terminal to compensate for the absence of host echo. Most operating systems allow the terminal operator to enable or disable host echo for his or her line. If it is impossible or inconvenient to disable host echo, the Set Output Network command can be used to prevent the echoed characters from reaching the screen during transmissions of data to the host (see Section 2.8.2). Contact the systems programmer or administrator of your computer installation for information on disabling host system echo.

Where it is possible to use either Full or Half Duplex, a user who alternated between Block and Character Modes could simplify the transition by setting the terminal to Half Duplex, and keeping host echo off.

## Sending Data to the Host

The Transmit Command, ESC 5 , causes data currently in display memory to be sent to the host computer. A number of terminal settings, described below, combine to determine exactly which of the characters on the screen are transmitted. If the programmable Start of Message character (SOM) has been defined, it is transmitted at the beginning of the set of characters being transmitted from the screen. By default, no SOM is used. Each line of characters transmitted from the screen, except for the last, is followed by the character or pair of characters currently defined as the End of Line character (EOL). The entire transmission is terminated by the programmable End of Message character (EOM). Both EOL and EOM default to carriage return (ASCII 013), so that a line of characters transmitted in Block Mode looks to the host exactly like the same line of characters followed by a carriage return.

Interactive users would issue this command from the keyboard, typically by striking programmable key F5 (labelled SEND on the front), which executes the Transmit command as its power-up default (see Section 2.4.1). Of course, any programmable key could be programmed to execute this command. Like all terminal commands, Transmit can also be sent from the host (see below).

The amount of data sent by a Transmit command is determined primarily by three factors: Transmit Extent, Transmit Termination and Transmit Initiation. Transmit Extent determines whether the scope of the transmission is the Window (0), Line (1), or Field (3) in which the cursor



is currently located. Transmit Termination, HDS Mode 16, specifies whether the transmission includes all of the data in the Window, Line, or Field (Reset), or only the data from the beginning of the Window, Line or Field up to (but not including) the current cursor location (Set).

Transmit Initiation, HDS Mode 116, applies only when the Transmit Extent is Window. It determines whether the transmission starts at the beginning of the window (Reset) or at the Start of Print/Transmit (Set). The Start of Print/Transmit is a marker that is set by positioning the cursor to the location of the first character to be transmitted, and then issuing the Start of Print/Transmit command, ESC ? . By default, this marker is set to the home position (line 1, column 1). Once defined, it retains its value until redefined by another Start of Print/Transmit or reset to its default value by a Define Window command. Start of Print/Transmit is maintained independently for each Window (see Section 2.7.4).

For example, the APL user imagined above might find it easier to manage the interaction sending data one line at a time, and would therefore enter the command ESC [ 1 \* y to set the Transmit Extent to Line. With HDS Mode 16 (Transmit Termination) in its Reset state, the entire line will be transmitted, regardless of where the cursor is when the Transmit command is issued. This is convenient, provided that the user remembers to keep unused parts of the line clear.

With the terminal configured in this way, the SEND key (programmable key F5) would be used in much the same way as the RETURN key in typical character mode interactions. Data and commands are sent to the host by typing them on the keyboard and then striking the SEND key. Thus typing

```
)LOAD EXAMPLE-1<SEND>
```

causes the character sequence )LOAD EXAMPLE-1<CR> to be transmitted to the host, exactly as would have been transmitted by typing this sequence in Character Mode. The <CR> appended by the Transmit command is the default value for the programmable End of Message character (EOM). As explained in Section 2.3.3, the user can specify some other character, pair of characters, or no character at all to be used as the EOM.

A more sophisticated user interface can be created by using multiple windows (Section 2.7.4) in conjunction with Block Mode. Define the first line of the screen as a command window, and the remainder of display memory (or the remaining 23 lines on the screen) as the editing window. Then set the Transmit Extent to Window. This provides the advantage of multiple line transmissions of text or data, and single line transmission of commands.

## 2.9.2 Block Mode Applications

Block Mode can be used to greatly increase the efficiency of applications such as data entry and retrieval and full-screen text editing. By making use of the terminal's local editing capabilities, block mode application programs offload some of the processing burden from the host to the terminal, while at the same time reducing the volume of data transmitted. Since keyboard input is transmitted only in response to the Transmit command, data is only transmitted when the terminal operator is satisfied that it is correct. Naturally, the gain in efficiency is greatest when the host system is heavily loaded, or data is transmitted between host and terminal at a relatively slow speed.

### Transmission Under Host Control

Application programs designed to use Block Mode generally do not allow the terminal operator to actually execute the Transmit command directly from the keyboard. Instead, when the data on the screen is ready to be transmitted to the host, the terminal operator strikes a transmit mode function key to signal the host, and the host issues the Transmit command. This has the advantage that the Transmit command is only issued when the application program and the host operating system are ready to receive data, thereby reducing the risk that data will be lost in transmission. Also, it allows the program to issue multiple Transmit commands to obtain a large amount of data in convenient installments.

In any case, the difference between executing the Transmit command from the keyboard and sending it from the host would be transparent to the terminal operator. Though any of the programmable keys could serve this purpose, it might be convenient to use F5 (labelled SEND on the front) in transmit mode to signal that data is ready for transmission. Since F5 powers up programmed to execute the Transmit command, the application would have to use the Program Function Key command (Section 2.4.2) to change it to transmit mode as part of the initialization procedure. The application designer could decide whether to use the key's default transmit sequence or some other sequence as the signal to the host to issue the Transmit command. A programmable key such as F7, which powers up in transmit mode, could be used without being specially programmed, provided that the application program was designed to recognize its default transmit sequence (which is `↑\ 007 <CR>` in the case of F7) as the signal to issue the Transmit command.

The important point is that the only way to send a signal to the host (while the terminal is in Block Mode) is by striking a transmit mode function key. Transmit mode function keys would therefore also be used to send other signals to the host, including requests to terminate processing, or proceed to a different point in the interaction, or abort a data entry or retrieval procedure, for example.

A block mode data entry application might go through the following steps to get data from the terminal operator:

1. Send prompt to terminal
2. Get response from terminal
3. If response is not transmit signal, go to ...
4. Send Transmit command ( ESC 5 ) to terminal
5. Get transmitted data from terminal

All of the keyboard input occurs between steps 1 and 2, including the use of local editing capabilities to make sure that the information is correct before the transmit signal is given. Normally the terminal operator is not aware of steps 3 - 5.

#### Transmit Delay

In some Half Duplex environments, the host system requires a certain amount of time to "turn around" between sending and receiving data. In the above example, this would mean that a certain amount of time must elapse between steps 1 and 2, and between steps 4 and 5. Since keyboard input follows 1, there is almost certainly going to be adequate turnaround time between 1 and 2. However, since the terminal responds to the Transmit command as soon as it is received, data transmitted by the terminal might reach the host system before the latter has turned around to receive input, and would therefore be lost. This problem can be avoided by using the Set Transmit Delay command, ESC [ dly \* z , which specifies the amount of time the terminal is to wait between receiving a Transmit command and executing it. As explained in Section 3.9, dly is the delay in 100 millisecond units. Consult your computer installation for information on host system communications.

## 2.10 Forms Handling

The Concept AVT has a wide range of features that support efficient handling of screen forms such as are often used in data entry and retrieval applications. Though the features described below are frequently used in combination, they are independent of one another, and can be used individually or with most other terminal features. Similarly, although many applications that use the features described below operate in Block Mode (Section 2.9), these features are available in Character Mode as well.

### 2.10.1 Fields and Protection

A screen form is often thought of as containing a number of "fields", each of which contains a single piece or kind of information. These fields are generally positioned on the screen in such a way as to aid visual differentiation of information. Various kinds of highlighting, such as reverse video, bold, underlining, and blinking, are used to promote identification of different kinds of information, call attention to special conditions, and reduce the likelihood of data entry errors.

A protected field is one in which data cannot be changed in any way (overwritten, erased, deleted). Protected fields are typically used in forms applications for constant data, such as headings and labels, that are not supposed to be altered by the terminal operator. Unprotected fields are set up to receive the terminal operator's input, and are not restricted in any way. It is possible to configure the terminal so that data coming from the host can overwrite fields that cannot be overwritten by the terminal operator (see Section 2.10.2 below).

As explained in Section 2.6.2, protection is implemented as a character attribute. As a result, a "protected field" is actually just a contiguous string of characters each of which has its protection attribute set on. Similarly, an unprotected field is just a contiguous string of unprotected characters. A protected field is delimited by either an unprotected character or the beginning or end of the window. An unprotected field is delimited by either a protected character or the boundary of the window.

**Note:** The end of a line within a window does not delimit a field. A field can "wrap around" the end of a line.

A protected field is created by setting protection on before writing the first character of the field, and setting it off again after writing the last character of the protected field. Protection can be set on either by using the appropriate Select Alternate Attribute command (Section 2.6.2) or by using the special Protection On command ( ESC V ). Use the appropriate Select Default Attribute command or the special Protection Off command ( ESC W ) to turn protection off.

**Note:** "Field" has no meaning apart from the transition between protected and unprotected characters. While many terminal functions (such as editing and transmission) can be made to operate on individual fields, the only way to specify a field is through the use of at least one protected character. See Section 3.6 on editing functions

and Section 3.9 on transmission.

#### 2.10.2 Protected Field Overwrite (HDS Mode 108)

Attempts to replace protected characters normally fail -- no change is made in the contents of display memory, and the cursor does not move. If the attempt was made from the keyboard, the bell sounds. This effect of protection can be overridden by setting HDS Mode 108 ( ESC [ = 108 h ).

Though the primary purpose of protection is to prevent keyboard input from modifying data on the screen, it is often convenient to allow protected fields to be overwritten by the application program. This would permit the application program to change the text in a protected field without having to first use Block Attribute Change to unprotect it. Since the status of HDS Mode 108 is stored as part of the attribute list (Section 2.6.3), the same field can be protected from keyboard input, but not protected from data sent by the host, provided that the keyboard and the host communication line use different attribute lists, and the keyboard's attribute list has HDS Mode 108 Reset, and Line 1's attribute list has HDS Mode 108 Set.

#### 2.10.3 Protected Field Erasure (HDS Mode 6)

To maintain VT100 compatibility, erasure of protected fields is allowed when the terminal is in factory default state. In many forms applications, it is more convenient to have Erase commands affect only unprotected characters. This can be achieved by Resetting HDS Mode 6 ( ESC [ = 6 1 ). With HDS Mode 6 Reset, homing the cursor ( ESC [ H ) and erasing the entire window ( ESC [ J ) has the effect of erasing all of the unprotected fields in the window.

As in the previous paragraph, it may be desirable to allow the application program to erase protected characters, while preventing the terminal operator from doing so. This could be achieved in exactly the same way, by simply adding HDS Mode 6 to the differences between Line 1's attribute list and the keyboard's attribute list.

#### 2.10.4 Scrolling (HDS Mode 104)

Normally, a line feed (ASCII 010) encountered when the cursor is on the last line of the window causes the data in the window to "scroll up", so that each line in the window moves up one, the top line disappears, and a blank line is created at the bottom of the screen. In forms applications this is generally not desirable, so scrolling is turned off by Setting HDS Mode 104 ( ESC [ = 104 h ). With this mode set, a line feed at the bottom of the window has no effect.

### 2.10.5 Tab Processing (HDS Mode 105)

The TAB key is generally used in forms applications to move the cursor to the beginning of the next unprotected field, rather than to the next text tab stop. To select this mode of operation for the TAB key (and for the Tab, Forward Tab, and Backward Tab commands), enter the Set Mode command for HDS Mode 105 ( ESC [ = 105 h ).

### 2.10.6 Auto Tabs (HDS Mode 106)

When this mode is set, the cursor automatically moves to the beginning of the next unprotected field as soon as a character is typed into the last position of the current (unprotected) field. Striking a cursor movement key in the last position of an unprotected field does not have this effect. This feature streamlines the user interface by making it unnecessary for the terminal operator to strike the TAB key to get to the next field. It similarly eliminates the need for Tab commands in data being sent from the host to fill unprotected fields. For this feature to work properly, the entire screen must be protected, except for the areas in which data is to be entered (see Block Attribute Change below). Also if the host line is using Auto Tabs, it must not suppress trailing blanks when sending data to the terminal.

### 2.10.7 Line Drawing

The Block Character Change command can be used to draw lines for borders, and fill rectangular areas with arbitrary characters. This command specifies a character (by its ASCII chart location) and the number of columns and lines to fill with that character, starting at the current cursor position. For example, the sequence of commands:

```
ESC ) 0          Define Alternate Character Set as VT100 and
                  Concept Special Graphics
↑N              Use Alternate Character Set
ESC | H         Home the cursor
ESC [ 11 ; 80 ; 2 p Draw two lines of horizontal bars (ASCII 011)
                  across the screen
ESC [ 0 ; 1 ; 24 p Draw a vertical bar (ASCII 000) to the bottom
                  of the screen
ESC [ 80 G      Position the cursor to column 80
ESC [ 0 ; 1 ; 24 p Draw a vertical bar (ASCII 000) to the bottom
                  of the screen
↑0              Use Normal Character Set
```

uses the VT100 and Concept Special Graphics character set (0) to draw a double border at the top, and single borders down the sides of the screen. Since the Block Character Change commands do not change the cursor position, the cursor would be left at the right end of the top line of the screen.

### 2.10.8 Block Attribute Change

Another terminal feature that is useful in creating screen forms is the ability to set the character attributes of a block of display memory by a single command. The Block Attribute Change command is analogous to the Block Character Change command in the way it specifies the rectangular area affected, except that the number of lines affected is given first. The way in which the attributes of that area are to be changed is specified by two lists: the first list contains the attributes that are used by the command; the second list indicates which of the attributes on the first list are to be set to their alternate states. Attributes on the first list but not on the second are set to their default states; attributes on the second list but not on the first are ignored.

For example, an application using Auto Tabs might begin creating a form by protecting the entire screen:

```
ESC [ H           Home cursor
ESC [ = 99 : 99 : 24 ; 80 ! q   Block Attribute Change
```

This command would leave all other attributes as they were. If it is necessary to set all attributes other than protection to their default states, this could be accomplished by the following command:

```
ESC [ = 1 ; 4 ; 5 ; 7 ; 8 ; 99 : 99 : 24 ; 80 ! q
```

The Block Attribute Change command could also be used to change the way a field is highlighted to indicate an error or other special condition. For example, a data entry application might create a screen form containing an 11 character unprotected field for social security number starting in column 48 of line 3. If an error were detected in the data typed into this field, the application could return the entire form to the terminal and highlight the invalid data by setting the attributes of just that field to blinking inverse video. After transmitting all of the data in the unprotected fields back to the terminal, the application program would position the cursor at the beginning of the field to be highlighted

```
ESC [ 3 ; 48 H
```

and then use the Block Attribute Change to select blinking (5) and inverse video (7) for the next 11 characters

```
ESC [ 5 ; 7 : 5 ; 7 : 1 ; 11 ! q
```





## Command Reference

### 3.1 Introduction

This chapter contains a detailed explanation of all Concept AVT terminal commands. Information on each command is presented in such a way as to highlight the most frequently needed material. Commands are grouped by terminal function (as they are on the Concept AVT Reference Card, DN 1301-8211-1). The entry for each command includes one or more of the following:

#### 3.1.1 Format of Command Entries

1. **Command Name**

2. **Command Sequence** - according to the following conventions:

- a. ESC stands for the escape message character (default is ^[, ASCII 027). Use CMD key to enter from keyboard. Use shifted CMD to enter ESC [ from the keyboard.
- b. Command sequences may contain no embedded spaces; blank space in command sequence representations is for clarity of graphic presentation only. Any spaces that should occur in terminal commands (or responses) are indicated by 'Sp'.
- c. Numbers and single characters are literals, and should be entered exactly as shown.
- d. Each string of three or more lower case alphabetic characters represents a parameter.

3. **Special Key**

If there is a special key that generates the command, the key top label is shown in braces {} following the command sequence. If the special key generates the command by itself, its label appears in lower case within the braces. If it generates the command only when used in conjunction with the shift key, the key top label appears in upper case. If the sequence is generated whether shifted or not, the key top label appears in lower case.

4. **Description of Command Function**

5. **Parameters**

The meaning of each recognized value, the default value assumed if the parameter is omitted, and the action taken if the value is not recognized are given for each parameter used in a command sequence. (See below for a detailed discussion of parameters.)

## 6. Interactions with Alternate Processing Modes

A number of terminal operating characteristics can be selected through the Set and Reset Mode commands. The behavior of many terminal commands is affected by the settings of these modes, and the interaction is described as part of the explanation of the command. Modes are identified by their HDS Mode number. A complete listing of Alternate Processing Modes, in HDS Mode number order, is given in Chapter 4.

## 7. Notes

Special considerations regarding the use and effects of the command, as well as possibly unexpected interactions with other commands.

## 8. Examples

Illustrations of typical or particularly interesting uses of the command. Once again, blank space in example command sequences is for graphic clarity only; actual spaces (i.e. ASCII 032) are represented by 'Sp'.

### 3.1.2 Types of Commands

The Concept AVT processes characters in accordance with American National Standards Institute (ANSI) standard X3.64-1979.

Terminal commands are either control codes or command sequences. Most commands can either be entered from the keyboard or sent to the terminal by a program running on the host computer, and the actual operation of the command often depends on its source in ways that are explained individually for each command. Control codes are entered from the keyboard by pressing the control key (CTRL) while simultaneously pressing another key. Command sequences are entered from the keyboard by striking the CMD key in the Cursor Pad (not the ESC key in the Main Pad) and then typing the remainder of the command sequence.

### 3.1.3 Command Sequences

Command sequences (not including control codes) can be divided into two groups: those beginning with ESC [ , the Command Sequence Introducer, and those beginning with just ESC. The structure of commands beginning with the command sequence introducer is discussed below. Commands beginning with just ESC are generally simpler, usually consisting of ESC followed by a single character (the only exceptions to this are the Define Alternate and Define Normal Character Set commands -- section 3.3).

## Note:

1. The key labelled ESC on the keyboard is NOT equivalent to the CMD key. The escape message character is only used from communication lines.
2. For convenience, typing shifted CMD is equivalent to typing unshifted CMD followed by [ (i.e., the way to produce the sequence ESC [ from the keyboard).
3. To abort a command sequence, type CTRL-X.

**3.1.4 Command Sequence Introducer      ESC [**

This two character sequence indicates the start of a control sequence. It is generated from the keyboard either by striking the two key sequence CMD [, or by striking the CMD while holding down the SHIFT key. Terminal commands that begin this way have the following ANSI defined structure:

**ESC [ d Pn ; Pn ... I F**

made up of the following parts:

ESC [    is the Command Sequence Introducer (CSI).

**d**    is an optional private parameter indicator. If it occurs in a command sequence, it indicates that the parameters following it do not have ANSI defined standard values. In the current implementation, the equal sign (=) indicates HDS private parameters, and the question mark (?) indicates DEC private parameters.

**Pn**    is a parameter consisting of one or more numeric values. Each parameter value is a string of digits (ASCII 048-057) representing a value 0 through 255. Leading zeros are ignored, and values greater than 255 are interpreted as 255. Each parameter is terminated by by one of the following: semi-colon (ASCII 059), colon (ASCII 058), the Intermediate Character, or the Final Character (see below).

**I**    is an Intermediate Character. If it occurs in a command, it is used in conjunction with the Final Character to specify the command. The intermediates used in the AVT are ! (ASCII 033) and \* (ASCII 042).

**F**    is the Final Character, used in conjunction with the Intermediate Character, if any, to specify the command.

### 3.1.5 Parameters

Parameters are optional. For each command, a default value is defined for each parameter. A parameter is assumed to have its default value if it is either omitted, or entered as zero (with the exception of the parameter `chr` in Change Message Character, Section 3.2). In general, default values have been chosen so that parameters can frequently be omitted (see the examples below).

Parameters are separated by semi-colons. If there is more than one parameter in a command, and one or more of the parameters may have multiple values, then the two parameter strings must be separated by a colon.

If more parameters are entered than the command calls for, excess parameters are ignored. If more than 15 parameters are entered for a command, those that are entered after the 15th one are ignored.

The following examples of the Position Cursor command (whose format is `ESC [ lin ; col H`) all have the same effect: they all position the cursor at line 1, column 10 of the current window:

<code>ESC [ 1 ; 10 H</code>	<code>lin</code> specified as 1; <code>col</code> specified as 10.
<code>ESC [ ; 10 H</code>	The semicolon indicates that <code>lin</code> has been omitted (and hence gets its default value of 1).
<code>ESC [ 0 ; 10 H</code>	As in most cases, 0 indicates a default parameter.
<code>ESC [ 00001 ; 0010 H</code>	Leading zeros are ignored.

### 3.1.6 Error Conditions

A terminal command will fail, i.e. be cancelled, ignoring the command sequence prior to the error condition, if any of the following is encountered:

1. A private parameter indicator (ASCII 060-063) anywhere in the command stream except immediately following the Command Sequence Introducer. (`ESC [`).
2. An unrecognized value for one or more parameters.
3. `↑X` or `↑Z`.

Note: Control codes other than `↑X` and `↑Z` are ignored when encountered within parameter strings.

For example, the following sequences fail (i.e., have no effect):

ESC [ 7 = 9 m	The private parameter indicator = does not immediately follow the ESC [.
ESC [ 5 ; 2 ! w	5 is not a recognized value for the first parameter, which, in this case, must be the number of a window (1 - 4).

### 3.2 General

#### ● Self Test      ESC [ test y

Performs internal tests, and displays the result on the Alert Message Status Line.

Parameter test specifies the test(s) to be performed. More than one test may be specified in the same command; values not listed below are ignored; 0 is the default.

0 ROM/RAM/NVM test. Display either NO ERROR or one or more of the following error codes on the Alert Message Status Line:

- 1-5 ROM (firmware) chip number in which error was found.
- 6 RAM (Display Memory) error.
- 7 NVM (Non-Volatile Memory) error.

1 Communications test. Requires a loopback connector (anything that connects pins 2 and 3, receive and transmit, together) on the keyboard communication line. The command must be received from the communication line, not the keyboard (that is, type 'ESC', not 'CMD'). If the request is received directly from the keyboard, it is ignored.

Each character is displayed on the screen twice, once as it is sent out and once as it is received. NO ERROR is displayed if the test is successful; otherwise error code 8 is displayed.

2 Character/attribute display. All character sets are displayed, using different combinations of display attributes (i.e., blinking, inverse video, half-bright, etc.), and NO ERROR is displayed on the Alert Message Status Line.

3 NVM display. The contents of Non-Volatile Memory are displayed on the screen, and NO ERROR is displayed on the Alert Message Status Line.

9 Repeat test(s) until error is found. Test(s) specified by other parameters are repeated until the terminal is reset or an error condition is encountered.

Example: To test communications on the keyboard's communication line install a 2-3 jumper on the keyboard communication line, and type ESC [ 1 y using the ESC key, not the CMD key.

#### ● Reset Terminal      ESC c      {RESET}

Resets the terminal to the configuration stored in NVM (the "power-up defaults"). This has the same effect as turning the terminal off and then on again. All terminal characteristics are changed (including window definitions, attribute lists, and modes). Both display memory and communication buffers are cleared, program-

mable function keys are reprogrammed to their factory default settings, and the Latent Expression is invoked.

Notes:

1. Since all communication buffers are cleared, some kind of delay mechanism (such as fill characters) should be used to ensure that the terminal has sufficient time to complete execution of the command before further data or commands are sent to it.
2. Since the Latent Expression is invoked, inclusion of this command (or any command that implicitly resets the terminal) in the Latent Expression will result in an endless loop that cannot be broken without physically resetting non-volatile memory.

● **Save Configuration in NVM**      **ESC [ save ~**

Saves terminal configuration (i.e. operating characteristics) in Non-Volatile Memory.

Parameter save specifies what is to be saved:

- 0 Save all aspects of terminal's current configuration, except contents of programmable function keys.
- 9 Restore all operating characteristics to factory default state, and reset terminal. See Reset Terminal command.

● **Change Message Character**      **ESC [ msgchr;chr;chr ! t**

Changes the ASCII character(s) used for a specified Message Character.

Parameters:

msgchr specifies the Message Character to change:

msgchr	REF	DEFAULTS		MEANING	MAX CHARS.
		CDE	SYM/LOC		
0	ESC	↑[	ESC 027	Escape	1
1	SOM	Not used	***	Start of Message	1
2	FKD	↑\	FS 028	Function Key Identifier	1
3	CPM	Not used	***	Cursor Pad End of Message	1
4	DLY	Not used	***	Delay Character	1
5	EOF	↑W	ETB 023	End of Field	2
6	EOL	↑M	CR 013	End of Line	2
7	EOM	↑M	CR 013	End of Message	2

Other values for msgchr cause the command to be ignored.

chr specifies the ASCII chart location of the character to use.

Allowable values are 000 - 127. Other values are ignored

(i.e., if the first chr is valid and the second is not, the Message character specified by msgchr is changed to the first chr only). If the first occurrence of chr is omitted, the Message Character specified by msgchr is not used.

NOTE: ESC cannot be disimplemented (i.e. changed to no character at all).

Examples:

ESC [ 5 ! t            Causes the EOF character to not be used

ESC [ 7;13;10 ! t      Causes the pair of characters CR/LF to be sent at the end of a transmission (EOM).

● **Fill Character**        ↑@  
**(alternate)**            RUB

Ignored by the terminal except in Transparent Mode (HDS Mode 3 set), when they are displayed. Any number of fill characters may be interspersed anywhere within data or terminal commands for timing purposes. The Fill Character should not be confused with the programmable message character DLY (Section 3.2).

● **Select Attribute List**      ESC [ alist;dev ! u

Causes a specified set (list) of display attributes to be applied to characters received from a specified device (or communication line).

Parameters:

alist specifies the attribute list to use. Recognized values are 1-4; the default is 1; other values cause the command to be ignored.

dev specifies the device or communication line for which the attribute list is being selected (default is 0):

0	Requesting device.	3	Line 3.
1	Line 1.	9	Keyboard.
2	Line 2.		

Other values for either parameter cause the command to be ignored.

● **Copy Attribute List**        ESC [ alist;dev ! v

Makes the attribute list used by device dev equal to the attribute list alist. Changes attributes applied to data coming from the specified device, as well as devices using the same attribute list as the specified device.

Parameters:



`alist` specifies the attribute list to copy. Allowable values are 1 - 4; the default is 1; other values cause the command to be ignored.

`dev` specifies the device whose attribute list is to be copied from the attribute list specified (default is 0):

0	Requesting device.	3	Line 3.
1	Line 1	9	Keyboard.
2	Line 2		

Other values cause the command to be ignored.

### ● **Select Window**      **ESC [ win;dev ! w**

Causes window `win` to be used for device `dev`. Data received from device `dev` is stored in `win`; data to be transmitted to `dev` comes from `win`. Selection of a different window may change the cursor position, window definition, scrolling region, and Start of Print/Transmit. See Section 2.1 and the Define Window command in Section 3.10.

#### Parameters:

`win` specifies the window to be used. Allowable values are 1 - 4; the default is 1; other values cause the command to be ignored.

`dev` specifies the device for which the window is to be used:

0	Requesting device.	3	Line 3.
1	Line 1	9	Keyboard.
2	Line 2.		

Other values cause the command to be ignored.

**Note:** Since all windows are by default defined as a single page of display memory, a window may be selected even though it has not explicitly been defined.

### 3.3 APL/ASCII (Character Sets)

#### ● Define Normal Character Set      ESC ( cset

Defines the character set to be selected by the Use Normal Character Set command.

Parameter cset specifies the character set:

- B,A First character set (ASCII).
- 0 Second character set (VT100 and Concept Special Graphics).
- 1 Third character set (HDS Block Graphics).
- 2 Fourth character set (APL).

Other values cause the command to be ignored.

Note: If the character set specified has not been installed on the terminal, attempts to use it will result in no characters being displayed on the screen, even though data is sent and received normally by the terminal.

#### ● Use Normal Character Set      ↑0

Selects the character set currently defined as the Normal Character Set. See Define Normal Character Set command. The factory default is 'B' (ASCII).

Note: If the Normal Character Set is currently defined as 'B' (ASCII), Character Overstrike (HDS Mode 102) is turned off.

#### ● Define Alternate Character Set      ESC ) cset

Defines the character set to be selected by the Use Alternate Character Set command.

Parameter cset specifies the character set. See Define Normal Character Set for recognized values.

#### ● Use Alternate Character Set      ↑N

Selects the character set currently defined as the Alternate Character Set. See Define Alternate Character Set. The factory default is 'B' (ASCII).

Note: If the Alternate Character Set is currently defined as '2' (APL), Character Overstrike (HDS Mode 102) is turned on.

### 3.4 Status Information

#### ● **Transmit Terminal Identifier**      **ESC [ 0 c**

Causes the terminal to send out an identifying sequence to the requesting device (or to the keyboard communication line, if the command comes from the keyboard). The terminal's response depends upon whether the Terminal Identifier (HDS Mode 123) is VT100 (reset state) or AVT (set state), and upon the amount of display memory the terminal has.

Responses:

```
ESC [ ? 1;2 c   VT100.
ESC [ = 1;1 c   Concept AVT with four pages of display memory.
ESC [ = 1;2 c   Concept AVT with eight pages of display memory.
```

Note: The command will be ignored if the terminal is in Local Mode (HDS Mode 111 set).

#### ● **Transmit Device Status**      **ESC [ 5 n**

Transmits the terminal's status to the requesting device (or to the keyboard communication device, if the command comes from the keyboard).

Responses:

```
ESC [ 0 n   Ready, no malfunctions detected
ESC [ 3 n   Malfunctions; retry
```

Note: The command is ignored if the terminal is in Local Mode (HDS Mode 111 set).

#### ● **Display Status Line**      **ESC [ devst;sline \* u**

Displays the status line specified by the parameters devst and sline.

Parameters:

devst specifies the device for which a status line is to be displayed (0, the default, means scroll status line).

```
1   Line 1.           3   Line 3.
2   Line 2.           9   Keyboard.
```

sline specifies the type of status line to display:

```
0,1  User (default).  4   Tabs.
2   Programmer       5   Message Characters.
3   Modes            6   Alert Message Line.
```

Invalid values for either parameter cause the command to be ignored.

Notes: Command is ignored if the terminal is in Setup Mode or a status line transmission is in progress (See Transmit Status Line command.)

● **Next Status Line**      **ESC [ \* u**      **{STATUS}**

Scrolls the existing status line to the next one, or displays the Keyboard User Status Line, if none was previously displayed.

The order in which status lines appear when scrolled:

1. Keyboard - User	9. Keyboard - Modes
2. Line 1      "	10. Line 1      "
3. Line 2      "	11. Line 2      "
4. Line 3      "	12. Line 3      "
5. Keyboard - Programmer	13. Tabs
6. Line 1      "	14. Message Characters
7. Line 2      "	15. Alert Message
8. Line 3      "	

Notes: Command is ignored if the terminal is in Setup Mode or a status line transmission is in progress (See Transmit Status Line command.)

● **Transmit Status Line**      **ESC [ dev;sline;beg;end \* t**

Transmits all or part of a status line to the requesting device (or to the keyboard communication line, if the command comes from the keyboard). Status line information is transmitted to the requesting line preceded by Start of Message (SOM) and followed by End of Message (EOM), if defined.

Parameters:

devst specifies the device whose status line is to be transmitted (default is 0):

0	Requesting device	2	Line 2.
1	Line 1.	3	Line 3.

sline specifies the kind of status line to transmit (default is 0):

0,1	User	4	Tabs
2	Programmer	5	Message Character
3	Modes	6	Alert Message

Other values for either parameter cause the command to be ignored.

beg Status line column position at which to begin transmission.  
 end Status line column position at which to end transmission.

If either beg or end is outside the range 1 - 132, or if beg is greater than end, the entire status line will be transmitted.

Note: All requests to display or clear status lines are ignored while status transmission is in progress. If the terminal is in Setup Mode, or a transmission is already in progress, the requested status line transmission is delayed.

Examples:

1. ESC [ \* t Transmits 'my' User Status Line. Terminal responds by transmitting to the requesting device the entire 132 characters of the User Status Line for that device (preceded by SOM and followed by EOM). This might be used in a multiple computer application when a computer needs to know which line it is physically attached to on the Concept AVT.
2. ESC [ 0;2;64;70 \* t Transmits columns 64 through 70 of the Programmer Status Line for the requesting device. The terminal responds by transmitting the requested information preceded by SOM and followed by EOM. The resulting string could be parsed by the program to determine if a prior Set Output Network command was successful.

Note: The Transmit Status Line command will be ignored if the terminal is in Local Mode (HDS Mode 111 set).

● **Toggle/Clear Status Line**      ESC [ tcs \* v {status}

When tcs is 0 or null, the currently displayed status line is toggled on or off, like depressing the STATUS key. When tcs is 1, the status line display is cleared. Other values for tcs are ignored.

Note: Command is ignored if terminal is in Setup Mode, or a status line transmission is in progress (see Transmit Status Line).

● **Set Background Status Line**      ESC [ bkg \* w

Sets the Background Status Line to the line of the specified window at which the cursor is currently located. The Background Status Line is displayed as the 25th line of the screen whenever no other status line is being displayed. It is continually updated to reflect changes due to scrolling or character input in the contents of the line to which it has been set.

To set the Background Status Line to a different line in the same window, the cursor must be positioned and the command re-issued.

Parameter `bkg` specifies the window whose current cursor line position defines the Background Status Line. If `bkg` is 0 or omitted, the Background Status Line is set to blank (i.e. not displayed).

Example:

```
ESC [ 4 * w  Sets the Background Status Line to be the line
              that window 4's cursor is currently on. This line will then
              be displayed when no status line is being displayed.
```

### ● **Transmit Answerback Message**      ↑E

Transmits the answerback message to the requesting device (or to the keyboard communication line if the command comes from the keyboard). See Section 3.21 for commands that define the answerback message. If no answerback message has been defined, or the terminal is in Local Mode (HDS Mode 111), this command is ignored.

Note: The exact text of the answerback message is all that is transmitted; programmable message characters are not added to the transmitted sequence.

### 3.5 Cursor Controls

#### ● Carriage Return      ↑M      {return}

Action taken depends on Linefeed Processing (HDS Mode 20):

Reset: Normal processing: a carriage return (↑M, ASCII 013) is sent to all devices in the output network of the device from which the carriage return is received. If the display is in the output network, the cursor is positioned at the beginning of the current line.

Set: When a carriage return is received from the keyboard, a line feed (↑J, ASCII 010) is added following the carriage return, so that the pair ↑M↑J is sent to all devices in the keyboard's output network. If the display is included in the output network, a New Line (see below) is executed, that is the cursor is positioned at the beginning of the next line.

Carriage returns received from devices other than the keyboard are processed as in Reset state.

#### ● Line Feed            ↑J            {line feed}

Action taken depends on Line Feed Processing (HDS Mode 20).

Reset: Line feeds are processed normally regardless of where they come from. If the display is in the output network of the device from which the line feed comes, the cursor is moved down one line in the current column position (i.e. and Index is performed).

Set: Line feeds coming from the keyboard are processed as in Reset state (a line feed (↑J, ASCII 010) is sent to all devices in the keyboard's output network). When a line feed is received from a device other than the keyboard, a carriage return (↑M, ASCII 015) is added, so that the pair ↑J↑M is sent to all devices in the output network of the device from which the line feed was received. If the display is in the output network, the cursor is positioned at the beginning of the next line (i.e. a New Line is performed).

Note: Operation of the Index and New Line commands depends upon whether Scrolling (HDS Mode 104) is On or Off.  
See Alternate Processing Modes, Section 4.4.

#### ● Index (Line Feed)      ESC D

Moves the cursor down one line in the same column. The action taken when the cursor is at the bottom of the window or scrolling region depends on whether Scrolling (HDS Mode 104) is on. If Scrolling is on (reset state), the window or scrolling region will scroll up a line, and the cursor will stay in the same column. If Scrolling is off (set state), the cursor will stay on the bottom line of the window or scrolling region, and no scrolling will occur.

● **Reverse Index**      **ESC M**

Moves the cursor up one line in the same column. The action taken when the cursor is at the top of the window or scrolling region depends on whether Scrolling (HDS Mode 104) is on. If Scrolling is on (reset state), the window or scrolling region will scroll down a line, and the cursor will stay in the same column at the top of the scrolling region. If Scrolling is off, (set state), the cursor will stay in the same column at the top of the window or scrolling region and no scrolling will occur.

● **New Line (Line Feed/CR)**      **ESC E**

Moves the cursor to the beginning of the next line. Action taken at the bottom of a window or scrolling region depends on Scrolling (HDS Mode 104). See Index command.

● **Cursor Up**      **ESC [ repeat A**      **{ ↑ }**

Moves the cursor up in the same column.

Parameter repeat specifies the number of lines upward the cursor is to move. If repeat is 0 or omitted, the cursor moves up one line.

Whether the cursor wraps around to the bottom of the window when it reaches the top of the window depends on whether Cursor Wraparound is on (HDS Mode 107).

● **Cursor Down**      **ESC [ repeat B**      **{ ↓ }**  
**(alternate)**      **ESC [ repeat e**

Moves the cursor down in the same column.

Parameter repeat specifies the number of lines the cursor is to move down. If repeat is 0 or omitted the cursor moves down one line.

Whether the cursor wraps around to the top of the window when it reaches the bottom of the window depends on whether Cursor Wraparound is on (see HDS Mode 107).

● **Cursor Right**      **ESC [ repeat C**      **{ → }**  
**(alternate)**      **ESC [ repeat a**

Moves the cursor to the right on the same line.

Parameter repeat specifies the number of columns the cursor is to move. If it is 0 or omitted, the cursor moves one column to right.

Whether the cursor wraps around to the beginning of the next line when it reaches the end of the current line depends on whether Cursor Wraparound is on (see HDS Mode 107).



● **Cursor Left**      **ESC [ repeat D**      { ← }

Moves the cursor to the left on the same line.

Parameter repeat specifies the number of columns the cursor is to move. If it is 0 or omitted, the cursor moves left one column.

Whether the cursor wraps around to the end of the previous line when it reaches the beginning of the current line depends on whether Cursor Wraparound is on (see HDS Mode 107).

● **Backspace**      ↑H      {back space}

Moves the cursor one column to the left on the same line.

Whether the cursor wraps around to the end of the previous line when it reaches the beginning of the current line depends on whether Cursor Wraparound is on (see HDS Mode 107).

Note: Even if Cursor Wraparound is on (HDS Mode 107 set), Backspace will not cause the cursor to wrap around from the home position (line 1 column 1 of the window) to the end of the window. In this respect, backspace differs from Cursor Left (above).

● **Cursor Up to Left Margin**      **ESC [ repeat F**

Moves the cursor to the beginning of the line and then up a specified number of lines.

Parameter repeat specifies the number of lines the cursor is to move up. If repeat is 0 or omitted, the cursor moves up one line.

Whether the cursor wraps around to the bottom of the window when it reaches the top of the window depends on whether Cursor Wraparound is on (see HDS Mode 107).

● **Cursor Down to Left Margin**      **ESC [ repeat E**

Moves the cursor to the beginning of the line and then down a specified number of lines.

Parameter repeat specifies the number of lines the cursor is to move down. If repeat is 0 or omitted, the cursor moves down one line.

Action taken at the bottom of the window or scrolling region depends on whether Cursor Wraparound (HDS Mode 107) is on.

● **Position Cursor**      **ESC [ lin;col H**  
    **(alternate)**      **ESC [ lin;col f**

Moves the cursor to a specific line and column position relative to the window or scrolling region.

**Parameters:**

`lin` specifies the line position.

`col` specifies the column position.

Line and column numbering begins with 1. If a parameter value exceeds the bounds of the window or scrolling region, it will take on the maximum value for the window or scrolling region. The default value of 1 is used when a parameter is 0 or omitted. If both parameters are 0 or omitted, the cursor is moved to the home position in the window or scrolling region.

Whether the cursor is positioned relative to the beginning of the window or scrolling region is determined by Cursor Addressing (HDS Mode 206). If Cursor Addressing is Window-Relative (reset state), the cursor is positioned in relation to the origin of the window; if it is Scrolling Region-Relative, (set state) the cursor is positioned relative to the origin of the scrolling region.

**Examples:**

1. "Home Cursor": `ESC [ H` moves the cursor to the home position in the window or scrolling region.
2. Assume that a scrolling region has been defined as lines 10 to 20 of the current window. When the command `ESC [ 5;10 H` is given, the effect depends on Cursor Addressing (HDS Mode 206):
  - Reset: The cursor moves to line 5, column 10 of the window.
  - Set: The cursor moves to line 14, column 10 of the window (i.e., line 5, column 10 of the scrolling region.)

● **Position Cursor to Column**      **ESC [ col G**  
**(alternate)**                      **ESC [ col `**

Moves the cursor along the current line to a specified column. See Position Cursor for further information.

● **Position Cursor to Line**      **ESC [ lin d**

Moves the cursor along the current column to a specified line. See Position Cursor command for further information.

● **End of Text**              **ESC 2**

Moves the cursor to the last non-space character in the window or scrolling region, depending on whether Cursor Addressing (HDS Mode 206) is Window-Relative (reset) or Scrolling Region-Relative (set).

● **Transmit Cursor Position**      **ESC [ 6 n**

Transmits the current cursor position to the requesting device.

Response: ESC [ lin;col R , where lin and col are three digit numbers (including leading zeros).

Whether the address transmitted is window-relative or scrolling region-relative depends on Cursor Addressing (HDS Mode 206).

● **Save Cursor Position and Attributes**      **ESC 7**

Saves the window-relative cursor position, state of Cursor Addressing (HDS Mode 206), video attributes, and character set. All of these can be restored by using the Restore Cursor command.

Note: Since the cursor position is saved relative to the window, it is possible to restore the cursor outside of the scrolling region when Cursor Addressing is Scrolling Region-Relative (HDS Mode 206 in set state).

● **Restore Cursor Position and Attributes**      **ESC 8**

Restores the window-relative cursor position, Cursor Addressing (HDS Mode 206), video attributes, and character set that were stored by the most recent Save Cursor command. See Save Cursor Position and Attributes.

● **Tab**      **↑I {tab}**

Executes a single Forward Tab. See Forward Tab command (below).

● **Forward Tab**      **ESC [ repeat I**

Moves the cursor forward a specified number of text tab stops or unprotected fields, depending on Tab Processing (HDS Mode 105).

Parameter repeat specifies text tab stop or unprotected field to which the cursor is to be moved. Action taken depends upon Tab Processing (HDS Mode 105):

Reset: (Text) Moves the cursor forward on the same line repeat text tab stops. If repeat is 0 or omitted, the cursor moves to the next text tab stop. If there are no more text tab stops between the cursor and the right margin of the window, the cursor moves to the rightmost position on the line.

Set: (Form) Moves the cursor to the beginning of the unprotected field that is repeat unprotected fields ahead of the current cursor position. If repeat is 0 or omitted, the cursor moves to the beginning of the next unprotected field. If there are no more unprotected fields in the window, the cursor moves to the end of the window.

● **Backward Tab**      **ESC [ repeat Z      {B TAB}**

Moves the cursor backward a specified number of text tab stops or unprotected fields, depending upon Tab Processing (HDS Mode 105).

Parameter repeat specifies the tab stop or unprotected field to which the cursor is to be moved. Action taken depends on Tab Processing (HDS Mode 105):

Reset: (Text) Moves the cursor backwards on the same line repeat text tab stops. If repeat is 0 or omitted, the cursor moves to the next text tab stop back. If there are no more text tab stops between the cursor and the left edge of the window, the cursor moves to the left edge of the window.

Set: (Form) Moves the cursor backward in the window to the beginning of the unprotected field repeat unprotected fields ahead of the current cursor position. If repeat is 0 or omitted, the cursor moves to the beginning of the previous unprotected field. If the cursor is in the first unprotected field in the window, the cursor moves to the beginning of the window.

● **Set Tab**      **ESC H**

Sets a text tab stop at the current cursor column.

● **Clear Tabs**      **ESC [ clrtab g**

Parameter clrtab specifies tabs to clear:

- 0 Clear text tab stop, if any, at the current cursor position (default).
- 2,3 Clear all text tab stops.

Other values cause the command to be ignored.

● **Tab Control**      **ESC [ tabctl W**

Sets or clears text tab stops.

Parameter tabctl specifies the action to be taken:

- 0 Set a tab stop at the current cursor column (default).
- 2 Clear the tab stop, if any, at the current cursor column.
- 4,5 Clear all tab stops.

Other values cause the command to be ignored.

## 3.6 Editing

### 3.6.1 Clearing Display Memory

Many editing functions, including insertion and deletion of characters and lines, involve "clearing" an area of display memory. The area cleared may be a single character, a line, or an entire window. An area of display memory is cleared by clearing each character location in accordance with the current Clear Characteristics. These include both a "clear character", and a list of character attributes known as the "clear list", used in conjunction with the "current attribute" in determining the way attributes are to be set when a character is cleared. See Section 3.7.1 for more information on character attributes. A location of display memory is cleared as follows:

1. The character is set to the current "clear character", which by default is a space (ASCII 032). This can be changed at any time by the Set Clear Characteristics command (below).
2. The display attributes are set by selecting some attributes from the "current attribute list" and some from the current "clear list", according to the last Set Clear Characteristic command issued.

If no such command has been issued, all attributes are taken from the "clear list", which by default has all attributes set to their default state (see example 1 below).

3. The Set Clear Characteristics command (below) determines both which attributes are to be taken from the clear list, and which attributes in the clear list are to be set to alternate state.

#### Examples:

1. In factory default state, all attributes in both the current list and the clear list are in default state. As characters are placed in display memory, they are displayed in normal brightness and video, with no underline, non-blinking and no protection. When an area of display memory is cleared, a space (ASCII 032) is placed in each character location, and all character attributes are set to those in the (default) clear list. This area of display memory appears blank (dark) on the screen.
2. If the current attribute list is changed to include inverse video (using the command ESC [ 7 m ), text in display memory appears on the screen as dark characters on a light ground (assuming that screen video is normal -- HDS Mode 205 reset), but otherwise as in example 1. Since no change has been made in the clear list, display memory is still cleared by being set to spaces displayed in normal video, so it appears blank (dark) on the screen.

3. Assuming the situation described in example 2, the clear characteristics can be changed so that when an area of display memory is cleared, it is set to inverse (as opposed to normal) video. This is done by using the Set Clear Characteristics Command to set the Normal/Inverse Video attribute (7) to its alternate state:

```
ESC [ = 1;4;5;7;8;99 : 7 : 32 ! r
```

This command specifies that all attributes (i.e., 1, 4, 5, 7, 8, and 99) are to be taken from the clear list, and that attribute 7 is to be set to its alternate state (inverse video) in the clear list. The effect of this command is to make cleared areas of display memory appear light on the screen.

4. Non-displayable characters (characters with attribute 8 set to alternate state) have special significance for certain editing commands explained below. To take advantage of these functions it might be desirable to have display memory clear to non-displayable spaces. This could be done by issuing the command.

```
ESC [ = 1;4;5;7;8;99 : 8 : 32 ! r
```

### ● Set Clear Characteristics      ESC [ atusd:atalt:chr ! r

Determines how display memory is "cleared". See the beginning of this section for a discussion of clearing display memory.

#### Parameters:

atusd specifies the attribute(s) to be taken from the "clear list"; attributes not listed are taken from the current attribute list.

atalt specifies the attributes in the clear list to be set to alternate state. Those attributes that are specified by atusd but not specified by atalt are set to default state. Attributes specified by atalt but not specified by atusd are ignored.

Attribute	Default State	Alternate State
1	Normal Brightness	Bold
4	No Underline	Underline
5	Non-blinking	Blinking
7	Normal Video	Inverse Video
8	Displayable	Non-displayable
(=) 99	No Protection	Protection

Note: When attribute 99 is specified for either atusd or atalt, the entire parameter string must be preceded by = .

chr specifies the ASCII chart location of the character to be used as the "clear character".

● **Form Feed**      **↑L**

Action taken depends on Form Feed Processing (HDS Mode 122):

**Reset:** Clears the current window, and positions the cursor in the home position (line 1, column 1) of the window or scrolling region (depending on HDS Mode 206).

**Set:** Executes an Index (moves cursor down one line in the same column). This is the factory default.

● **Select Editing Extent**      **ESC [ extent Q**

Determines the area affected by character insertion and deletion (see Delete Character and Toggle Insert Mode commands).

Parameter extent specifies the area affected:

- 0    Insert/Delete in Window (default). Character insertion and deletion affects the entire window, causing character movement wrapping around the ends of lines within the window.
- 1    Insert/Delete in Line. Character insertion and deletion affects only characters in the current line.
- 3    Insert/Delete in Field. Character insertion and deletion affects only characters in the current unprotected field.

**Note:** If both 1 and 3 are specified, the result is the same as for 3 (Insert/Delete in Field), except that characters do not wrap around ends of lines in multiline fields.

● **Toggle Insert Mode**      **ESC I**      **{insrt}**

Switches Replace/Insert Characters (HDS Mode 4) between Replace and Insert. In Insert Mode, character insertion begins immediately ahead of the current cursor position.

The effect of character insertion on characters following the current cursor position depends on whether the Editing Extent is window (default), line, or field (see the Select Editing Extent command).

extent	action taken
window	All characters from the point of insertion to the next non-displayable space or the end of the window are shifted forward away from the cursor, wrapping around the end of the window as necessary. Characters shifted beyond the end of the window are lost.

**line** All characters from the point of insertion to the next non-displayable space or the end of the line are moved forward away from the cursor. Characters shifted beyond the end of the line are lost.

**field** All characters from the point of insertion to the next non-displayable space or the end of the field are moved forward away from the cursor. If both line and field have been specified by the Select Editing Extent command, characters shifted forward away from the cursor will not wrap around lines within a multiline field.

● **Delete Character**      **ESC [ repeat P**      **{del char}**

Deletes a specified number of characters, beginning with the current cursor position.

Parameter repeat specifies the number of characters to delete. If repeat is 0 or omitted, one character is deleted.

The effect of this command on the remaining characters depends on whether the Editing Extent is window (default), line, or field (see the Select Editing Extent command).

extent	action taken
window	All characters from the point of deletion to the next non-displayable space or the end of the window are shifted backward toward the cursor, wrapping around the edges of the window as necessary.
line	All characters from the point of deletion to the next non-displayable space or the end of the line are moved toward the cursor.
field	All characters from the point of deletion to the next non-displayable space or the end of the field are moved backward toward the cursor. If both line and field have been specified by the Select Editing Extent command, the movement of characters toward the cursor will not wrap around lines within a multiline field.

● **Insert Line**      **ESC [ repeat L**      **{ins l}**

Inserts a specified number of lines at the current cursor position by moving the text down the required number of lines and leaving the cursor at the beginning of the first line inserted. The attributes of the cleared line(s) inserted are determined by the current Clear Characteristics (see Section 3.6.1).

Parameter repeat specifies the number of lines to insert. If it is 0



or omitted, 1 line is inserted.

Note:

1. This command is ignored if the cursor is outside of the Scrolling Region. Such a condition does not ordinarily occur, since the Scrolling Region is usually the same as the window.
2. The attributes of the cleared lines inserted are determined by the current Clear Characteristic (see beginning of this section).

● **Delete Line**      **ESC [ repeat M      {DEL L}**

Deletes a specified number of lines, beginning with the line at which the cursor is positioned and proceeding down. Lines below those just deleted are scrolled up to fill the gap, and cleared lines are created at the bottom of the Scrolling Region. The cursor is positioned at the beginning of the line it was on when the command was entered.

Parameter repeat specifies the number of lines to delete. If repeat is 0 or omitted, one line is deleted.

Note:

1. This command is ignored if the cursor is outside of the Scrolling Region. Such a condition does not ordinarily occur, since the Scrolling Region is usually the same as the window.
2. The attributes of the cleared lines inserted at the bottom of the scrolling region are determined by the current Clear Characteristic (see beginning of this section).

● **Erase in Window**      **ESC [ erase J      {ERAS}**

Clears characters in the current window. See beginning of this section for an explanation of clearing characters.

Parameter erase specifies the area in which characters are to be cleared:

- 0 From cursor to end of window, inclusive. (default)
- 1 From beginning of window to cursor, inclusive.
- 2 All of window.

Other values cause the command to be ignored.

Whether protected characters (i.e. characters in protected fields) are erased depends upon the status of HDS Mode 6 (Erase Protected Characters). If this mode is set, protected characters are erased; otherwise they are not.

Note: The effect of this command does NOT depend on whether Cursor

Addressing is window-relative or scrolling region-relative (HDS Mode 206).

● **Erase in Line**      **ESC [ erase K {eras}**

Clears characters within the current line. See Section 3.6.1 for an explanation of clearing characters.

Parameter erase specifies the area affected:

- 0 From cursor to end of line inclusive (default).
- 1 From beginning of line to cursor inclusive.
- 2 Entire line.

Other values cause the command to be ignored.

Whether protected characters are erased depends on Erase Protected Characters (HDS Mode 6). If this mode is set, protected characters are erased; otherwise they are not.

● **Erase in Field**      **ESC [ erase 0**

Clears characters within the current field. See Section 3.6.1 for an explanation of clearing characters.

Parameter erase specifies the area affected:

- 0 From cursor to end of field, inclusive (default).
- 1 From beginning of field to cursor, inclusive.
- 2 Entire field.

Other values cause the command to be ignored.

A field is a string of one or more unprotected characters, bounded by either one or more protected characters, or the beginning or end of the current window.

Note: This command is ignored if the cursor is currently in a protected field, even if Erase Protected Characters is allowed (HDS Mode 6 set).

● **Set Margin Bell**      **ESC [ offset ! y**

Sets the column position at which the margin bell rings. Once this command has been entered, the bell will ring once each time character input moves the cursor past the position specified.

Parameter offset specifies the column, counting from the righthand boundary of the window, at which the bell is to ring. If offset is 0 or omitted, the margin bell is turned off.

## Examples:

1. ESC [ 8 ! y sets the margin bell to ring when character input moves the cursor past the eighth column from the righthand edge of the window. For the default window of 80 columns, this means that the bell rings when the cursor moves past the 72nd column. If the window were defined as 132 columns wide, the bell would ring at column 124.
2. ESC [ ! y turns off the margin bell.

Note: The bell does not ring when the cursor moves past the specified column position as a result of cursor movement commands.

### 3.7 Display Controls

#### 3.7.1 Character Attributes

Associated with each character location in display memory is a set of "attributes" that determine how the character is to be displayed on the screen, as well as how it is processed by certain terminal functions. Each attribute is in either Default or Alternate state. Factory defaults are designed so that, unless some command is issued to select Alternate rather than Default attributes, characters are displayed "normally" on the screen, i.e., with all of the attributes described below in their default states.

Note: Each attribute is independent of the others, and the selection of attributes is additive.

The only attribute available for overstruck characters is Reverse Video (7). Other characters may use any combination of attributes.

- 1   Default:    Normal Brightness.  
    Alternate: Bold. Light areas of the screen, whether characters or background, are presented in greater intensity than normal.
  
- 4   Default:    No underline.  
    Alternate: Underline. All characters (including spaces) appear underlined. If Underline Attribute Processing (HDS Mode 115) is reset, each underlined character is followed by a backspace and an underline character (ASCII 095) when transmitted.
  
- 5   Default:    Non-Blinking.  
    Alternate: Blinking. Character blinks on and off, while background and underline (if any) remain constant.
  
- 7   Default:    Normal Video. Character and background are same as Screen Video (HDS Mode 205).  
    Alternate: Inverse Video. Character and background are opposite to Screen Video. If Screen Video is light characters on dark background, then characters with inverse video appear as dark characters on light background. If Screen Video is dark characters on light background, characters with inverse video appear as light characters on dark background.
  
- 8   Default:    Displayable. Character is displayed in accordance with other attributes.  
    Alternate: Non-Displayable. Character is not displayed on the screen, but is otherwise processed normally. See Editing Commands for special processing in regard to non-displayed spaces.

(=)99 Default: No Protection. Characters are processed normally.  
 Alternate: Protection. Characters with this attribute cannot normally be overwritten, erased, or deleted (though this protection can be overridden by setting appropriate Alternate Processing Modes).

Note: if attribute 99 is specified in any parameter list, the entire list must be preceded by an equal sign ( = ).

**Attribute Table**

Attribute	Default State	Alternate State
1	Normal Brightness	Bold
4	No Underline	Underline
5	Non-blinking	Blinking
7	Normal Video	Inverse Video
8	Displayable	Non-displayable
(=) 99	No Protection	Protection

During normal character input, the combination of attributes associated with a given character in display memory is determined by the Attribute List being used by the device from which the characters were received (generally referred to as the "current attribute list"). The association of an Attribute List with a device is determined by the Select Attribute List command. The particular combination of attributes in a given list is determined by Select Alternate Attribute and Select Default Attribute commands received from the device currently associated with the Attribute List in question.

Ordinarily, when a character in display memory is replaced by a new character, the attributes associated with the old character are replaced by the current attributes (i.e. the attributes in the list associated with the device from which the character was received). However, by setting Character/Attribute Replacement (HDS Mode 120) to Character Only (set state), it is possible to have the attributes remain constant while the characters change.

The display attributes of a given area of display memory may also be determined by commands (such as Delete Character) that "clear" display memory. (See Section 3.6.1 for a discussion of clearing display memory.) The attributes associated with a cleared area of display memory may be specified via the Set Clear Characteristics command (Section 3.6).

● **Select Alternate Attributes**      **ESC [ atalt m**

Adds the alternate state of specified attributes to the current attribute list.

Parameter `atalt` specifies the attribute(s) to be set to alternate state. If `atalt` is 0 or omitted, all attributes are set to their default states; otherwise, only attributes specified are affected by the command. Allowable values are listed in the Attribute Table at the beginning of this section; all other values for `atalt` are ignored.

Parameters are applied incrementally, in the order in which they appear in the command sequence. Do not use values other than 7 (Reverse Video) for overstruck characters.

Examples:

1. `ESC [ 7 m` Adds inverse video to the attribute list. If the attribute list already included bold, the result would be bold, inverse video.
2. `ESC [ 0 ; 7 m` Sets all attributes to default state, and then sets attribute 7 to alternate state (inverse video).

### ● **Select Default Attributes**      `ESC [ atdef ! {`

Selects the default state for specified attributes. Only attributes specified are affected.

Parameter `atdef` specifies attributes to be set to default state. Allowable values are listed in the Attribute Table at the beginning of this section; other values are ignored. Do not use values other than 7 (Reverse Video) for overstruck characters.

### ● **Block Attribute Change**      `ESC [ atusd:atalt:lns;cls ! q`

Changes the attributes of a rectangular block of display memory.

Parameters:

`atusd` specifies the attributes that are to be changed. Allowable values are listed in the Attribute Table at the beginning of this section. Other values are ignored. Do not use values other than 7 for overstruck characters.

`atalt` specifies the attributes to be set to alternate state. Allowable values are listed in the Attribute Table at the beginning of this section. Attributes specified in `atusd` but not in `atalt` are changed to their default state. Attributes specified in `atalt` but not in `atusd` are ignored.

`lns` specifies the number of lines, starting from the current cursor over which attributes are to be changed. The default for `lns` is 1. If `lns` is greater than the number of lines from the cursor to the end of the window, attributes are changed up to the end of the window.

`cls` specifies the number of columns, starting from the current

cursor position, over which attributes are to be changed. The default for `cls` is 1. If `cls` is greater than the number of columns from the cursor to the right hand edge of the window, attributes are changed up to the right hand edge of the window.

Note: The area over which attributes are changed is limited only by the window, not by the Scrolling Region (if it has been defined as different from the window).

● **Block Character Change**      **ESC [ chr;cols;lins p**

Repeats a single character over a specified rectangular area of display memory. Cursor does not move, and nothing is transmitted.

Parameters:

`chr` specifies the ASCII chart location of the character to be repeated. If `chr` is omitted, ASCII chart location 000 (null) is used. For values greater than 127, the ASCII chart location is computed as `chr modulo 128`.

`cls` specifies the number of columns, starting from the current cursor over which to generate the character. The default for `cls` is 1. If `cls` is greater than the number of columns from the cursor to the right edge of the window, the character is repeated up to the right edge of the window (that is, characters do not wrap around).

`lins` specifies the number of lines, starting from the current cursor position, over which to generate the character. The default for `lins` is 1. If `lins` is greater than the number of lines from the cursor to the bottom of the window, the character is repeated up to the end of the window.

Note:

1. Characters generated by this command overwrite protected characters and are not affected by Auto Tabs (HDS Mode 106).
2. The area over which characters are changed is limited only by the window, not by the Scrolling Region (if it has been defined as different from the window).
3. The attributes of the characters generated depends on Character/Attribute Replacement (HDS Mode 120). If it is reset, all characters generated are displayed using the current set of attributes. If it is set, each character generated is displayed using the set of attributes that was used to display the character it replaced in display memory.

4. Displayable versions of control codes can be generated using this command.

Examples:

1. ESC [ 42;35;10 p Generates a block of asterisks (\*, ASCII 042) 35 columns by 10 lines, beginning at the cursor and proceeding down and to the right.
2. ESC [ 45;132 p Fills the remainder of the current line with dashes (-, ASCII 045). Note that this command would have this effect regardless of the current window definition, since 132 is the maximum window width. (Parameter lns is omitted, so the default value 1 is used.)
3. ESC [ 013;1;1 p Displays the transparent mode representation of a carriage return (ASCII 013) in the current cursor position.

● **Transparent Mode On      ESC Q**

Puts the terminal in Transparent Mode (HDS Mode 3 set). In this mode, control codes (such as carriage return, back space, form feed, etc.) are processed by displaying a special control code symbol rather than executing the appropriate terminal action. The control code symbols that appear in transparent mode are shown in the ASCII Character Set chart in Appendix A.

All control characters are displayed, including those normally only used for communications controls, such as NUL (ASCII 000), RUB (ASCII 127), XON (ASCII 017) and XOFF (ASCII 019). This also includes the escape character; however, if an escape is received, it is buffered until the next character is received from the device. If the next character is an R, Transparent Mode is turned off.

Note: Transparent Mode can also be turned on by using the Set command for Transparent/Control Code Processing (HDS Mode 3).

● **Transparent Mode Off      ESC R**

Takes the terminal out of Transparent Mode. When terminal is not in transparent mode, control characters are processed normally (carriage return moves the cursor to the beginning of the current line, ↑H moves the cursor one column to the left, etc.)

● **Protection On      ESC V**

Turns Protection on in the requesting device's attribute list. This has the same effect as using the Select Alternate Attribute command to add protection to the current attribute list ( ESC [ = 99 m ).



**● Protection Off      ESC W**

Turns Protection off in the requesting device's attribute list. This has the same effect as using the Select Default Attributes command to specify No Protection in the current attribute list ( ESC [ = 99 ! { ).

### 3.8 Multiple Devices

#### ● **Print/Line 3 Control**      **ESC [ media i**

Manages the flow of data between the terminal and devices connected to Line 2 (Printer Port) and Line 3. When the command is entered from the keyboard (either by striking a special key, or by keying in the command sequence), the bell rings if the printer or other device is unavailable.

Parameter media specifies the action to be taken:

- 0 Print window to cursor position. Attaches printer (or other device) connected to Line 2, transmits data in current window to Line 2, and detaches printer (or other device). Can be entered from the keyboard using the PRINT SCRN key (unshifted).

Data is transmitted starting at the beginning of the window or Start of Print/Transmit, depending on Transfer Initiation (HDS Mode 116), and ends with the character just ahead of the cursor.

If Form Feed Prior to Print is on (HDS Mode 117 set), the text to be transmitted is preceded by a Form Feed (↑L, ASCII 012) and a one-half second delay to allow hard-copy devices time to complete the form feed. Each line, including the last, is followed by a carriage return (ASCII 013) and a line feed (ASCII 010).

Note:

1. Control Codes (ASCII 000 - 031 and 127) are replaced by spaces (ASCII 032) before being sent to the printer.
2. If Transmit Unprotected/All (HDS Mode 1) is Reset, all protected characters are replaced by spaces.
- 2 Send to Line 3. Functions just like the Transmit command, except that data is always sent to Line 3 (rather than to the requesting device). Control Codes (ASCII 000 - 031 and 127) are sent as is. See Transmit command (Section 3.9 for details of interactions with Alternate Processing Modes.
- 4 Detach Printer. Detaches the printer (or other device) connected to Line 2. Removes Line 2 from all output networks (see Set Output Network). This command can be entered from the keyboard using the shifted PRINT key.
- 5 Attach Printer (Line 2). Attaches the Printer (or other device connected to Line 2), and adds it to the output network for every device that has the display in its output network. If this command comes from the keyboard, and the printer (or other device) connected to Line 2 is not available, the keyboard bell will sound. If the command

comes from some other device, the command will be ignored if the printer is unavailable. This command can be entered from the keyboard using the PRINT key (unshifted).

- 6 Detach Line 3. Detaches the device connected to Line 3, and removes it from all output networks.
- 7 Attach Line 3. Attaches the device connected to Line 3, and adds it to the output network for every device that has the display in its output network. If this command comes from the keyboard and the device connected to Line 3 is not available, the keyboard bell will sound. If the command comes from some other device, the command will be ignored if the device is not available.
- (=) 8 Print to End of Window. Functions exactly like 0 (Print Window to Cursor Position) except that the transmission of data ends at the end of the window rather than at the cursor position. This command can be entered from the keyboard using shifted PRINT SCRN.
- (=) 9 Print Line. Functions exactly like 0, except that only the line on which the cursor is currently located is transmitted.

Note: 8 and 9 are HDS Private Parameters, and must be preceded by an equal sign ( = ).

### ● Set Output Network      ESC [ output : dev t

Specifies where data received from a given line or device is to be sent. In factory default state, data from all communication lines is sent to the screen; keyboard input is sent to Line 1 (the default Keyboard Communication Line), but not to the screen.

Parameters:

output specifies the line(s) or device(s) to which data is to be sent.

0	No devices (default).	3	Line 3.
1	Line 1.	9	Display (i.e. window used by device dev).
2	Line 2.		

Other values cause the command to be ignored.

dev specifies the device to which the command applies, i.e., the the device from which data is received. The default is 0.

0	Requesting device.	3	Line 3.
1	Line 1	9	Keyboard.
2	Line 2		

Other values cause the command to be ignored.

## Examples:

1. ESC [ 0:1 t sets the output network for Line 1 to be no devices. This means that characters (both data and control codes) are effectively ignored, since they are neither displayed on the screen nor sent to any other device.
2. The pair of commands ESC [ 1;9 : 9 t and ESC [ 9 : 1 t establishes the appropriate network for "half duplex" operation. The first command causes data from the keyboard to be sent to both Line 1 and the Display. The second command causes data from Line 1 to be sent to the Display. The same effect could be achieved by using Setup Mode to set Half Duplex (see Section 1.9).
3. ESC [ 2 : 3 t causes characters from Line 3 to be sent to Line 2 only. This configuration might be used to transfer data from a host connected to Line 3 to a printer (or diskette) connected to Line 2 while the keyboard and screen are being used to interact with a host connected to Line 1.

Note: This command does not generate any direct response to the user. If it is necessary for a program to determine whether the command was successful, the program can request a partial transmission of the Programmer Status Line, and directly examine the output network for the device in question. See the Transmit Status Line command (section 3.4).

● **Set Keyboard Communication Device**      **ESC [ devcm z**

Determines the communication line with which the keyboard communicates, that is, the line to which keyboard input is sent when the terminal is in Remote Mode (HDS Mode 111), and to which commands controlling data transmission (Section 3.9) apply when issued from the keyboard. For example, striking the SEND key causes data to be transmitted to the Keyboard Communication Device. Similarly, the contents of a programmable key set to "transmit" mode are sent to the Keyboard Communication Device.

Parameter devcm specifies the line. The default is 1.

- 1 Line 1.
- 2 Line 2.
- 3 Line 3.

Other values cause the command to be ignored.

### 3.9 Transmission Control

#### ● **Set Parity/Parity Checking**      **ESC [ parity;check;devcm \* p**

Controls the generation of parity bits in data transmitted to the line specified, as well as the checking of parity on data received from the line specified. See Appendix D.

Parameters:

par specifies the type of parity generated on output:

0	No parity bit (default).	3	Mark parity (1).
1	Even parity	4	Space parity (0).
2	Odd parity		

Other values cause the command to be ignored.

NOTE: If No Parity Bit (0) is selected, one less bit will be transmitted than for all other parity settings.

check specifies whether parity is checked on input:

0	No Parity Checking (default)
1	Parity Checking. Incoming characters with parity errors are converted to tildas ('~', ASCII chart location 126).

Other values cause the command to be ignored.

devcm specifies the line to which the command applies:

0	Requesting device, or keyboard communication device (if the command comes from the keyboard)	1	Line 1.
		2	Line 2.
		3	Line 3.

The default is 0; values not listed cause the command to be ignored.

#### ● **Set Baud Rate**      **ESC [ baud;devcm \* r**

Determines the speed at which data is transmitted and received on the communication line specified by devcm. Different baud rates can be specified for each line.

Parameters:

baud specifies the baud rate (the default is 0):

0	50	4	150	8	1800	12	4800
1	75	5	300	9	2000	13	7200
2	110	6	600	10	2400	14	9600
3	134.5	7	1200	11	3600		

The default is 0; values not listed cause the command to be ignored.

devcm specifies the device to which the command applies:

0	Requesting device, or keyboard communication device (if the command comes from the keyboard)	1	Line 1.
		2	Line 2.
		3	Line 3.

The default is 0; values not listed cause the command to be ignored.

### ● **Set Stop Bits**      **ESC [ sbit;devcm \* s**

Determines the number of stop bits to be used in transmitting data to the line specified. See Appendix D.

Parameters:

sbit specifies the number of stop bits to use:

0,1	One stop bit (default).
2	Two stop bits.

Other values cause the command to be ignored.

devcm specifies the device to which the command applies:

0	Requesting device, or keyboards communication device (if the command comes from the keyboard)	1	Line 1.
		2	Line 2.
		3	Line 3.

The default is 0; values not listed cause the command to be ignored.

### ● **CTS/RTS (Transmit) Protocol**      **ESC [ xmit;devcm \* x**

Causes Clear to Send/Request to Send hardware protocol to be used in transmissions to a line specified.

Parameters:

xmit specifies the operation of CTS/RTS signals:

0	Protocol off (default): RTS (Request to Send) is always held high, and CTS (Clear to Send) is not required for transmission.
1	Protocol on. RTS is only held high if a device is networked to the requesting device. CTS must be high for any transmission to occur over the line.

Other values cause the command to be ignored.

devcm specifies the device to which the command applies:

0	Requesting device, or keyboard communication device (if the command comes from the keyboard)	1	Line 1.
		2	Line 2.
		3	Line 3.

The default is 0; values not listed cause the command to be ignored.

### ● **Buffer Overflow (Receive) Protocol**      **ESC [ buff;devcm \* q**

Determines whether software buffer overflow control (XON/XOFF) is to be used on input for a specified device.

Parameters:

buff specifies whether buffer overflow control is to be used:

0	No software buffer overflow control (default).
1	Input buffer overflow controlled as follows: when the input buffer is half full, XOFF (↑S, ASCII 019) is sent to the host. When the buffer is down to one quarter full, XON (↑Q, ASCII 017) is sent to the host. Only devices for which buffer overflow control has been specified are affected.

Other values cause the command to be ignored.

devcm specifies the device to which the command applies:

0	Requesting device, or keyboard communication device (if the command comes from the keyboard)	1	Line 1.
		2	Line 2.
		3	Line 3.

The default is 0; values not listed cause the command to be ignored.

**Note:** This command only determines whether the terminal sends XON/XOFF to the host computer. By default, the terminal responds to incoming XOFF (↑S) by ceasing transmission (both Block and Character Mode) to the line from which the XOFF was received until an XON (↑Q) is received from the same line. See Appendix H for information on disabling this response.

● **Stop Transmission**      ↑S

Causes the terminal to cease transmission (both Block and Character Mode) to the communication line from which the command was received. This command is normally used by a host computer or a peripheral device (such as a printer) to prevent overflow of input buffers. It has no effect when entered from the keyboard, and is not affected by the Buffer Overflow (Receive) Protocol command. See Appendix H.

● **Resume Transmission**      ↑Q

Causes the terminal to resume transmission to the line from which it was received. It is normally used to resume transmission that has been stopped by a Stop Transmission command. If transmission has not been stopped, this command has no effect. See Appendix H.

● **Transmit**      ESC 5      {send}

Transmits a portion of the window to the requesting device, or to the keyboard communication line, if the command is issued from the keyboard.

The data transmitted depends on the current Transmit Extent (see below), Transmit Initiation (HDS Mode 116), Transmit Termination (HDS Mode 16), Start of Print/Transmit (see below), and Transmit All/Unprotected (HDS Mode 1).

Data from the window is preceded by the programmable Start of Message (SOM) character (if it is in use), and followed by the programmable End of Message (EOM) character. Each line is followed by the programmable End of Line (EOL) character. When transmitting only unprotected fields (HDS Mode 1 reset), the end of each (unprotected) field is followed by the programmable End of Field (EOF) character, except in the case of a field overlapping the end of the line, in which case a programmable End of Line (EOL) is sent.

Notes:

1. Command is ignored if terminal is in Local Mode (HDS Mode 111 set).
2. Programmable message characters SOM, EOM, EOF, and EOL are transmitted only if they are currently defined to be some ASCII character (SOM is not used by default). Also, EOM, EOF, and EOF can be defined to be pairs of characters. See Change Message Character.

● **Set Transmit Extent**      ESC [ tfrext \* y

Determines how much data is to be sent as the result of a Transmit command.



Parameter `trfext` specifies the scope of the transmission:

- 0 Window (default). Transmission begins at the beginning of the window, or at the Start of Print/ Transmit, depending on Transfer Initiation (HDS Mode 116). Transmission ends at the end of the window or at the cursor position, depending on Transfer Termination (HDS Mode 16).

Note: The effect of this parameter does NOT depend on whether Cursor Addressing is window-relative or scrolling-region relative (HDS Mode 206).

- 1 Line. Transmission begins at the beginning of the current line, and ends at the end of the line or at the cursor position, depending on Transfer Termination (HDS Mode 16).
- 3 Field. Transmission begins at the beginning of the current field and ends at the end of the field or at the cursor position, depending on Transfer Termination (HDS Mode 16). See the beginning of the Editing Commands section for a discussion of fields.

Note: A separate Transmit Extent is maintained for each window definition.

### ● **Set Transmit Delay**      **ESC [ dly \* z**

Determines the length of the time delay prior to the execution of transmission commands (such as Transmit, Printer/Line 3 Control).

Parameter `dly` specifies delay in 100 millisecond units. The default is 0 (no delay).

### ● **Break**      **ESC [ devcm \* ~**      **{break}**

Causes a break to be initiated for device `devcm`:

Parameter `devcm` specifies the device to which the command applies:

- |   |                          |   |         |
|---|--------------------------|---|---------|
| 0 | Requesting device, or    | 1 | Line 1. |
|   | keyboards communication  | 2 | Line 2. |
|   | device (if the command   | 3 | Line 3. |
|   | comes from the keyboard) |   |         |

A 'break' condition involves holding the transmit line for the device high for approximately 300 milliseconds.

Note: Command is ignored if terminal is in Local Mode (HDS Mode 111 set).

**● Start of Print/Transmit      ESC ?**

Defines the Start of Print/Transmit as the current cursor position.

A separate Start of Print/Transmit is maintained for each window definition. It is automatically reset to the window's home position when the window is redefined.

### 3.10 Screen Controls

● **Scroll Down**      **ESC [ repeat T      {scrol}**

Moves the screen ahead with respect to display memory, so that a later portion of display memory appears on the screen.

Parameter repeat specifies the number of lines the display is to move. The default is 1.

If Cursor Addressing is Window-Relative (HDS Mode 206 reset), the screen will not move below the last line of the window currently used by the keyboard or the end of display memory. If Cursor Addressing is Scrolling Region-Relative (HDS Mode 206 set), the display will not move below the last line of the scrolling region currently used by the keyboard, or the end of display memory.

● **Scroll Up**      **ESC [ repeat S      {SCROL}**

Moves the screen backward with respect to display memory, so that an earlier portion of display memory appears on the screen.

Parameter repeat specifies the number of lines the display is to move. The default is 1.

If Cursor Addressing is Window-Relative (HDS Mode 206 reset), the display will not move above the first line of the window currently used by the keyboard or the beginning of display memory. If Cursor Addressing is Scrolling Region-Relative (HDS Mode 206 set), the display will not move above the first line of the scrolling region currently used by the keyboard, or the beginning of display memory.

● **Page Down**      **ESC [ repeat V      {page}**

Moves the screen ahead with respect to display memory by a specified number of pages. A single Page Down command has exactly the same effect as scrolling down 24 lines. See Scroll Down.

Parameter repeat specifies the number of pages. The default is 1.

● **Page Up**      **ESC [ repeat U      {PAGE}**

Move the screen backward with respect to display memory by a specified number of pages. A single Page Up command has exactly the same effect as scrolling up 24 lines. See Scroll Up.

Parameter repeat specifies the number of pages. The default is 1.

**● Set Top of Screen      ESC [ lin ! s**

Sets the top of the screen to a specified line in display memory. Cursor is moved, if necessary to keep it visible, provided that it remains within the window in use by the keyboard.

Parameter `lin` specifies the line of display memory that is to appear at the top of the screen. Line numbering begins at 1 (default), and is relative to the beginning of physical display memory (not the window used by the keyboard). Invalid values (see below) cause the command to be ignored. See Set Display Pages.

Note: There must be at least 23 lines between the line specified and the end of physical display memory; and at least one line of the window used by the keyboard must be within the area displayed. Otherwise the command will be ignored.

**● Define Scrolling Region      ESC [ top;bot r**

Defines the top and bottom limits of the scrolling region.

Parameters:

`top` specifies the first line of the scrolling region. Line numbering begins with 1, and is relative to the window. The default is the beginning of the window. Values greater than the end of the window cause the command to be ignored.

`bot` specifies the last line of the scrolling region. Line numbering begins with 1, and is relative to the window. The default is the end of the window. If `bot` is less than `top`, or greater than the number of lines in the window, the command is ignored.

A number of terminal functions are affected by the definition of the scrolling region, including scrolling, cursor movement, and the insertion and deletion of characters and lines. However, since the scrolling region is by default equal to the window, there is usually no need to worry about scrolling regions. The concept of scrolling region only comes into play when this command has been used to explicitly define a scrolling region that is different from the window.

The scrolling region is automatically redefined to the entire window size when a Define Window command is executed.

Examples:

1. `ESC [ 10;20 r` Defines a scrolling region as lines 10 through 20 of the window.
2. `ESC [ r` Defines the maximum size scrolling region, i.e. equal to the entire window. This is functionally equivalent to not having defined a scrolling region at all.

Note: The DEC VT100 terminal does not allow a 1-line scrolling region to be defined (that is, top must be less than, and not equal to, bot). This restriction does not apply in the Concept AVT.

● **Set Display Pages**      **ESC [ pages ! p**

Determines the amount of memory to allocate to display memory. Pages of memory not allocated to display memory are available to store data for programmable keys (Section 3.12). The number of pages currently allocated to display memory is shown in the Modes Status Line (Appendix B).

Parameter pages specifies the number of pages to allocate to display memory. If pages is 0 or omitted, the maximum allowed by the terminal configuration (4 or 8 pages) is allocated. Values other than 0 - 4 or 0 - 8 (depending on the terminal configuration) cause the the command to be ignored.

Note: This command does an implicit Reset Terminal. As a result, all display memory and communication buffers are cleared, and the Latent Expression is invoked. This has two important implications:

1. Some kind of delay mechanism should be used to allow the terminal time to complete this command before additional data or commands are sent.
2. Since the Latent Expression is invoked, Set Display Pages must not be included in the Latent Expression. The result would be a loop that cannot be broken without disassembling the terminal.

● **Define Window**      **ESC [ top;bot;lft;rgt w**

Determines the dimensions of the current window, that is, the window in use by the requesting device. See Sections 2.1 and 3.2.

Parameters:

top specifies the first line of the window; it defaults to 1, and must not be greater than last line of display memory available. See Set Display Pages.

bot specifies the last line of the window; it defaults to, and must not be greater than, the last line of display memory available. bot must be greater than or equal to top.

lft specifies the left most column of the window; it defaults to 1, and must be less than or equal to the current screen display width (80 or 132).

rgt specifies the right most column of the window; it defaults to, and must not be greater than, the current screen display

width (80 or 132). rgt must be greater than or equal to lft.

Invalid values for any parameters cause the command to be ignored.

Note: Various terminal characteristics are affected by the Define Window command:

1. Start of Print/Transmit is set to (1,1), the home position.
2. The scrolling region is defined to be the entire window.
3. The saved cursor position is set to (1,1), the home position.
4. The status of Cursor Addressing (HDS Mode 206) saved by the Save Cursor command (Section 3.5) is reset to Window-Relative.
5. The cursor is moved to (1,1), the home position in the new window.

Examples:

1. ESC [ w Defines a "maximum" window. All parameters are omitted, so all take on default values.
2. ESC [ 1;24 w Defines a 'one-page' window (24 lines long and 80 or 132 columns wide, depending on the current display width). Only top and bot are specified: lft and rgt default to 1 and screen display width respectively.
3. ESC [ 25;48;41 w Defines a 24 line window, extending from lines 25 to 48 of display memory, and covering only the right portion of the screen (from columns 41 to 80 or 132, depending on screen display width).

### 3.11 Keyboard Controls

#### ● **Lock Keyboard**      **ESC `**

Disables keyboard input. No input of any kind, including control codes, and command sequences, is accepted. Striking any key causes the keyboard bell to ring. Keyboard remains locked until an Unlock Keyboard or Reset Terminal command is received from the keyboard communication line.

Note: The only way to unlock the keyboard from the terminal is to turn the terminal off and then back on again, since the reset key is disabled along with the rest of the keyboard.

#### ● **Unlock Keyboard**      **ESC b**

Enables keyboard input. This command must come from the keyboard communication line if the keyboard is locked. If the keyboard is not locked, it has no effect.

#### ● **Cursor Pad Key Settings**      **ESC [ crsop;crsky;crslv ! z**

Determines the mode of operation of keys in the Cursor Pad. The actual sequence of characters executed and/or transmitted by cursor pad keys depends upon whether Cursor Pad Operation (HDS Mode 201) is Normal (default) or Application. See Cursor Pad Operation table below.

Parameters:

crsop specifies the mode of operation:

0	Execute (default).	2	Transmit and execute.
1	Transmit.	3	Disabled (bell rings, input is ignored).

Other values cause the command to be ignored.






crsky specifies the key(s) affected. The default is 0, which specifies all keys, all levels. See Cursor Pad Operation table for other recognized values. Other values cause the the command to be ignored.

crslv specifies which levels of the keys specified by crsky are affected:

0	All levels, i.e., shifted and unshifted (default).
1	Unshifted only.
2	Shifted only.

Other values cause the command to be ignored.

**Cursor Pad Operation**

LEVEL	crsky/LABEL	FUNCTION EXECUTED	NORMAL	APPLICATION
Both	1 SETUP	Enter Setup Mode	ESC ;	ESC ;
Shifted Unshifted	2 CMD [ 2 CMD	Command Sequence Introducer	ESC [ ESC :	ESC O ESC :
Shifted Unshifted	3 RESET 3 BREAK	Reset Terminal Break	ESC c ESC [ * ~	ESC c ESC O * ~
Both	4 	Home Cursor	ESC   H	ESC O H
Shifted Unshifted	5 PRINT 5	Detach Printer Attach Printer	ESC [ 4 i ESC [ 5 i	ESC O 4 i ESC O 5 i
Shifted Unshifted	6 PRINT 6 SCRN	Print to End of Window Print to Cursor	ESC [ = 8 i ESC [ i	ESC O = 8 i ESC O i
Shifted Unshifted	7 7 PAGE	Page Up Page Down	ESC [ V ESC [ U	ESC O V ESC O U
Both	8 	Cursor Up	ESC [ A	ESC O A
Shifted Unshifted	9 9 SCROL	Scroll Up Scroll Down	ESC [ S ESC [ T	ESC O S ESC O T
Both	10 	Cursor Down	ESC [ B	ESC O B
Both	11 	Cursor Right	ESC [ C	ESC O C
Both	12 	Cursor Left	ESC [ D	ESC O D
Shifted Unshifted	13 13 STATUS	Scroll Status Line Toggle Status Line	ESC [ * p ESC [ * v	ESC O * p ESC O * v
Shifted	14 B TAB	Backward Tab	ESC [ Z	ESC O Z

● **Ring Keyboard Bell**      G

Causes the keyboard bell to ring. Has no effect if Keyboard Bell is Off (HDS Mode 112 reset).



### 3.12 Function Keys and Stored Data

The commands in this section allow the user or programmer to store and manipulate data and/or terminal commands in local (terminal) memory. See the table of Programmable Keys at the end of this Section. There are two ways to program function keys and other stored data fields: The Program Function Key command incorporates the text to be programmed directly into the command; the Display/Edit Function Key/Stored Data command permits use of the display screen to compose and edit the text to be programmed.

The amount of terminal memory available for use in this way depends on the initial terminal configuration (4 or 8 pages), and the amount of memory currently allocated to display (see Set Display Pages). When the maximum allocation is made to display memory (factory default), four page terminals provide 271 characters for programmable functions, and eight page terminals provide 1103 characters. For each page of memory not allocated to display, approximately 3888 characters of memory become available for programmable functions (for four and eight page terminals alike).

#### ● **Program Function Key**      **ESC [ key;actn u del msg del**

Stores terminal commands and/or text in an area of terminal memory associated with a specific function key. Once stored, the contents may be executed and/or transmitted either by pressing the specified function key, or by issuing an Execute Function Key/Stored Data command (see below).

Parameters:

- key specifies the number of the key to be programmed. Recognized values for key are shown in the table of Programmable Keys at the end of this section. 0, meaning all keys, shifted and unshifted, can be used with actn = 2, 3, or 9 to do "mass reprogramming". Other values cause the command to be ignored. For convenience, the function key itself may be pressed when programming keys F1 - F12 (shifted and unshifted) and PF1 - PF4 from the keyboard.
- actn specifies the type of action to be taken when the key is pressed (either from the keyboard or by a program):
  - 0 Execute sequence specified by msg as if it were keyed in on the keyboard (default).
  - 1 Transmit sequence specified by msg to requesting device, or to keyboard communication device, if function is invoked by pressing the key.

- 2 Execute "default execute" sequence as if it were keyed in on the keyboard.
- 3 Transmit "default transmit" sequence to requesting device, or to keyboard communication device, if function is invoked by pressing the key.
- 9 Perform function corresponding to power-up state (that is, default execute or default transmit, as specified in Programmable Keys table).

Note: Only values 0 and 1 allow the user to enter text to be stored. Values 2,3 and 9 specify which of the alternate pre-programmed functions to invoke ( default execute or default transmit).

`del` is a delimiter character used to mark the beginning and the end of the sequence to be stored (used only when `actn` is 0 or 1). Any ASCII character may be used, provided that it does not occur in the string of characters being stored. This is an exception to the rule that parameters must be numeric.

Note: Beginning and ending delimiters must match: the terminal will continue to interpret all input as part of the text to be stored until the ending delimiter is encountered, or no more memory is available for programmable functions. Mistyping one of the delimiters can lead to a situation in which the terminal appears to be "hung", since it interprets input as part of the text to be stored, and does not respond. ↑X will not cancel the command at this point, so the only way to recover is to reset the terminal.

`msg` is the actual string of characters to be stored (used only when `actn` is 0 or 1). This is an exception to the rule that parameters must be numeric.

Note: When programming the key to execute (i.e., when `actn` = 0), ESC is interpreted as CMD, and ESC ESC is interpreted as ESC.

#### Examples:

1. ESC [ 7;0 u /Example 1/ Programs the key labeled F7 on top to "execute" the text 'Example 1'. Striking F7 now has the same effect as keying in 'Example 1'. Slash (/) is the delimiter.
2. ESC [ 0;2 u Programs ALL programmable keys to transmit their default transmit sequences to the keyboard communication line (see table of Programmable Keys at the end of this chapter).

3. ESC [ 31;0 u /ESC [ 2;9 ! w/ Programs shifted F1 to execute the Select Window command ESC [ 2;9 ! w . Striking shifted F1 now has the same effect as typing the 7 key sequence CMD [ 2 ; 9 ! w (it selects Window 2 for the keyboard).

● **Program Alert Line Message**      **ESC [ 97 u del alert del**

Stores text to be displayed on the Alert Message Status Line in response to the appropriate Execute Function Key/Stored Data command (see below).

Parameters:

**del** is a delimiter used to mark the beginning and end of the text to be stored. It may be any ASCII character that does not occur in the text to be stored. This is an exception to the rule that parameters must be numeric. See note on this parameter under Program Function Key.

**alert** is the actual text to be displayed on the Alert Status Line. A maximum of 40 characters may be used. This is an exception to the rule that parameters must be numeric.

● **Program Answerback Message**      **ESC [ 98 u del ans del**

Stores text to be transmitted to the requesting device (the keyboard communication device, if the request comes from the keyboard) in response to the Transmit Answerback Message command (i.e., ↑E).

Parameters:

**del** is a delimiter used to mark the beginning and end of the text to be stored. It may be any ASCII character that does not occur in the text to be stored. This is an exception to the rule that parameters must be numeric. See note on this parameter under Program Function Key.

**ans** is the actual text to be transmitted. A maximum of 20 characters may be used. This is an exception to the rule that parameters must be numeric.

● **Program Latent Expression**      **ESC [ 99 u del latexp del**

Stores a command sequence to be invoked automatically whenever the terminal is powered up or reset (explicitly or implicitly).

Parameters:

`del` is a delimiter used to mark the beginning and end of the text to be stored. It may be any ASCII character that does not occur in the text to be stored. This is an exception to the rule that parameters must be numeric. See note on this parameter under Program Function Key.

`latexp` is the actual sequence of be invoked. Strike the ESC key (Main Pad) once to enter the ESC at the beginning of a terminal command; strike the ESC key twice to enter the escape message character. A maximum of 80 characters may be used. This is an exception to the rule that parameters must be numeric.

Note: since the Latent Expression is invoked automatically, it should not contain any command that resets the terminal, either implicitly or explicitly. If any such command is included in the Latent Expression, the terminal will go into an endless loop the next time it is reset (implicitly or explicitly) or powered up. The only way to break out of this loop is to disassemble the terminal and physically reset Non-Volatile Memory. Commands that reset the terminal are Reset Terminal (ESC c), Reset Terminal/NVM to Factory Defaults (ESC [ 9 ~), and Set Display Pages (ESC [ pages ! p).

### ● Display/Edit Function Key      ESC [ field;oper v

Enables the user to edit and program the contents of function keys and other stored data fields.

Parameters:

`field` specifies the function key or stored data field to be edited. Allowable values are the key numbers shown in the table of Programmable Keys, plus:

98 Answerback message.      99 Latent Expression.

Note: The actual function key may be used to specify function keys F1 - F12 (shifted and unshifted) and PF1 - PF4.

`oper` specifies the operation to be performed:

0 Display stored data (default). Data is displayed in the following format: | m text | where m may have one of four values:

X Execute a programmed sequence.  
 T Transmit a programmed sequence.  
 x Execute a default sequence.  
 t Transmit a default sequence.

The message is displayed as if the terminal were in Transparent Mode, so all control characters are displayed (but the terminal is not actually in Transparent Mode). When field specifies a programmable key, an Alert Message Status Line appears indicating the key being displayed, and the amount of memory available for programming keys.

- 1 Setup for edit. Clears the screen of all characters, protected and unprotected, and displays the stored data as described in the previous paragraph, except that the terminal is actually in Transparent Mode (HDS Mode 3 set).

Note: The terminal remains in Transparent Mode, which facilitates editing programmed text. User or program must explicitly take the terminal out of transparent mode (ESC R).

- 2 Program from screen. This allows programming of a function key, the latent expression, or the answerback message from the screen. The contents of the key or message must be displayed on the screen in the following format:

```
del opr msg del
```

del is any character that does not appear in the data to be stored.

opr (used only for programming function keys) is either X (execute) or T (transmit). If any other character is encountered, execute is assumed, and the character is included as part of the key message. Strike the ESC key (Main Pad) once to enter the ESC at the beginning of a terminal command; strike ESC twice to enter the escape message character.

msg is the actual sequence of character to be stored. Control characters must be entered in Transparent Mode.

The cursor must be positioned at the first delimiter when the command is entered.

● **Execute Function Key/Stored Data**      **ESC [ field ! x**

Causes the execution and/or transmission (as appropriate) of a function key, the latent expression, or the answerback message.

Parameter field specifies the stored data field to invoke. Recognized values include the key numbers shown in the table of Programmable Keys (at the end of this section), plus:

- 98 Answerback message. The answerback message is transmitted to the requesting device, or to the keyboard communication device if the command came from the keyboard.
- 99 Latent Expression. The latent expression is treated as if it had been entered from the keyboard or received from a communication line, depending on its source.

The effect of using this command to invoke a programmable key depends on the source of the command as well as how the key was programmed to operate. If the command comes from the keyboard, the effect is same as if the function key were pressed. If the requesting device is other than the keyboard, and the key was programmed to transmit, the stored data associated with the key are sent to the requesting device. If the requesting device is other than the keyboard, and the key was programmed to execute, the contents of the key are treated as though received from the requesting device.

● **Program Numeric Pad - Application Mode**      **ESC =**

Programs all keys in the Numeric Pad to transmit Default Transmit sequences. See the table of Programmable Keys below.

● **Program Numeric Pad - Numeric Mode**      **ESC >**

Programs all keys in the Numeric Pad to their power-up states. See the table of Programmable Keys below.

**Programmable Keys****Unshifted Function Keys:**

KEY LABEL TOP (FRONT)	KEY NUMBER	POWER-UP STATE	DEFAULT EXECUTE	DEFAULT TRANSMIT
F1 (INSERT)	001	Execute	Toggle insert mode	FKD 001 EOM
F2 (DEL CHAR)	002	"	Delete character	FKD 002 EOM
F3 (INS L)	003	"	Insert line	FKD 003 EOM
F4 (ERAS)	004	"	Erase line	FKD 004 EOM
F5 (SEND)	005	"	Send	FKD 005 EOM
F6	006	Transmit	Nothing	FKD 006 EOM
F7	007	"	"	FKD 007 EOM
F8	008	"	"	FKD 008 EOM
F9	009	"	"	FKD 009 EOM
F10	010	"	"	FKD 010 EOM
F11	011	"	"	FKD 011 EOM
F12	012	"	"	FKD 012 EOM

**Shifted Function Keys**

F1 (INSRT)	031	Execute	Toggle insert mode	FKD 031 EOM
F2 (DEL CHAR)	032	"	Delete character	FKD 032 EOM
F3 (DEL LINE)	033	"	Delete line	FKD 033 EOM
F4 (ERASE)	034	"	Erase window	FKD 034 EOM
F5 (SEND)	035	"	Send	FKD 035 EOM
F6	036	Transmit	Nothing	FKD 036 EOM
F7	037	"	"	FKD 037 EOM
F8	038	"	"	FKD 038 EOM
F9	039	"	"	FKD 039 EOM
F10	040	"	"	FKD 040 EOM
F11	041	"	"	FKD 041 EOM
F12	042	"	"	FKD 042 EOM

Note: FKD = Function Key Identifier (default: ↑\, ASCII chart location 028). EOM = End of Message (default: ↑M, ASCII chart location 013).

**Programmable Keys (continued)****Numeric Keypad Keys \***

KEY LABEL	KEY NUMBER	POWER-UP STATE	DEFAULT EXECUTE	DEFAULT TRANSMIT
PF1	101	Transmit	P	ESC O P
PF2	102	"	Q	ESC O Q
PF3	103	"	R	ESC O R
PF4	104	"	S	ESC O S
ENTER	105	Execute	Carriage return	ESC O M
, (comma)	106	"	comma	ESC O l
TAB	107	"	Horizontal tab	ESC O N
- (dash)	108	"	dash	ESC O m
. (period)	109	"	period	ESC O n
0	110	"	0	ESC O p
1	111	"	1	ESC O q
2	112	"	2	ESC O r
3	113	"	3	ESC O s
4	114	"	4	ESC O t
5	115	"	5	ESC O u
6	116	"	6	ESC O v
7	117	"	7	ESC O w
8	118	"	8	ESC O x
9	119	"	9	ESC O y

**\* Notes on Numeric Pad keys:**

1. Keys in the Numeric Pad are NOT functionally equivalent to number keys in the Main Pad. Number keys in the Main Pad have different shifted and unshifted values, and are not programmable (and therefore do not operate in both execute and transmit modes).
2. The operation of keys in the Numeric Pad depends upon the Program Numeric Pad commands as well as the Program Function Key command (see Function Keys, Stored Data section of Command Summary):
  - a. Individual Numeric Pad keys are fully programmable via the Program Function Key command.
  - b. The Program Numeric Pad - Application command (ESC =) reprograms all Numeric Pad keys to Default Transmit operation.
  - c. The Program Numeric Pad - Numeric command (ESC >) reprograms all Numeric Pad keys to Power-up State.



## Alternate Processing Modes

### 4.1 Introduction

With regard to a number of terminal operating characteristics, it is possible to choose between two alternatives: SET and RESET. There are three groups of such characteristics: ANSI defined modes, HDS private modes, and DEC private modes. For each group, there is a single SET command that selects the SET state of one or more modes, and a single RESET command that selects the RESET state of one or more modes. All of these commands use a parameter string to specify the feature(s) to be set or reset. Any one of these commands may contain as many as 15 parameters -separated by semicolons (;) -- as long as all of the modes referenced belong to the same group. For convenience, the HDS group includes equivalents for all ANSI and DEC modes.

### 4.2 Set and Reset Mode Commands

	ANSI Group	DEC Group	HDS Group
Set	ESC [ ansi h	ESC [ ? dec h	ESC [ = hds h
Reset	ESC [ ansi l	ESC [ ? dec l	ESC [ = hds l

Note: (1) The final character in each of the RESET commands is the lower case letter 'l' (ASCII chart location 108).

(2) ansi represents a string of ANSI mode numbers; dec represents a string of DEC mode numbers; hds represents a string of HDS mode numbers.

### 4.3 Defaults

The factory default is indicated for each mode. Factory defaults have been chosen in the interests of VT100 compatibility. Since the settings of these modes is stored in Non-Volatile Memory, the user is free to choose the most convenient combination as the "power-up" defaults. See the Save Configuration command in section 3.2. The current status of each mode is shown on the Modes Status Line (see Appendix B.3). Reset state is indicated by 0, and set state is indicated by 1.

#### 4.4 Examples

1. ESC [ 20 h (or ESC [ = 20 h ) Selects the Set state of ANSI Mode 20 (or HDS Mode 20, which is equivalent).
2. ESC [ = 16 ; 123 h Selects the Set state of HDS Modes 16 and 123. Notice that the equal sign is required because 123 is a HDS private parameter (even though there is an equivalent ANSI Mode for 16).
3. ESC [ ? 2 1 (or ESC [ = 202 1 ) Selects the Reset state for DEC Mode 2 (or HDS Mode 202, which is equivalent).

#### 4.5 Listing of Alternate Processing Modes

HDS Mode numbers are listed at the left; equivalent ANSI and DEC mode numbers, where available, are given in parentheses following the name of the mode. An asterisk ( \* ) following the mode name indicates that the mode can be changed in Setup Mode (see sections 1.2 and 1.9).

##### 1 Transmit Unprotected/All (ANSI 1)

Reset: Transmit only unprotected characters (default).

Set: Transmit all characters, whether protected or not.

##### 2 Keyboard Lock (ANSI 2)

Reset: Keyboard not locked; keyboard input permitted (default).

Set: Keyboard locked; no keyboard input accepted. Striking any key causes the bell to ring. Command to reset this mode must come from the host; otherwise terminal must be turned off and turned on again.

##### 3 Transparent/Control Code Processing (ANSI 3)

Reset: Control codes executed normally (default).

Set: Transparent Mode: control codes displayed but not executed.

##### 4 Replace/Insert Characters (ANSI 4)

Reset: Replacement Mode (default): existing characters are replaced by new character input.

Set: Insert Mode: existing characters are shifted forward as new characters are input. See Editing section.

##### 6 Erase Protected Characters (ANSI 6)

Reset: Protected characters not affected by Erase commands.

Set: Protected characters erased along with unprotected characters (default).

**12 Full/Half Duplex \* (ANSI 12)**

Reset: Half Duplex: keyboard input sent to the screen.

Set: Full Duplex (default): keyboard input not sent to the screen.

**16 Transmit Termination (ANSI 16)**

Reset: Transmission ends at the end of the window, line, or field depending on the current Transmit Extent (default). See Set Transmit Extent command, Section 3.9.

Set: Transmission ends at current cursor position.

**20 Line feed Processing (ANSI 20)**

Reset: Line feed (default).

Set: New Line (see Line Feed command in Cursor Controls section).

**101 ASCII/APL \***

Reset: ASCII character set with character overstrike off (default).

Set: APL character set with character overstrike on.

**102 Character Overstrike**

Reset: Character Overstrike off (default).

Set: Character Overstrike on.

**103 Display Width \***

Reset: 80 Columns (default).

Set: 132 Columns.

Note: data currently displayed is preserved when switching from 80 to 132.

**104 Scrolling (Line Feed)**

Reset: Window (or Scrolling Region) scrolls in response to Line Feed New Line, and Index commands issued when cursor is on the last line of the window (or Scrolling Region), or to a Reverse Index issued when the cursor is on the first line (default). See Cursor Addressing, HDS Mode 206.

Set: Window (or Scrolling Region) does not scroll.

**105 Tab Processing**

Reset: Text Mode (default): Tab command moves cursor to next text tab stop.

Set: Form Mode: Tab command moves cursor to next field, that is, unprotected character. See Tab command in section 3.5.

**106 Auto Tabs**

Reset: Auto Tabs off (default).

Set: Auto Tabs on. When the last character before a protected field is received, a Tab command is automatically executed, moving the cursor to the beginning of the next unprotected field.

**107 Cursor Wraparound \***

Reset: Cursor Wraparound off (default). Cursor movement commands do not move the cursor past the boundaries of the window or Scrolling Region.

Set: Cursor Wraparound on. Cursor movement commands cause the cursor to "wrap around" the edges of the window (or scrolling region, depending on Cursor Addressing, HDS 206).

**108 Protected Field Overwrite**

Reset: Not Allowed (default). Attempts to overwrite protected characters are ignored. When they come from the keyboard, the bell rings.

Set: Allowed. Protected characters can be overwritten, deleted and shifted as the result of character insertion.

**109 Caps Lock \***

Reset: Off (default). Characters are displayed exactly as they are entered or received, in upper case or lower case.

Set: On. Lower case alphabetic characters entered from the keyboard are converted to upper case for display and transmission.

Note: Characters occurring in terminal command sequences are not affected, nor are non-alphabets.

**110 Character/Block Transmit \***

Reset: Character Mode (default). Keyboard input is transmitted to the keyboard communication line as it is entered.

Set: Block Mode. Keyboard input is displayed as it is entered, but only transmitted in response to a Transmit command (see section 3.9).

Note: This mode has no affect when terminal is in Local Mode (HDS Mode 111 set).

**111 Remote/Local \***

- Reset:** Remote Mode (default). Keyboard input can be sent to the keyboard communication line; characters received from the keyboard communication line are displayed on the screen. Keyboard input is sent to the screen only when in Half Duplex (HDS Mode 12 reset).
- Set:** Local Mode. Keyboard input is displayed but not sent to the keyboard communication line. Terminal commands received from the keyboard communication line are executed, but no data received from the keyboard communication line is displayed. Most commands requiring transmission are ignored.

**112 Keyboard Bell**

- Reset:** Keyboard Bell Off. Bell only rings to signal error conditions (such as aborted terminal commands, keyboard lock, etc).
- Set:** Keyboard Bell On (default). Bell rings normally, i.e. in response to ^G, etc., as well as in response to error conditions.

**113 Alert Line Display**

- Reset:** Automatic (default). Alert line is displayed automatically when the keyboard is locked, when a self test is requested, when the self test at power-up or reset fails, and when using the Display/Edit Function Keys/Stored Data command, as well as in response to a Display Status Line command specifying the alert line (see Section 3.4).
- Set:** On request. Alert line displayed only when specifically requested by a Display Status Line command.

**114 Trailing Spaces on Output**

- Reset:** Transmit. Trailing spaces are treated as ordinary characters.
- Set:** Suppress (default). Spaces that would come at the end of a block transmission (data transmitted in response to a Transmit or Print/Line 3 Control command) are not transmitted.

**115 Underline Attribute Output Processing**

- Reset:** Transmit (default). When a character with the underline attribute is transmitted by the terminal, it results in the sequence <character> <backspace> <underline>.
- Set:** Suppress. The underline attribute is ignored when transmitting characters from the terminal.

**116 Transfer Initiation**

**Reset:** Beginning of window (default). Transmission starts at the beginning of the window in response to Transmit and Print/Line 3 Control commands.

**Set:** Start of Print/Transmit. Transmission begins at the Start of of Print/Transmit for the window. See Start of Print/Transmit command.

**117 Form Feed Prior to Print**

**Reset:** Off. No form feed generated automatically by Print commands.

**Set:** On (default). A form feed (^L) is generated by the terminal when a print command is issued, either from the keyboard (using the PRINT SCRN key) or by the host computer.

**118 Protected Field Display**

**Reset:** As specified (default). Protected characters displayed according to their display attributes.

**Set:** Half-bright. Protected characters are displayed as half-bright regardless of their specified brightness.

**119 Cursor Representation \***

**Reset:** Flashing underline.

**Set:** Flashing block (default).

**120 Character/Attribute Replacement**

**Reset:** Both (default). When a character is replaced, the new character takes on the current attributes.

**Set:** Character only. When a character is replaced, the new character retains the attributes of the character replaced. See Section 3.7.1.

**121 ASCII Underline on Input**

**Reset:** Attribute. When an underline is encountered, the underline attribute (4) is turned on for the character position where the underline was typed. Note: this applies only when the character set in use is ASCII.

**Set:** Character (default). The underline character is treated as an normal ASCII character, so it replaces the character over which it is typed.

**122 Form Feed Processing**

Reset: Clear screen. Move cursor to Home position, and then clear to end of window (or scrolling region, depending on Cursor Addressing (HDS Mode 206)).

Set: Index (default). Execute an Index (line feed).

**123 Terminal Identifier**

Reset: VT100 (default). Terminal identifies itself as a VT100 to requesting device. Response is: ESC [ ? 1;2 c

Set: AVT. Terminal identifies itself as an AVT to requesting device. Response is ESC [ = 1;1 c for 4 page terminals, and ESC [ = 1;2 c for 8 page terminals.

**201 Cursor Pad Operation \* (DEC 1)**

Reset: Normal (default). Cursor Pad keys transmit normal sequences when in transmit mode.

Set: Application. Cursor Pad keys transmit special application sequences when in transmit mode. See Cursor Pad Operation table in section 3.11.

**202 ANSI/VT52 Compatibility \* (DEC 2)**

Reset VT52. Terminal responds to and generates command sequences as if it were a Digital Equipment Company VT52 terminal. See Appendix E.

Set: ANSI (default). Terminal responds to and generates ANSI standard sequences.

Note: Once the terminal is in VT52 mode, it does not respond to any ANSI standard commands, so the Reset Mode command cannot be used to return to ANSI mode. The only way to do so is to use the VT52 command Return to ANSI Mode ( ESC < ).

**203 Display Width (DEC 3)**

Reset: 80 columns (default).

Set: 132 columns.

Note: All data is lost when this mode is used to switch between display widths. Also, all windows are redefined to be 24 lines. This mode is provided solely for VT100 compatibility. HDS Mode 103 allows display width to be changed non-destructively and is recommended.

**205 Screen Video \* (DEC 5)**

Reset: Light characters on a dark background (default).

Set: Dark characters on a light background.

Note: This is a global characteristic of the terminal, and should not be confused with character attribute 7 (Normal/Inverse Video). The setting of character attribute 7 determines how characters are to be displayed relative to screen video: Normal (default) means "same as the screen"; Inverse (alternate) means "opposite to the screen". For example, if HDS Mode 205 is Reset, Inverse Video (attribute 7 set to alternate) means dark characters on a light background. But if HDS Mode 205 is set, Inverse Video means light characters on a dark background. See Section 1.9.

**206 Cursor Addressing (DEC 6)**

Reset: Window-relative (default). All commands involving cursor addressing interpret the line and column positions of the cursor as relative to the window. See section 1.5.

Set: Scrolling Region-relative. All commands involving cursor addressing interpret the line and column positions of the cursor as relative to the Scrolling Region. See section 1.5.

Note: Since the Scrolling Region is by default equal to the window, this mode is only of concern when the Scrolling Region has been explicitly defined as different from the window. See the Define Scrolling Region command (section 1.2). Scrolling Regions provide limited functionality by comparison to windows, and are only supported to provide VT100 compatibility.

**207 Character Wraparound \* (DEC 7)**

Reset: Off (default). Character input does not wrap around the end of a line. When the cursor reaches the end of a line, subsequent characters replace the last character on the line, and the cursor remains where it is.

Set: On. Character input wraps around the end of a line. When the cursor is at the rightmost column of a line, the next character typed replaces the rightmost character in the line, and the cursor automatically moves to the first position of the next line. When the cursor is at the end of the window, it wraps around to the home position (line 1 column 1); and when it is at the home position, it wraps around to the end of the window.



# APPENDIX A

## Character Sets

### A.1 ASCII (80 Columns)

000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
N	A	X	X	F	E	R	P	B	H	L	Y	F	G	O	S
016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
E	R	2	B	4	K	N	E	N	E	B	E	E	B	B	B
032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
.	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
p	q	r	s	t	u	v	w	x	y	z	{		}	~	⊗



**A.3 APL (80 Columns)**

000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
0	1	2	3	4	5	6	7	8	9	:	:	R	n	≡	.
016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
0	1	2	3	4	5	6	7	8	9	:	:	R	n	≡	.
032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
	⋄	)	<	⌊	⌋	⌈	⌉	∧	∨	∗	÷	.	+	.	/
048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
0	1	2	3	4	5	6	7	8	9	(	[	,	x	:	\
064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
⋄	α	⊥	∩	∪	ε	⋄	▽	△	⊙	⊖	⊕	⊗	⊘	⊙	⊚
080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
*	?	ρ	⌈	⌊	⌋	⌈	⌉	⊖	⊕	⊗	⊘	⊙	⊚	⊛	⊜
096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
⊜	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
P	Q	R	S	T	U	V	W	X	Y	Z	{	⌈	⌋	⊙	⊚



**A.5 VT100 and Concept Special Graphics (132 Columns)**

000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127



## APPENDIX B

### Status Lines

#### B.1 User Status Line\*

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
KB	50	HDX	1S	NO	REM	BLK	U/L	lin: col	ASC	top: bot: lft: rgt	ANS	S	C	V	a	CT	VPPPPP	INS/OFF

START/ REF	LENGTH	DESCRIPTON	EXPLANATION
A	2 2	Status line identifier	KB = keyboard; L1 - L3 = Lines 1 - 3.
B	5 4	Baud rate	See Section 1.6.1.
C	10 3	Duplex	HDX = half duplex; FDX = full duplex.
D	14 2	Stop bits	1S = 1 stop bit; 2S = 2 stop bits.
E	17 2	Parity	NO = none; EV = even; OD = odd;
F	20 3	Local/remote	MK = mark(1); SP = space (0).
G	24 3	Block/character mode	LOC = local; REM = remote.
H	28 3	Caps lock	BLK = block; CHR = character.
I	32 7	Cursor position	U/L = upper & lower; CAP = caps only.
J	40 3	Char. set (ASCII/APL)	lin = line #; col = column #.
K	44 15	Window definition	APL and values beginning with 0 have Character Overstrike on.
L	60 3	ANSI/VT52 mode	top = top; bot = bottom; lft = left; rgt = right.
M	64 1	Screen width	ANS = ANSI; V52 = VT52 mode.
N	66 1	Cursor representation	Place holder only (see Section 1.9).
O	68 1	Video/reverse video	"
P	70 1	Wraparound	"
Q	72 2	Cursor keypad operation	A = Character and cursor wrap on; a = Character or cursor wrap off.
R	75 6	Software version	CT = one or more keys not in execute mode; CE = all cursor pad keys in execute mode (see Section 1.9).
S	82 7	Insert mode indicator	V = Product ID; P's = PROM version numbers. Only visible when Screen Width is 132.

\* All fields except I, R, and S can be changed in Setup Mode. See Section 1.9.

B.2 Programmer Status Line

AB	CDE	F	G	H	I	J	K	L	M	N	O	P	Q
K1	B(B 1-atalt m			1-top; bot; lft; rgt w		top; bot r	lin; col ? . o/p t	lin; col H	dly	lin	M	atusd; atalt; chr ! r	

START/REF	LENGTH	DESCRIPTION	EXPLANATION	
A	2	1	Line identifier	K = keyboard; 1-3 = Lines 1-3.
B	3	1	Keyboard Comm. Device	1-3 = Lines 1-3.
C	5	1	Normal character set	See Section 3.3 for values.
D	6	1	Character set in use	Indicated by direction of arrow.
E	7	1	Alternate character set	See Section 3.3 for values.
F	9	1	Attribute list selected	1 - 4
G	11	13	Alternate attribute(s)	atalt = attributes on list that are in alternate state (see Section 3.7); m = Select Alternate Attributes command identifier.
H	25	1	Window number	1 - 4.
I	27	16	Window definition	top = top line; bot = bottom line; lft = left column; rgt = right column; w = Define Window command identifier.
J	45	8	Scrolling region	top = top line; bot = bottom line; r = Define Scrolling region command identifier.
K	54	8	Start of print/transmit	lin = line; col = column position; ? = Set Start of Print/Transmit command identifier.
L	64	8	Output network	o/p = device(s) to which data is sent; t = Set Output Network command identifier (see Section 3.8).
M	73	8	Cursor position	lin = line; col = column position; H = Position Cursor command identifier.
N	85	3	Transmission delay	dly = number of 100 millisecond units.
O	90	3	Background status line	lin = line number.
P	95	1	Display memory	M = number of pages (4 or 8).
Q	98	29	Clear characteristics	atusd = attributes on clear list; atalt = attributes on clear list set to alternate state; chr = clear character; !r = Clear Characteristics command identifier (Section 3.6).



## B.3 Modes Status Line

ID	a-123	b-12345	c-1234	d-123	e	f-123456	g-123 4	h-123456	i-12345	j
KB	a-10	b-00000	c-0000	d-001	e-027	f-000010	g-01101	h-11010	i-00008	j-001

START/ REF	LENGTH	DESCRIPTION	EXPLANATION	HDS MODE	
ID	2	2	Line identifier	KB L1 L2 L3	
a-1	8	1	Buffer overflow (receive) control	0 = off; 1 = on.	
a-2	9	1	CTS/RTS (transmit) protocol	0 = off; 1 = on.	
a-3	10	1	Reserved		
b-1	14	1	Tab processing	0 = text; 1 = form;	105
b-2	15	1	Scrolling	0 = on; 1 = off	104
b-3	16	1	Protected field overwrite	0 = not allowed; 1 = allowed.	108
b-4	17	1	Char./attribute replacement	0 = both; 1 = character only.	120
b-5	18	1	Protected field display	0 = normal; 1 = bold.	118
c-1	22	1	Cursor addressing	0 = window relative; 1 = scroll. region relative.	206
c-2	23	1	Numeric pad operation	0 = numeric; 1 = application.	
c-3	24	1	Cursor keypad operation	0 = normal; 1 = application.	201
c-4	25	1	Terminal identifier	0 = VT100; 1 = AVT; 2 = GVT	123
c-5	29	1	Transparent (control code processing)	0 = take action; 1 = display.	3
d-1	30	1	Parity checking	0 = off; 1 = on.	
d-2	31	1	Formfeed processing	0 = clear screen; 1 = index. (line feed)	122
e	35	3	Escape message character	ASCII chart location.	

## Modes Status Line (cont'd)

REF	START/ LENGTH	DESCRIPTION	EXPLANATION	HDS MODE
f-1	41 1	Character wraparound	0 = off; 1 = on.	207
f-2	42 1	Cursor wraparound	0 = off; 1 = on.	107
f-3	43 1	Linefeed processing	0 = linefeed; 1 = new line (line feed/CR)	20
f-4	44 1	Autotabs	0 = off; 1 = on.	106
f-5	45 1	Keyboard bell	0 = on; 1 = off.	112
f-6	46 1	Replace/insert characters	0 = replace; 1 = insert.	4
g-1	50 1	Character overstrike	0 = off; 1 = on.	102
g-2	51 1	ASCII underline method	0 = attribute; 1 = character.	121
g-3	52 2	Editing extent	00 = window; 01 = field/wind. 10 = line; 11 = field/line	
g-4	54 1	Erase protected characters	0 = no; 1 = yes.	6
h-1	58 1	Transmit unprotected/all	0 = unprot. only; 1 = all.	1
h-2	59 1	Trailing blank transmission	0 = transmit; 1 = suppress.	114
h-3	60 1	Underline attribute processing	0 = transmit; 1 = suppress.	115
h-4	61 1	Formfeed prior to print	0 = on; 1 = off.	117
h-5	62 1	Transfer extent	0 = window; 1 = line; 3 = field.	
i-1	66 1	Transfer termination	0 = end of field, line or window; 1 = cursor position.	16
i-2	68 1	Transfer initiation	0 = start of print/transmit; 1 = beginning of window.	116
i-3	69 1	Display width (columns)	0 = 80; 1 = 132.	103
i-4	70 1	Alert line display	0 = automatic; 1 = on request	113
i-5	71 1	Display memory (pages)		
j	75 3	Start of screen	Line number of top of screen	

#### B.4 Programmable Message Character Status Line

SOM = ***	FKD = 028	CPM = ***	DLY = ***	EOF = 023; ***	EOL = 013; ***	EOM = 013; ***
-----------	-----------	-----------	-----------	----------------	----------------	----------------

- See Change Message Character command in Section 3.2 for a table of programmable message characters
- Numbers represent ASCII chart location of character.
- \*\*\* indicates that Message Character is not used.
- Values shown above are Factory Defaults.

#### B.5 Tabs Status Line

1234.678901.3456789.12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
--

Period indicates position of text tab stop. In above example, tabs are set at the fifth, 12th, and 20th columns. (The Factory Default is tab stops every eight columns.)

#### B.6 Alert Message Status Line

A	B	C	D	E
LOCK	error	K: key	BLK: blk, chs CHAR	Alert Line Message

REF	DESCRIPTION	EXPLANATION
A	Keyboard Lock	Blank if Unlocked.
B	Self-test result	NO ERROR or error codes 1 - 8 (see Section 3.2 on Self Test).
C	Key being programmed	See Display/Edit Function Key/Stored Data command, Section 3.12.
D	Programmable memory available	blk = number of blocks (256 chrs/block); characters. chrs = number of characters.
E	Alert line message	Actual text of message currently defined (if any).



## APPENDIX C

### Summary of Control Codes and Escape Sequences

The tables below summarize in numerical order the control code and escape sequence functions. 'LOCATION' is the ASCII chart location of the control code or command identifier; 'CHAR' is the equivalent ASCII character sequence; 'NAME' is the name of the function; and 'PAGE' is the number of the page in Chapter 3 where the command is formally defined.

#### C.1 Control Codes

LOCATION	CHAR	NAME	PAGE
000	↑@	Fill Character	3-8
001	↑A		
002	↑B		
003	↑C		
004	↑D		
005	↑E	Trans. Answerback Msg.	3-14
006	↑F		
007	↑G	Ring Keyboard Bell	3-48
008	↑H	Backspace	3-17
009	↑I	Tab	3-19
010	↑J	Line Feed	3-15
011	↑K	Vertical Tab	3-15
012	↑L	Form Feed	3-23
013	↑M	Carriage Return	3-15
014	↑N	Use Alt. Char. Set (APL)	3-10
015	↑O	Use Nor. Char. Set (ASC)	3-10
016	↑P		
017	↑Q	Resume Transmission	3-40
018	↑R		
019	↑S	Stop Transmission	3-40
020	↑T		
021	↑U		
022	↑V		
023	↑W		
024	↑X		
025	↑Y		
026	↑Z		
027	↑[		
028	↑\		
029	↑]		
030	↑^		
031	↑_		
127	RUB	Fill Character	3-8

\* Illegal/Reserved (ANSI)

## C.2 ESC Followed by a Single Character

LOCATION	CHAR	NAME	PAGE
000	↑ @		
⋮		*	
031	↑ _		
032	sp		
033	!		
034	"		
035	#		
036	\$		
037			
038	&	*	
039	'	*	
040	(	Def. Norm. Character Set	3-10
041	)	Def/ Alt. Character Set	3-10
042	*	*	
043	+	*	
044	,		
045	-		
046	.	*	
047	/	*	
048	0		
049	1	Toggle Insert	3-23
050	2	End of Text	3-18
051	3		
052	4		
053	5	Transmit	3-40
054	6		
055	7	Save Cursor	3-19
056	8	Restore Cursor	3-19
057	9		
058	:		
059	;		
060	<		
061	=	Program Num. Pad. Appl.	3-54
062	>	Program Num. Pad. Num.	3-54
063	?	Start of Print/Transmit	3-42

\* Illegal/Reserved (ANSI)

## C.2 ESC Followed by a Single Character (cont'd)

LOCATION	CHAR	NAME	PAGE
064	@	*	
065	A	*	
066	B	*	
067	C	*	
068	D	Index (Line Feed)	3-15
069	E	New Line (Line Feed/CR)	3-16
070	F		
071	G		
072	H	Set Tab	3-20
073	I		
074	J		
075	K		
076	L		
077	M	Reverse Index	3-16
078	N		
079	O		
080	P		
081	Q	Transparent Mode On	3-32
082	R	Transparent Mode Off	3-32
083	S		
084	T		
085	U		
086	V	Protection On	3-32
087	W	Protection Off	3-33
088	X	*	
089	Y	*	
090	Z	Transmit Terminal ID	3-6
091	[	Control Seq. Introducer	3-3
092	\		
093	]		
094	^		
095	_		
096	`	Lock Keyboard	3-47
097	a	Halt Terminal	H-1
098	b	Unlock Keyboard	3-47
099	c	Reset Terminal	3-6
100	d		
:	:	*	
127	RUB		

\* Illegal/Reserved (ANSI)

## C.3 Sequences Preceded by ESC [

LOCATION	CHAR	NAME	PAGE
000	↑@		
⋮	⋮	*	
063	?		
064	@		
065	A	Cursor Up	3-16
066	B	Cursor Down	3-16
067	C	Cursor Right	3-16
068	D	Cursor Left	3-17
069	E	Cursor Down-Left Margin	3-17
070	F	Cursor Up-Left Margin	3-17
071	G	Position Cursor-Column	3-17
072	H	Position Cursor	3-17
073	I	Forward Tab	3-19
074	J	Erase in Window	3-25
075	K	Erase in Line	3-26
076	L	Insert Line	3-14
077	M	Delete Line	3-25
078	N		
079	O	Erase in Field	3-26
080	P	Delete Character	3-24
081	Q	Select Editing Extent	3-23
082	R		
083	S	Scroll Up	3-43
084	T	Scroll Down	3-43
085	U	Page Down	3-43
086	V	Page Up	3-43
087	W	Tab Control	3-20
088	X		
089	Y		
090	Z	Backward Tab	3-20
091	[	*	
092	\	*	
093	]	*	
094	^	*	
095	-	*	

\* Illegal/Reserved (ANSI)



## C.3 Sequences Preceded by ESC [ (cont'd)

LOCATION	CHAR	NAME	PAGE
096	`	Pos. Cursor-Col.(Alt.)	3-18
097	a	Cursor Right (Alt.)	3-16
098	b		
099	c	Transmit Terminal ID	3-6
100	d	Position Cursor-Line	3-18
101	e	Cursor Down (Alt.)	3-16
102	f	Position Cursor (Alt.)	3-17
103	g	Clear Tabs	3-20
104	h	Set Mode	4.1
105	i	Print/Line 3 Control	3-34
106	j	*	
107	k	*	
108	l	Reset Mode	4.1
109	m	Select Alt. Attributes	3-29
110	n	Trans. Device Status	3-11
111	o		
112	p	Block Character Change	3-31
113	q		
114	r	Define Scrolling Region	3-44
115	s		
116	t	Set Output Network	3-35
117	u	Program Function Key	3-49
118	v	Display/Edit Function Keys/Stored Data	3-52
119	w	Define Window	3-45
120	x		
121	y	Self Test	3-6
122	z	Keyboard Comm. Device	3-36
123	{		
124			
125	}		
126	~	Save Configuration (NVM)	3-7
127	RUB	*	

\* Illegal/Reserved (ANSI)

## C.4 Sequences Preceded by ESC [ with ! as an Intermediate

LOCATION	CHAR	NAME	PAGE
33 000	!^@		
:		*	
33 111	! o		
33 112	! p	Set Display Pages	3-45
33 113	! q	Block Attribute Change	3-30
33 114	! r	Set Clear Characteristic	3-22
33 115	! s	Set Top of Screen	3-44
33 116	! t	Change Message Character	3-7
33 117	! u	Select Attribute List	3-8
33 118	! v	Copy Attribute List	3-8
33 119	! w	Select Window	3-9
33 120	! x	Execute Function Key/ Stored Data	3-53
33 121	! y	Set Margin Bell	3-26
33 122	! z	Cursor Pad Key Settings	3-47
33 123	! {	Sel. Default Attributes	3-30
33 124	!		
33 125	! }		
33 126	! ~		
33 127	! RUB	*	

\* Illegal/Reserved (ANSI)

## C.5 Sequences Preceded by ESC [ with \* as an Intermediate

LOCATION	CHAR	NAME	PAGE
42 000	*↑@		
⋮		*	
42 111	* o		
42 112	* p	Parity/Parity Checking	3-37
42 113	* q	Buffer Overflow(Receive) Protocol	3-39
42 114	* r	Set Baud Rate	3-37
42 115	* s	Set Stop Bits	3-38
42 116	* t	Transmit Status Line	3-12
42 117	* u	Display Status Line	3-11
42 118	* v	Toggle/Clear Status Line	3-13
42 119	* w	Set Background Stat Line	3-13
42 120	* x	CTS/RTS (Xmit) Protocol	3-38
42 121	* y	Set Transmit Extent	3-40
42 122	* z	Set Transmit Delay	3-41
42 123	* {	Host Overflow Control	H-1
42 124	*		
42 125	* }		
42 126	* ~	Break	3-41
42 127	* RUB	*	

\* Illegal/Reserved (ANSI)



## APPENDIX D

### Communications Interfaces

#### D.1 Basic Communications

The Concept AVT transmits ASCII coded data in asynchronous format. Each character is transmitted serially and is composed of four (4) logical parts:

1. Start bit (one bit, always present)
2. Data bits (seven bits, always present)
3. Parity bit (zero or one bit, depending on settings)
4. Stop bits (one or two bits, depending on settings)

The various possible bit configurations and resultant character 'sizes' are shown in the table below:

Parameters Set		Number of Bits Generated				
Parity	Stop Bits	Start	Data	Parity	Stop	Total
None	1	1	7	0	1	9
None	2	1	7	0	2	10
Odd	1	1	7	1	1	10
Odd	2	1	7	1	2	11
Even	1	1	7	1	1	10
Even	2	1	7	1	2	11
Mark	1	1	7	1	1	10
Mark	2	1	7	1	2	11
Space	1	1	7	1	1	10
Space	2	1	7	1	2	11

Signals are provided which conform to EIA standards for interfacing to data communications equipment. The back panel on the terminal contains two or three 25 pin RS-232-C type connectors. They are male for Lines one (1) and three (3) and female for Line two (2); a six foot straight-through male-female extension cable is provided (for use with Line 1).

The table below lists the pin configuration, EIA circuit name, CCITT V.24 circuit name, and signal descriptions for all three communications interfaces.

## Communications Interfaces Pin Assignments

Line	Pin Number	EIA Circuit	CCITT V.24	Description
1	1	AA	101	Protective Ground
	2	BA	103	Transmitted Data
	3	BB	104	Received Data
	4	CA	105	Request to Send (RTS)
	5	CB	106	Clear to Send (CTS)
	7	AB	102	Signal Ground
	20	CD	108	Data Terminal Ready (DTR)
2	1	AA	101	Protective Ground
	2	BB	104	Received Data
	3	BA	103	Transmitted Data
	4	CB	106	Clear to Send (CTS)
	5	CA	105	Request to Send (RTS)
	6	CC	107	Data Set Ready (DSR)
	7	AB	102	Signal Ground
	8	CF	109	Carrier Detect (CD)
3	1	AA	101	Protective Ground
	2	BA	103	Transmitted Data
	3	BB	104	Received Data
	4	CA	105	Request to Send (RTS)
	5	CB	106	Clear to Send (CTS)
	7	AB	102	Signal Ground
	20	CD	108	Data Terminal Ready (DTR)

## Circuit Description

Protective Ground (AA,101) This conductor is electrically bonded to the machine frame.

Transmitted Data (BA,103) This conductor transmits data from the terminal to a modem, computer interface, or other communications device. The circuit is held in a marking condition during intervals between characters and at all times when no data are being transmitted. If the CTS/RTS is ON for this line, the Clear to Send circuit (CB,106) must be ON for data transmission to occur.

Received Data (BB,104) Signals on this circuit are received input from the modem, computer interface, or other communication device.

Request to Send (CA,105) When ON, this signal, which is generated by the terminal, indicates that the terminal is prepared to transmit data. The Request to Send line is maintained in an ON condition by the terminal if either of these conditions applies:

- The CTS/RTS Protocol is OFF for this line (see Section 3.9).
- This line on the output network of some line or device (see Set Output Network, Section 3.8).

Clear to Send (CB,106) When on, this signal, which is generated by the data communications equipment, indicates the data set is ready to transmit data. If the CTS/RTS Protocol is ON for this line, this signal is required for transmission of data and for successful 'attaching' of this line as an output device (via the Set Output Network, Print/Line 3 Control commands in Section 3.8, for example).

Signal Ground (AB,102) This conductor establishes the common ground reference potential for all interchange circuits. This conductor is internally connected to protective ground (CA,101).

Data Terminal Ready (CD,108) The signal, provided by the terminal, is used to control switching of the data communications equipment to the communications channel. This circuit is held in the ON condition at all times when the terminal is powered up.

Data Set Ready (CC,107) and Carrier Detect (CF,109) These signals, provided by the terminal, are used to control switching of the data communications equipment to the communications channel. These circuits are internally jumpered together and are held in the ON condition at all times when the terminal is powered up.

## D.2 Current Loop (20 Milliamp) Interface Option

When the Concept AVT is ordered with the 20ma current loop option, it can be configured to operate in any of three modes - 20ma active, 20ma passive, of RS-232 compatible. The mode selection is accomplished by setting two sets of DIP switches which are located on the interface option circuit board (mounted on the main circuit board inside the terminal case). The DIP switches are located on the top right hand corner of the circuit board in locations 'S1' and 'S2' (S1 is on top). DIP switch S1 controls the 'receive' line mode and S2 controls the 'transmit' line mode. The following tables specify the switch configurations for the three modes of operation: a '1' indicates the switch; should be on; a '0' indicates the switch should be off; and an 'X' indicates the switch is not used and may be in any position.

### DIP Switch S1 (Receive)

	1	2	3	4	5	6	7	8
RS-232	0	0	0	0	0	0	1	X
20ma Passive	0	1	0	1	0	1	0	X
20ma Active	1	0	1	0	1	1	0	X

## DIP Switch S2 (Transmit)

	1	2	3	4	5	6	7	8
RS-232	0	0	0	0	0	1	X	X
20ma Passive	0	1	0	1	0	0	X	X
20ma Active	1	0	1	0	1	0	X	X

The communication interface connector on terminals equipped with the 20ma current loop option is the standard 25-pin male connector, as defined above. The current loop signals are on the following pins:

Terminal Transmit + Pin 18  
Terminal Transmit - Pin 25  
  
Terminal Receive + Pin 10  
Terminal Receive - Pin 11



## Appendix E

### VT52 Terminal Type Operation

The following information applies only when the terminal has been put in VT52 Mode, which is done by using the Reset Mode command for HDS Mode 202 ( ESC [ = 202 1 ), or DEC Mode 2 ( ESC [ ? 2 1 ).

Note: Once the terminal is in VT52 Mode, it recognizes ONLY the commands listed below. As a result, the ONLY way to return to ANSI Mode is by issuing the Return to ANSI Mode command ( ESC < ) shown below.

#### E.1 Commands Recognized in VT52 Mode

Cursor Up	ESC A
Cursor Down	ESC B
Cursor Right	ESC C
Cursor Left	ESC D
Reverse Line Feed	ESC I
Cursor Home	ESC H
Direct Cursor Address	ESC Y lin col (see note A)
Erase to End of Line	ESC J
Erase to End of Window	ESC K
Select Special Graphics Character Set	ESC F
Select ASCII Character Set	ESC G
Program Numeric Pad to Application Mode	ESC = (see E.2)
Numeric Pad to Normal Mode	ESC > (see E.2)
Identify Terminal Type	ESC Z (response: ESC / Z)
Return to Ansi Mode	ESC <

Note A: lin and col are one- and two-character codes that specify the line and column positions respectively according to their ASCII chart locations as follows (ASCII chart locations shown in parentheses):

1 = ! (33)	95 = ↑A Sp (1 32)	190 = ↑B Sp (2 32)
2 = " (34)	96 = ↑A ! (1 33)	191 = ↑B ! (2 33)
3 = # (35)	97 = ↑A " (1 34)	192 = ↑B " (2 34)
⋮	⋮	
94 = ` (126)	189 = ↑A ` (1 126)	

#### E.2 Numeric Pad - Numeric and Application Modes

The Program Numeric Pad to Application Mode command programs all keys in the numeric pad to Default Transmit operation, as described below. The Program Numeric Pad to Numeric Mode command programs all keys in the numeric pad to Default Execute operation as described below.

Note: Putting the terminal in VT52 Mode does not by itself affect the operation of numeric pad keys. Therefore numeric pad keys may be in any combination of states upon entering VT52 Mode, depending upon whether and how they were programmed while in ANSI Mode. See section 3.12 on programmable keys.






KEY*	DEFAULT EXECUTE	DEFAULT TRANSMIT
PF1	ESC P	ESC P
PF2	ESC Q	ESC Q
PF3	ESC R	ESC R
PF4	ESC S	ESC S
ENTER	<CR>	ESC ? M
, (comma)	,	ESC ? l
TAB	TAB	ESC ? N
- (dash)	-	ESC ? m
.(period)	.	ESC ? n

KEY*	DEFAULT EXECUTE	DEFAULT TRANSMIT
0	0	ESC ? p
1	1	ESC ? q
2	2	ESC ? r
3	3	ESC ? s
4	4	ESC ? t
5	5	ESC ? u
6	6	ESC ? v
7	7	ESC ? w
8	8	ESC ? x
9	9	ESC ? y

\* No difference between shifted and unshifted.

### E.3 VT52 Cursor Keypad Sequences

The following table shows the sequences transmitted by the cursor pad keys when the terminal is in VT52 Mode.

	Key	Sequence	Key	Sequence	Key	Sequence
Shifted	SETUP	ESC ;	CMD [	ESC	RESET	ESC c
Unshifted			CMD	ESC :	BREAK	ESC * ~
Shifted		ESC H	PRINT	ESC 4 i	PRINT	ESC = 8 i
Unshifted				ESC 5 i	SCRN	ESC = i
Shifted	PAGE	ESC V		ESC A	SCROLL	ESC S
Unshifted		ESC U		ESC A		ESC T
Shifted		ESC D		ESC B		ESC C
Unshifted		ESC D		ESC B		ESC C

## **Appendix F**

### **Timing Considerations**

In general, the CONCEPT terminal can process incoming data at a rate which does not require the insertion of fill characters or the execution of program delays. However, there are a few operations which involve moving a significant amount of data around and for which timing considerations are important.

#### **F.1 Fill Characters**

A user's program can either send the specified number of 'fill' characters or delay for the equivalent amount of time. The table on page F-3 shows the approximate number of timing characters required for the most time consuming terminal commands under "worst case" conditions. Since the number of fill characters is proportional to the baud rate, the user may extrapolate for baud rates not listed in the accompanying tables. The column headed '132 Col' gives a rough factor used to determine the corresponding number of fill characters that must be used in when the display width is 132 column.

NUL or ↑@ (ASCII 000) and RUB (ASCII 127) are always treated as fill characters by the terminal. The programmable delay character DLY (not used by default) can also be used for this purpose. Unless the terminal is in Transparent Mode, these characters are completely ignored. When the terminal is in Transparent Mode, fill characters are displayed on the screen. They are never networked.

#### **F.2 Input Buffers**

Each communications line in the terminal has an input buffer (256 characters for Line 1; 128 characters for Lines 2 and 3). This buffer is used to store incoming data if the terminal is 'busy' at the time the character is received. This will happen whenever the terminal is processing a command sequence which takes more than one character time to complete. To a degree, this allows the user to ignore timing considerations, since the timing charts assume a worst case situation in which the input buffer is totally full.

For example, assume we are executing a command from Line 1 that takes 200 character times to complete. Since the input buffer on Line 1 is 256 characters long, approximately 56 more characters could be received without the need for fill characters or delays. However, if 100 more characters were sent, some portion of them would be lost.

It is strongly recommended that all users adhere to the timing requirements stated in this Appendix. A system MAY work without fill characters, but there are no guarantees. A change in the workload on the host computer could cause a new problem which may not have been detected before. Problems arise when the user changes from a 300 baud, 4 page application (where the timing considerations are relatively minimal) to a 1200 baud, 8 page application, which may take 4-8 times as many fill characters depending on the commands used.

### F.3 Buffer Overflow Control

Buffer Overflow Control provides one method for users to ignore timing considerations. If enabled, the terminal will transmit XOFF (↑S, ASCII 019) when an input buffer is half full. When the buffer drops back to one quarter full, XON (↑Q, ASCII 017) is sent. If this protocol is supported by the host computer, the programmer will be able to ignore fill character requirements. This protocol operates on each communications line separately so that multiple computer applications can be supported.

The user should be sure that the particular application being run will ignore XON/XOFF characters transmitted by the CONCEPT terminal. Various text editors, for example, use ↑S and ↑Q as commands. In these cases Buffer Overflow control may be disabled and re-enabled for other applications.

Command/Condition	1200	9600	132 col
Self Test - 4 page terminal	140	1400	1.5
Self Test - 8 page terminal	240	2400	1.5
Reset Terminal, NVM, Set Display Pages - per page	5	40	1.0
Save Configuration (NVM)	3	25	1.5
Set/Reset Alternate Processing Modes	1	5	1.5
Use Alternate/Normal Character Sets	0	1	1.5
Define Alternate/Normal Character Sets	0	1	1.5
APL/ASCII Mode	0	1	1.5
Line Feed, Index, New Line			
- scrolling full-width window, per page scrolled	0	1	2.0
- scrolling non-full-width window, per page scrolled	4	32	2.0
End of Text - per page scanned	6	48	1.5
Tab, Back Tab - text tab processing over 1 line	0	4	2.0
Tab, Back Tab - form tab processing over 1 line	1	8	2.0
Form Feed (Clear Screen) - per page cleared	4	38	2.0
Insert/Delete Characters - per 80 characters moved	1	4	2.0
Insert/Delete Line, Reverse Index			
- full-width 24 line window, repeat=1	0	4	2.0
- full-width 24 line window, repeat=24	10	72	2.0
- full-width 192 line window, repeat=1	1	8	2.0
- full-width 192 line window, repeat=24	18	144	2.0
- non-full-width window, per page scrolled, repeat=1	5	40	2.0
Erase in Window - per page erased	12	96	2.0
Erase in Line - per line cleared	1	6	2.0
Set Clear Characteristics	1	4	2.0
Select Alternate/Default Attributes	1	4	2.0
Block Attribute Change - per page changed	8	64	1.5
Block Character Change - per line changed	1	5	1.5
Set Output Network, Keyboard Comm. Device	1	6	1.5
Define Window, Define Scrolling Region	0	1	2.0
Cursor Pad Key Settings	0	1	2.0
Display Function Keys/Stored Data - per 80 characters	3	24	2.0
Program Numeric Pad - Application/Numeric	2	8	1.5
Display Width Mode Changed (worst case)			
- 4 page terminal	48	512	1.5
- 8 page terminal	144	1536	1.5

Note: For "Insert/Delete Line", multiple values are given to illustrate the timing needed. Note that the timing increases at a rate equal to about 75% of the change in the repeat factor; however, for equal repeat factors, the timing increases at a rate equal to about 33% of the increase in the number of lines moved.



## Appendix G

### Factory Default Settings

To return terminal to factory default state, enter ESC [ 9 ~

Numbers in parentheses following the name of a feature are HDS Mode numbers. The letter in the 'List' column indicates whether the setting of the feature is stored in the attribute (A), device (D), terminal (T), or window (W) list.

Terminal Feature	Factory Default Condition	List
<b>General</b>		
Full/Half Duplex (12)	Full	T
Block/Character Mode (110)	Character	T
Remote/Local Mode (111)	Remote	T
Caps Lock (109)	Off (Upper and lower case)	T
Screen Video (205)	Light characters/dark background	T
ANSI/VT52 Operation (202)	ANSI	T
Terminal Identifier (123)	VT100	T
Alert Line Displayable (109)	On request	T
Background Status Line	None	T
<b>Keyboard</b>		
Keyboard Lock (2)	Unlocked	T
Cursor Keys	↑ , ↓ , → , and ← in Trans. the rest in Execute	T
Cursor Pad Operation (207)	Normal	T
Numeric Pad	Numeric	T
Bell (112)	On	T
<b>Character Set (ASCII/APL)</b>		
Normal Character Set	ASCII	T
Alternate Character Set	ASCII	T
Character Set in Use	Normal	A
ASCII/APL (101)	ASCII	A
Character Overstrike	Off	A
<b>Message Characters</b>		
Escape (ESC)	ASCII 027	A
Start of Message (SOM)	Not used	T
Function Key Identifier (FKD)	ASCII 028	T
Cursor Pad End of Message (CPM)	Not used	T
Delay Character (DLY)	Not used	T
End of Field (EOF)	ASCII 023	T
End of Line (EOL)	ASCII 013	T
End of Message (EOM)	ASCII 013	T

Terminal Feature	Factory Default Condition	List
Attribute List	All devices use List 1	D
Attributes (same for all lists)		
Normal/Bold (1)	Normal	A
Underline (4)	No Underline	A
Blinking (5)	No Blinking	A
Video (7)	Same as Screen Video	A
Displayable/Non-displayable (8)	Displayable	A
Protection (99)	No Protection	A
Protected Field Display (108)	According to attributes	A
Window List	All devices use List 1	D
Display Memory		
Pages allocated to display	All available (4 or 8)	T
Display Width (103,203)	80 columns	T
Start of Screen	Line 1	W
Window Definition (all windows)	Lines 1-24; columns 1-80	W
Scrolling Region	Lines 1-24	W
Cursor		
Cursor Position	Line 1, Column 1 (home)	W
Cursor Addressing (206)	Window relative	W
Cursor Representation (119)	Flashing Block	T
Cursor Wraparound (107)	Off	A
Tab Stops	Every 8 columns (9,17,25...129)	T
Special Character Processing		
Control Code Processing (3)	Execute (Transparent Mode off)	A
Character Wraparound (207)	Off	A
Line Feed Processing (20)	Line Feed	A
Scrolling (Line Feed) (104)	On (Reset)	A
Tab Processing (105)	Text	A
Auto Tabs (106)	Off	A
Replace Char./Attribute (120)	Both	A
ASCII Underline on Input (121)	Character	T
Form Feed Processing (122)	Index (Line Feed)	T
Editing		
Replace/Insert Character (4)	Replace	A
Editing Extent	Line	A
Erase Protected Characters (6)	Yes	A
Protected Field Overwrite (108)	Not allowed	A
Clear Character	Space (ASCII 032)	T
Clear Attributes to use	All from Attribute List	T
Clear Attribute List	All default attributes	T
Margin Bell Offset	Margin bell not used	T



Terminal Feature	Factory Default Condition	List
Networking		
Output Networks	Keyboard to Line 1 (Full Duplex)	D
Keyboard Communication Line	Lines 1, 2, and 3 to screen Line 1	T
Transmission		
Baud Rate	Line 1: 9600. Lines 2 and 3: 300	D
Stop Bits	2	D
Parity	No parity bit used	D
Parity Checking on Input	Off	D
Transmission Delay	None	T
CTS/RTS Protocol	Line 1: Off. Lines 2 and 3: On.	D
Buffer Overflow Control	Lines 1: On. Lines 2 and 3: Off.	D
Start of Print/Transmit	Line 1, column 1 (home)	W
Transmit Protected/All (1)	All characters transmitted	A
Transmit Extent	Window	W
Transmit Termination (16)	End of Line, Window or Field	W
Transmit Initiation (116)	Beginning of Window	W
Trailing Spaces on Output (114)	Suppress	T
Underline Attribute (115)	Transmit as 008 followed by 095	T
Form Feed Prior to Print (117)	Yes	T



## Appendix H

### Software Version 33333

This appendix describes enhancements and modifications included beginning with version 33333 of the terminal software. The software version number is shown near the right end of the User Status Line (see Appendix B.1) as follows:

A33333 Concept AVT or AVT-APL with 8 pages of display memory  
B33333 Concept AVT or AVT-APL with 4 pages of display memory  
C33333 Concept GVT or GVT-APL with 8 pages of display memory  
D33333 Concept GVT or GVT-APL with 4 pages of display memory

The first three changes explained below apply to all of the products listed above. The last two apply only to the Concept GVT and GVT-APL.

#### H.1 Halt Terminal       ESC a       {ESC}

This function provides a way of stopping a Print or Transmit operation that is in progress without completely resetting the terminal. Striking the ESC key shifted (or sending ESC a from the line) stops all terminal functions currently in progress without altering the contents of display memory or programmable keys. All input and output buffers are cleared. All cursors are homed, and the Start of Screen set to the first line of display memory. No window definitions are changed, but all scrolling regions are reset to the entire window. No other terminal operating characteristics are affected.

The shifted ESC key is functionally part of the Cursor Pad, and can be set to one of four operational modes: execute, transmit, execute and transmit, disabled (see Sections 2.4.4 and 3.11). When using the Cursor Pad Keys Settings command (Section 3.11), the shifted ESC key is identified as key number 15. In Transmit or Execute and Transmit mode, shifted ESC transmits the sequence ESC a regardless of whether the Cursor Pad Operation (HDS Mode 201) is Normal (Reset) or Application (Set).

#### H.2 Host Overflow Control       ESC [ hctl;devcm \* {

This command can be used to prevent the terminal from responding to XOFF (↑S, ASCII 019) transmitted by the host computer. In factory default state, the terminal responds to XOFF by ceasing all transmission of data to the line from which the XOFF was received, until an XON (↑Q, ASCII 017) is received from the same line. Both Block and Character Mode transmission are affected (see Section 2.9).

Although this response to XOFF facilitates control of communication between terminal and host, it can lead to problems if XOFF is used by host software for purposes other than buffer overflow control. In particular, if the host sends XOFF but does not follow it with XON, and the terminal is in character mode, the terminal will ignore keyboard input while waiting for the host to

send XON. When this happens, the terminal appears to be "hung".

In terminals with software versions earlier than 33333, terminal response to XOFF from the host can be effectively disabled by defining the programmable delay character as ASCII 019 (↑S), as explained in Section 2.3.4. However, the Host Overflow Control command provides a better solution (since it applies only to specific lines), and should be used if available.

This command uses two parameters:

hctl specifies whether the terminal is to respond to XOFF from the line specified by devcm.

- 0 Ignore XOFF (↑S, ASCII 019) received from line devcm (default). XOFF is then effectively a delay character.
- 1 Respond to XOFF from line devcm by ceasing all transmission to line devcm

devcm specifies the line to which the command applies.

- 0 Requesting device or Keyboard           1   Line 1
- Communication Device (default)       2   Line 2
- 3   Line 3

For example, the command

```
ESC [ 0 ; 1 * {
```

causes the terminal to ignore XOFF received from Line 1. Since 0 is the default value for hctl, ESC [ ; 1 \* { would have the same effect. If the command were sent from Line 1, it could be shortened to ESC [ \* { , since devcm defaults to requesting device.

To make the terminal respond to XOFF from Line 1 once again, the command would be

```
ESC [ 1 ; 1 * {       ( ESC [ 1 * { from Line 1)
```

Note: This command has no effect on whether the terminal sends XOFF/XON to the host computer to prevent overflow of the terminal's input buffer. Use of XOFF/XON for this purpose is controlled by the Buffer Overflow (Receive) Protocol command (Section 3.9).

### H.3 Keyboard Communication Device

A minor modification has been made in the operation of the Set Keyboard Communication Device command ( ESC [ devcm z ). This command is now ignored if CTS/RTS (transmit) protocol is in effect for line devcm, but no high signal is detected on the CTS pin of the line connected to devcm. See Section 3.9 and Appendix D.

This modification has the effect of making it impossible to try to send data to a device that is not available when CTS/RTS protocol is in use.

#### H.4 GVT Terminal Identifier (C33333 and D33333 only)

Concept GVT (and GVT-APL) terminals now transmit special sequences in response to the Transmit Terminal ID command ( ESC [ 0 c ) that make it possible for host software to distinguish between Concept AVT and Concept GVT terminals. When HDS Mode 123 is Set, the following sequences are transmitted:

ESC [ = 2 ; 1 c	Concept GVT (or GVT-APL) with 4 pages of memory
ESC [ = 2 ; 2 c	Concept GVT (or GVT-APL) with 8 pages of memory

When HDS Mode 123 is Reset (factory default state), both Concept AVT and Concept GVT terminals identify themselves as VT100's by transmitting the sequence ESC [ ? 1 ; 2 c . See Section 3.4 for information on the Transmit Terminal ID command.

#### H.5 Line 3 Stop Bits and Parity (C33333 and D33333 only)

Concept GVT terminals now ignore all attempts to change the Stop Bit and Parity settings for Line 3, either through the Set Parity and Set Stop Bit commands, or in Setup Mode. Since Line 3 is used exclusively for communication between the graphics processor and the main terminal board, there would be no reason for the user to change any of the Line 3 communication parameters. This modification is designed to prevent problems from arising due to inadvertent changes in these settings.



- Alert Line (see also Status Lines) 2-27; 3-51
- Alternate Processing Modes 2-4; 3-2; 4-1
- Alternate Character Set (see: Character Set)
- Answerback Message 2-11,27; 3-14,51
- APL
  - in Setup Mode 1-17; 2-32
  - , Translation from ASCII 2-1
  - Overstrike 2-32; 4-3
  - Character Set A-3
  - APL/ASCII Mode 4-3
- Application Mode (Numeric Pad) 2-24; E-1
- ASCII Character Set A-1
- ASCII/APL Chart Location 2-1
- Attribute Lists 2-36; 3-8
- Attributes 2-33; 3-28
  - Attribute Selection 2-35; 3-29,30
  - Blinking 2-34
  - Block Attribute Change 2-37,63; 3-30
  - Bold 2-33
  - Inverse Video 2-34
  - Non-Display 2-34
  - Protection 2-34
  - Underline 2-34
- Auto Tabs 2-62; 4-4
- Background Status Line 2-10; 3-13
  - (see also: Status Lines)
- Backspace 1-11; 3-17
- Backward Tab 1-11; 3-20
- Baud Rate 1-6,16; 3-37
- Blinking Characters (see: Attributes)
- Block Attribute Change (See: Attributes)
- Block Character Change 2-62; 3-31
- Block Mode 2-55; 4-4
  - Applications 2-58
  - Interactive Uses 2-55
  - Transfer of Data Between Hosts 2-51
- Bold Characters (see: Attributes)
- Break Key 1-12; 3-41
- Buffer Overflow Control 2-17; 3-39; F-2
  - Response to  $\uparrow$ S/ $\uparrow$ Q from Host 2-18; 3-39; H-1
- Carriage Return (see also: Line Feed) 3-15
- Change Message Character 2-18; 3-7
- Character
  - /Attribute Replacement 2-37; 4-6
  - Mode 2-55
  - Processing 2-5
  - Set, Normal and Alternate 2-31; 3-10
  - Sets 2-31; 3-10; A
  - Set Self-Test 2-14; 3-6
- Checking Out the Terminal 1-1; 2-13
- Clear Characteristics 2-30,37; 3-22
- Clear Tabs 3-20
- Clear to Send/Request to Send (see: CTS/RTS)
- CMD key 1-12; 2-3
- Command(s), Terminal 2-1, 3-1
  - Sequence Introducer 3-3
  - Sequences 2-1,2,3; 3-2
  - Recognized in VT52 Mode E-1
- Communication(s)
  - Control Codes 2-7
  - Considerations 2-16
  - Interfaces D-1
  - Lines and Devices 2-7
  - Self-Test 2-14; 3-6
  - Setting Up 1-6
  - with Multiple Hosts 2-49
  - with a Printer 1-7
- Control Codes 2-2,6; C-1
- Copy Attribute List 2-37; 3-8
- CTS/RTS (Transmit) Protocol 2-45,49; 3-38; D-2
- Current Loop Interface D-3
- Cursor 1-4
  - Control Commands 2-41; 3-15
  - Down 3-16
  - Down to Left Margin 3-17
  - Left 3-17
  - Pad Keys 1-12; 2-25; 3-47,48
  - Position (Transmit) 3-19
  - Positioning Commands 3-17,18
  - Right 3-16
  - Up 3-16
  - Up to Left Margin 3-17
- Cursor Pad Keys 1-12; 3-48
  - Execute, Transmit, and Disable 2-25; 3-47
- Default(s) (see: Factory Defaults)
  - Power-up Defaults (see: NVM)
  - values for parameters 2-4; 3-4
- Default Execute Function Keys 1-15; 2-21; 3-55
- Default Transmit Function Keys 1-16; 2-21; 3-55
- Delete Character 2-29; 3-24
- Delete Line 2-29; 3-25
- Device (see: Communication)
- Device Status (Transmit) 3-11
- Display Memory (see: also Windows)
  - 1-4; 2-38; 3-9,43,44,45
  - Allocating Display Memory 2-38; 3-45
  - Clearing Display Memory 2-29,30; 3-21
- Display Status Line 2-10; 3-11
- Display Width (80 or 132 Columns) 1-18; 2-38; 4-3,7
- Display/Edit Function Key 2-22; 3-52
- Duplex, Half and Full 1-6,16; 2-48; 4-3
- Echo from Host System 2-56
- Editing 2-28,55; 3-21
  - Extent 2-29; 3-21,23
- End of Field (EOF) 2-16; 3-7,40
- End of Line (EOL) 2-15; 3-7,40

- End of Message (EOM) 2-15; 3-7,12,40
- End of Text 3-18
- Erase (see also: Clear) 2-29; 3-21,22
  - in Field 2-29; 3-26
  - in Line 2-29; 3-26
  - in Window 2-29; 3-25
- Errors in Commands 2-7,48; 3-4
- ESC [ 2-7; 3-3
- Escape Message Characters (ESC) 2-7,16; 3-2
- Escape Sequences (see also: Commands) 2-3,7,46; C-2
- Execute Function Key/Stored Data 3-53
- Factory Default Settings G-1
  - Reset to 2-12; 3-7
- Fields (see also: Protection) 2-60
- Fill Character 2-7,14,18; 3-8; F-1
- Form Feed 3-23; 4-7
- Forms Handling 2-60
- Forward Tab 1-11; 2-62; 3-19; 4-3
- Function Keys (see Programmable Function Keys) and Stored Data 3-49
- Half Bright (see: Attributes, Bold)
- Halt Terminal H-1
- Host Overflow Control H-1;
  - (see: Buffer Overflow Control)
- Host Communications 1-6
- Index (Line Feed) 3-15
- Input Buffers F-1
- Insert Line 2-29; 3-24
- Insert Mode 2-28; 3-23; 4-2
  - Toggle Insert Mode 2-28; 3-23
- Inverse Video (see: Attributes)
- Keyboard 1-10
  - Communication Line 2-8; 3-36
  - Controls 3-47
  - Cursor Pad 1-12; 2-25
  - Lock 3-47
  - Operation with Display 1-4
  - Numeric Pad 1-12; 3-54
- Latent Expression 2-25; 3-51
- Line (see: Communication)
- Line Drawing 2-62
- Line Feed 3-15; 4-3
- Local Mode 1-3; 2-13; 4-5
- Margin Bell 3-26
- Memory (see also: Display Memory)
  - , Display 1-4,17
  - for Programmable Keys 3-45,49
  - , Non-Volatile (see: NVM)
- Message Characters (Programmable) 2-15; 3-7
  - Cursor Pad End of Message (CPM) 2-15; 3-7
  - Delay (DLY) 2-15,18; 3-7; F-2
  - End of Field (EOF) 2-16; 3-7,40
  - End of Line (EOL) 2-15; 3-7,40
  - End of Message (EOM) 2-15; 3-7,12,40
  - Escape (ESC) 2-7,16; 3-2
  - Function Key Identifier 2-15,21; 3-7,55
  - Start of Message (SOM) 2-15; 3-7,12,40
- Modes (see: Alternate Processing Modes)
- Multiple Computer Applications 2-49
  - Data Transfer Between Hosts 2-52
  - Output Network 2-49; 3-35
- Networking 2-47; 3-34
- New Line (Line Feed/CR) 3-16; 4-3
- Next Status Line 3-12
- Non-Volatile Memory (NVM) 1-3,7; 2-9,11; 3-7
- Output Network 2-49; 3-35
- Page Down 3-43
- Page Up 3-43
- Parameters 2-3; 3-1,4
- Parity/Parity Checking 1-6,17; 3-37
- Peripherals (see: Networking) 1-6,9; 2-45; 3-34
- Power-up 1-1 (see also: Reset Terminal)
- Printer Port (Line 2) 1-7; 2-45,46; 3-34
  - Print/Line 3 Control 3-34
  - PRINT key 1-13; 2-46
  - PRINT/SCRN key 1-13; 2-47
- Programmable Function Keys 1-15; 2-20; 3-49
  - Execute and Transmit 2-20
  - Defaults 2-21; 3-55
  - Keys (Table of) 3-55
  - Programmed by NVM 2-26
- Protection
  - Attribute 2-34
  - and Fields 2-60
  - Protected Field Erasure 2-61; 4-2
  - Protected Field Overwrite 2-61; 4-4
  - Protection Off 3-33
  - Protection On 3-32
- Requesting Device 2-8
- Reset Terminal 3-6
  - to Factory Default State 2-12; 3-7
  - to Power-Up State 2-11; 3-6
- Restore Cursor Position and Attributes 3-19
- Resume Transmission (XON) 3-40
- Reverse Index 3-16
- Reverse Video (Screen Video) 1-18; 2-34,35; 4-8
- Ring Keyboard Bell 3-48
- ↑S (ASCII 019) (see: Buffer Overflow Control)
- Save Configuration in NVM 3-7
- Save Cursor Position and Attributes 3-19
- Screen Video (see: Reverse Video)
- Scroll Down 2-39,40; 3-43



- Scroll Up 2-39,40; 3-43
- Scrolling 1-5
  - No Scrolling (Forms) 2-61; 3-15; 4-3
- Select Window 2-42; 3-9
- Self Test 2-13; 3-6
- Sending Data to the Host 2-56; 3-40
- Set and Reset Modes 2-4; 4-1
- Setting Up Communications 1-6
- Setup Mode 1-2,16
- Shift
  - In (↑N) 2-31; 3-10
  - key 1-11
  - Out (↑O) 2-31; 3-10
- Special Keys 3-1
- Start of Print/Transmit 2-56; 3-42
- Status Information 2-9; 3-11
  - Transmission to Host 2-10
  - Device Status 3-11
- Status Lines 2-9
  - Alert Message 2-27; 3-51; B-4
  - in Setup Mode 1-16
  - Modes B-3
  - Programmable Message Character B-5
  - Programmer 2-35,47; B-2
  - Tabs 1-2,19; B-5
  - User 1-2; 2-9,39; B-1
- Stop Bits 1-6,17; 2-17; 3-38
- Stop Transmission (XOFF) 3-40
  - (see: Buffer Overflow Control)
- Tab 3-20
  - Processing 2-62
  - Status Lines 1-19
- Terminal Commands (see: Commands) 2-1
  - on Programmable Keys 2-23
- Terminal Identifier 2-10; 3-11; H-3
- Tests (see: Self Test)
- Timing Considerations F-1
- Toggle/Clear Status Line 3-13
- Top of Screen 3-44
- Trailing Blank Suppression 4-5
- Transmit 3-40
  - Delay 2-59; 3-41
  - Extent 2-56; 3-40
  - Mode (see: Programmable Keys)
  - Status Line 3-12
  - under Host Control 2-58
- Transparent Mode 2-14; 3-32
- Trouble Shooting 2-13
- Underline
  - Attribute 2-33
  - Input Processing 4-6
  - Output Processing 4-5
- Unpacking 1-1
- User Status Lines 1-16
- VT100 and Concept Special Graphics A-2
- VT100 Compatibility 1-9; 2-32,38,41,61; 3-45; 4-7
- VT52 Terminal Type Operation 1-18; E-1
- VT52 Cursor Keypad Operation E-2
- Window(s) (see also: Display Memory) 2-40
  - Definition 2-42; 3-45
  - Lists 2-42;
  - Multiple Window Applications 2-43,53
  - Selection 2-41; 3-9
- XOFF/XON (see: Buffer Overflow Control)

