Part No. 070-4526-01
Product Group 18

4105
COMPUTER DISPLAY TERMINAL

Please Check for
CHANGE INFORMATION
at the Rear of This Manual

First Printing APR 1983
Revised SEP 1983

Tektronix®
COMMITTED TO EXCELLENCE

MANUAL REVISION STATUS

PRODUCT: 4105 Computer Display Terminal

This manual supports the following versions of this product: Firmware Version: 1 and up. Current Version: 2.

REV DATE	DESCRIPTION
APR 1983	Original Issue
JUN 1983	Revised: pages iii, 1-2, 1-3, 2-1, 2-4, 2-5, 2-6, 3-2, 3-3, 3-8, 3-10, 3-11, 4-1, 4-6 through 4-9, 4-12, 4-14, 4-19, 5-1, 5-3, 5-5, 5-9, 5-10, 5-12, 5-14 through 5-17, 5-19, 5-22, 5-23, 5-24, 5-26 through 5-30, 5-32 through 5-35, 5-37 through 5-41, C-1 through C-4, D-2, D-3, D-4, and IDX-2.
SEP 1983	Rewritten to incorporate JUN 1983 revisions and add Version 2 Enhancements.

CONTENTS

Section 4 (cont) **Page**

Section 5 (cont) **Page**

ILLUSTRATIONS

TABLES

Section 1

INTRODUCTION

The TEKTRONIX 4105 Computer Display Terminal is a color graphics and text editing terminal that lets you display graphics as well as use a variety of text editors.

THE MANUAL PACKAGE

This manual contains the reference information needed to develop and maintain applications software for the TEKTRONIX 4105 Computer Display Terminal.

Related manuals include:

- The *4105 Computer Display Terminal Operators Manual*, which describes the terminal, its use, and the self-test features.
- The *4105 Computer Display Terminal Reference Guide*, which contains essential reference material in a condensed form.

WHERE TO LOOK FOR INFORMATION

This manual is organized as follows:

- Section 1, *Introduction*, contains introductory information about the terminal and tells you where to look for more information.
- Section 2, *Communications*, discusses the concepts of terminal communications with the host computer. Topics discussed are reports, communications settings, and handshaking protocols.
- Section 3, *The Graphics Terminal*, discusses controlling the keyboard and display from the applications program, graphic input and output, and using the color display.
- Section 4, *Screen Editor Support*, discusses text editing concepts. This section also contains detailed command descriptions for the ANSI X3.64 and VT52 text editing commands. If you are not familiar with this terminal, be sure to read the Syntax Conventions portion of this section; it will help you to better understand the command descriptions that follow in this section.
- Section 5, *4100-Style Commands, Reports, and Parameter Types*, contains an alphabetically organized dictionary of all the terminal commands except ANSI X3.64 commands. If you are not familiar with this terminal, be sure to read the "Command Conventions" portion of this section; it will help you to better understand the command descriptions that follow in this section.
- Appendices include an ASCII Chart, Alternate Character Sets, Error Codes, Parameter Default Values, the Tektronix Color Standard, Examples of Integer Parameters, a glossary, and an index.

FEATURES

The following paragraphs describe some of the terminal's features.

Color. You can select from a palette of 64 colors, with up to eight different colors displayed in the graphics area and an additional eight colors in the dialog area at one time. Use color to highlight lines, panels, and text.

Graphics. You can draw solid or dashed lines in color. You can fill panels with solid colors, color shading, or with a variety of patterns. You can add labels or text to the graphics in different sizes, colors, and rotation angles.

Graphic Input. You can use most graphic input application programs written for Tektronix 4010 Series terminals without modification.

Alphanumerics. You can display uppercase and lowercase characters. The characters have definable attributes, including character color, background character cell color, underline, and blink. The characters are displayed in 5-by-7 dot matrices in a 6-by-12 dot character cell.

Alternate Character Sets. You can select from eight alternate character sets: ASCII, United Kingdom, French, Swedish, Danish/Norwegian, German, a special supplementary character set, and a special rulings character set.

The Dialog Area. The terminal includes a user-definable dialog area that lets you display host communications without interfering with the graphics on the screen. You can select a column width of 80 or 132 characters and you can adjust the area height from 2 to 30 lines. The dialog area's scrollable memory is adjustable from two to as many lines as there is memory available. The visibility of the dialog area can be controlled with the Dialog key (toggles between visible and not visible).

Nonvolatile Memory. This portion of the terminal's memory lets you configure the terminal for a particular application and then save the operating parameters in a part of the terminal's memory that is not erased even when the power is turned off. When you turn the power on again, the terminal automatically uses the saved parameter values.

Commands. The terminal has three types of commands. The first type lets you produce color graphics and control and program the keyboard and display; these commands are compatible with Tektronix 4010 Series graphics applications programs. The second type of command lets you insert, delete, and erase characters and lines and to otherwise manipulate text in the dialog area; the terminal is compatible with screen editors that transmit ANSI X3.64 commands or VT52 commands (commands that follow the VT52 command syntax). The third type lets you enter English-style terminal control commands from the keyboard while in Setup mode.

Computer Interface. The terminal uses a full-duplex serial RS-232-C interface with data rates to 38400 baud.

Software Compatibility. The following software packages are compatible with this terminal:

- Existing programs written for 4010 Series terminals
- TEKTRONIX PLOT 10 Interactive Graphics Library
- TEKTRONIX PLOT 10 Easy Graphing II
- SASGRAPH, from the SAS Institute, Inc
- DISSPLA, from ISSCO (Integrated Software Systems Corporation)
- Certain editors designed for use with VT100 terminals, such as EDT, VI, and EMACS

Color Hardcopy. The terminal is compatible with the TEKTRONIX 4695 Color Graphics Copier.

Keyboard. The terminal has a low-profile detachable keyboard. The keys include ASCII uppercase, lowercase, and control characters; BREAK and ERASE keys; a 14-key numeric keypad; and four special function keys and eight programmable function keys. Most keys have N-key rollover. All keys are programmable except SHIFT, CTRL, and CAPS LOCK and you can set MOST keys to repeat when held down for more than one-half second.

The Display. The terminal incorporates a 60-Hz non-interlaced raster-scan display. The color CRT has 480-by-360 pixel resolution.

THE TERMINAL'S PROGRAMMING MODEL

You can configure the terminal to perform many different functions. Once you've set the operating parameters or characteristics of the terminal to best perform the desired function, you can save these parameters in the terminal's memory so that they are not erased when the terminal is turned off. In this way, the terminal can remain in the same configuration for as long as you want.

This terminal is really two different terminals in one package:

- A graphics display terminal that is compatible with Tektronix 4010 Series graphics applications programs. It also has an interactive color interface (used to adjust the terminal's color map or displayed colors), a scrollable dialog area for host communications while displaying graphics information, and a subset of the Tektronix 4110 Series command set.
- A text entry and editing terminal that is compatible with both the American National Standards Institute (ANSI) X3.64 and the International Organization for Standardization (ISO) 6429 standards, and VT52-style commands. These standards define a set of terminal functions that let you manipulate and edit computer text files. You can configure the terminal to run most popular screen editors.

Modes of Operation

The terminal has four major modes of operation:

- Ansi mode. In this mode, the terminal understands the syntax of the ANSI X3.64 text editing commands only. One specific implementation of Ansi mode is Edit mode, in which the operating characteristics of the terminal have been set to specific values. The *Screen Editor Support* section of this manual contains a discussion of Ansi and Edit modes, and the descriptions of the ANSI commands.
- Tek mode. In this mode, the terminal understands the syntax of the graphics and terminal control commands. *The Graphics Terminal* and *4100-Style Commands, Reports, and Parameter Types* sections of this manual contain discussions of this mode and descriptions of the commands available in this mode. For added versatility, the parameters set in Tek mode carry over appropriately into Ansi mode.
- VT52 mode. In this mode, the terminal is compatible with programs using VT52-style commands.
- Setup mode. In this mode, the terminal accepts Setup mode commands, which the user enters from the terminal keyboard. *The Graphics Terminal* and *4100-Style Commands, Reports, and Parameter Types* sections of this manual contain a discussion of this mode and descriptions of the commands available in this mode.

The terminal also has many different submodes that let you perform different functions. Figure 1-1 shows the relationship of these submodes to the four major modes.

Terminal Commands

The terminal can accept four different types of commands, each with its own syntax. Commands can only be executed in the terminal mode to which they belong.

- Setup mode commands. These commands must be typed on the keyboard while in Setup mode. Setup mode commands have English-style names and parameters. For example: **CODE TEK** is the Setup mode command entered to select Tek mode.
- Ansi mode commands. These commands are used for text editing. These commands can only be entered with the terminal in Ansi mode. You can type these commands on the keyboard. For example: **^{E}cM** is the ANSI command that performs a reverse index function, and **^{E}c[5B** is the Ansi mode command that moves the alpha cursor 5 lines toward the bottom of the screen. Some of these commands also have equivalent commands in Setup mode.
- 4100-Style (Tek mode) commands. These commands are used for graphics and terminal control. This class of commands lets you display graphic information on the screen and control the terminal. You can enter these commands on the keyboard, and they can only be entered with the terminal in Tek mode. Most of these commands also have equivalent commands in Setup mode. For example: **^{E}cLL5** (**DALINES 5** in Setup mode) is the 4100-style command that sets five visible lines in the dialog area.
- VT52 commands. These commands are used for text editing and can only be entered with the terminal in VT52 mode.

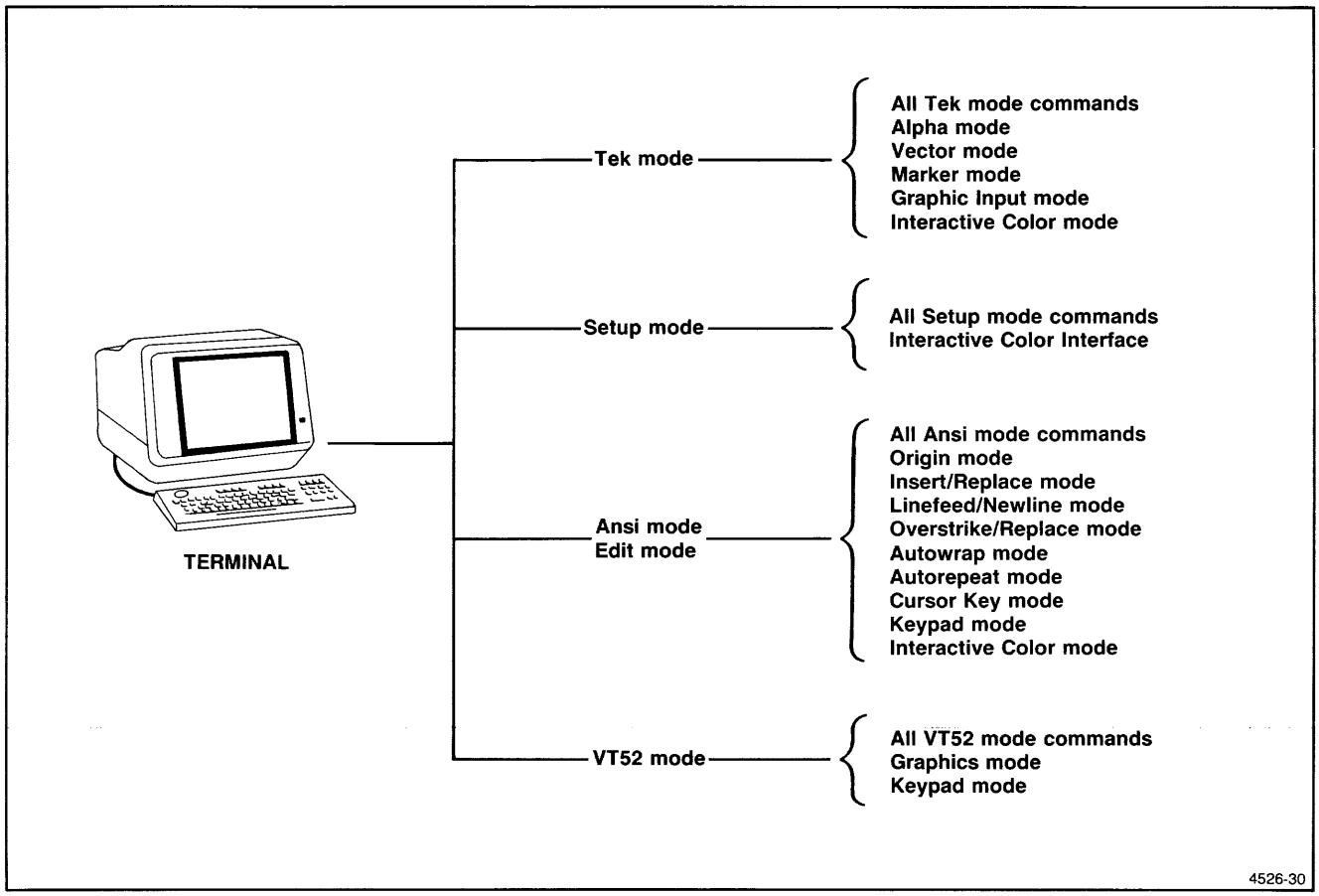


Figure 1-1. Terminal Modes.

Section 2

COMMUNICATIONS

This section explains communication between the terminal and a host computer. The topics covered are:

- *Communications Settings*
- *The Communications Input Queue and Handshaking Protocols*
- *Requesting Reports From the Terminal*

COMMUNICATIONS SETTINGS

This subsection explains:

- How to set the terminal's standard communication settings
- How to use full-duplex communications

You can set all of the terminal communications settings by sending commands from the applications program, or the operator can set them by typing Setup mode commands on the keyboard. These settings can be saved in the terminal's nonvolatile memory and remembered even if the terminal is turned off.

The following paragraphs contain an explanation of each communication setting and the name of the command used to change the setting.

BAUD RATES

You can set the terminal's host-to-terminal and terminal-to-host data transmission baud rates with the SET BAUD RATES command. The term *baud rate* refers to the rate (in bits per second) that the terminal transmits or receives data. Normally, the operator sets these rates using the Setup command BAUD; however, you can issue a SET BAUD RATES command from the host.

Communications with the host cannot take place unless the terminal's baud rates are correctly set.

TRANSMIT RATE LIMITS

You can specify a transmit data-rate limit — a maximum speed for terminal-to-host communications — that is less than the rate at which the terminal sends each individual character. A transmit data rate limit of 300, for instance, means that the terminal, in sending characters to the host, will space those characters apart for an average data rate of 300 bits per second. This limit is useful at high terminal-to-host baud rates, where the host computer's input processor can't accept characters at the full data rate. Use the SET TRANSMIT RATE LIMIT command to set this limit.

SET ECHO

Except in Setup mode, when you type on the terminal's keyboard, the characters go to the host. They do not necessarily appear on the terminal's screen. They only appear on the screen if (1) the host (or modem) sends the same characters back to the terminal — provides a remote echo — or (2) the terminal provides its own local echo of the transmitted characters.

The operator can specify whether the terminal provides a local echo by using the ECHO command in Setup mode. The SET ECHO command can also be issued by the host.

FULL-DUPLEX DATA COMMUNICATIONS

The terminal uses full-duplex data communications. To use full-duplex data communications with remote echo, the terminal's local echo must be turned off. (Enter an ECHO NO command in Setup mode.)

PARITY

The terminal's parity setting controls how the terminal sets the eighth bit (parity bit) in each character it sends to the host. Table 2-1 shows the possible parity settings and how they work.

Table 2-1

PARITY BIT DESCRIPTION

Parity	Description
Low Parity	When the terminal sends a character to the host, it sets the parity bit to 0.
Odd Parity	When sending a character to the host, the terminal sets the parity bit so that there are an odd number of 1's in the character's eight bits.
Even Parity	When sending a character to the host, the terminal sets the parity bit so that there are an even number of 1's in the character's eight data bits.
High Parity	Sets the parity bit to 1 in each character it transmits to the host.
Data Parity	The parity bit is used for encoding data, the same as are the other seven bits in each eight-bit character. Data parity is used in the terminal to send eight-bit data to the hardcopy unit.

The host can control the terminal's parity setting with the SET PARITY command.

STOP BITS

While communicating with the host, the terminal sends and receives each character serially, as a sequence of ten or eleven bits. This is called *asynchronous serial* data communications. The first bit for each character is a start bit, always a 0 (or space) bit. The next seven bits determine the particular ASCII character, after which comes a parity bit, as described earlier. The character ends with one or two stop bits, which are always 1 (or mark) bits. The communications line then remains in the marking condition until the start bit for the next character is detected.

While receiving characters from the host, the terminal will always respond, regardless of whether the host sends one or two stop bits in each character.

While transmitting characters to the host, the terminal includes one or two stop bits in each character it transmits. Set the number of stop bits with the SET STOP BITS command.

BREAK TIME

Pressing the BREAK key sends a break signal to the host. In full-duplex communications, the break sets the communication line in a "space" condition for the length of the break.

As shipped from the factory, the break signal is set to last 200 milliseconds; this is adequate for most host computers (refer to your host's documentation to determine how the break signal is interpreted). The SET BREAK TIME command lets you change this value for hosts for which 200 milliseconds is too short or too long. For hosts that do not accept break signals, you can set the break time to 0, disabling the break feature.

COPING WITH $^{D}_{T}$ FILLER CHARACTERS

The Problem. Some host computers intersperse $^{D}_{T}$ characters (also known as Delete or Rubout characters) among the characters they send to a terminal. The host inserts these filler characters and the applications program has no control over them. Since the terminal interprets $^{D}_{T}$ as a valid character in integer and xy-coordinate parameters, these extra $^{D}_{T}$ characters can cause problems. Integer and xy-coordinate parameters are described in the *4100-Style Commands, Reports, and Parameter Types* section of this manual.

The Remedy. The terminal includes two features that help solve this problem. First, the terminal accepts the two-character sequence ^{E}**c?** as a synonym for $^{D}_{T}$. Second, the IGNORE DELETES command causes the terminal to ignore any $^{D}_{T}$ characters that the host sends it. The terminal does not, however, ignore ^{E}**c?** sequences.

Thus, if your host uses $^{D}_{T}$ as a filler character, you should do the following two things:

- Write your device driver routines so that they always send ^{E}**c?** when they would otherwise send the $^{D}_{T}$ character. Change routines that issue integer and xy-coordinate parameters to output ^{E}**c?** instead of $^{D}_{T}$.
- Execute an IGNORE DELETES command and then execute the SAVE NONVOLATILE PARAMETERS command to cause the terminal to remember this condition, even if it is turned off.

THE COMMUNICATIONS INPUT QUEUE AND HANDSHAKING PROTOCOLS

THE INPUT QUEUE

The terminal's input queue (or buffer) accumulates characters received from the host. When characters arrive faster than the terminal can process them, the terminal stores them in its input queue until it has a chance to process them, or until the memory allocated for that queue is exhausted. (If the queue memory is exhausted, incoming characters are lost.)

For instance, the terminal cannot display characters while it is erasing its screen. Therefore, while the screen is being erased (as, for instance, in response to the S Erase, D Erase, and G Erase keys), any characters coming from the host are stored in the input queue until the erase operation is finished. When the screen erasure is complete, the terminal reads the characters from the queue and displays them.

While in Setup mode, the terminal does not display characters coming from the host. Instead, such characters accumulate in the input queue. The terminal waits to process those characters until the operator terminates Setup mode.

As shipped from the factory, the input queue can hold up to 300 ASCII characters; however, you can change this value with the SET QUEUE SIZE command.

THE NEED FOR HANDSHAKING

The terminal can display simple alphanumeric text and graphics only up to a maximum continuous data rate of 19200 baud. At higher data rates, or for more complex operations, use handshaking protocol to prevent the input queue from overflowing. Even at slow data rates, it may be prudent to use a handshaking protocol.

You can simply issue a REPORT 4010 STATUS command from time to time, and wait to receive the reply before sending more characters to the terminal. Alternatively, you can use Flagging mode or Prompt mode. The type of flagging you select depends on what your host supports.

Flagging

Flagging is used to prevent the input queue from overflowing by allowing the terminal to start and stop transmissions. With flagging enabled, the data communications at high data rates can be accomplished without overflowing the input queue.

Two types of flagging are available: DC1/DC3 software flagging and DTR/CTS hardware flagging.

DC1/DC3 Flagging. DC1/DC3 flagging uses the transmission or reception of the DC3 and DC1 control characters to inhibit or enable the transmitting device. DC3 is the stop character and DC1 is the start character. This allows you to use flagging with devices that do not have control of the RS-232-C DTR and CTS lines. Note that if a communications error causes a DC1 or DC3 character to be garbled in transmission, the handshake with the host computer may stop communications.

DTR/CTS Flagging. In DTR/CTS flagging, the terminal indicates that it wants to transmit data by asserting RTS (Request To Send). If the host is ready to receive the data, it asserts CTS (Clear To Send). The terminal can transmit only when CTS is asserted.

If the terminal transmits characters faster than the host can process them, the host can drop CTS. When the host is ready to receive more characters, it asserts CTS, and the terminal resumes its transmission.

When receiving characters from the host, the terminal uses the DTR (Data Terminal Ready) signal line in the same way that the host uses the CTS line. If the host is sending characters faster than the terminal can process them, the terminal drops DTR. The host stops transmitting to the terminal. When the terminal is ready for more characters, it asserts DTR, and the host resumes its transmission to the terminal.

You can set the terminal's Flagging mode using the SET FLAGGING MODE command. Alternatively, the operator can set Flagging mode using the Setup mode FLAGGING command (see the *4105 Computer Display Terminal Operator's Manual* for details).

NOTE

DTR/CTS flagging is usually not practical when the terminal is connected to the host with a telephone line modem. In such circumstances you should use DC1/DC3 flagging because the host does not have direct access to the DTR and CTS signal lines.

Prompt Mode

Prompt mode protocol is useful for preventing the host's input queue from overflowing when the terminal has too much data to send to the host. However, Prompt mode does not protect the terminal's input queue from overflowing when the host sends data to the terminal. For that, use Flagging mode.

Prompt Mode Operation. When the terminal is in Prompt mode, it waits to send each line of text until it receives a *prompt string* from the host. The prompt string is a sequence of characters, determined by the most recent SET PROMPT STRING command.

Upon receiving the prompt string, the terminal waits for the transmit delay and then sends one line of text to the host. Here, *one line of text* means all the characters it has to send, up to and including the next EOM (End of Message) character or EOL (End of Line) string — usually $^{C}_{R}$. This is described in more detail later in this section.

The prompt string must be the last characters the terminal receives, or the terminal will not recognize it as a prompt.

The following steps summarize Prompt mode operation:

1. The terminal sends a line of text, up to and including the EOM character or EOL string that marks the end of the line.
2. The terminal receives a prompt string from the host.
3. The terminal waits for the transmit delay.
4. Steps 1 through 3 are repeated again and again, until you terminate Prompt mode.

End of a Line of Text. As just mentioned, the end of a line of text occurs when the terminal encounters an EOM character or EOL string in the data it is sending to the host. The EOL strings are typically ^{C}R (carriage return) characters, although you can choose other characters or character sequences with the SET EOL STRING command.

LINES OF TEXT AND THE TRANSMIT DELAY

The preceding description of Prompt mode mentioned the concept of a line of text in data being transmitted to the host, and the transmit delay that occurs after the terminal sends each such line of text. The following description explains these concepts in more detail.

The Output Queue

The terminal's output queue holds any characters that are waiting to be transmitted to the host. When the operator types on the keyboard, the characters he or she types go into the output queue. Likewise, when the terminal has a report message to send to the host, the characters of that message go into the output queue.

The terminal sends the characters in the output queue to the host a line at a time; that is, it reads characters from the output queue and sends them to the host until it encounters the end of a line. Then it waits for a short time (the *transmit delay*) before sending the next line of text. If in Prompt mode, the terminal also waits to receive a prompt from the host before sending the next line of text.

If there are no characters waiting in the output queue, then each character the operator types enters the output queue and is immediately sent to the host. (This is true even if the terminal is in Prompt mode.) Characters wait in the output queue only if the operator types faster than the terminal's transmit rate. When the operator presses RETURN, a ^{C}R goes into the output queue and is sent to the host. Since ^{C}R is the usual EOM character, this also marks the end of a line of text. The terminal then waits a short time before sending the next character typed. This transmit delay, however, is usually so short that it is imperceptible to the operator.

A *line of text* means "all the characters waiting to be transmitted, up to and including the next EOM character or EOL string." As the terminal is shipped from the factory, its only EOM character is ^{C}R, and the EOL string consists of one character, ^{C}R. Thus, if the terminal is set as it is when shipped from the factory, a *line of text* means "all characters waiting to be transmitted, up to and including the next ^{C}R character."

REQUESTING REPORTS FROM THE TERMINAL

This subsection explains how you can request reports from the terminal.

The applications program can cause the terminal to transmit reports by sending report commands. When the terminal receives one of these commands, it transmits a report containing the requested information back to the program. Each of these commands causes the terminal to send a particular report. It is the program's responsibility to interpret these reports. The syntax of each report is explained in the *4100-Style Commands, Reports, and Parameter Types* section of this manual.

The terminal ends each line of information in the report with an EOM character. The terminal actually substitutes its current EOL string for each EOM character that is sent. The default EOL string is ^{C}R, although other characters may be selected with the SET EOL STRING command.

REPORT COMMANDS

The following paragraphs contain a brief explanation of the five reports that you can request.

Graphic Cursor Position. The 4010 GIN Report transmits the position of the graphic cursor to the host. This report is generated in response to an ENABLE 4010 GIN command, which allows the operator to position the graphic cursor and then press a key to transmit the coordinates of that location.

Alphatext Cursor Position. The CPR (Cursor Position Report) reports the location of the Alphatext cursor in the dialog area. This report is generated in ANSI format in response to a DSR (Device Status Report) command. You must select Ansi mode (by using the SELECT CODE command) before any DSR commands can be executed. The EOL string is not appended to the Cursor Position Report.

Terminal Settings. The Terminal Settings Report transmits the parameter values for the terminal setting to the host. The REPORT TERMINAL SETTINGS command causes the terminal to generate a Terminal Settings Report. The REPORT TERMINAL SETTINGS command allows you to use an inquiry-code parameter. You can enter the op-code for a particular command, and the Terminal Settings Report will report the values for that command. You can also enter special inquiry codes that report the current configuration of your terminal.

Error Messages. The Error Message Report reports the eight most recently detected error codes, their severity level, and how many times each error code was detected. This report is generated by the REPORT ERRORS command.

4010 Status. The 4010 Status Report reports the position of the graphic cursor. This report, when generated, terminates 4010 GIN mode and the terminal is placed in Alpha mode. This report is generated by the REPORT 4010 STATUS command.

Each time one of these reports is generated (except for the Cursor Position Report), the terminal enters Bypass mode. In Bypass mode the terminal will not accept commands or any other host input except the bypass-cancel character. The terminal remains in Bypass mode until it receives the bypass-cancel character or the Cancel key is pressed. Refer to the *4100-Style Commands, Reports, and Parameter Types* section of this manual for more information about the BYPASS CANCEL CHARACTER command.

Section 3

THE GRAPHICS TERMINAL

This section explains how to use the terminal for graphics display. Its principal topics are:

- *Using the Display* — displaying graphics and the host-operator dialog.
- *Graphic Input* — the process in which the terminal lets the operator choose a position on the screen and then reports to the host what location was chosen.
- *Macros* — character strings stored in the terminal that can be retrieved for a variety of applications.
- *Pixel Operations* — the additional capabilities offered by the optional pixel ROM pair.

While in Tek mode, the terminal responds to a full set of commands capable of creating graphics displays. In addition, Tek mode lets both the host and operator use macros.

To put the terminal in Tek mode, the host issues a SELECT CODE command that specifies Tek mode. The complete set of graphics commands is now available to the host. An operator can use the graphics commands at any time, regardless of whether the terminal is in Tek mode, by pressing the Setup key to put the terminal in Setup mode.

USING THE DISPLAY

The terminal displays two types of information:

- Host-operator dialog
- Graphics

The *host-operator dialog* consists of messages from the host and the operator's responses. *Graphics* is the screen's pictorial information. Usually, a host program directs the two types of information to separate areas of the screen. The area of the screen where the host-operator dialog appears is the *dialog area*. The area where graphics is displayed is the *graphics area*.

THE DIALOG AREA

Think of the dialog area as a writing surface that sits in front of the graphics area. Unlike the graphics area, which always fills the entire screen, the dialog area can vary in size from two lines to the entire screen. As explained later under "Colors and Transparency," the dialog area need not completely cover the graphics area, and you can make the dialog area transparent.

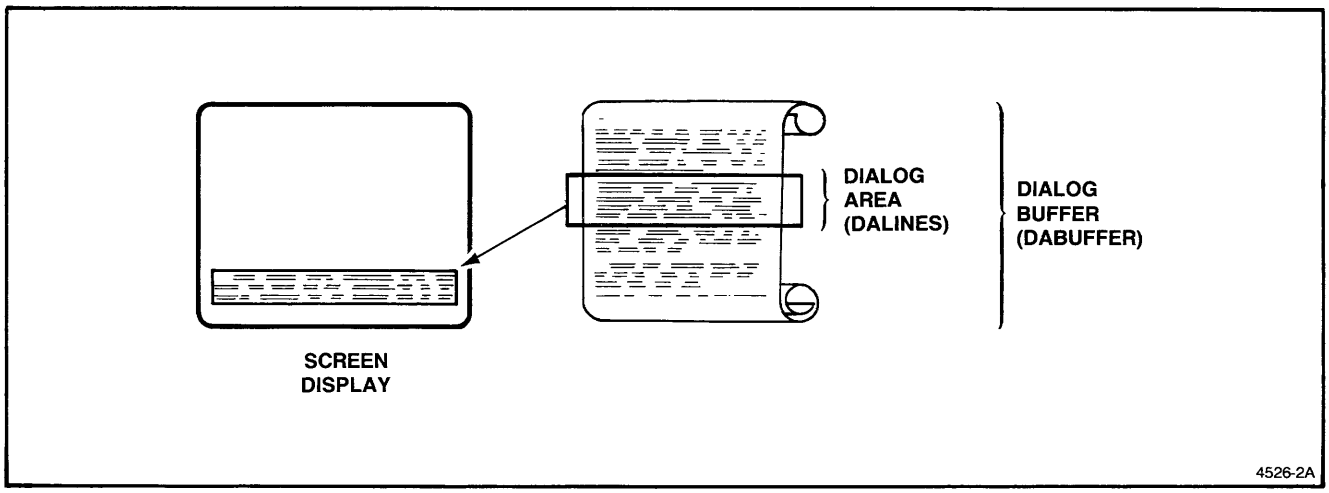


Figure 3-1. The Dialog Area and Buffer.

Either a host program or the operator can set the size of the dialog area by specifying the number of dialog lines. The operator uses the DALINES command; a host program uses the SET DIALOG AREA LINES command. Figure 3-1 shows a representation of the dialog area.

Dialog Buffer

The terminal can save more lines of dialog than what the dialog area shows. The saved lines, which include the lines displayed, are the *dialog buffer*. A host program sets the number of lines saved with the SET DIALOG AREA BUFFER SIZE command; the operator uses the Setup command DABUFFER. The operator can scroll the lines of the dialog buffer, moving different portions of the dialog buffer into view (see Figure 3-1).

Alphatext

Alphatext is the type of text the terminal uses to display messages from the host. The normal mode for the terminal when it is not in Ansi mode is *Alpha mode*. In Alpha mode, the terminal displays all characters it receives from the host, unless the characters are a command to the terminal. (The terminal can recognize commands from the host because they always begin with ^{E}C or a control character.)

Enabling the Dialog Area

When the dialog area is *enabled*, all alphatext is sent to the dialog buffer. When the dialog area is disabled, the terminal emulates a Tektronix 4010 Series terminal, which does not have a dialog area — alphatext appears on the screen at locations determined by previous commands or display operations. A host program enables or disables the dialog area with the ENABLE DIALOG command; the operator uses the Setup command DAENABLE.

Making the Dialog Area Visible

The dialog area does not appear on the screen unless it is *visible*. If the dialog area is enabled, the dialog is saved in the dialog buffer; but dialog is not displayed unless the dialog area is both enabled and visible. If the dialog area is enabled but *invisible*, changing the dialog area to *visible* displays dialog that accumulated while the dialog area was invisible. A host program sets dialog area visibility with SET DIALOG VISIBILITY; the operator uses the Setup command DAVISIBILITY.

Colors and Transparency

The SET DIALOG AREA INDEX command (Setup command DAINDEX) sets these colors in the dialog area:

- Character color — the color for alphatext in the dialog area
- Character background — the color of the cell surrounding each character
- Dialog background — the color of the dialog area before anything is written on it

Figure 3-2 shows where the dialog area uses these colors.

Both the dialog background and character background can be *transparent*. Graphics behind a transparent area can be seen. For example, when character background is transparent, alphatext appears as if it is written on a piece of glass in front of the graphics.

The details of specifying colors are explained later in this section.

Dialog Area Commands

Table 3-1 lists the commands that control the dialog area. Most of these commands can be either sent by the host or entered by the operator as a Setup command. The SAVE NONVOLATILE PARAMETERS command saves the settings made by these commands; these settings are then retained even when the terminal is turned off.

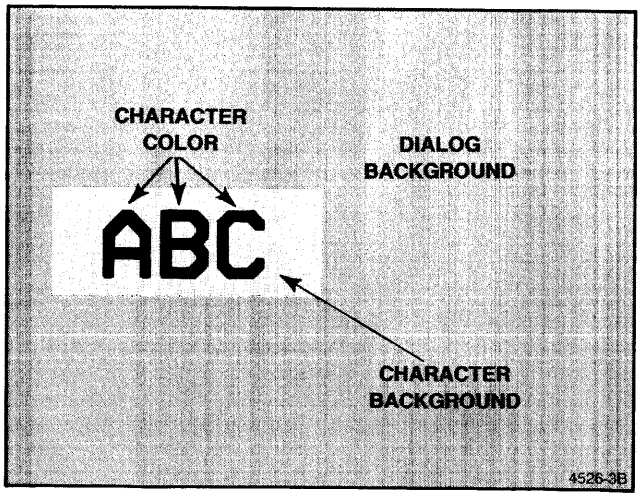


Figure 3-2. Colors in the Dialog Area.

Table 3-1

COMMANDS AFFECTING THE DIALOG AREA

Host Command	Setup Command	Function
ENABLE DIALOG AREA	DAENABLE	Directs alphatext to either dialog area or graphics area
SET DIALOG AREA BUFFER SIZE	DABUFFER	Sets the maximum number of dialog lines the terminal remembers
SET DIALOG AREA LINES	DALINES	Sets the maximum number of dialog lines visible at one time
SET DIALOG AREA INDEX	DAINDEX	Sets color of dialog area background and text
SET DIALOG AREA COLOR MAP	DACMAP	Defines the colors of the eight dialog area indices
SET DIALOG AREA WRITING MODE	DAMODE	Sets space and underscore to either replace or overstrike other characters
SET DIALOG AREA VISIBILITY	DAVISIBILITY	Makes dialog visible or invisible
CLEAR DIALOG SCROLL	CLEARDIALOG or Shift-D Eras key	Deletes all dialog

DISPLAYING GRAPHICS INFORMATION

This discussion tells how a program can draw pictures in the graphics area. Subjects covered are:

- **Terminal Space**. How to specify screen positions.
- **Windows**. How to specify the region you want displayed.
- **Lines**. How to draw lines and how to make them solid, dashed, or colored.
- **Markers**. How to mark screen positions with small symbols.
- **Panels**. How to fill an area with a pattern or color.
- **Graphtext**. How to put text in your pictures.

Terminal Space

Graphics are created on a conceptual two-dimensional surface called *terminal space*. All or part of terminal space might be displayed in the graphics area depending on the window you display (windows are explained later in this section).

Commands use *terminal space coordinates* to specify a location in terminal space. These coordinates are paired integers; each integer pair specifies a single screen position. The first value in each pair is the x-coordinate and specifies how far left or right. The second value is the y-coordinate and specifies how far up or down. Figure 3-3 shows the positions of certain coordinate pairs.

When you send terminal space coordinates to the terminal, they must be encoded in a special way. See "XY-Coordinates in Host Syntax" in Section 5 for instructions.

You can use any integer from 0 through 4095 for either the x- or y-coordinate. This range covers the entire terminal space. Each unit in this coordinate system is a *terminal space unit* (TSU).

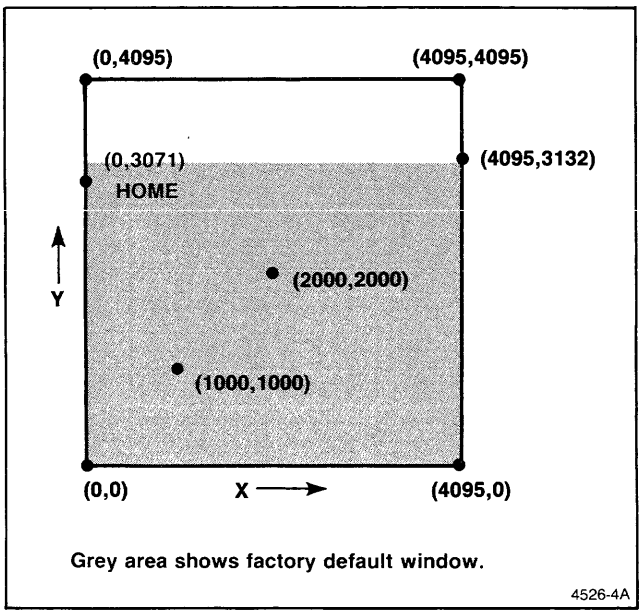


Figure 3-3. Terminal Space Coordinates.

Current Graphics Position. There is always one position in terminal space that is the *current graphics position*. This is often the position that was specified in the last graphics display operation and is usually the default position the terminal uses when none is specified. When the terminal is first turned on, position (0,3071) is the current position. This position is called the *home* position and is near the upper-left corner of the screen when the factory default window is used.

Windows

A *window* is a rectangular area in terminal space that you specify that you want to fill the screen. Specify the coordinate of the lower-left and upper-right corners of the window with the SET WINDOW command. These positions are placed at the corresponding corners of the screen. The display is linearly distorted unless the window has the 3:4 aspect ratio of the screen. (Aspect ratio is the ratio of height to width.) Figure 3-4 shows different windows displaying the same terminal space.

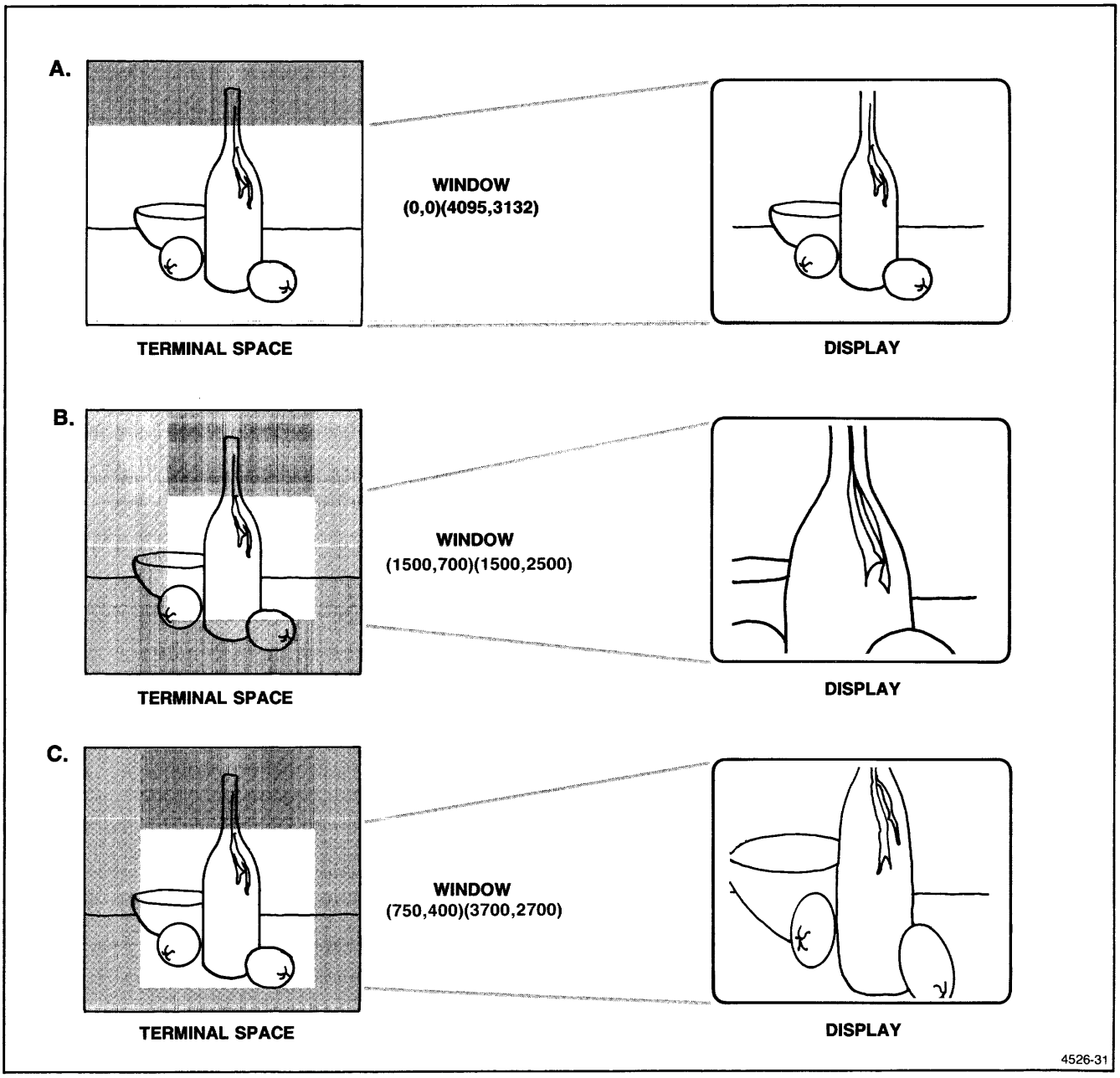


Figure 3-4. Examples of Windows.

To ensure that the display is not distorted, specify only the height or width of the window and specify zero for the other dimension. The terminal automatically replaces the zero height or width with a value that maintains a proper aspect ratio. Since you specify the corners of a window with xy-coordinates, a zero width means you use the same x-coordinate for both corners. A zero height means you use the same y-coordinate for both corners.

Since the terminal displays the dialog area in front of the graphics area, all or part of the window may be obscured by opaque portions of the dialog area. As explained earlier in this section, you can avoid obscuring the graphics area by issuing commands that make the dialog area invisible or transparent.

Lines

To draw lines, you can use either of two methods. One method uses a special mode, Vector mode, in which the terminal interprets messages from the host as encoded positions for line segment endpoints. The other method uses MOVE and DRAW commands to create lines.

Vector Mode. The ENTER VECTOR MODE command puts the terminal in Vector mode. When in this mode, the terminal interprets all characters the host sends, except for characters forming commands, as xy-coordinates. (Commands all begin with ^{E}C or a control character.) The first coordinate pair specifies where the line starts. When the terminal receives a second coordinate pair, it draws a line to that position and makes it the current position. As the terminal receives subsequent coordinates, it draws a line from the current position to the specified position. This continues until the terminal leaves Vector mode.

At times, you might want to start a line at a new position without drawing a line to that position. (This is analogous to lifting the pen from the paper in a drawing and moving it to a new location.) To do this, send another ENTER VECTOR MODE command. A new line will be started at the next position you specify.

At times, you might want to draw a line from the current position when the terminal is not yet in Vector mode. To do this, issue an ENTER VECTOR MODE command followed by the ^{B}L character. The terminal's bell sounds and a line will be drawn from the current position to the next position you specify.

MOVEs and DRAWs. The MOVE-and-DRAW method of creating lines uses individual commands to create lines. The terminal does not need to be in Vector mode to execute these commands.

The MOVE command sets the current position, but does not draw a line to this position. The command's effect is analogous to lifting a pen from paper and placing it at a new position in a drawing.

The DRAW command draws a line from the current position to the position specified in the DRAW command. The current position then becomes the position specified in the command.

Figure 3-5 compares how a line is created using Vector mode and how it is created using MOVEs and DRAWs.

Line Attributes. Before drawing a line on the screen, you can set both a line style and a line index for the line. The seven available line styles are shown with the SET LINE STYLE command description. The line index determines the line's color. With the SET LINE INDEX command you can assign any of the available color indices to the line before drawing it. (Color indices are explained later in this section.)

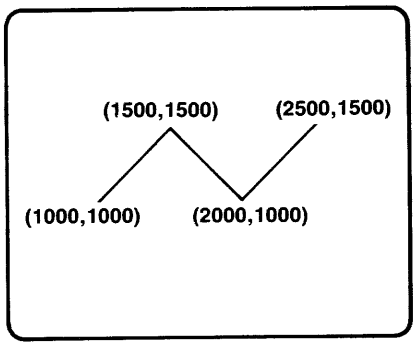
Markers

A *marker* is a small, predefined symbol that marks a position in terminal space. Before displaying a marker, you can choose any of eleven marker types (shown in the description of the SET MARKER TYPE command). You then use either of two methods to display markers. One method uses Marker mode. The other method uses an individual command, MARKER, to create each marker.

Marker Mode. The ENTER MARKER MODE command puts the terminal in Marker mode. The terminal then interprets all characters
it receives, except for those in commands, to be xy-coordinates. It places a marker at each specified position.

The only way to leave Marker mode is to issue the ENTER ALPHA MODE command. Alpha mode is the only mode you can enter directly from Marker mode.

DRAW MARKER Command. The MARKER command specifies a single position. The terminal displays, at the specified position, the type of marker selected in the most recent SET MARKER TYPE command. The terminal does not need to be in Marker mode when it executes the DRAW MARKER command.



MOVE and DRAW Method

Command Description	After Encoding
MOVE 1000,1000	^{E}cLF'`z'Z
DRAW 1500,1500	^{E}cLG+`w+W
DRAW 2000,1000	^{E}cLG'`z/T
DRAW 2500,1500	^{E}cLG+`w3Q

Vector Mode Method

Command Description	After Encoding
ENTER VECTOR MODE	^{G}S
1000,1000	'`z'Z
1500,1500	+`w+W
2000,1000	'`z/T
2500,1500	+`w 3Q
ENTER ALPHA MODE	^{U}S

4526-6A

Figure 3-5. Two Methods for Displaying a Line.

Panels

A panel is an area in terminal space that can be filled with any of 148 predefined patterns. Figure 3-6 shows some of these patterns.

To create a panel:

1. Send the SELECT FILL PATTERN command and specify the fill pattern you want to use. See this command's description for the available choices.

2. Send the BEGIN PANEL BOUNDARY command. This command specifies the position for the boundary line's starting position and sets whether the panel's boundary line should be displayed in the finished panel.

3. Define the panel's boundary line. Do this in either of two ways:

   a. Put the terminal in Vector or Marker Mode and then send the coordinates of boundary segment endpoints.

   b. Send MOVE and DRAW commands.

   You do not need to define the last segment that closes the panel.

4. Send the END PANEL command. This automatically creates a boundary line segment back to the starting position and displays the panel with its fill pattern. The current position is set to the panel's starting position.

If the terminal is in Marker mode during this process, markers are not displayed.

If the terminal is in Vector mode, the boundary line is displayed as the panel is created. Depending on the choice specified in the BEGIN PANEL BOUNDARY command, the boundary line is either displayed in the finished panel or covered by the fill pattern.

If a panel's boundary crosses itself as in Figure 3-6C, the terminal uses this rule to determine which areas are inside the panel: If, starting from a point distant from the panel, you cross the boundary an odd number of times to get to the area, the area is inside the panel; if you cross the boundary an even number of times, the area is outside the panel. Only areas inside the panel are filled.

As shown in Figure 3-6B, a panel can have more than one boundary. To make such a panel, issue additional BEGIN PANEL BOUNDARY commands before ending the panel. When you issue an additional BEGIN PANEL BOUNDARY command, this closes the boundary being created and starts a new boundary at the specified position. The panel, defined by multiple boundaries, is not filled until the terminal receives the END PANEL command.

Graphtext

Graphtext is a special type of text for use in the graphics area. You should always use graphtext in the graphics area because of its versatility and because it leaves the dialog area undisturbed. Graphtext can be resized, rotated, and written in different directions. The terminal lets you put alphatext in the graphics area, but this feature is provided primarily for compatibility with Tektronix 4010 Series terminals.

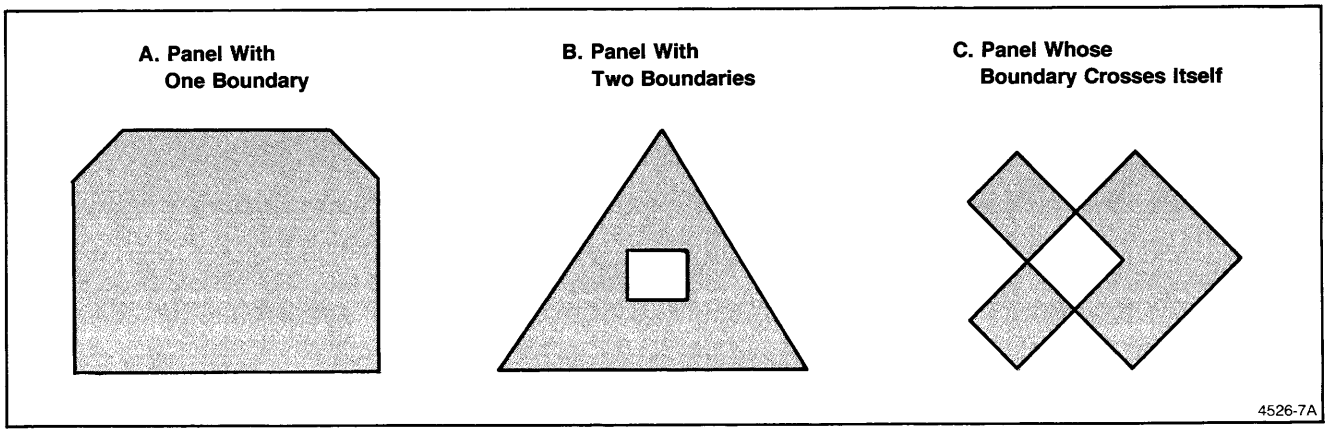


Figure 3-6. Examples of Panels.

Before you send the GRAPHIC TEXT command (the command that displays graphtext), you can set several characteristics of graphtext with other commands. See Figure 3-7 for a list of these commands and examples.

Only printable characters can be in the string you specify as graphtext. This includes the ASCII characters ^{S}P through ~ (ASCII decimal equivalents 32 through 126). The terminal generates an error for characters outside this range.

COLOR DISPLAY

When a command specifies the color of something on the screen, it does not state an absolute color, such as light-green or bluish-grey. Instead, it specifies an integer called a *color index*. Either a host program or the operator can select the color a particular color index is associated with. For example, when the terminal is shipped from the factory, Color Index 2 is set to mean "red"; so, anything you assign to Index 2 is displayed in red. If you then change Index 2's definition to "blue", anything displayed in Index 2 is blue.

Color Indices

A color index must be an integer in the range 0 through 7. Since all graphics area colors are specified with color indices, you can use eight different colors in the graphics area at any one time. The dialog area has its own set of eight color indices.

The color a color index produces can be set in several ways. When the terminal is shipped from the factory, the color indices are assigned default colors. A host program can assign colors with the SET SURFACE COLOR MAP command and the SET DIALOG AREA COLOR MAP command. The operator can assign colors with the color interface (see the *4105 Operators Manual*), the CMAP Setup command, and the DACMAP Setup command.

The SAVE NONVOLATILE PARAMETERS command saves the settings for indices so that they are retained even when the terminal is turned off.

Colors for color indices are defined by three characteristics: hue, lightness, and saturation. Each of these characteristics is identified by an integer. See the color cone in Appendix F for an illustration of these characteristics. Note that insufficient saturation and extreme values for lightness can obscure colors.

Characteristic	Command	Examples			
Rotation	SET GRAPHTEXT ROTATION	xABC	ABC (x)	CBA x	ABC (x)
		0°	90°	180°	270°
		Character Path is RIGHT in all examples.			
Character Path	SET CHARACTER PATH	xABC	C B A (x)	CBA x	x A B C
		Right	Up	Left	Down
		Rotation = 0° in all examples.			
Character Size	SET GRAPHTEXT SIZE	Height { ABC (Width)			

Note: x indicates current graphics position.

4526-8B

Figure 3-7. Graphtext Characteristics.

Dither Patterns

A predefined set of panel fill patterns, called *dither patterns*, mimic colors by displaying patterns of adjacent different-colored pixels. (A pixel is the smallest display element.) The pattern of different-colored pixels looks like a solid color. The dither patterns are identified by fill patterns 50 through 174.

Using Colors

To color objects on a display, you simply use commands to assign a color index to each object. For lines, set the color index with SET LINE INDEX before creating the line. For panels, use SELECT FILL PATTERN to choose either a solid color (identified by a color index) or another pattern. For text in the graphics area, use SET TEXT INDEX before creating the text. The SET DIALOG AREA INDEX command sets the color of text in the dialog area, character background, and dialog area background.

Assigning a new color to a color index immediately changes the color of all items assigned to that color index.

Effective Color Displays

Color is effective for:

- Separating categories of information
- Helping the operator locate a particular object in a cluttered display
- Cuing the operator that information has changed
- Making small symbols (such as markers) more obvious

Before assigning colors, first consider how the operator will use the display and then decide whether color will be helpful. In an uncluttered, structured display, color adds little to the operator's understanding. As the number of colors increases, the operator time required to respond to any one color increases. The chance for confusion among colors also increases.

Keep these points in mind when assigning colors:

- The four most useful colors are red, green, yellow, and blue. If you need additional colors, use orange, yellow-green, blue-green, and violet.
- Blue text and symbols are harder to read than text and symbols in other colors. (Blue is easier to see when its lightness level is greater than 50%.)
- Use the same color the same way in different displays; this creates easier and quicker recognition of the color's use.

GRAPHICS INPUT

Graphics Input (GIN) mode allows an operator to "point" to a graphics location on the terminal's display. In GIN mode, when the operator presses a key, the terminal sends the xy-coordinates of the desired location to the host.

Some commands used in GIN mode have "4010" in their title. This indicates that these commands originated in Tektronix 4010 Series terminals and differentiates the commands from input commands used by other Tektronix terminals.

These steps describe what typically happens in GIN mode:

1. The host sends an ENABLE 4010 GIN command to the terminal. The terminal is now in GIN mode.
2. The terminal displays the graphic cursor and the operator moves the cursor to the desired position by pressing the joydisk.
3. The operator presses any key that normally produces a character.
4. The terminal sends a 4010 GIN REPORT to the host. This report tells which key the operator pressed and the xy-coordinates of the graphics cursor.
5. The graphics cursor location becomes the current position and the terminal enters Alpha mode.

If the dialog area is not enabled, the alphatext cursor is set to the graphics cursor position.

If the dialog area is enabled, the terminal ignores the PAGE command and the ^{C}R character while in GIN mode. Pressing the G Eras key with the dialog area enabled erases the graphics area, but leaves the terminal in GIN mode. If the dialog area is not enabled, the G Eras key, PAGE command, or ^{C}R character terminate GIN mode and put the terminal in Alpha mode; no report is sent to the host.

While the terminal is sending a report to the host, the terminal is in Bypass mode. While in this mode, the terminal ignores input from the host. After the message is sent, the terminal returns to Alpha mode. (Bypass mode is explained in the *Communications* section.)

If a program needs the current position of the graphics cursor, but does not want to give the operator the opportunity to move the graphics cursor, it sends the REPORT 4010 STATUS command. The terminal reports the xy-coordinates of the graphics cursor immediately without giving the operator the chance to move the cursor.

MACROS

A *macro* is a string of characters identified by an integer called the *macro number*. Either the host or the operator can assign a string to a macro number with the DEFINE MACRO command. The terminal then stores the macro along with its identifying macro number.

After a macro is defined, a host program or an operator can issue an EXPAND MACRO command, which specifies a macro number. The terminal looks up the macro assigned to that macro number and uses that string. This process of looking up a macro number and using the stored macro is called *expanding a macro*.

For example, a host program can define macro 301 to be the string "Type any key to continue." Then, anytime the program needs this string displayed, instead of sending the entire message, it could just issue the terminal an EXPAND MACRO command specifying macro number 301. When the terminal receives this command, it looks up macro 301 in its memory and displays the string it finds. If it had found that macro 301 was a terminal command, it would execute that command.

Macros are also used another way. Each key or key combination indicates a particular macro number (details are explained later in this discussion). If a macro is defined for a key's macro number, when the operator presses that key, the terminal looks up the macro and uses the string of characters it finds instead of the character the key normally produces. A macro that can be expanded by pressing a key is called a *key macro*.

HOST MACROS

Host macros are the group of macros that can be expanded only in the EXPAND MACRO command. If the macro is a terminal command, when the macro is expanded, the terminal executes the command. If the macro is not a command, it is displayed on the screen, just like any message from the host.

Macros are efficient in many operations. For example, instead of sending a long string of characters, the host can send the much shorter EXPAND MACRO command. This saves data transmission time and ensures that the terminal receives the same string each time a particular macro is specified.

KEY MACROS

Each key or key combination is matched to macro numbers as shown in the tables with the DEFINE command description in Section 5. For keys that produce ASCII characters, the associated macro number is the ASCII decimal equivalent of that character. For example, the ASCII decimal equivalent of the uppercase P is 80. If macro number 80 has been defined, pressing uppercase P produces that macro. The uppercase P key is now *defined*.

In addition to being expanded by a keystroke, key macros can be expanded by the EXPAND MACRO command just as host macros are.

When the terminal first enters Edit mode, key macros are disabled. To enable them while in Edit mode, use the ENABLE KEY EXPANSION command.

Key Macro LEARN

You can easily define key macros from the keyboard by using the LEARN or NVLEARN Setup command. This method allows you to define a key by pressing the ASCII keys that make up the sequence instead of having to enter key codes.

Disabling Key Macros

The ENABLE KEY EXPANSION command (Setup command KEYEXPAND) lets you choose when you want key macros in effect. When key macros are not in effect, all characters produce their normal character; their are no special key definitions in effect. The key macros are still saved, however, and can be put back into effect by either the operator or a host program. If you disable key macros, they can still be expanded by the EXPAND MACRO command sent from the host.

Keeping a Key Macro Local

You might want a macro to only be displayed on the screen, or you might want a key macro to be interpreted as a command to the terminal. In these cases, you do not want the macro sent to the host, but want it executed locally. To create such a macro, follow these steps:

1. Send the SET KEY EXECUTE CHARACTER command to the terminal. This command defines one character to be the *key-execute character*. When this character is in a macro it means that following characters should be used locally until the key-execute character again appears in the string.

2. Send the DEFINE MACRO command to define the macro. Put a key-execute character at the beginning and the end of the string.

When the defined key is pressed, the characters enclosed by the key-execute characters are used locally. If these characters are a terminal command, they are executed. If they are not a terminal command, they are displayed as if the operator typed them. The characters are not sent to the host.

Key-execute characters in macros expanded by an EXPAND MACRO command have no special meaning; they are treated just like other characters in the macro. Macros expanded by the EXPAND MACRO command are always treated as if they were sent by the host.

When defining a macro to be used locally, make certain you include the second key-execute character. Otherwise, following key macros are used locally until the terminal encounters another key-execute character.

VOLATILE AND NONVOLATILE MACROS

The terminal can store a macro in two ways. A *volatile macro* is stored in volatile memory — the macro is not retained when the terminal is turned off or reset. A *nonvolatile macro* is stored in nonvolatile memory — when the terminal is turned off or reset, the macro is retained. When the terminal is turned on or reset, all nonvolatile macros are in effect.

A macro can be stored in both volatile and nonvolatile memory at the same time. When a macro is expanded, the terminal first checks volatile memory for the macro. If the macro is not found in volatile memory, the terminal continues its search in nonvolatile memory.

The DEFINE MACRO command defines only the volatile version of a macro. The DEFINE NONVOLATILE MACRO command defines both the volatile and nonvolatile version of a macro. To actually save the nonvolatile version in nonvolatile memory, you must issue a SAVE NONVOLATILE PARAMETERS command before turning off or resetting the terminal.

PIXEL OPERATIONS

NOTE

Pixel operations require that you have 4105 Option 30 Pixel ROMS or 4105F30 Pixel ROMS. You can see if your terminal has these ROMs installed by using the STATUS PIXELS Setup Command.

Pixel operations give the user a much faster way to place images on the screen or to modify images currently there. You can use pixel commands to quickly display and transmit very complex images — much more quickly than with line or panel graphics commands. Unlike line graphics that make up images with a series of MOVEs and DRAWs, pixel operations compose images with groups of pixels.

A *pixel* is the smallest screen element that a terminal can address. If you display a single pixel on the screen, you will see a tiny dot of color. If all the pixels in a given area are the same color, then the area is that color. But if the pixels in an area alternate between two or more different colors, the area appears to be a color different from any of the individual pixels.

Each pixel on the screen corresponds to a memory location in a special memory area called *raster memory*. As the terminal scans across the raster memory, it reads the number stored in each location. That number represents a *color index* in the color map. The terminal reads color indices from raster memory, looks in the color map for the entry corresponding to that index, and displays the pixel using the color mix defined by that entry. Thus, the screen display is a visual copy of the contents of the raster memory; and by changing the contents of raster memory, you can change the image on the screen. One method of doing this is to use *pixel operations*.

WAYS OF USING PIXEL OPERATIONS

All pixel writing operations use an *ALU mode* parameter, set by the BEGIN PIXEL OPERATIONS command. The ALU mode determines how the new color index information in a pixel writing command will affect the pixel information currently stored in raster memory. Depending on the ALU mode, a pixel writing command may erase the screen, replace one image with another, or combine images (see the BEGIN PIXEL OPERATIONS command description for details).

One useful feature of pixel operations is the off-screen memory. This is an area of raster memory that is not visible on the screen but can be written to and copied from. Off-screen memory is useful as a scratch pad, or as a storage area where you can store images for later retrieval. For example, you could store a special graphics character font in off-screen memory and retrieve each character as it was needed with a PIXEL COPY command.

You should be aware that the results of pixel commands differ significantly between different Tektronix terminals, primarily because of the different dimensions of raster memory. If you write a program for this terminal that uses pixel commands, you may need to rewrite the program so that it works on another Tektronix terminal. This is especially true if your program uses the off-screen memory.

WRITING INTO THE PIXEL VIEWPORT

You can access the individual pixels of the terminal's raster memory using the terminal's pixel commands. These commands fall into two categories: preparation commands and pixel-writing commands. There are three preparation commands: BEGIN PIXEL OPERATIONS, SET PIXEL VIEWPORT, and SET PIXEL BEAM POSITION; and there are four pixel-writing commands: RASTER WRITE, RUNLENGTH WRITE, PIXEL COPY, and RECTANGLE FILL. The following examples briefly explain each of these commands, with the exception of PIXEL COPY. The PIXEL COPY command copies pixels from one rectangular region in raster memory space to another rectangular region. Refer to the command descriptions in for additional information about these commands.

Figure 3-8 shows commands that define a pixel viewport and write some color indices into that pixel viewport. (The commands are shown in both Setup syntax and Host syntax.) Let's consider these commands, one by one.

PXBEGIN. In this example, no parameter values are given. Therefore, the PXBEGIN (BEGIN PIXEL OPERATIONS) command assumes default parameters 1, 11, and 6. These parameters specify Graphics Surface 1 as the pixel-writing surface, set the ALU mode to 11 (Replace), and specify that 6 bits per pixel be used to store each color index.

PXVIEWPORT 100 100 109 109. The PXVIEWPORT (SET PIXEL VIEWPORT) command defines a rectangular region on the pixel-writing surface. In this example, the lower-left and upper-right corners of the pixel viewport are at (100,100) and (109,109), respectively. These coordinates are in 480-by-360 raster memory space.

PXRECTANGLE 100 100 109 109 0. The PXRECTANGLE (RECTANGLE FILL) command sets all pixels within a rectangular region to the same color index. In this example, the command clears the pixel viewport by setting all pixels in that region to Color Index 0.

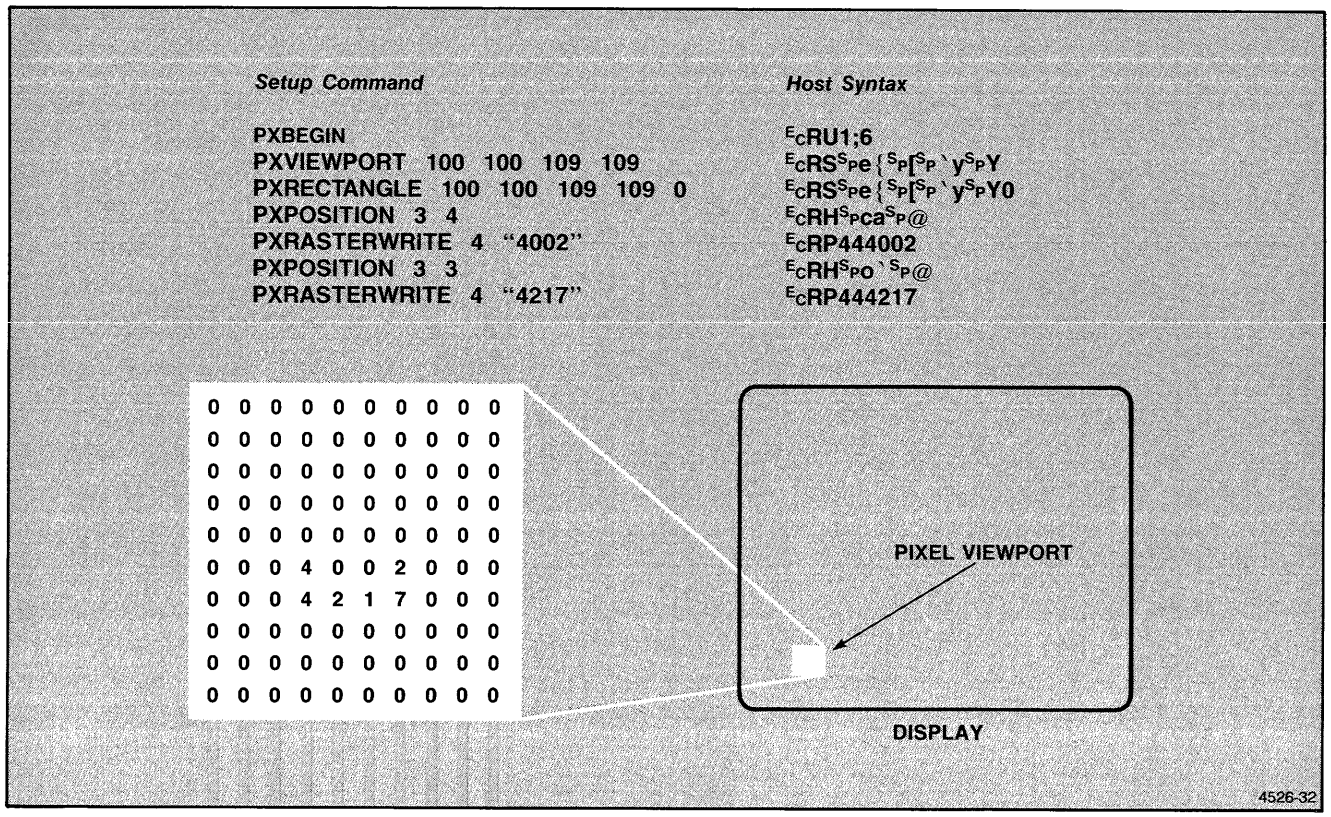


Figure 3-8. Writing Into the Pixel Viewport Using RASTER WRITE.

PXPOSITION 3 4. The PXPOSITION (SET PIXEL BEAM POSITION) command moves the pixel beam to the point (3,4). These coordinates are relative to the lower-left corner of the pixel viewport.

PXRASTERWRITE 4 "4002". The PXRASTERWRITE (RASTER WRITE) command writes the Color Indices 4, 0, 0, and 2 into four successive pixels. At the end of this command, the pixel beam position is at (7,4). These coordinates are relative to the lower-left corner of the pixel viewport.

PXPOSITION 3 3. This moves the pixel beam to the position (3,3). These coordinates are relative to the lower-left corner of the pixel viewport.

PXRASTERWRITE 4 "4217". This writes the Color Indices 4, 2, 1, and 7 into four successive pixels. At the end of this operation, the pixel beam position is at (7,3). These coordinates are relative to the lower-left corner of the pixel viewport.

You can use PXRUNLENGTHWRITE (RUNLENGTH WRITE) commands as well as PXRASTERWRITE commands to write in the pixel viewport. PXRUNLENGTHWRITE (using an array of specially encoded integers called *runcodes*) specifies the number of pixels in a sequence, and sends the same color index to each pixel in the sequence. The next three paragraphs and Figure 3-9 explain how this is done.

As in Figure 3-8, Figure 3-9 uses PXBEGIN (this time with bits-per-pixel set to 3) and PXVIEWPORT commands to define a pixel viewport that is ten pixels wide and ten pixels high. As before, a PXRECTANGLE command is used to set all pixels in the pixel viewport to Color Index 0. This time, however, there is no PXPOSITION command, so the pixel beam position starts at the upper-left corner of the pixel viewport.

In this example (Figure 3-9), the RUNLENGTH WRITE command has four runcodes in its integer-array parameter. Each runcode is a single number into which two other numbers are packed. The runcodes are packed using the form:

Runcode = (number-of-pixels) * (2^n) + (color-index)

where n = number-of-bits-per-pixel

The *bits-per-pixel* parameter from the most recent BEGIN PIXEL OPERATIONS command supplies the value for n unless that parameter is 4 or 6; then the value of n is 3.

The first code, 160, calls for 20 pixels of Color Index 0 (20 × (2^3) + 0 = 160). The second code, 243, calls for 30 pixels of Color Index 3 (30 × (2^3) + 3 = 243). The third code, 160, calls for 20 pixels of Color Index 0; it is the same as the first code. The fourth code, 246, calls for 30 pixels of Color Index 6 (30 × (2^3) + 6 = 246).

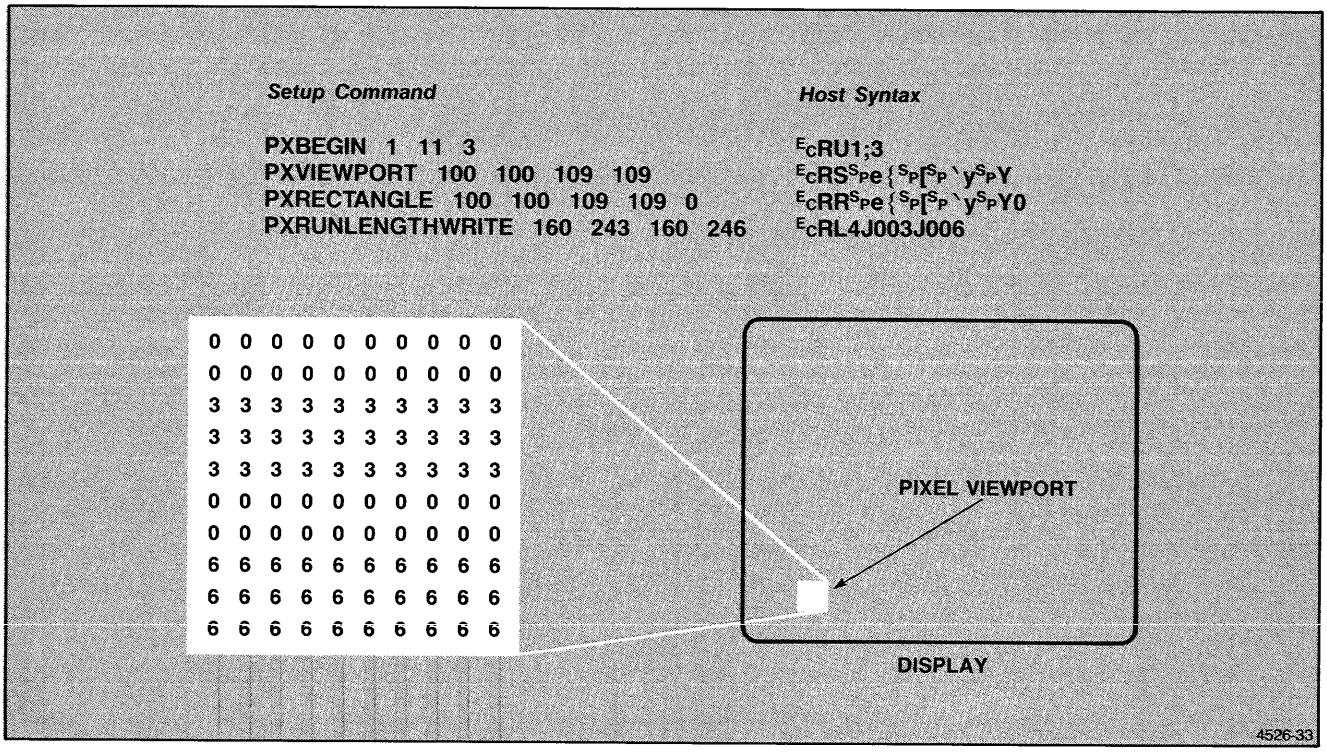


Figure 3-9. Writing into the Pixel Viewport Using RUNLENGTH WRITE.

Section 4

SCREEN EDITING SUPPORT

INTRODUCTION

This section covers the text entry and editing functions of the terminal. Explanations include:

- A discussion of how to use this terminal with screen editing programs
- A description of the syntax conventions used in the command descriptions
- A detailed description of each command

SCREEN EDITING CONCEPTS

Screen editing programs allow you to view and edit a computer file. This terminal has a fully addressable cursor and can insert, erase, or delete characters or lines. You can use any existing screen editing program with this terminal as long as it uses commands that are compatible with the ANSI X3.64 command set.

The terminal's command set is a subset of the ANSI X3.64 and ISO 6429 standards. This command set is compatible with most screen editing programs that issue ANSI X3.64 commands.

SCREEN EDITING FEATURES

The cursor is displayed on the screen as an underline at the current cursor position. When you execute text manipulation commands, they affect the characters displayed on the screen relative to the position of the cursor. The line of text on which the cursor is located is referred to in this manual as the *current line*. The position of the cursor on that line is referred to as the *cursor position*.

You can set the display characteristics of the cursor with the ACURSOR Setup mode command. This command lets you select the cursor blink and color. Refer to the *4100-Style Parameter Types, Commands, and Reports* section of this manual for more information.

Each character on the screen has a unique horizontal row and vertical column address. You can specify exact cursor positions using these row and column addresses.

You can use the SGR (Select Graphic Rendition) command (described later in this section) to select the display style of the text from the following:

- Normal — White characters on a transparent background
- Bold — Characters displayed in Color Index 2 (default is red)
- Underscore — Underlined characters
- Blink — Blinking characters
- Reverse video — Black character on a white background
- Colors — Choice of eight colors for characters, character background, and dialog background

OPERATING MODES

The terminal has four major operating modes and many submodes that determine how commands affect the display. The major modes are Ansi mode, Setup mode, Tek mode, and VT52 mode. All screen editing must be done in Ansi mode or VT52 mode. Figure 4-1 shows the relationship of these modes to each other.

To use a screen editing program:

- The terminal's dialog area must be visible.
- The terminal must be in Ansi mode so it can accept ANSI commands (use the SELECT CODE command with the Ansi parameter), or in VT52 mode so that it can accept VT52 commands (use the SELECT CODE command with the VT52 parameter).

You can set each Ansi submode individually, or you can use Edit mode. Entering Edit mode automatically sets all Ansi submodes so that the terminal is compatible with most VT100 software. When you enter Edit mode, the terminal automatically:

- Enters Ansi mode
- Sets Origin mode to Absolute
- Sets Insert/Replace mode to Replace
- Sets the dialog buffer to 24 lines
- Makes the dialog area visible
- Defines a scrolling region the size of the visible dialog buffer (24 lines)
- Disables all macro expansion

You can enable macro expansion with the KEYEXPAND YES Setup mode command.

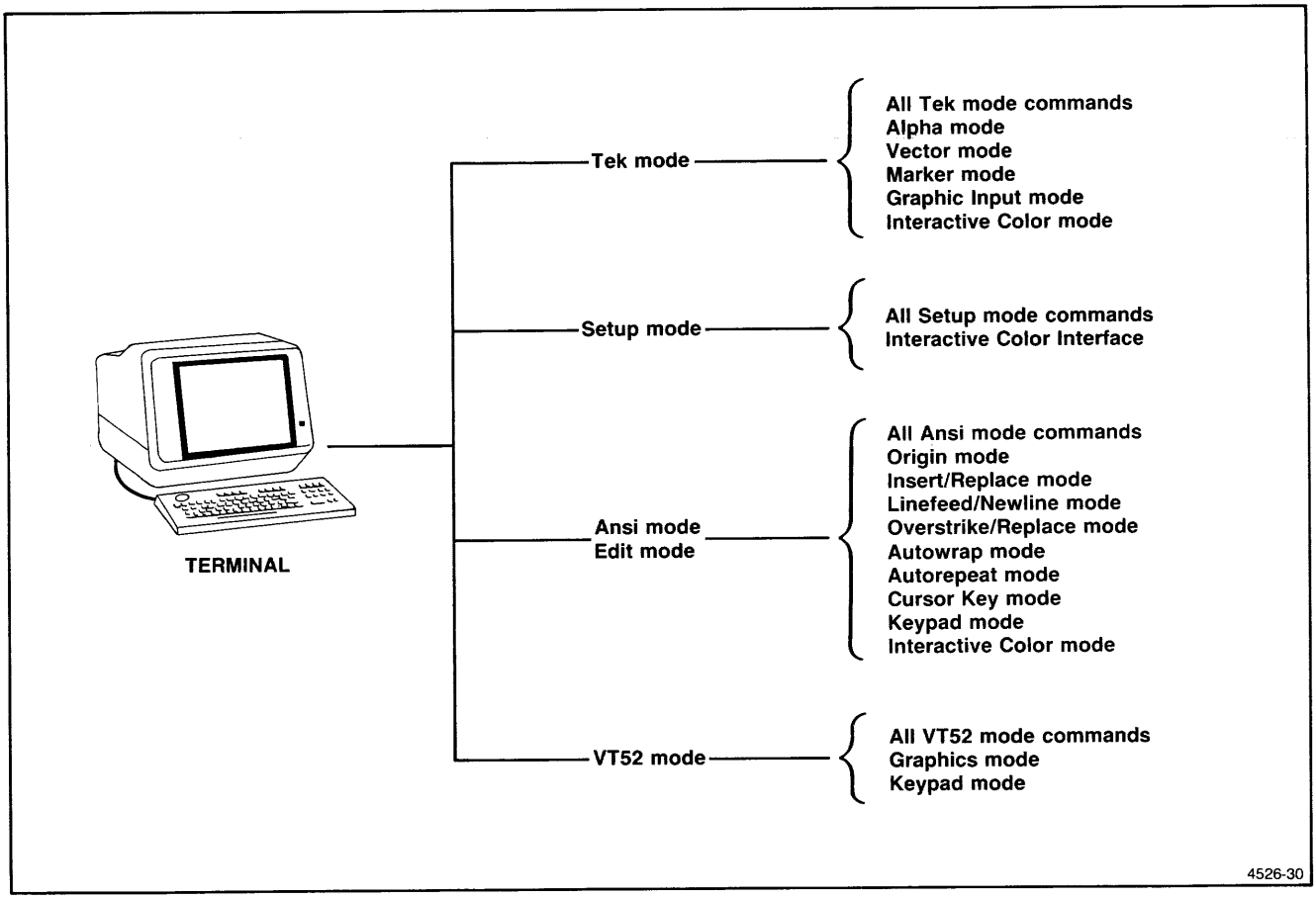


Figure 4-1. Terminal Modes.

While in Ansi or Edit mode, you can set or reset the submodes to change the operating characteristics of the terminal:

- Origin mode — Determines how the cursor is addressed in the fixed and scrolling regions of the dialog buffer.
- Insert/Replace mode — Determines how text is entered at the cursor position. In Insert mode, entered characters are inserted preceding the cursor position, which moves the character at the cursor position and all characters to the right of the cursor toward the right of the screen. In Replace mode, entered characters replace existing characters.
- Linefeed/Newline mode — Determines the position of the cursor when a linefeed is performed. In Linefeed mode, when the cursor is moved down a line with a ^{L}F character, it remains in the same column on the new line. In Newline mode, the ^{L}F character moves the cursor to the beginning of the new line. Linefeed/Newline mode does not affect cursor movement commands such as CUD (Cursor Down) and IND (Index).
- Overstrike/Replace mode — Determines how ^{S}P (Space) and _ (Underscore) are handled on the terminal screen. In Replace mode, ^{S}P and _ are displayed as visible characters. In Overstrike mode, ^{S}P moves the cursor one character to the right and _ underlines the current character.
- Autowrap mode — Determines what the cursor does at the end of a line. When this mode is set, characters entered in Column 80 wrap around to the next line. When this mode is reset, characters entered in Column 80 write over any existing character in Column 80.
- Autorepeat mode — Enables the Autorepeat feature of the keyboard. When this mode is set, keyboard keys repeat when held down. When this mode is reset, keyboard keys do not repeat when held down.
- Cursor Key mode — Controls the effect of function keys F1 through F4.
- Keypad mode — Causes the keypad to send either numeric or application codes when pressed.
- Column mode — Specifies whether the dialog area will be 80 or 132 columns wide. Only 80 of the 132 columns are displayed at any time, but scrolling horizontally will bring any of the columns into view.
- Screen mode — Reverses the colors on the display.

While in VT52 mode, you can enter or exit two submodes by issuing one of the commands associated with those modes:

- Graphics mode — Entering this mode selects the Rulings characters as the G0 character set; exiting this mode restores the character set that was in effect (as G0) prior to entering this mode.
- Alternate Keypad mode — Entering this mode gives the numeric keypad and function keys F5 — F8 meanings different than the factory default or programmed meanings; exiting this mode restores the factory default or programmed meanings.

If you switch back and forth between Ansi and VT52 modes, the numeric keypad retains its mode selection, although the mode name changes. Therefore, if the keypad sends numeric codes while in Ansi mode, then it sends factory default meanings when it enters VT52 mode. If the keypad sends applications codes while in Ansi mode, then it sends the alternate keypad meanings in VT52 mode (see the VT52 command description ENTER ALTERNATE KEYPAD MODE).

THE DIALOG AREA

All text editing and entry occur in the *dialog area*. The dialog area is a display area of the terminal that is separate from the graphics area. The ANSI commands work only in the dialog area. Before executing any program that uses the dialog area, make it visible with the SET DIALOG AREA VISIBILITY command or by pressing the Dialog key.

Since the syntax of the Ansi, Tek, and VT52 mode commands are not compatible, you must send the SELECT CODE command to the terminal with a parameter of 1 (to select Ansi mode) or 2 (to select Edit mode). Alternatively, the operator can type CODE EDIT or CODE ANSI in Setup mode. Once you have done this, you can use ANSI commands to manipulate the text in the dialog area.

The *dialog buffer* can have as many lines as there is memory available to hold them, with up to 30 lines visible on the screen at any one time. You can set the number of lines in the dialog buffer with the SET DIALOG AREA BUFFER SIZE command. You can set the number of lines that are visible with the SET DIALOG AREA LINES command. If the dialog buffer has more than 30 lines, the excess lines are not visible on the screen, but can be scrolled into view.

The ANSI command set allows the terminal to have up to two *fixed regions* and one *scrolling region* on the screen at the same time. This can be useful when, for example, your screen editor program has a status or message line at the top or bottom of the text editing region; the editing region is the scrolling region and the status lines are the fixed regions.

You can set the *edit margins* of the scrolling region with the SET TOP AND BOTTOM MARGINS command. If the edit margins coincide with the terminal screen size, then there are no fixed regions and the scrolling region contains the entire dialog buffer. Otherwise, the lines between and including the edit margins are in the scrolling region, and the lines outside the scrolling region are in the fixed regions.

The cursor can be moved from the scrolling region into a fixed region only by entering a CUP (Cursor Position) or HVP (Horizontal and Vertical Position) command while in Origin mode Absolute. Figure 4-2 shows how the dialog buffer and edit margins relate to the terminal display.

If the cursor is visible, the dialog buffer is rolled up or down as cursor movement commands are performed, so the cursor always remains in view. The exceptions to this are the SU (Scroll Up) and SD (Scroll Down) commands (and the equivalent joydisk movements). In scrolling, the dialog buffer is rolled up or down and the cursor maintains its position with respect to the dialog buffer, therefore, the cursor may move out of view.

If the cursor is not visible, cursor movement commands will not cause the dialog buffer to be rolled up or down.

Origin mode determines how the cursor is addressed in these fixed and scrolling regions. In Origin mode Absolute, you can address any character position relative to the upper-left character position of the dialog buffer (character address Row 1, Column 1). The scrolling region and the dialog buffer size are limited to the size of the terminal screen (30 lines maximum). In Origin mode Relative, you can address any character position in the scrolling region relative to the upper-left character position of the scrolling region. If edit margins have not been set, then the scrolling region is the same size as the dialog buffer (up to 49 lines).

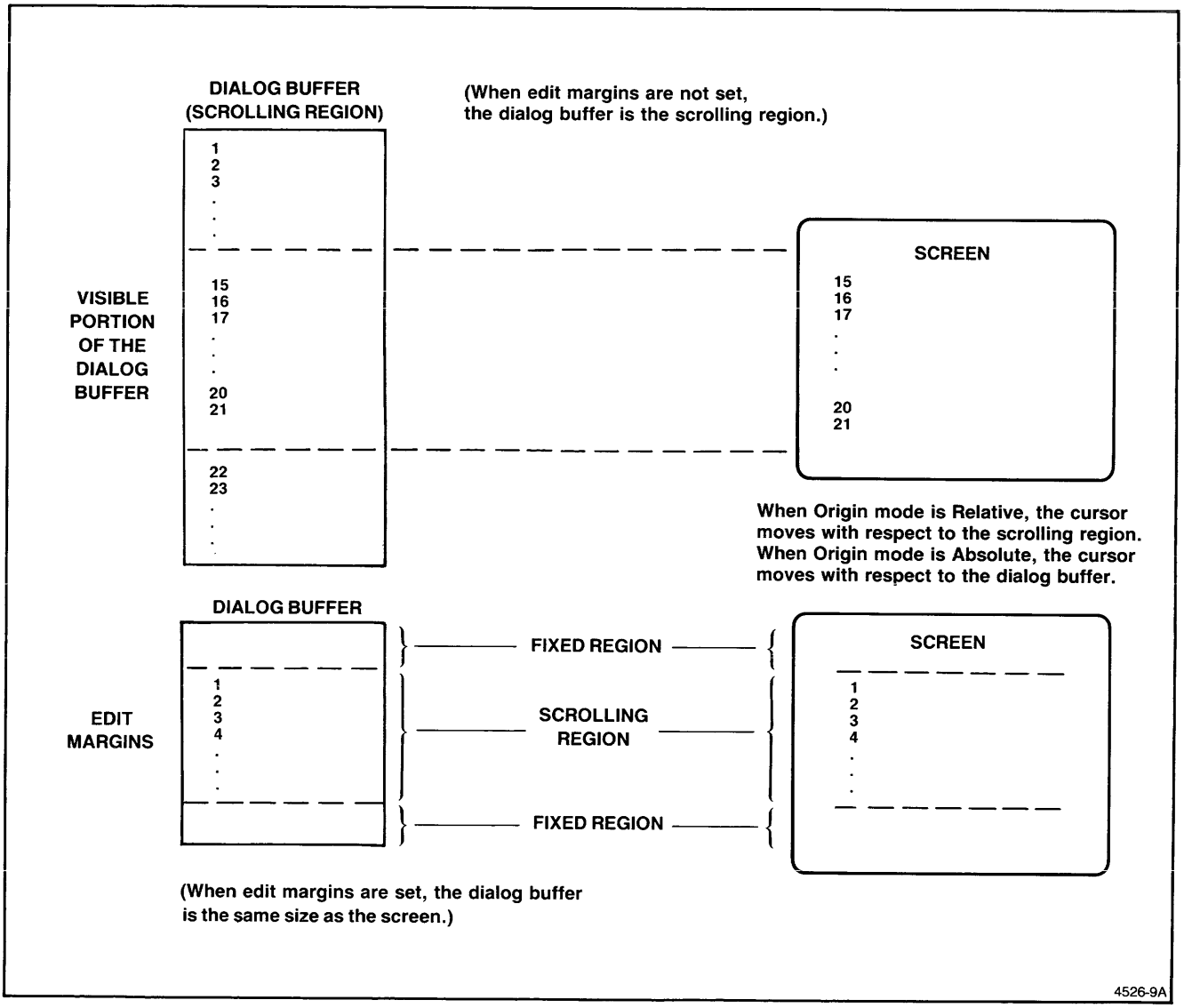


Figure 4-2. Fixed and Scrolling Regions in the Dialog Area.

ANSI MODE COMMANDS

The Ansi mode commands allow you to:

- Position the cursor
- Configure the different regions by setting the edit margins
- Delete characters and lines
- Erase characters and lines
- Insert characters and lines
- Set the attributes of displayed characters
- Control functions and modes

Cursor movement commands, such as CUU (Cursor Up) and CUP (Cursor Position), allow you to move the cursor relative to its current position or directly to a specified screen location.

Deletion commands, such as DCH (Delete Character) and DL (Delete Line), allow you to delete characters or lines in the dialog buffer.

Erasure commands, such as ED (Erase in Display), allow you to erase or blank characters or lines in the dialog buffer.

Insertion commands, such as IC (Insert Character) and IL (Insert Line), allow you to insert characters or lines in the dialog buffer.

Other commands allow you to save and restore the cursor position, change the appearance of displayed text, set and reset modes (such as Insert/Replace or Overstrike/Replace mode), or receive status reports from the terminal.

VT52 MODE COMMANDS

The VT52 mode commands allow you to:

- Position the cursor
- Erase portions of text
- Control functions and modes

Cursor movement commands move the cursor up, down, left, and right with respect to the dialog buffer.

Erasure commands erase portions of lines or portions of the screen from the current position to the end of the line or bottom of the screen.

Some commands, such as IDENTIFY, control functions; others, such as ENTER GRAPHICS MODE, control modes.

EDITING ON THIS TERMINAL

Using an Existing Editor That Understands ANSI X3.64 or VT52 Commands

You can use any existing screen editing program with this terminal as long as it sends commands to the terminal that are compatible with the ANSI X3.64 command set or the VT52 command set. You may not be able to use all of your editor's features with this terminal, and your editor may not be able to use all of this terminal's features. The detailed command descriptions provided later in this section explain how the terminal responds to each particular ANSI or VT52 command.

If your editing program is compatible with VT100 terminals, enter the SELECT CODE command with **Edit** as a parameter, which configures the terminal to run most VT100 applications programs.

If your editing program is compatible with VT52 terminals, enter the SELECT CODE command with **VT52** as a parameter, which configures the terminal to run most VT52 applications programs. The terminal should be in Edit mode prior to selecting VT52 mode so that the terminal characteristics set by Edit mode will remain in effect.

The documentation supplied with your particular editor should define any deviation from ANSI X3.64.

Designing an Editor to Work With This Terminal

If you are designing a new text editor for this terminal, you should carefully study the detailed command descriptions provided later in this section. The editing program must be designed to send the appropriate terminal command to perform the desired screen function.

Terminal Initialization File

Most screen editing programs use some type of terminal initialization file that allows them to work with more than one type of terminal. This file is basically a command translation table that tells the editing program what particular commands the terminal requires to perform the editing functions. Refer to the documentation supplied with your editor for more information.

ANSI AND VT52 COMMAND AND REPORT DESCRIPTIONS

Command Conventions

The following points summarize the syntax used in the command descriptions in this section:

- The command name is shown in all uppercase characters.
- Characters shown in bold type are those that you must enter exactly as shown.
- In Host syntax, enter parameters on the same line and separate them with semicolons. If you are entering more than one parameter, any parameters that begin with **?**, <, =, or >should be at the beginning of the list of parameters.
- In Setup syntax, enter parameters on the same line and separate them with a space or a comma.
- Most commands take integer values for their parameters. Integers do not need to be packed. The valid range is 0—32767. If you specify a value higher than is reasonable for a particular parameter, the parameter defaults to the highest value that it can accept. For example, if you specify column 200, the parameter defaults to 132.
- Two commands, RM (Reset mode) and SM (Set mode), take string parameters. The valid character strings are shown with each command.
- Multiword items are joined by hyphens. For example, *descriptive-parameter-name* and *optional-parameter* are single items described by two or more words.
- Default parameter values, if any, are shown at the end of each parameter description. Appendix D contains a table that lists all of the parameters and their default values.

Figure 4-3 illustrates the command description format.

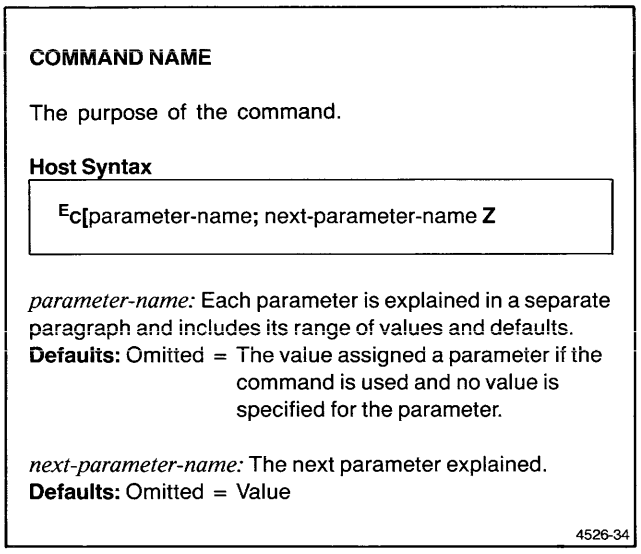


Figure 4-3. Command Description Format for ANSI and VT52 Commands.

ANSI COMMANDS

$^{B}_{L}$ (Bell Character)

Sounds the terminal's bell.

$^{B}_{S}$ (Backspace Character)

Moves the cursor left one position.

The $^{B}_{S}$ character moves the cursor one character position to the left. If the cursor is already at Column 1, then $^{B}_{S}$ has no effect.

CBT (Cursor Backward Tab)

Moves the cursor backwards to a preceding tab stop on the current line.

Host Syntax

$^{E}_{C}$[number-of-preceding-tab-stops **Z**

number-of-preceding-tab-stops: Enter 1 to move the cursor to the preceding tab stop. A value greater than 1 (*n*) moves the cursor to the *n*th preceding tab stop on the current line. If there are less than *n* preceding tab stops, the cursor moves to Column 1 of the current line.
Default: Omitted or 0 = 1

CHT (Cursor Horizontal Tab)

Moves the cursor forward to a following tab stop on the current line.

Host Syntax

$^{E}_{C}$[number-of-following-tab-stops **I**

number-of-following-tab-stops: Enter 1 to move the cursor to the next tab stop. A value greater than 1 (*n*) moves the cursor to the *n*th next tab stop on the current line. If there are less than *n* following tab stops, the cursor moves to the end of the current line.
Default: Omitted or 0 = 1

$^{C}_{N}$ (Cancel Character)

Cancels an Ansi mode command in progress.

CPR (Cursor Position Report)

Provides the row and column address of the current cursor position.

Host Syntax

$^{E}_{C}$[row ; column **R**

The CPR (Cursor Position Report) message is sent from the terminal to the host in response to a DSR (Device Status Report) command with 6 for a parameter.

If Origin mode is Relative (TEKOM set), the coordinates reported are row, column coordinates in the scrolling region. Row 1, Column 1 is the upper-left corner of the scrolling region.

If Origin mode is Absolute (TEKOM reset), the coordinates reported are row, column coordinates of the dialog buffer. Row 1, Column 1 is the upper-left corner of the dialog buffer.

The terminal does not enter Bypass mode for the CPR.

$^{C}_{R}$ (Carriage Return Character)

Moves the cursor to the first column in the current line.

If Carriage Return/Linefeed ($^{C}_{R}{}^{L}_{F}$) mode is set, a linefeed action is also performed.

CUB (Cursor Backward)

Moves the cursor left the specified number of columns.

Host Syntax

^{E}C[number-of-columns **D**

number-of-columns: The number of columns to move the cursor toward the left side of the screen. The cursor does not move beyond Column 1.
Default: Omitted or 0 = 1

If Column mode is set to 132, the cursor may disappear from the screen. This command will not scroll horizontally to keep the cursor in view.

CUD (Cursor Down)

Moves the cursor down the specified number of lines.

Host Syntax

^{E}C[number-of-lines **B**

number of lines: The number of lines to move the cursor down.
Default: Omitted or 0 = 1

If Origin mode is Absolute (TEKOM reset), the cursor moves with respect to the dialog buffer.

If Origin mode is Relative (TEKOM set), the cursor moves with respect to the scrolling region.

CUF (Cursor Forward)

Moves the cursor the specified number of columns to the right.

Host Syntax

^{E}C[number-of-columns **C**

number-of-columns: The number of columns to move the cursor toward the right side of the screen. The cursor does not move beyond the rightmost margin.
Default: Omitted or 0 = 1

If Column mode is set to 132, the cursor may disappear from the screen. This command will not scroll horizontally to keep the cursor in view.

CUP (Cursor Position)

Moves the cursor to a specified row and column.

Host Syntax

^{E}C[row-number; column-number **H**

row-number: The destination row for the cursor.
Default: Omitted or 0 = 1

column-number: The destination column for the cursor.
Default: Omitted or 0 = 1

If Origin mode is Relative (TEKOM set), the cursor moves with respect to the scrolling region. Row 1, Column 1 is the upper-left corner of the scrolling region.

If Origin mode is Absolute (TEKOM reset), the cursor moves with respect to the dialog buffer. Row 1, Column 1 is the upper-left corner of the dialog buffer.

CUU (Cursor Up)

Moves the cursor upward the specified number of lines.

Host Syntax

Ec[number-of-lines **A**

number-of-lines: The number of lines to move the cursor toward the top of the screen.
Default: Omitted or 0 = 1

If Origin mode is Absolute (TEKOM reset), the cursor moves with respect to the dialog buffer.

If Origin mode is Relative (TEKOM set), the cursor moves with respect to the scrolling region.

DA (Device Attributes)

Tells the terminal to report what type of terminal it is.

Host Syntax

Ec[device-status-request **c**

device-status-request: This parameter is always 0.
Default: Omitted = 0

The host sends this command with a parameter of 0 to the terminal asking it to identify what type of terminal it is. The terminal sends back to the host the report **Ec[?1;2c**, which says that the terminal is similar to a VT100 with Advanced Video Option. This means that the terminal includes:

- 132 Column mode
- Bold, blink, underline, and reverse image character attributes

If the host echos this report back to the terminal, the terminal ignores the echo.

DCH (Delete Character)

Deletes the character at the cursor and characters following the cursor depending on the value of the parameter.

Host Syntax

Ec[number-of-characters **P**

number-of-characters: The number of characters to delete.
Default: Omitted or 0 = 1

Any characters to the right of the deleted characters are moved left by the same number of character positions; thus the gap is filled and the remainder of the line to the right of the last character is filled with spaces.

Only characters on the current line are affected by this command.

DL (Delete Line)

Deletes the specified number of lines starting with the current line.

Host Syntax

Ec[number-of-lines **M**

number-of-lines: The number of lines to delete.
Default: Omitted or 0 = 1

All lines following the deleted lines (including those in the invisible portion of the scroll) are shifted in a block toward the line containing the cursor. The cursor does not change position. If fixed and scrolling regions have been defined, this command only affects lines in the region that the cursor is currently in. For example, if the cursor is in the top fixed region, only the lines in the top fixed region are affected.

DMI (Disable Manual Input)

Disables the keyboard.

Host Syntax

```
EC `
```

This command is the equivalent of the Ansi mode SM command with the KAM parameter or of the 4100-style command LOCK KEYBOARD with a parameter of 1.

DSR (Device Status Report)

Generates a CPR (Cursor Position Report) or a DSR (Device Status Report).

Host Syntax

```
EC[ status n
```

status: Determines whether the DSR is a command from the host or a report from the terminal (see Table 4-1 for parameter values).

When the host sends a DSR command with a parameter of 5, the terminal responds with a DSR message telling whether or not there is a malfunction. When the host sends a DSR command with a parameter of 6, the terminal responds with a CPR (see CPR command for the meaning of the response). If the terminal receives a DSR with a parameter value of 0 or 3 (which could be the echo of a report it has sent to the host), the terminal ignores that DSR.

Table 4-1

DEVICE STATUS REPORT PARAMETERS

Parameter	Meaning
0	Report from terminal: ready — no malfunctions detected
3	Report from terminal: malfunction — try again
5	Command from host: report status using a DSR
6	Command from host: report cursor position using a CPR (see CPR command)

ECH (Erase Character)

Erases the specified number of characters starting at the cursor position.

Host Syntax

```
EC[ number-of-characters X
```

number-of-characters: The number of characters to erase.
Default: Omitted or 0 = 1

Characters are erased, not deleted. When a character is erased, its character cell is cleared (replaced with the current erase color index). The cursor location remains unchanged.

The effect of the ECH command is not confined to the current line. For example, if the cursor is in Column 41, and an ECH command with a parameter of 45 is issued, the 45 characters at and following the cursor position are erased. This is true even if this means erasing characters on following lines and into the fixed region from within the scrolling region.

ED (Erase in Display)

Erases part or all of the current display, according to the parameter.

Host Syntax

```
EC[ erase-extent J
```

erase-extent: Enter 0 to erase text from the cursor position to the end of the dialog buffer. Enter 1 to erase text from the beginning of the dialog buffer to the cursor position. Enter 2 to erase the entire dialog buffer.
Default: Omitted = 0

The cursor does not change position.

EL (Erase in Line)

Erases part or all of the current line, according to the parameter.

Host Syntax

^{E}C[erase-extent **K**

erase-extent: Enter 0 to erase text from the cursor position to the end of the line. Enter 1 to erase text from the beginning of the line to the cursor position. Enter 2 to erase the entire line.
Default: Omitted = 0

EMI (Enable Manual Input)

Enables the keyboard.

Host Syntax

^{E}cb

This command is the equivalent of the ANSI-style RM command with the KAM parameter or of the 4100-style LOCK KEYBOARD command with a parameter of 0.

$^{F}_{F}$ (Formfeed Character)

The terminal inserts $^{F}_{F}$ into the dialog area and advances the cursor position. This character indicates the start of a new page to a hardcopy unit (see command description for Tek mode command SET DIALOG HARDCOPY ATTRIBUTES).

$^{H}_{T}$ (Horizontal Tab Character)

Advances the cursor forward to the next horizontal tab stop on the current line.

If there are no horizontal tab stops to the right of the cursor position, the cursor moves to the last column of the line.

The tab stops that are in effect when the terminal is turned on are at columns: 1, 9, 17, 25, 33, 41, 49, 57, 65, 73, 81, 89, 97, 105, 113, 121, 129. These power-up tab stops can be changed and saved with the SAVE NONVOLATILE PARAMETERS command.

HTS (Horizontal Tab Set)

Sets a tab stop at the current cursor location or sets the tab stops for the entire screen.

Host Syntax

^{E}cH

Setup Syntax

TABS list-of-tab-stops

list-of-tab-stops: A list of numbers representing the column numbers where you want tab stops set.
Default: Factory default = 1, 9, 17, 25, 33, 41, 49, 57, 65, 73, 81, 89, 97, 105, 113, 121, 129.

The tab stops that are in effect when the terminal is turned on are those stops that have been saved in nonvolatile memory.

HVP (Horizontal and Vertical Position)

Moves the cursor to a specified row and column of the screen.

Host Syntax

^{E}C[row-number ; column-number **f**

row-number: The destination horizontal row number for the cursor.
Default: Omitted or 0 = 1

column-number: The destination vertical column number for the cursor.
Default: Omitted or 0 = 1

If Origin mode is Relative (TEKOM set), the cursor moves with respect to the scrolling region. The cursor does not move beyond the top and bottom of the scrolling region.

If Origin mode is Absolute (TEKOM reset), the cursor moves with respect to the dialog buffer. "Row 1, Column 1" is the upper-left corner of the dialog buffer. The cursor does not move beyond the top and bottom of the dialog buffer.

ICH (Insert Character)

Inserts the specified number of Space characters at the cursor position.

Host Syntax

^{E}c[number-of-characters @

number-of-characters: The number of Space characters to insert.
Default: Omitted or 0 = 1

The character currently at the cursor position and all other characters to the right of the cursor are shifted *n* columns to the right. Characters shifted off the end of the line are lost. The cursor position remains unchanged.

IL (Insert Line)

Inserts the specified number of blank lines in place of the current line.

Host Syntax

^{E}c[number-of-lines L

number-of-lines: The number of lines to insert.
Default: Omitted or 0 = 1

The current line and all following lines are shifted down. Lines scrolled below the bottom margin are lost. The cursor position does not change.

If fixed and scrolling regions have been defined, this command only affects lines in the region that the cursor is currently in (if the cursor is in the scrolling (nonfixed) region, only the lines in the scrolling region are affected).

IND (Index)

Moves the cursor down one line without affecting its character position on the line.

Host Syntax

^{E}cD

If the cursor is on the last line of the scrolling region, a blank line is added to the end of the scrolling region and a line is removed from the beginning of the scrolling region. The cursor can index into the scrolling region from the top fixed region, but cannot index into the bottom fixed region. An index on the last line of the bottom fixed region has no effect.

^{L}F (Linefeed Character)

Moves the cursor position down one line.

If LNM (Linefeed/Newline mode) is reset, then ^{L}F has exactly the same effect as the IND (Index) command; it advances the cursor to the same position on the following line of text. If the cursor is on the last line of the scrolling region, a blank line is added to the end of the scrolling region and a line is removed from the beginning of the scrolling region.

If LNM (Linefeed/Newline mode) is set, then ^{L}F has the same effect as ^{C}R IND: it advances the cursor position to the first character position on the following line.

NEL (Next Line)

Moves the cursor to the start of the next line.

Host Syntax

^{E}cE

The NEL (Next Line) command has the same effect as ^{C}RIND (a Carriage Return character followed by an IND (Index) command); the cursor moves to the first character position on the next line.

REPORT SYNTAX MODE

Sends a Terminal Settings Report to the host.

Host Syntax

^{E}c#!0

The Terminal Settings Report contains the terminal's syntax mode status.

This command has the same effect as a REPORT TERMINAL SETTINGS command issued for the SELECT CODE command (as if **^{E}cIQ%!** was sent from the host). See the REPORT TERMINAL SETTINGS command in Section 5 for additional information.

This command is recognized in all modes: Ansi, Edit, Tek, and VT52.

RI (Reverse Index)

Moves the cursor position up one line without affecting the cursor position on the line.

Host Syntax

^{E}cM

If the cursor is on the first line of the scrolling region, a new line is added to the beginning of the scrolling region and a line is removed from the end of the scrolling region.

RIS (Reset to Initial State)

Resets certain terminal attributes to power-up condition.

Host Syntax

^{E}cc

When the terminal receives this command, it:

- Erases the screen
- Positions the dialog area cursor at the Home position
- Sets Insert/Replace mode to Replace
- Clears edit margins
- Turns off the character graphic rendition feature
- Selects default G0 or G1 character set
- Shifts in the G0 character set
- Enables the dialog area
- Makes the dialog area visible

RM (Reset Mode)

Resets one or more modes.

Host Syntax

^{E}c[mode **l**

mode: The terminal mode that you want to reset.
Default: Omitted = 0 (Causes an error)

Each mode is specified by a separate parameter. You can reset more than one mode by entering more than one parameter separated by semicolons. If you are resetting more than one mode, any parameters that begin with **?**, **<**, or **>** should be at the start of the list of parameters. Refer to Table 4-2 for an explanation of these parameters and what they mean when reset (using the RM command) and set (using the SM command). A mode is reset until you set it again with an SM (Set Mode) command.

Table 4-2

RESET MODE AND SET MODE COMMAND PARAMETERS

Parameter	Mode Name	Effect When Reset (RM Command)	Effect When Set (SM Command)
2	KAM (Keyboard Action mode)	Enables keyboard.	Disables keyboard.
4	IRM (Insert/Replace mode)	Causes each entered character to overwrite or overstrike the character at the current position. Can also be reset with the INSERTREPLACE REPLACE Setup command.	Inserts each entered character at the current cursor position and characters at and to the right of the cursor position are shifted to the right. Can also be set with the INSERTREPLACE INSERT Setup command.
12	SRM (Send/Receive mode)	Enables the terminal's local echo (equivalent to ECHO YES command in Setup mode).	Disables the terminal's local echo (equivalent to ECHO NO command in Setup mode).
20	LNM (Linefeed/ Newline mode)	Determines the effect of the ^{L}F character. ^{L}F moves the cursor down one line, without changing the column position. Can also be reset with the LFCR NO Setup command.	Causes ^{L}F to move the cursor to the beginning of the next line. Can also be set with the LFCR YES Setup command.
<1	TEKORM (Overstrike/ Replace mode)	Inserts the ^{S}P (space character) and (underscore character) as normal characters. Can also be set with the DAMODE REPLACE Setup command.	Causes each entered ^{S}P to advance the cursor to the next character position. Causes each entered to underline the current character. Can also be reset with the DAMODE OVERSTRIKE Setup command.
?1	TEKCKM (Cursor Key mode)	Causes function keys F1 through F4 to transmit ANSI cursor control commands. If these keys are programmed and key expansions are enabled, they transmit their programmed values. Table 4-3 shows the codes that the keys transmit in this mode. Can also be reset with the CURSORKEYMODE NO Setup command.	Causes function keys F1 through F4 to transmit application program codes. Table 4-3 shows the codes that the keys transmit in this mode. Can also be set with the CURSORKEYMODE YES Setup command.
?2	TEKANM (Ansi/ VT52 mode)	Puts the terminal in VT52 mode.	Not a valid command.
?3	TEKCOLM (Column mode)	Specifies 80-column width. Resetting Column mode erases the contents of the dialog area and resets margins. This command does not affect Origin mode, tabs, character attributes, or any other screen attributes. Also can be reset with COLUMNMODE 80 Setup command.	Specifies 132-column width. The terminal displays only 80 of the 132 columns at any time, but horizontal scrolling can bring any of the columns into view. Setting Column mode erases the contents of the dialog area and resets margins, but does not affect Origin mode, tabs, character attributes, or any other screen attributes. When Column mode is set to 132, and you make the dialog area visible with the DIALOG key, an automatic horizontal scroll may occur; if the cursor is in columns 1—80, the dialog area will scroll right to bring the leftmost column into view; if the cursor is in columns 81—132, the dialog area will scroll left to bring the rightmost column into view. Can also be set with COLUMNMODE 132 Setup command.

(continued)

Table 4-2 (cont)

RESET MODE AND SET MODE COMMAND PARAMETERS

Parameter	Mode Name	Effect When Reset (RM Command)	Effect When Set (SM Command)
?5	TEKSCNM (Screen mode)	Reverses the colors on the display from Reverse to Normal. If Screen mode is already Normal, resetting has no effect. Can also be reset with SCREENMODE NORMAL Setup command.	Reverses the colors on the display from Normal to Reverse. If Screen mode is already Reverse, setting Screen mode has no effect. Can also be set with SCREENMODE REVERSE Setup command.
?6	TEKOM (Origin mode)	Places the terminal in Origin mode Absolute. Moves the cursor to Row 1, Column 1 of the dialog area, and reduces the dialog buffer size to the screen size. Can also be reset with the ORIGINMODE ABSOLUTE Setup command.	Places the terminal in Origin mode Relative. Moves the cursor to Row 1, Column 1 of the scrolling region. Can also be set with the ORIGINMODE RELATIVE Setup command.
?7	TEKAWM (Autowrap mode)	Disables the Autowrap function of the terminal. The cursor does not wrap around to the next line when characters are entered in the last column. Can also be reset with the AUTOWRAP NO Setup mode command.	Enables the Autowrap feature of the terminal. Can also be set with the AUTOWRAP YES Setup command.
?8	TEKARM (Autorepeat mode)	Disables the Autorepeat function of the terminal. Can also be reset with the AUTOREPEAT NO Setup command.	Enables the Autorepeat feature of the terminal. Can also be set with the AUTOREPEAT YES Setup command.

Table 4-3

CURSOR KEY MODE CODES

Function Key	Codes Sent When Set	Codes Sent When Reset
F1	^{E}cOA	^{E}c[A
F2	^{E}cOB	^{E}c[B
F3	^{E}cOD	^{E}c[D
F4	^{E}cOC	^{E}c[C

SCS (Select Character Set)

Selects default and special character sets.

Host Syntax

```
Ec character-set
```

Setup Syntax

```
SELECTCHARSET G0 character-set

or

SELECTCHARSET G1 character-set
```

character-set: The parameter from Table 4-4 that designates the desired character set as either the G0 or G1 character set. Do not include parenthesis in Setup syntax.

The terminal allows you to access two different character sets by using the $^{S}_{I}$ (selects the current G0 character set) and $^{S}_{O}$ (selects the current G1 character set) commands. First you must designate which of the eight possible character sets you want to use and then you can easily switch between them.

When the terminal is turned on, the character set associated with the particular keyboard is designated as the G0 character set and also as the G1 character set. This command allows you to designate different G0 or G1 sets.

Table 4-4 shows the parameters needed to select a particular character set. Appendix B contains tables that list the contents of each character set.

Table 4-4

SELECT CHARACTER SET COMMAND PARAMETERS

Parameter to Designate a G0 Character Set	Parameter to Designate a G1 Character Set	Character Set Designated
(A	)A	United Kingdom
(B	)B	U.S. (ASCII)
(G	)G	Swedish
(K	)K	German
(f	)f	French (See note)
('	)'	Danish/Norwegian
(0	)0	Ruling Set
(3	)3	Supplementary

NOTE

Please use **(f** *or* **)f** *to designate the French character set. These are the current standard escape sequence parameters. However, for compatability with programs designed for an earlier version of the French character set, the terminal does accept* **(R** *and* **)R** *as synonyms for* **(f** *and* **)f,** *respectively.*

SD (Scroll Down)

Scrolls lines down.

Host Syntax

```
Ec[ number-of-lines T
```

number-of-lines: The number of lines to scroll toward the bottom of the screen.
Default: Omitted or 0 = 1

The SD command shifts all lines in the dialog buffer down the specified number (*n*) of rows. The *n* lines at the bottom margin are rolled out of sight and *n* lines are rolled into view at the top margin.

SELECT CODE

Causes the terminal to recognize Ansi, Tek, or VT52 mode command syntax. Also used to select Edit mode.

Host Syntax

^{E}**c%!** syntax

Setup Syntax

CODE syntax

syntax: **0** selects Tek mode syntax. **1** selects Ansi mode syntax for Ansi mode. **2** selects Ansi mode syntax for Edit mode. **3** selects VT52 mode syntax. In Setup mode, enter **TEK**, **ANSI**, **EDIT**, or **VT52**. **Default:** Omitted = 0 (Tek mode)

The syntax of Tek, Ansi, and VT52 mode commands are not compatible. If you are using commands from one mode and want to execute one or more commands from another mode, you must issue the Select Code command with the appropriate parameter.

This command is recognized in all major modes: Ansi, Setup, Tek, and VT52.

Edit mode allows the terminal to be used with VT-100 application programs. Selecting this mode sets the following terminal characteristics:

- Selects ANSI X3.64 syntax
- Sets Origin mode to Absolute
- Sets the dialog buffer to 24 lines
- Sets Insert/Replace mode to Replace
- Enables the dialog area and makes it visible
- Defines a scrolling region the size of the dialog buffer (24 lines)
- Disables all programmed keys (the programmed meanings remain available; use the Setup command **KEYEXPAND yes** to enable them)

SGR (Select Graphic Rendition)

Invokes the character display style specified by the parameters.

Host Syntax

^{E}**c[** graphic-rendition **m**

Setup Syntax

TEXTRENDITION graphic-rendition

graphic-rendition: The style in which text is displayed in the dialog area.
Default: Omitted = 0

In Host syntax, you can set more than one display attribute by entering more than one parameter separated by semicolons. In Setup mode, parameters must be separated by one or more spaces. If you are entering more than one parameter, any parameters that begin with **?**, **<**, **=** , or **>** should be at the start of the list of parameters. Refer to Table 4-5 for an explanation of these parameters. All characters after you send the SGR command are displayed as specified by the SGR parameters until you execute another SGR command.

In Table 4-5, *foreground index* is the color index of the characters as they are displayed on the screen. *Background index* is the color of the screen on which the characters are displayed. *Erase index* is the color with which the characters are erased when a delete or erase command is issued.

Table 4-5

SGR COMMAND PARAMETERS

Parameter	Description
0	Characters displayed in specified color indices with all other attributes (blink, bold, underscore, reverse) turned off.
1	Bold or increased intensity. Bold text is displayed in Color Index 2. Color Index 2 defaults to red. Character background remains the same as current background index.
4	Underscore text.
5	Slow blink (less than 150 blinks per minute).
7	Reverse video text. Foreground and background color indices are swapped.
24	No underscore. Cancels the effect of SGR with *graphic-rendition* set to 4.
25	No blink. Cancels the effect of SGR with *graphic-rendition* set to 5.
27	Positive video text. Cancels the effect of SGR with *graphic-rendition* set to 7.
30	Black display. Sets current character color index to 0, which defaults to black.
31	Red display. Sets current character color index to 2, which defaults to red.
32	Green display. Sets current character color index to 3, which defaults to green.
33	Yellow display. Sets current character color index to 7, which defaults to yellow.
34	Blue display. Sets current character color index to 4, which defaults to blue.
35	Magenta display. Sets current character color index to 6, which defaults to magenta.
36	Cyan display. Sets current character color index to 5, which defaults to cyan.
37	White display. Sets current character color index to 1, which defaults to white.
39	Default display color, Color Index 1, which defaults to white.
40	Transparent background. Sets current character background color index to 0, which defaults to transparent.

Table 4-5 (cont)

SGR COMMAND PARAMETERS

Parameter	Description
41	Red background. Sets current character background color index to 2, which defaults to red.
42	Green Background. Sets current character background color index to 3, which defaults to green.
43	Yellow background. Sets current character background color index to 7, which defaults to yellow.
44	Blue background. Sets current character background color index to 4, which defaults to blue.
45	Magenta background. Sets current character background color index to 6, which defaults to magenta.
46	Cyan background. Sets current character background color index to 5, which defaults to cyan.
47	White background. Sets current character background color index to 1, which defaults to white.
49	Default background color, Color Index 0, which defaults to transparent.
< *index*	Character color. Parameter specifies the color index. Indices from 0 to 7 select colors in the terminal's color map. Indices greater than 7 default to 7.
= *index*	Background color. Parameter specifies the character background color index. Index 0 means that the graphics area shows through. Background indices from 1 to 7 represent colors in the terminal's color map. Indices greater than 7 default to 7.
> *index*	Erase color. Parameter specifies the dialog area erase color index. Erase Index 0 always means that the graphics area shows through. Indices from 1 to 7 select colors in the terminal's color map. Indices greater than 7 default to 7.

^{S}I (Shift In Character)

Invokes the current G0 character set.

The terminal allows you to access two different character sets by using the ^{S}I (selects the current G0 character set) and ^{S}O (selects the current G1 character set) commands.

The ^{S}I command invokes the current G0 character set installed in the terminal. This may be the 94 graphic characters from the ASCII character set, or the corresponding 94 characters from the United Kingdom, French, Swedish, Danish/Norwegian, German, supplementary, or special rulings character sets; each of these character sets is designated by connecting a standard or optional keyboard to the terminal or by using the SCS (Select Character Set) command. Appendix B of this manual lists these character sets.

To select the G1 character set, use the ^{S}O (Shift Out) character.

SL (Scroll Left)

Moves the contents of the visible dialog area to the left.

Host Syntax

^{E}C[number-of-columns^{S}P@

number-of-columns: Specifies the number of columns to scroll left.
Default: Omitted or 0 = 1

The SL command moves the entire contents of the visible dialog area to the left by the specified number of columns. You can scroll horizontally only when Column mode is set to 132. Since the cursor moves with the text, the cursor may disappear from the screen. Unlike vertical scrolling, the terminal will not automatically scroll left or right to keep the cursor in view.

To scroll horizontally, you must give the SCROLL LEFT or SCROLL RIGHT command (or you can use the joydisk). However, an automatic scroll left may occur when you make the dialog area visible with the DIALOG key; if the cursor is in Columns 81—132, the terminal will scroll left to bring the rightmost column into view.

SM (Set Mode)

Sets one or more modes.

Host Syntax

^{E}C[mode **h**

mode: The terminal mode or modes that you want to set.
Default: Omitted = 0 (not a valid parameter)

Each mode to be set is specified by a separate parameter. You can set one or more modes by entering more than one parameter separated by semicolons. If you are setting more than one mode, any parameters that begin with **?**, **<**, **=**, or **>** should be at the start of the list of parameters. A mode remains set until you reset it with an RM (Reset Mode) command. Refer back to Table 4-2 (under RM command) for an explanation of the *mode* parameters.

^{S}O (Shift Out Character)

Invokes the G1 character set.

The terminal allows you to access two different character sets by using the ^{S}I (selects the current G0 character set) and ^{S}O (selects the G1 character set) commands.

The ^{S}O command invokes the the G1 character set. When a keyboard is plugged into the terminal, the character set associated with that keyboard is designated as both the G0 and the G1 set. You may use the SCS (Select Character Set) command to designate a different character set than the one associated with the current keyboard. Appendix B lists the contents of all the available character sets.

To select the G0 character set, use the ^{S}I (Shift In) character.

^{S}P (Space Character)

Moves the cursor one character position to the right or inserts a ^{S}P (Space) character.

If Overstrike mode is enabled (TEKORM is reset), ^{S}P moves the cursor one character to the right. The ^{S}P character does not replace the character, if any, at the cursor position.

If Replace mode is enabled (TEKORM is set), ^{S}P replaces any character at the cursor position.

SR (Scroll Right)

Moves the contents of the visible dialog area to the right.

Host Syntax

^{E}c[number-of-columns ^{S}P **A**

number-of-columns: Specifies the number of columns to scroll right.
Default: Omitted or 0 = 1

The SR command moves the entire contents of the visible dialog area to the right by the specified number of columns. Since the cursor moves with the text, the cursor may disappear from the screen. Unlike vertical scrolling, the terminal will not automatically scroll left or right to keep the cursor in view.

To scroll horizontally, you must give the SCROLL LEFT or SCROLL RIGHT command (or you can use the joydisk). However, an automatic scroll right may occur when you make the dialog area visible with the DIALOG key; if the cursor is in Columns 1—80, the terminal will scroll right to bring the rightmost column into view.

SU (Scroll Up)

Scrolls lines up.

Host Syntax

^{E}c[number-of-lines **S**

number-of-lines: The number of lines to scroll toward the top of the screen.
Default: Omitted or 0 = 1

The SU (Scroll Up) command shifts all lines in the dialog buffer upward by the specified number (n) of rows. The n lines at the top margin are rolled out of sight and n lines are rolled into view at the bottom margin.

TBC (Tab Clear)

Clears one or more tab stops.

Host Syntax

^{E}c[tab-clear-extent **g**

tab-clear-extent: Clears the tabs as indicated by the parameter values. Refer to Table 4-6 for an explanation of these parameters.
Default: Omitted = 0

Table 4-6

TAB CLEAR COMMAND PARAMETERS

Parameter	Description
0	Clears the horizontal tab stop at the cursor position
2	Clears all horizontal tab stops
3	Clears all horizontal tab stops

TEKDHL (Double Height Line)

Causes the line containing the cursor to become the top or bottom half of a double-height, double-width line.

Host Syntax

Top Half	Bottom Half
^{E}C#3	^{E}C#4

Both lines that receive these commands must contain the same characters. Since using double-width characters halves the number of characters per line, characters to the right of screen center are lost if the line was previously single width.

To make an exact hardcopy of a double-height, double-width line, you must make a screen copy. Making a dialog copy will copy each character of the top-half line as a regular size character followed by a space; the bottom-half line becomes a blank line. (See the Tek mode HARDCOPY command description for additional details about making screen and dialog copies.)

This command affects only the current line. The line will retain this attribute until the line is deleted or until the terminal receives another line attribute command (TEKDHL, TEKDWL, or TEKSWL).

TEKDWL (Double Width Line)

Causes the line containing the cursor to become a double-width, single-height line.

Host Syntax

^{E}C#6

This command affects only the current line. The line will retain this attribute until the line is deleted or until the terminal receives another line attribute command (TEKDHL, TEKDWL, or TEKSWL).

Since using double-width characters halves the number of characters available per line, characters to the right of screen center are lost if the line was previously single width.

To make an exact copy of a double-width line, you must make a screen copy. Making a dialog copy will copy each character in the line as a regular size character followed by a space. (See the Tek mode HARDCOPY command description for additional details about making screen and dialog copies.)

TEKID (Identify Terminal)

Tells the terminal to report what type of terminal it is.

Host Syntax

^{E}cZ

This command causes the same response as the Ansi mode DEVICE ATTRIBUTES (DA) command with a parameter of 0.

NOTE

The TEKID command is provided in Ansi mode only for compatibility with programs written for VT100 terminals. Avoid using this command if you can; its use violates ANSI and ISO standards.

TEKKPAM (Keypad Application Mode)

Places the terminal in Keypad Application mode.

Host Syntax

$^{E}_{C}=$

Setup Syntax

KEYPADMODE APPLICATION

The TEKKPAM command causes the numeric keypad to send characters distinct from the numeric keys on the main keyboard. This means that when you press the 6 key on the numeric keypad, a different code is generated than when you press the 6 key on the main keyboard. Refer to Table 4-7 for an explanation of these codes.

When the terminal is turned on, it is in Keypad Numeric mode.

TEKKPNM (Keypad Numeric Mode)

Places the terminal in Keypad Numeric mode.

Host Syntax

$^{E}_{C}>$

Setup Syntax

KEYPADMODE NUMERIC

This command causes the keys on the numeric keypad and function keys F5 through F8 to return to their default meanings, as shown in the righthand column of Table 4-7. If the keys have been programmed and key expansions are enabled, the keys transmit their programmed meanings instead.

When the terminal is turned on, it is Keypad Numeric mode (keys produce their default meanings).

Table 4-7

NUMERIC KEYPAD PROGRAMMING CODES

Numeric Keypad Key	Characters Sent in Application Mode	Characters Sent in Numeric Mode
0	^{E}cOp	0
1	^{E}cOq	1
2	^{E}cOr	2
3	^{E}cOs	3
4	^{E}cOt	4
5	^{E}cOu	5
6	^{E}cOv	6
7	^{E}cOw	7
8	^{E}cOx	8
9	^{E}cOy	9
-	^{E}cOm	-
,	^{E}cOl	,
.	^{E}cOn	.
ENTER	^{E}cOM	$^{C}_{R}$
F5	^{E}cOP	^{E}cOP
F6	^{E}cOQ	^{E}cOQ
F7	^{E}cOR	^{E}cOR
F8	^{E}cOS	^{E}cOS

TEKRC (Restore Cursor)

Restores the cursor position, graphic rendition, character set, and Origin mode previously saved using the TEKSC (Save Cursor) command.

Host Syntax

$^{E}_{C8}$

If the TEKSC (Save Cursor) command is not used first, then TEKRC (Restore Cursor) returns the cursor to the Home position and restores the power-up graphic rendition, character set, and Origin mode.

TEKSC (Save Cursor)

Saves the cursor position, graphic rendition, character set, and Origin mode.

Host Syntax

^{E}C7

The TEKSC (Save Cursor) command saves information about the cursor position, graphic rendition, character set, and Origin mode in the terminal's memory. This saved information may be restored using the TEKRC (Restore Cursor) command.

TEKSTBM (Set Top and Bottom Margins)

Sets the dialog buffer's edit margins.

Host Syntax

^{E}C[top-margin ; bottom-margin r

Setup Syntax

EDITMARGIN top-margin, bottom-margin

top-margin: The row number of the top margin.
Default: Omitted or 0 = 1

bottom-margin: The row number of the bottom margin.
Default: Omitted or 0 = The number of lines visible in the dialog buffer (DALINES).

The parameter value for the top margin specifies which row of the dialog buffer becomes the top line of the scrolling region. Similarly, the value for the bottom margin specifies the row of the dialog buffer for the bottom line of the scrolling region.

The rows in the dialog buffer above the top margin and the rows below the bottom margin become fixed regions. No scrolling actions occur in the fixed regions.

TEKSWL (Single Width Line)

Causes the line containing the cursor to become a single-width, single-height line.

Host Syntax

^{E}C#5

The cursor retains its current column number. This is the default for all new lines in the dialog area. This command affects only the current line. The line will retain this attribute until the line is deleted or until the terminal receives another line attribute command (TEKDHL, TEKDWL, or TEKSWL).

^{V}T (Vertical Tab Character)

The ^{V}T character moves the cursor down one line without affecting the cursor position on the line.

_ (Underscore Character)

If Overstrike mode is enabled (TEKORM is reset), the current character is underlined and the cursor advances to the next character.

If Replace mode is enabled (TEKORM is set), the underscore character replaces the current character.

VT52 COMMANDS

The VT52 commands that follow can be executed only while the terminal is in VT52 mode. You can put the terminal in VT52 mode by:

- Entering **CODE VT52** while in Setup mode
- Sending an RM command (Ec**[?21**) from the host while in Ansi mode
- Sending a SELECT CODE command (Ec**%!3**) from the host while in Tek or Ansi mode

Once the terminal is in VT52 mode, it will recognize only VT52 commands (which are explained here), and the commands SELECT CODE and REPORT SYNTAX MODE, both of which work in all modes.

CURSOR DOWN

Moves the cursor down one line without moving it horizontally.

Host Syntax

EcB

The cursor moves with respect to the dialog buffer and stops at the last row of the dialog buffer. However, if margins are set and the cursor is within the scrolling region, the cursor stops at the bottom margin of the scrolling region.

CURSOR LEFT

Moves the cursor one column to the left.

Host Syntax

EcD

The cursor does not move beyond the leftmost column (Column 1).

This command works just like the Ansi mode command CUB (Cursor Backward) with a parameter of 1.

CURSOR RIGHT

Moves the cursor one column to the right.

Host Syntax

EcC

The cursor does not move beyond the rightmost column. If Column mode is set to 132, the cursor may disappear from the screen. This command will not scroll horizontally to keep the cursor in view.

This command works just like the Ansi mode command CUF (Cursor Forward) with a parameter of 1.

CURSOR TO HOME

Moves the cursor to the Home position.

Host Syntax

EcH

The home position is Row 1, Column 1 of the dialog buffer.

CURSOR UP

Moves the cursor up one line without moving it horizontally.

Host Syntax

EcA

The cursor moves with respect to the dialog buffer and stops at the first row of the dialog buffer. However, if margins are set and the cursor is within the scrolling region, the cursor stops at the top margin of the scrolling region.

DIRECT CURSOR ADDRESS

Moves the cursor to the specified line and column.

Host Syntax

^{E}cY line; column

line: An ASCII character that represents the line position number plus 31. The maximum line range is 96, even if the dialog buffer is larger.

column: An ASCII character that represents the column number plus 31. The maximum column range is 80, even if Column mode is set to 132.

For example, to move the cursor to Line 3, Column 1, give the command ^{E}cY"^{S}P since the decimal equivalent of " is 34 (3 + 31) and the decimal equivalent of ^{S}P is 32 (1 + 31). If a parameter is out of range, the cursor will not change position for that parameter. However, the cursor will move to the other parameter position if it is within the range.

ENTER ALTERNATE KEYPAD MODE

Causes the numeric keypad keys and Function Keys F5 — F8 to assume their Alternate Keypad mode meanings (shown in Table 4-8).

Host Syntax

^{E}c=

Any other meanings you program into these keys cannot be used as long as the terminal is in Alternate Keypad mode.

This command works like the Ansi mode command TEKKPAM (Keypad Application Mode).

Table 4-8

ALTERNATE KEYPAD PROGRAMMING CODES

Numeric Keypad Key	Characters Sent as Factory Default	Characters Sent in Alternate Keypad Mode
0	0	^{E}c?p
1	1	^{E}c?q
2	2	^{E}c?r
3	3	^{E}c?s
4	4	^{E}c?t
5	5	^{E}c?u
6	6	^{E}c?v
7	7	^{E}c?w
8	8	^{E}c?x
9	9	^{E}c?y
–	–	^{E}c?m
,	,	^{E}c?l
.	.	^{E}c?n
ENTER	^{C}R	^{E}c?M
F5	^{E}cP	^{E}cP
F6	^{E}cQ	^{E}cQ
F7	^{E}cR	^{E}cR
F8	^{E}cS	^{E}cS

ENTER ANSI MODE

Places the terminal in Ansi mode.

Host Syntax

^{E}c<

The terminal will interpret all subsequent commands according to ANSI Standard X3.64.

ENTER GRAPHICS MODE

Causes the Rulings character set to be selected as the G0 character set.

Host Syntax

^{E}cF

The terminal will remain in Graphics mode until you issue an EXIT GRAPHICS MODE command. If you issue the ENTER ANSI MODE command while the terminal is still in Graphics mode, the terminal will exit Graphics mode before it exits VT52 mode.

ERASE TO END OF LINE

Erases all characters from the cursor to the end of the current line.

Host Syntax

^{E}cK

The cursor position does not change.

This command works like the Ansi mode command EL (Erase in Line) with a parameter of 0.

ERASE TO END OF SCREEN

Erases all characters from the cursor to the end of the screen.

Host Syntax

^{E}cJ

The cursor position does not change.

This command works like the Ansi mode command ED (Erase in Display) with a parameter of 0. It erases text from the cursor position to the end of the dialog buffer, so it makes no difference if margins are set.

EXIT ALTERNATE KEYPAD MODE

Causes the numeric keypad keys and Function Keys F5 – F8 to assume their factory default meanings, or their programmed meanings if they have been programmed.

Host Syntax

^{E}c>

Factory default meanings are shown in Table 4-8 (under ENTER ALTERNATE KEYPAD MODE).

This command works like the Ansi mode command TEKKPNM (Keypad Numeric Mode).

EXIT GRAPHICS MODE

Restores the G0 character set that was in effect before the current ENTER GRAPHICS MODE command was issued.

Host Syntax

^{E}cG

IDENTIFY

Identifies the terminal to the host.

Host Syntax

^{E}cZ

When the host issues this command, the terminal sends its identifier escape sequence ^{E}c/Z to the host.

REVERSE LINEFEED

Moves the cursor up one line without affecting the cursor position on the line.

Host Syntax

^{E}cI

Follow the ^{E}c with an uppercase i.

Section 5

4100-STYLE PARAMETER TYPES, COMMANDS, AND REPORTS

INTRODUCTION

This section contains detailed descriptions of the terminal's 4100-style parameter types, commands, and reports. The explanations are for both the terminal's Host syntax and Setup mode syntax.

The terminal executes 4100-style commands when it is in Tek mode, which you specify with the SELECT CODE command.

The first part of this section includes descriptions of parameter types associated with 4100-style commands. The parameter types are explained in alphabetical order. Commands issued from the host use special encoding for parameter values. Setup mode commands use simple English words or numbers as parameters.

The second part of this section contains explanations of 4100-style commands and of reports associated with certain commands. Most commands have two forms: one used for sending the command from the host, and the other used locally in the terminal's Setup mode. Since the host form is in a cryptic opcode format, it is given a *descriptive* command name to clarify what the command does. The associated Setup mode command accomplishes the same function locally; it has an English-style name that identifies the command's purpose. The command explanations are given alphabetically according to the descriptive command name.

4100-STYLE PARAMETER TYPES

The following explanations document the various types of values used for the 4100-style parameters.

Character Array Parameters in Host Syntax

The character array parameter type allows you to send a list of characters as one parameter. This parameter type occurs only in the Host syntax form of a command.

The first character in the array is an encoded integer indicating the length of the array. The integer is followed by the ASCII characters in the array.

The following example shows how the phrase **PRESS RETURN KEY** would be sent as a character array:

A0PRESS RETURN KEY

Character Parameters in Setup Syntax

The Setup mode form of commands have three kinds of character parameters: *small integer*, *string*, and *character*.

The Small Integer Parameter. The *small integer* parameter type corresponds to all characters listed on a standard ASCII chart. A *small integer* parameter has a range of $^{N}_{U}$ to $^{D}_{T}$; it has an ADE range of 0 to 127. When you specify a *small integer* parameter, you can use either the actual character or its ADE value.

The String Parameter. Some commands use an undelimited *string*. The word "undelimited" means that the parameter does not require special characters to distinguish it from other character strings. A *string* parameter requires that you specify actual characters rather than ADE values.

The Character Parameter. The *character* parameter type corresponds to the printing characters listed on a standard ASCII chart. A *character* parameter has a range of $^{S}_{P}$ to ~ (tilde); it has an ASCII decimal equivalent (ADE) range of 32 to 126. When you specify a *character* parameter, you can use either the actual character or its ADE value.

All three of these parameter types can be used as delimited parameters. When parameters are delimited, the first and last characters have a special meaning to distinguish that group of characters from others. A delimited parameter can only include ASCII characters, not ADE values.

For example, in the Setup syntax of the SET EOF STRING command, you specify up to ten *small integer* characters to indicate the end-of-file marker. The delimiter can be any character — it is always the first character after the space following the command name. The delimiter is followed by up to ten alphanumeric characters and that group of characters is terminated by the same delimiter. The following command illustrates a delimited character parameter:

EOFSTRING 'THE END'

Integer Parameters in Host Syntax

In Host syntax, integers are encoded as a series of one, two or three characters. For example, Figure 5-1 illustrates that the integer –2413 is encoded as:

BV-

First, the decimal numeral must be translated into binary. Then the digits of the binary numeral are encoded (see Figure 5-1).

Hi-I: There may be zero, one or two Hi-I characters, depending on the value of the integer. The Hi-I characters are:

1 d d d d d d

The first bit is always set to 1; the next six bits are the next six most significant bits in the binary numeral. In the example in Figure 5-1, the Hi-I character is:

1 0 0 0 0 1 0

This corresponds to the **B** character on an ASCII chart.

The second Hi-I character is:

1 0 1 0 1 1 0

This corresponds to the **V** character on an ASCII chart.

Lo-I: The Lo-I character may be the only character. It contains the sign bit and the four least-significant bits of the integer:

0 1 s d d d d

The first two bits are always set to 0 and 1; the third bit represents the sign of the integer. A negative integer is represented by a 0; a positive integer is represented by a 1. The next four characters are the least significant bits of the integer.

In the example in Figure 5-1, Lo-I would be:

0 1 0 1 1 0 1

This corresponds to the - (hyphen) character on an ASCII chart.

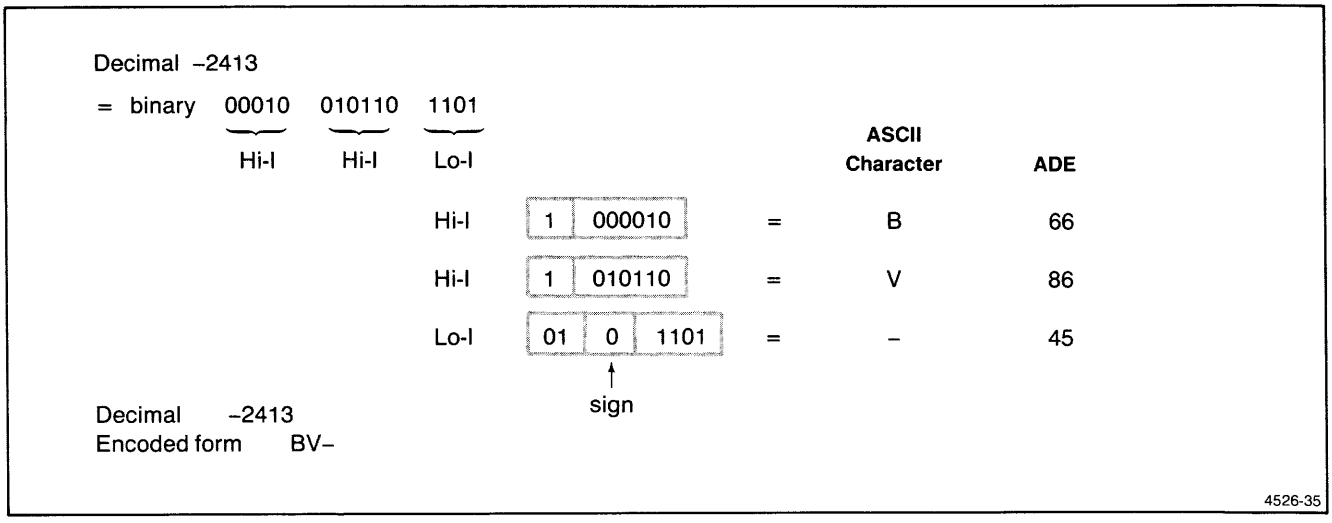


Figure 5-1. How Integers Are Encoded.

Integer Report Parameters in Host Syntax

When the terminal sends integers to the host, it sends them encoded as three characters in an integer report. The way the terminal encodes integer reports is similar, but not identical, to the way the host encodes integer parameters (see the immediately preceding paragraphs). Integer report parameters have these characteristics:

- The terminal may include an EOL string, but sends it only if there is no other way to avoid exceeding the current maximum line length.
- Unlike integer parameters, integer reports always have three characters.
- The encoding scheme for an integer report's Lo-I character is identical to that of the integer parameter.
- The encoding scheme for an integer report's two Hi-I characters uses a different "offset" than for an integer parameter. In an integer parameter's Hi-I character, 64 is added to a six-bit binary numeral to form the ASCII decimal equivalent of the Hi-I character (this is the 1 seen at the beginning of the Hi-I's in Figure 5-1). In an integer report's Hi-I characters, 32 (rather than 64) is added to the six-bit binary numeral.

If, for example, we use the binary representation of –2413 shown in Figure 5-1, the first Hi-I character will be:

Six-bit binary		000010
Add offset of 32	+	0100000
	=	0100010
	=	ASCII character "

The second Hi-I character will be:

Six-bit binary		010110
Add offset of 32	+	0100000
	=	0110110
	=	ASCII character **6**

The Lo-I character is encoded in the same way as the integer parameter and will be:

		0101101
	=	ASCII character **-**

Thus, if the terminal sends the number –2413 to the host as an integer report, it encodes that number as "**6-**.

Integer Array Parameters in Host Syntax

Some commands take integer array parameters. Integer array parameters consist of sequences of integer parameters. The first integer tells how many items are in the array; subsequent integers represent individual array items. You encode each item in the array the same way you encode a single integer, as described under "Integer Parameter Type in Host Syntax."

For example, you would send the array of integers (1, 5, –1, 16) to the terminal as follows:

integer array: (1,5,–1,16)	=	integer:4 (the count of 4)
		integer:1
		integer:5
		integer:-1
		integer:16
	=	**415!A0**

Integer Parameters in Setup Syntax

Whenever you use an integer parameter in Setup syntax, simply enter the value. For example, to set both the transmit and receive rate to 2400 using the BAUDRATE Setup command, enter:

BAUDRATE 2400 2400

Key Specifiers in Setup Syntax

The *key specifier* parameter type is used in the DEFINE MACRO and DEFINE NONVOLATILE MACRO commands.

See Table 5-2 (under DEFINE MACRO) for the ASCII decimal equivalents (ADE) and macro numbers referred to in the following paragraphs.

Key specifiers for keys labeled with ASCII characters can be the ASCII character itself or a two- or three-digit ADE value or a macro integer number.

Function keys are represented by macro numbers or by a two-character abbreviation. The abbreviations F1 through F8 designate the unshifted versions of the function keys; S1 through S8 specify the shifted versions of those keys. The Ctrl and Shift-Ctrl versions of those keys are specified by macro numbers only.

The numeric keypad, special function keys, and joydisk are specified by macro numbers.

Keywords in Setup Syntax

The *keyword* parameter type is used in Setup mode only. Keyword parameters specify what action you want a command to perform. When you use a keyword, you can spell out the entire word or just as many characters as necessary to distinguish that keyword from other keywords associated with the command.

XY-Coordinates in Host Syntax

The host must encode each xy-coordinate into one to five ASCII characters. The valid range of xy-coordinates is 0 to 4095. The xy-coordinates (53,1000), for example, would be encoded and sent to the terminal as the following characters:

'az^{SP} M

Figure 5-2 illustrates how the encoding process works. First, the decimal numeral is translated into a 12-bit binary numeral. Then the digits of the binary numeral are encoded. In the encoding process, the bits are divided into *Hi, Lo,* and *Extra* bytes.

In the example in Figure 5-2, decimal 53 translates to binary:

00000 01101 01

Decimal 1000 translates to binary:

00111 1101 00

The digits are encoded and transmitted in the order in which the following explanations are given:

Hi-Y: Contains the most significant five bits of the binary numeral representing the y-coordinate of the xy-parameter. The seven-bit Hi-Y character (excluding the parity bit) has the following format:

0 1 y y y y y

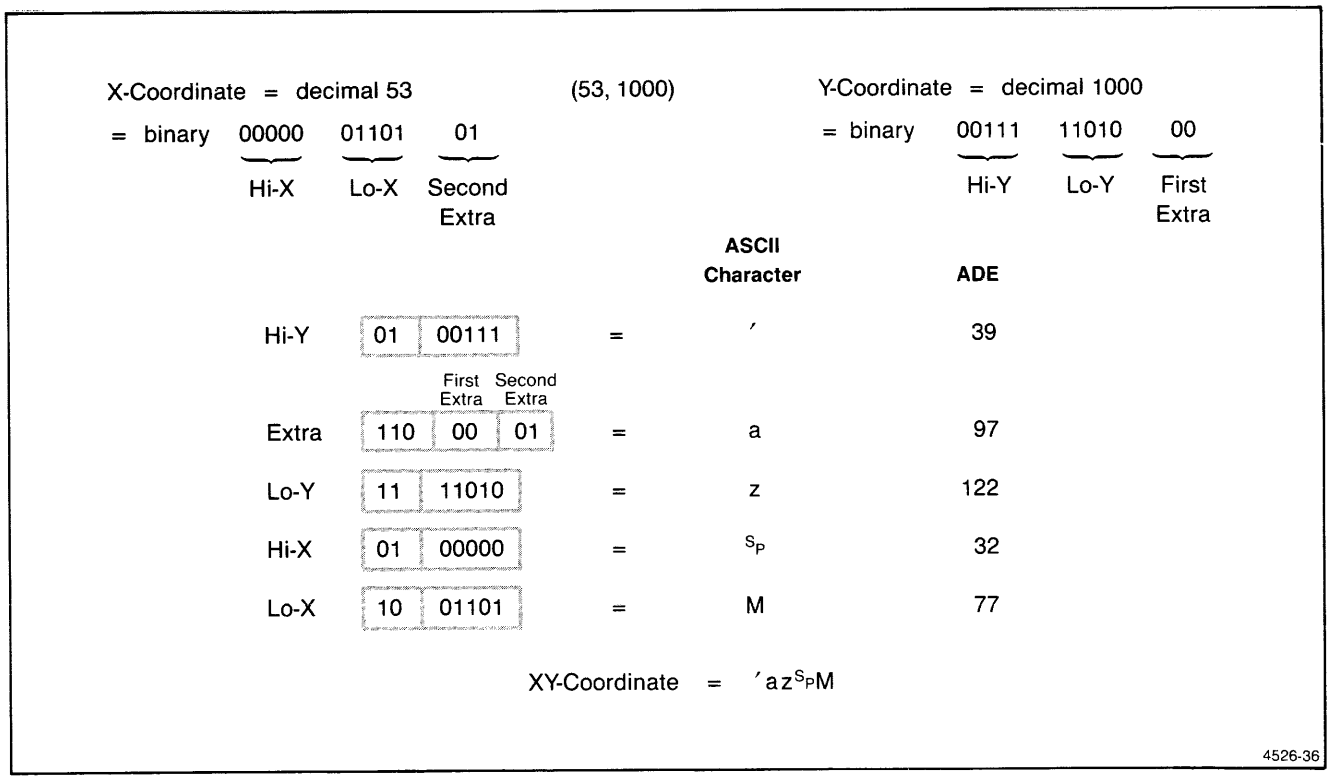


Figure 5-2. How XY-Coordinates Are Encoded.

The first two bits must be set to **0 1**, as shown. The characters "y y y y y" represent the five most significant bits of the y-coordinate. In the example in Figure 5-2, this byte is:

0 1 0 0 1 1 1

This corresponds to the apostrophe (') character.

You can omit the Hi-Y byte if the high-order five bits of the y-coordinate has not changed since the last xy-coordinate was sent.

Extra: Contains the least-significant two bits of the x- and y-coordinates. The seven-bit ASCII character has the following format:

1 1 0 y y x x

The first three bits must be set to **1 1 0**, as shown. The characters "y y x x" represent the two least significant bits of the y- and x-coordinates, respectively. In the example in Figure 5-2, this byte is:

1 1 0 0 0 0 1

This corresponds to the letter **a** on the ASCII chart.

You can omit the Extra byte if you only want to send coordinates with 10 bits of precision or if the two least significant bits of the x- or y-coordinate have not changed since the last xy-coordinate. If you do send the Extra byte, you must follow it with a Lo-Y bite.

Lo-Y: Contains the intermediate five bits of the y-coordinate. The seven-bit ASCII character has the following format:

1 1 y y y y y

The first two bits must be set to **1 1**, as shown. The characters "y y y y y" represent the five intermediate bits of the y-coordinate. In the example in Figure 5-2, this byte is:

1 1 1 1 0 1 0

This corresponds to the letter **z** on the ASCII chart.

You can omit the Lo-Y byte if the following conditions are all true:

- These five bits have not changed since the last xy-coordinate.
- You are not sending the Extra byte.
- You are not sending the Hi-X byte.

Hi-X: Contains the most significant five bits of the x-coordinate of the xy-parameter. The seven-bit ASCII character has the following format:

0 1 x x x x x

The first two bits must be set to **0 1**, as shown. The characters "x x x x x" represent the five most significant bits of the x-coordinate. In the example in Figure 5-2 this byte is:

0 1 0 0 0 0 0

This corresponds to the ASCII $^{S}{}_{P}$ (space) character.

You can omit the Hi-X character if the x-coordinate's most significant bits have not changed from the last xy-coordinate sent. If you do send the Hi-X character you must precede it with the Lo-Y byte.

Lo-X: Contains the intermediate five bits of the x-coordinate. The seven-bit ASCII character has the following format:

1 0 x x x x x

The first two bits must be set to **1 0**, as shown. The characters "x x x x x" represent the five intermediate bits of the x-coordinate. In the example in Figure 5-2, this byte is:

1 0 0 1 1 0 1

This corresponds to the letter **M** on the ASCII chart.

This character must always be sent, since it terminates the xy-parameter sequence.

NOTE

Since the Lo-Y and Extra bytes each have high-order bits of "1 1", $^{D}{}_{T}$ (binary 1 1 1 1 1 1 1) could appear as a Lo-Y or Extra byte. This presents a problem for hosts that use $^{D}{}_{T}$ as a filler character.

If your host uses $^{D}{}_{T}$ as a filler, have the host do the following things:

- *Send the two-character string $^{E}{}_{C}$? instead of $^{D}{}_{T}$ since the terminal recognizes that string as a synonym for $^{D}{}_{T}$.*
- *Send the terminal an IGNORE DELETES command to cause the terminal to ignore all $^{D}{}_{T}$ characters sent from the host.*

4100-STYLE COMMAND AND REPORT DESCRIPTIONS

This part of the section describes each of the terminal's 4100-style commands and reports. The commands are presented alphabetically according to the descriptive names associated with the Host syntax. Figure 5-3 shows the format in which commands are explained.

Command Conventions

There are four different kinds of 4100-style escape sequence commands:

- A single character
- The ^{E}c character followed by one character
- The ^{E}c character followed by two characters
- The ^{E}c character followed by two characters and one or more parameters

Figure 5-3 illustrates the general syntax conventions used in this section of the manual. Almost every command in this section has a Host syntax form; many commands have both a Host syntax form and an English-style Setup syntax form. Some commands have just a Setup syntax form.

The following points summarize the format illustrated in Figure 5-3.

- The descriptive command name is shown in all uppercase characters.
- Characters shown in bold type are those you must enter exactly as shown.
- No spaces are allowed between the ^{E}c and following letters or parameters.
- Parameter names are shown on separate lines and are separated by commas to make the command syntax easier to read. However, when entering commands, follow these rules:
  - In Setup mode, enter parameters on the same line. The first character after the command name must be a space; use one or more spaces or one comma to separate parameters.
  - In Host syntax, enter parameters on the same line with no spaces between any of the parameters.
- Multiword items are joined by hyphens. For example, *parameter-name* and *next-parameter-name* are single items described by two or more words.
- Three periods following a parameter (. . .) indicate that it can be repeated.

DESCRIPTIVE COMMAND NAME

A short statement describing the purpose of the command.

Host Syntax

^{E}**cQQ** parameter-name,
next-parameter-name, . . .

Setup Syntax

SETUPNAME parameter-name,
next-parameter-name, . . .

parameter-name: parameter type; what it does. Explanations include the range.

Defaults: Factory = The value assigned a parameter when the terminal is shipped from Tektronix; parameters can be restored to this value by the Extended Self-Test program.

Power-Up = The value assigned a parameter when the terminal is first turned on. The term "Saved in memory" indicates that the parameter takes on the value saved in the terminal's nonvolatile memory.

Omitted = The value assigned a parameter if the command is used and no value is specified for the parameter.

next-parameter-name: The next parameter is explained.

Defaults: Factory = Value
Power-Up = Value
Omitted = Value

4526-37

Figure 5-3. Command Description Format for 4110-Style Commands.

BEGIN PANEL BOUNDARY

Starts a panel definition.

Host Syntax

```
ESCLP first-point,
      draw-boundary
```

Setup Syntax

```
BEGINPANEL first-point,
           draw-boundary
```

first-point: xy-coordinate; indicates the first point in a panel boundary. The valid range of values is 0 through 4095 for both the x- and y-coordinate.
Default: Omitted = (0,0)

draw-boundary: integer; 0 specifies that the panel boundary should be covered by the fill pattern, 1 specifies that the boundary should be displayed in the finished panel.
Default: Omitted = 0

After issuing BEGIN PANEL BOUNDARY, you can define the boundary of the panel in either of two ways:

- Put the terminal in either Vector or Marker mode and then send the coordinates of the boundary line endpoints.
- Use MOVE and DRAW commands to define the boundary line.

You do not need to create the panel's last boundary segment. When the terminal receives END PANEL, it closes the panel and fills it with the fill pattern specified in the most recent SELECT FILL PATTERN.

You can define a panel when the terminal is in Marker mode. However, you cannot draw a marker during a panel definition when the terminal is in Marker mode.

Multiple Panel Boundaries. You can create a panel with multiple boundaries, as shown in Figure 5-4. To do this, send another BEGIN PANEL BOUNDARY when you want to start the second boundary (do not use END PANEL to close the first boundary). The second BEGIN PANEL BOUNDARY closes the first boundary and starts another boundary at the specified position. When you issue END PANEL, the last boundary is closed and the entire panel defined by the multiple boundaries is filled.

BEGIN PIXEL OPERATIONS (Requires Optional Pixel ROMs)

Sets three parameters for use in subsequent RASTER WRITE, RUNLENGTH WRITE, RECTANGLE FILL, and PIXEL COPY commands.

Host Syntax

```
ESCRU surface-number,
      ALU-mode,
      bits-per-pixel
```

Setup Syntax

```
PXBEGIN surface-number,
        ALU-mode,
        bits-per-pixel
```

surface-number: integer; valid values are –1, 0, and 1; specifies the surface on which subsequent pixel operations commands will write (or read) their data. Since Surface 1 is the terminal's only surface, all parameter values mean Surface 1. (Other Tektronix 4100 Series terminals may have additional surfaces, so the surface parameter is included here for compatibility).
Defaults: Factory = 1
Power-Up = 1
Omitted = 0

ALU-mode: integer; valid values are 0, 7, 11, 12, or 15; specifies arithmetic logic unit (ALU) writing mode. 0 means no change to the existing *ALU-mode*. 7, 11, 12, and 15 specify what function RASTER WRITE, RUNLENGTH WRITE, RECTANGLE FILL, and PIXEL COPY commands will use to modify the existing contents of the terminal's raster memory buffer. Table 5-1 lists the function that each *ALU-mode* parameter value selects.
Defaults: Factory = 11
Power-Up = 11
Omitted = 0

bits-per-pixel: integer; specifies the number of bits used to encode the color index for each pixel in subsequent RASTER WRITE and RUNLENGTH WRITE commands. Valid values are 0, 1, 2, 3, 4, and 6; 0 means no change. The command descriptions for RASTER WRITE and RUNLENGTH WRITE explain in greater detail how to use this parameter.
Defaults: Factory = 6
Power-Up = 6
Omitted = 0

Table 5-1

ALU VALUES

ALU Mode	Function	Application
0	no change	
7	A XOR B (exclusive OR)	Fast Erase mode. Writes an image that you can later completely remove by repeating the same operation. That is because A = (A XOR B) XOR B.
11	B	Replace mode. Replaces the existing image with the new pixels. This is the default *ALU-mode*.
12	A AND B	Performs a logical AND function on the pixel color indices and displays the results.
15	A OR B	Writes the new image on top of the existing image. Pixel values of 0 in the command string do not affect raster memory.

CANCEL

Resets several terminal parameters and modes.

Host Syntax

^{E}cKC

Setup Syntax

CANCEL

When you issue this command (which has the same effect as pressing the Cancel key) it:

- Puts the terminal into Alpha mode and terminates all of the following modes:
  - Vector mode
  - Marker mode
  - GIN mode
  - Bypass mode
  - Prompt mode
  - Snoopy mode
- Unlocks the keyboard
- Terminates the copy or hardcopy functions if either is currently in progress
- Empties the input/output queues

CLEAR DIALOG SCROLL

Clears (erases) the dialog buffer.

Host Syntax

^{E}cLZ

Setup Syntax

CLEARDIALOG

After the dialog buffer is cleared, the cursor returns to the home position (upper-left corner of dialog area).

Issuing CLEAR DIALOG SCROLL has the same effect as pressing the terminal's D Eras key.

COPY

Sends data from the host directly to an attached hardcopy unit.

Host Syntax

```
EcJC source,
     separator,
     destination
```

Setup Syntax

```
COPY HO: TO HC:
```

source: character array (string in Setup syntax); the only valid value is **HO:** (in uppercase or lowercase).

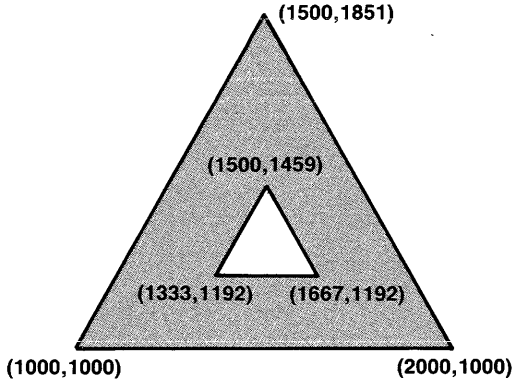
separator: character array (string in Setup syntax); If you use it, the only valid value is **TO** (in uppercase or lowercase). Otherwise, specify an empty array.

destination: character array (string in Setup syntax); the only valid value is **HC:** (in uppercase or lowercase).

After the terminal receives this command, it passes all data it receives from the host directly to the hardcopy port. The host is responsible for sending data that is meaningful to the hardcopy unit. The copy terminates when the terminal detects an EOF (end-of-file) string or when the operator presses the Cancel key. For example:

EcJC3HO:2to3HC:

As in all Host syntax character arrays, the integers in this command indicate the length of the array that follows.



Setup Command	Encoded Command From Host
BEGINPANEL 1000,1000,1	EcLP''z'Z1
DRAW 1500,1851	EcLF.In+W
DRAW 2000,1000	EcLF''z/T
DRAW 1000,1000	EcLF''z'Z
BEGINPANEL 1333,1192,1	EcLP)aj*M1
DRAW 1500,1459	EcLF+II+W
DRAW 1667, 1192	EcLF)cj-@
END PANEL	EcLE

4526-14A

Figure 5-4. Creating a Panel With Multiple Boundaries.

CRLF

Specifies whether a $^{C}_{R}$ character also implies an $^{L}_{F}$.

Host Syntax

^{E}cKR crlf-mode

Setup Syntax

CRLF crlf-mode

crlf-mode: integer (keyword in Setup syntax); 0 or **no** means that $^{C}_{R}$ does not imply $^{C}_{R}$ $^{L}_{F}$, 1 or **yes** means that $^{C}_{R}$ does imply $^{C}_{R}$ $^{L}_{F}$.
Defaults: Factory = 0 (no)
Power-Up = Saved in memory
Omitted = 1 (yes)

The term "$^{C}_{R}$ does not imply $^{C}_{R}$ $^{L}_{F}$" means that the $^{C}_{R}$ character is not interpreted as a carriage return and linefeed combination.

The term "$^{C}_{R}$ implies $^{C}_{R}$ $^{L}_{F}$" means that when the terminal receives a $^{C}_{R}$ from the host or if the terminal is in Local mode and you press the Return key, the terminal interprets the $^{C}_{R}$ as a carriage-return and line-feed combination. However, when you press the Return key, the $^{L}_{F}$ character is not sent to the host.)

Notice that if the parameter is omitted in Setup syntax it defaults to yes (the equivalent of 1 in Host syntax), but if it is omitted in Host syntax it defaults to 0 (the equivalent of keyword no).

DEFINE MACRO

Creates and deletes macros, which can be expanded on command from the host, at the keyboard, or both.

Host Syntax

^{E}cKD macro-number,
macro-contents

Setup Syntax

DEFINE macro-number,
string

macro-number: integer (key specifier or integer in Setup syntax); specifies the name of the macro being defined. Valid values in the range –150 through 32767. A value of –1 deletes all volatile macros.
Default: Omitted = 0

macro-contents: integer array; specifies ASCII decimal equivalent (ADE) integers that represent the characters defining the macro. (Host syntax only.)
Default: Omitted = Empty array

string: delimited string; defines the macro. To use $^{C}_{R}$ or the special editing characters in the macro definition, you must precede them with the *literal character;* see SET EDIT CHARS for information on the literal character. (Setup syntax only.)

After a macro is defined, you can expand it either with EXPAND MACRO (for any macro) or by pressing a key corresponding to the macro number (for macros numbered –150 through 143, excluding –1).

The following example shows how to define Macro 500 as "XYZ".

1. Get ADE values for X, Y, and Z.

Character	ADE
X	88
Y	89
Z	90

2. Convert the integers 500, 88, 89, and 90 to the terminal's integer format.

Integer	Converted Format
500	_4
88	E8
89	E9
90	E:

3. Issue this command:

^{E}cKD_43E8E9E:

The integer **3** indicates that three values follow. This convention is followed for all integer arrays.

Volatile and Nonvolatile Macros. *The Graphics Terminal* section explains the difference between volatile and nonvolatile macros. DEFINE MACRO defines and deletes only volatile macros. To define a nonvolatile macro, use DEFINE NONVOLATILE MACRO followed by SAVE NONVOLATILE PARAMETERS.

Deleting Macros. To delete a specific macro in Setup syntax, enter DEFINE MACRO but don't include *string* after the macro number. In Host syntax, indicate a length of 0 for *macro-contents*. The following example shows the command in Host syntax form that deletes macro 500 (500 is _4 in integer format):

^{E}cKD_40

To delete all macros, specify macro number –1. The following Host syntax command deletes all macros (–1 is ! in integer format):

^{E}cKD!0

Key Macros. Table 5-2 shows how integers from –150 to 143 (excluding –1) correspond to the terminal's keys. Macros in this range are called *key macros*. Note that the integer for most keys is the ADE of the character the key normally produces.

As shown in Table 5-2, each key is associated with up to four macros: unshifted, shifted, Ctrl, and Ctrl-shifted (in some cases two or more of the unshifted/shifted positions access the same macro).

Note that when you define a key in Setup syntax you can either enter the key's ASCII decimal equivalent or just press the key (provided this key normally produces an ASCII character). Enter the macro's definition as a delimited string rather than an array of ADE integers. For example, to define the Q key to send the characters QUIT, enter:

DEFINE Q "QUIT"

Key-Execute Character. A special character, called the *key-execute character*, can be included in a key macro to cause a macro to be used only at the terminal instead of being sent to the host. Enclose the string you want used locally between two occurrences of the key-execute character named in the last SET KEY EXECUTE CHARACTER.

If the characters between the key-execute characters form a valid command, the terminal executes the command. Otherwise, the macro is displayed as if the operator typed it.

Table 5-2

MACRO NUMBERS INVOKED BY KEYS (North American Keyboard)

Keys	Key Label	Un-Shift	Shift	Ctrl	Ctrl-Shift
Standard Keyboard and Function Keys	{ [	91	123	27	27
	! 1	49	33	49	33
	@ 2	50	64	50	0
	# 3	51	35	51	35
	$ 4	52	36	52	36
	% 5	53	37	53	37
	^ 6	54	94	54	30
	& 7	55	38	55	38
	* 8	56	42	56	42
	(9	57	40	57	40
	) 0	48	41	48	41
	_ -	45	95	45	31
	+ =	61	43	61	43
	}]	93	125	29	29
	Rub Out	127	–34	–35	–36
	Esc	27	–37	–38	–39
	~ |	124	126	124	126
	Q	113	81	17	17
	W	119	87	23	23
	E	101	69	5	5
	R	114	82	18	18
	T	116	84	20	20
	Y	121	89	25	25
	U	117	85	21	21
	I	105	73	9	9
	O	111	79	15	15

(continued)

Table 5-2 (cont)

MACRO NUMBERS INVOKED BY KEYS (North American Keyboard)

Keys	Key Label	Un-Shift	Shift	Ctrl	Ctrl-Shift
Standard Keyboard and Function Keys (cont)	P	112	80	16	16
	\	92	96	28	28
	Back Space	8	-40	-41	-42
	Line Feed	10	-43	-44	-45
	Tab	9	-46	-47	-48
	A	97	65	1	1
	S	115	83	19	19
	D	100	68	4	4
	F	102	70	6	6
	G	103	71	7	7
	H	104	72	8	8
	J	106	74	10	10
	K	107	75	11	11
	L	108	76	12	12
	: ;	59	58	59	58
	" '	39	34	39	34
	Return	13	-49	-50	-51
	Z	122	90	26	26
	X	120	88	24	24
	C	99	67	3	3
	V	118	86	22	22
	B	98	66	2	2
	N	110	78	14	14
	M	109	77	13	13
	< ,	44	60	44	60
	> .	46	62	46	62
	? /	47	63	47	63
	Space Bar	32	-52	-53	-54
	F1	128	136	-2	-10

Table 5-2 (cont)

MACRO NUMBERS INVOKED BY KEYS (North American Keyboard)

Keys	Key Label	Un-Shift	Shift	Ctrl	Ctrl-Shift
Standard Keyboard and Function Keys (cont)	F2	129	137	-3	-11
	F3	130	138	-4	-12
	F4	131	139	-5	-13
	F5	132	140	-6	-14
	F6	133	141	-7	-15
	F7	134	142	-8	-16
	F8	135	143	-9	-17
	G Eras Dialog	-111	-117	-123	-129
	Cancel Setup	-112	-118	-124	-130
	D Copy S Copy	-113	-119	-125	-131
	Color Select	-114	-120	-126	-132
	D Eras S Eras	-115	-121	-127	-133
	Break	-116	-122	-128	-134
Numeric Keypad Keys	0	-55	-69	-83	-97
	1	-56	-70	-84	-98
	2	-57	-71	-85	-99
	3	-58	-72	-86	-100
	4	-59	-73	-87	-101
	5	-60	-74	-88	-102
	6	-61	-75	-89	-103
	7	-62	-76	-90	-104
	8	-63	-77	-91	-105
	9	-64	-78	-92	-106
	.	-65	-79	-93	-107
	,	-66	-80	-94	-108
	-	-67	-81	-95	-109
	Ent	-68	-82	-96	-110
Joydisk Positions	Right	-135	-139	-143	-147
	Up	-136	-140	-144	-148
	Left	-137	-141	-145	-149
	Down	-138	-142	-146	-150

Each key-execute character the terminal encounters in a macro switches how key macros are used. If the terminal is sending macros to the host, the key-execute character means "use what follows locally." If the terminal is currently using macros locally, the key-execute character means "send what follows to the host."

Always include the second key-execute character in the macro. If you omit the second key-execute character, subsequent macros are expanded at the terminal, even if they are intended for the host. This would continue until the terminal expands a macro that includes the key-execute character.

DEFINE NONVOLATILE MACRO

Creates and deletes *both* the volatile and nonvolatile version of a macro.

Host Syntax

EcKO macro-number, macro-contents

Setup Syntax

NVDEFINE macro-number, string

macro-number: integer (key specifier or integer in Setup syntax); specifies the name of the macro being defined. Valid values in the range –150 through 32767. A value of –1 deletes all macros, volatile and nonvolatile.
Default: Omitted = 0

macro-contents: integer array; specifies ASCII decimal equivalent (ADE) integers that represent the characters defining the macro. (Host syntax only.)
Default: Omitted = Empty array

string: delimited string; defines the macro. To use **C**R or the special editing characters in the macro definition, you must precede them with the *literal character;* see SET EDIT CHARS for information on the literal character. (Setup syntax only.)

This command has the same effect as DEFINE MACRO, except it defines both the volatile and nonvolatile version of a macro, rather than just the volatile version. *The Graphics Terminal* section explains the difference between volatile and nonvolatile macros.

NOTE

This command must be followed by SAVE NONVOLATILE PARAMETERS to actually save or delete a macro in nonvolatile memory.

To delete a single nonvolatile macro, use DEFINE NONVOLATILE MACRO specifying an array length of 0 for *macro-contents*, and then use SAVE NONVOLATILE PARAMETERS.

DRAW

Draws a vector from the current graphics position to the specified location.

Host Syntax

EcLF position

Setup Syntax

DRAW position

position: xy-coordinate; indicates the point to draw to. The valid range of values is 0 through 4095 for both the x- and y-coordinate.
Default: Omitted = (0,0)

DRAW has the same effect as sending the terminal an xy-coordinate when the terminal is in Vector mode. The terminal can execute DRAW when it is in any mode except Ansi or Bypass mode.

The terminal draws the vector in the current line style and line index. Use SET LINE STYLE to set the line style. The line index is set with SET LINE INDEX.

DRAW does not affect the terminal's mode. For example, if the terminal was in Alpha mode when it received the DRAW command, it stays in that mode.

See the section called *The Graphics Terminal* for examples of how to create a line.

DRAW MARKER

Draws a marker at a specified location.

Host Syntax

^{E}cLH marker-position

Setup Syntax

MARKER marker-position

marker-position: xy-coordinate; indicates the point where the marker is drawn. Valid range of values is 0 to 4095 for both the x- and y-coordinate.
Default: Omitted = (0,0)

DRAW MARKER has the same effect as sending the terminal an xy-coordinate when the terminal is in Marker mode. The terminal can be in any mode except Ansi or Bypass mode when it executes DRAW MARKER.

The marker specified by the most recent SET MARKER TYPE is drawn in the current line index; SET LINE INDEX sets the line index.

DRAW MARKER does not affect the terminal's mode. For example, if the terminal is in Vector mode when it receives the DRAW command, it stays in that mode.

ENABLE DIALOG AREA

Enables or disables the dialog area.

Host Syntax

^{E}cKA enable-mode

Setup Syntax

DAENABLE enable-mode

enable-mode: integer (keyword in Setup syntax); 0 or **no** disables the dialog area, 1 or **yes** enables the dialog area.
Defaults: Factory = 1 (yes)
Power-Up = Saved in memory
Omitted = 1 (yes)

When the dialog area is enabled and the terminal is in Tek mode, all alphatext is directed to the dialog buffer. (For the text to be seen, the dialog area must also be made visible; see SET DIALOG AREA VISIBILITY.)

When the dialog area is disabled and the terminal is in Tek mode, the terminal emulates Tektronix 4010 Series terminals, which do not have a dialog area. The terminal displays alphatext at the current position in the graphics area.

In Ansi mode, the terminal automatically directs text to the dialog area, regardless of whether or not the dialog area is enabled.

Table 5-3 summarizes the effects of enabling and disabling the dialog area.

Table 5-3

EFFECTS OF ENABLE DIALOG AREA

Feature	Dialog Area Disabled	Dialog Area Enabled
Destination of Alphatext	Current position in terminal space	Current alpha cursor position in dialog area
G Eras Key, S Eras Key, and PAGE Command	Erases the graphics area of the screen Takes the terminal out of GIN mode Resets the terminal to Line Style 1 Sets current position to home position (0,3071) Puts terminal in Alpha mode	Erases the graphics area
$^{C}_{R}$ Character	Puts terminal in Alpha mode Performs a carriage return action Resets the terminal Line Style to 1 Takes the terminal out of GIN mode	If the terminal is in Alpha mode, performs a carriage return in the dialog area No action if the terminal is in Vector or Marker mode

ENABLE KEY EXPANSION

Enables or disables key macros.

Host Syntax

^{E}**cKW** switch

Setup Syntax

KEYEXPAND switch

switch: integer (keyword in Setup syntax); 1 or **yes** enables key expansion, 0 or **no** disables key expansion.
Defaults: Factory = 1 (yes)
Power-Up = 1 (yes)
Omitted = 1 (yes)

With DEFINE MACRO, you can define a key so that it produces a character (or characters) other than the character it normally produces. ENABLE KEY EXPANSION enables or disables alternate key definitions.

When key expansion is disabled, each key produces its normal character; that is, when you press a defined key, its macro is not expanded. When key expansion is enabled, macros are expanded as usual.

This command does not delete macros. All key macros remain in memory and you can reenable them at any time.

This command does not affect how the host uses macros. Even when key expansion is disabled, the host can still issue EXPAND MACRO to expand any macros, including those associated with keys.

ENABLE 4010 GIN

Puts the terminal in Graphic Input (GIN) mode.

Host Syntax

E_C S_B

When Graphics Input (GIN) mode is enabled, the graphics cursor appears on the screen. The terminal is now in GIN mode. The operator uses the joydisk to position the cursor and then presses any alphanumeric key. The terminal sends to the host a 4010 GIN report indicating the xy-coordinates of the cursor position and the identity of the key pressed. The graphics cursor then disappears and the terminal leaves GIN mode with the current position set to the cursor position.

4010 GIN Report

When the operator presses an alphanumeric key to send the cursor position to the host program, the terminal generates a 4010 GIN report. This report tells the host program which key the operator pressed and the location of the crosshair cursor in terminal space.

NOTE

The 4010 GIN report regards terminal space as a 1024x1024 area rather than the 4096x4096 area used when specifying locations for display. Reported coordinate values must be multiplied by 4 to give coordinates consistent with those used in other commands.

The characters in the report are:

1. The ASCII character corresponding to the key the operator pressed.
2. The Hi-X character — the five most significant bits of x's binary value preceded by 01.
3. The Lo-X character — the five least significant bits of x's binary value preceded by 01.
4. The Hi-Y character — the five most significant bits of y's binary value preceded by 01.
5. The Lo-Y character — the five least significant bits of y's binary value preceded by 01.

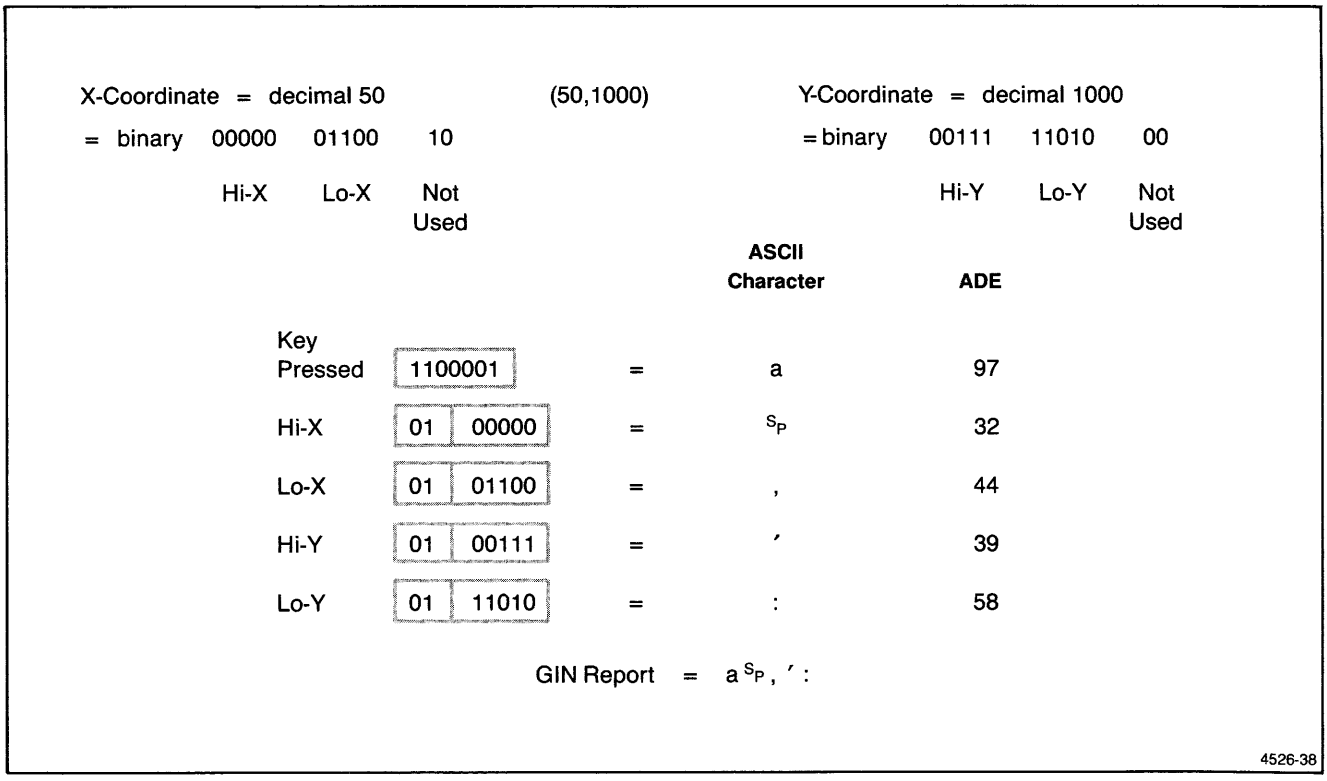


Figure 5-5. The GIN Cursor Position Report.

Since only the ten most significant bits of the x- and y-coordinate are reported, the reported values are an approximation of the graphics cursor position. After a host program converts reported coordinates by multiplying the values by 4, the derived values can differ from the true values by up to three units.

Example

If the operator positions the crosshair cursor at the point (50,1000) and then presses the lowercase "a" key, the following report is sent (in 1024x1024 terminal space):

a ^{S_P} **,':**

Figure 5-5 and the following explanations show why these five characters are sent.

ASCII Character. The first character in the report is **a**, since that was the key the operator pressed.

Hi-X Character. The x-coordinate in this example is 50, which converts to binary 0000 0011 0010. The Hi-X character always has the prefix 01 and is followed by the five high-order bits in the binary numeral. So in this case, it is:

0100000

This is the space (^{S_P}) character.

Lo-X Character. This character has the prefix 01 followed by the five least significant bits of the x-coordinate. This produces:

0101100

This is a comma (,).

Hi-Y Character. The y-coordinate in this example is 1000, which converts to binary 0011 1110 1000. The Hi-Y character always has the prefix 01 and is followed by the five high-order bits in the binary numeral. In this case, it is:

0100111

This corresponds to the apostrophe (') character.

Lo-Y Character. The Lo-Y character in the report consists of the prefix 01 followed by the five least significant bits in the binary numeral:

0111010

This corresponds to the colon (:) character.

END PANEL

Terminates a panel definition.

Host Syntax

^{E}cLE

Setup Syntax

ENDPANEL

The panel boundary is closed, the panel is filled, and the current graphics position is set to the panel's first boundary point.

ENTER ALPHA MODE

Puts the terminal in Alpha mode.

Host Syntax

^{U_S}

When the terminal is in Alpha mode, it interprets and displays ASCII characters as text. Alpha mode is the terminal's power-up default. The text is displayed in the dialog area if the dialog area is enabled and visible. The ^{U_S} character has an ADE value of 31.

The terminal leaves Alpha mode when it receives an ENTER VECTOR MODE or ENTER MARKER MODE command.

ENTER BYPASS MODE

Puts the terminal in Bypass mode.

Host Syntax

$^{E}_{C}$ $^{C}_{N}$

When the terminal is in Bypass mode, characters coming from the host are ignored until the terminal receives the *bypass-cancel character*. The terminal automatically enters Bypass mode when it sends reports to the host except when the bypass-cancel character is a null (see SET BYPASS CANCEL CHARACTER).

If the host echoes characters to the terminal as the host receives the characters, the terminal would receive its own report and try to execute it as a command. Bypass mode prevents this from occuring.

ENTER BYPASS MODE allows you to use Bypass mode in other circumstances where you find it useful.

ENTER MARKER MODE

Puts the terminal in Marker mode.

Host Syntax

$^{F}_{S}$

When the terminal is in Marker mode, it interprets ASCII characters as xy-coordinates and draws markers at the specified coordinates. (SET MARKER TYPE specifies the kind of marker the terminal draws.) The $^{F}_{S}$ character has an ADE value of 28.

See *XY-Coordinates in Host Syntax* at the beginning of this section for details on how to send those xy-coordinate parameters.

The terminal cannot go directly from Marker mode to Vector mode; it must first be placed in Alpha mode, then in Vector mode.

The use of Marker mode is explained and illustrated in *The Graphics Terminal* section.

ENTER VECTOR MODE

Puts the terminal in Vector mode.

Host Syntax

$^{G}_{S}$

When the terminal is in Vector mode, it interprets ASCII characters as xy-coordinates specifying coordinates to which a vector is to be drawn or a point to which the graphics position is to be moved. The cursor *moves* to the first xy-coordinate specified; it draws a vector to subsequent points. The $^{G}_{S}$ character has an ADE value of 29.

To *draw* rather than move, when specifying the first coordinate after entering Vector mode, include the $^{B}_{L}$ character immediately after the $^{G}_{S}$ character.

See *XY-Coordinates in Host Syntax* at the beginning of this section for details on how to send those xy-coordinate parameters.

EXPAND MACRO

Causes the terminal to expand a macro.

Host Syntax

^{E}**cKX** macro-number

Setup Syntax

EXPAND macro-number

macro-number: integer; indicates the macro to expand. Valid ranges are –150 through –2 and 0 through 32767. **Default:** Omitted = 0

When a macro is expanded, the terminal looks up the macro's definition and responds as if that string was sent from the host. See DEFINE MACRO for information on how to define a macro's contents.

FACTORY

Temporarily sets all volatile and nonvolatile parameters to factory default values and takes the terminal out of Setup mode.

Setup Syntax

FACTORY

To permanently change parameters to factory default, use the SAVE NONVOLATILE PARAMETERS (NVSAVE) command after entering FACTORY.

GRAPHIC TEXT

Writes a string of graphtext starting at the current graphics position.

Host Syntax

EcLT text

Setup Syntax

GTEXT text

text: character array (delimited string in Setup syntax); indicates the ASCII characters to be displayed. Each character must have an ADE in the range 32 through 126 ($^{S}_{P}$ through ~).
Default: Omitted = Empty string

The terminal displays the text so that the current position is aligned with the first character cell. After the text is displayed, the current position is updated to align with the last character of the displayed text. The character path determines the exact positions. See SET CHARACTER PATH for more information.

If the string is too long to fit in the 4096x4096 terminal space, the characters are clipped at the edge of terminal space. The current position is set to the edge of terminal space where the overflow occurred.

Graphtext cannot be included in a panel. Therefore, if you attempt to use GRAPHIC TEXT between BEGIN PANEL BOUNDARY and END PANEL commands, an error results and the panel is closed as if END PANEL had been executed.

The Ansi mode command SCS (SELECT CHARACTER SET) determines the character set used for graphtext. See the *Screen Editor Support* Section for information on character sets.

These 4100-style commands affect how graphtext is displayed: SET CHARACTER PATH, SET GRAPHTEXT ROTATION, SET GRAPHTEXT SIZE, SET GRAPHICS AREA WRITING MODE, SET TEXT INDEX, SET SURFACE COLOR MAP, and SET VIEW ATTRIBUTES.

HARDCOPY

Causes an attached hardcopy unit to make a copy of the terminal's screen or dialog area.

Host Syntax

EcKH hardcopy-code

hardcopy-code: integer; 0 and 1 copy the entire screen, 2 produces a positive hardcopy image of the entire screen, 3 copies only the dialog area. (To copy only the graphics area, make the dialog area invisible and use 0 or 1.)
Default: Omitted = 0

If you use 0, 1, or 3, the copy will be a negative image of the display (white areas copy black, black areas copy white), which is the way a hardcopy normally appears and typically is most useful. If you prefer a positive hardcopy image, use *hardcopy-code* 2 in the command. On the keyboard, pressing Ctrl-S Copy makes a negative image copy.

HELP

Displays a message that shows the escape sequence, Setup name, and parameter types for a command or cluster of commands.

Setup Syntax

HELP name

name: string; either a Setup mode command name or the name of a cluster of commands for which you want information. The cluster names are COMMUNICATIONS, GIN, HARDCOPY, and PIXELS.
Default: Omitted = All commands

Enter this command without a parameter to display all of the terminal's commands. In addition to the commands, the listing shows the kinds of parameters the commands take.

You can stop scrolling by pressing Ctrl-S. Press Ctrl-Q to continue scrolling.

IGNORE DELETES

Determines whether the terminal ignores the $^{D}_{T}$ character.

Host Syntax

^{E}**cKI** ignore-deletes-mode

Setup Syntax

IGNOREDEL ignore-deletes-mode

ignore-deletes-mode: integer (keyword in Setup syntax); 0 or **no** causes the terminal not to ignore $^{D}_{T}$ characters, 1 or **yes** causes the terminal to ignore $^{D}_{T}$ characters.

Defaults: Factory = 0 (no)
Power-Up = Saved in memory
Omitted = 1 (yes)

See the heading "Coping With $^{D}_{T}$ Filler Characters" in the *Communications* section for further explanation of the use of this command.

LEARN / NVLEARN

Provides an easy method for programming keys.

Setup Syntax

LEARN or **NVLEARN**

Based on which command you enter, LEARN OR NVLEARN, a macro or a nonvolatile macro is defined. Other than that, the two commands work the same.

In Setup mode enter:

LEARN $^{C}_{R}$ or **NVLEARN** $^{C}_{R}$

This starts a learn function and displays the following message:

Press the key to be defined:

This key can be any key on the keyboard (except CANCEL, CAPS LOCK, CTRL, and SHIFT), including function keys, the BREAK key, a shifted space bar, etc. If you press an ASCII key, the terminal echoes the ASCII character on the screen. If you press a function key or one of the special key-codes, such as a shifted return key, the terminal echoes the decimal macro number. Then the terminal displays the following message:

Enter definition. (F1 terminates definition, F2 deletes last character)

Now simply type in the ASCII definition; no delimiters are necessary. Each ASCII character you enter is added to the definition until you press F1, which ends the definition and returns control to Setup mode. Pressing F2 deletes the last character entered in the definition. If you press any other non-ASCII key during the definition phase, the bell rings and the terminal ignores that key-press.

You can cancel the LEARN definition before ending it (with F1) by pressing the CANCEL key.

If there is not enough memory available to store a macro definition when you issue the LEARN command, the terminal returns control to Setup mode after displaying this message:

Error: Not enough memory available.

If, however, memory fills up *while* you are writing the definition, then pressing more ASCII keys will only ring the bell and the keys will be ignored.

LFCR

Specifies whether or not a $^{L}_{F}$ character also implies a $^{C}_{R}$.

Host Syntax

EcKF lfcr-mode

Setup Syntax

LFCR lfcr-mode

lfcr-mode: integer (keyword in Setup syntax); 0 or **no** means that $^{L}_{F}$ does not imply $^{L}_{F}$ $^{C}_{R}$, 1 or **yes** means that $^{L}_{F}$ does imply $^{L}_{F}$ $^{C}_{R}$.
Defaults: Factory = 0 (no)
Power-Up = Saved in memory
Omitted = 1 (yes)

The term "$^{L}_{F}$ implies $^{L}_{F}$ $^{C}_{R}$" means that when the terminal receives the $^{L}_{F}$ from the host or you type $^{L}_{F}$ with the terminal in Local mode, the terminal displays it as if a $^{L}_{F}$ and $^{C}_{R}$ were received. However, when you press the Line Feed key, the $^{C}_{R}$ character is not sent to the host.

LOCAL

Specifies whether terminal is in Local mode.

Setup Syntax

LOCAL local-mode

local-mode: keyword; **no** takes the terminal out of Local mode, **yes** puts the terminal into Local mode.

LOCK KEYBOARD

Disables or enables the keyboard.

Host Syntax

EcKL lock-mode

lock-mode: integer; 0 enables (unlocks) the keyboard), 1 disables (locks) the keyboard.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

Locking the keyboard prevents the operator from entering data when important information is being sent from the host.

When the keyboard is locked, all keys except the Cancel and Break keys are inoperative. The terminal bell rings whenever the operator presses any key except Cancel or Break. The operator can unlock the keyboard by pressing the Cancel or Break key.

MACRO STATUS

Displays the definition of one or all macros.

Setup Syntax

MACROSTATUS macro-number

macro-number: integer in the range –150 through 32767 or a key specifier; specifies the macro whose definition you want displayed. The parameter –1 or the keyword **all** displays all macros.
Default: Omitted = All

This command shows the defintion of the volatile form of a macro. This form may or may not match the nonvolatile form, depending on whether the macro has been redefined since the terminal was powered-up or reset.

To see the definition of a nonvolatile macro, issue MACROSTATUS immediately after you power up or reset the terminal.

MOVE

Sets the current graphics position without drawing a vector.

Host Syntax

^{E}**cLG** position

Setup Syntax

MOVE position

position: xy-coordinate; specifies the screen position to set the current graphics position to.
Default: Omitted = (0,0)

This command does not change the display. It is analogous to lifting a pen from the paper in a drawing and moving it to a new location.

The section titled *The Graphics Terminal* gives examples of how to use this command with the DRAW command to create lines.

PAGE

Erases the screen (except the dialog area).

Host Syntax

$^{E}_{C}$ $^{F}_{F}$

This command has the same effect as pressing the terminal's G Eras key.

If the dialog area is enabled, the terminal just erases the graphics portion of the screen.

If the dialog area is not enabled, the terminal erases the graphics portion of the screen:

- Resets the current line style to 0 (solid lines)
- Terminates 4010 GIN mode
- Returns the graphics beam to its home position (0,3071)
- Puts the terminal in Alpha mode

PIXEL COPY (Requires Optional Pixel ROMs)

Copies pixels from one rectangular region to another.

Host Syntax

^{E}**cRX** destination-surface, destination-lower-left-corner, first-source-corner, second-source-corner

Setup Syntax

PXCOPY destination-surface, destination-lower-left-corner, first-source-corner, second-source-corner

destination-surface: integer; names the surface to which pixels are to be copied. The values –1, 0, and 1 are valid, but since Surface 1 is the terminal's only surface, all valid values select Surface 1.
Default: Omitted = 0 (Surface 1)

destination-lower-left-corner: xy-coordinate; specifies the lower-left corner of a rectangular region on the destination surface in raster memory space. This destination region is the same width and height as the source region specified by the source corners. The range of values for x is 0 through 511; for y, 0 through 359.
Default: Omitted = (0,0)

first-source-corner: xy-coordinate; one corner of a rectangular region on the current pixel surface. The PIXEL COPY command copies the pixel at this corner to the lower-left corner of the destination region. Then it copies each remaining pixel in the source region onto a corresponding pixel in the destination region. The valid range of values for x is 0 through 511; for y, 0 through 359.
Default: Omitted = 0,0

second-source-corner: xy-coordinate; the corner opposite the first source corner in the source region. The valid range of values for x is 0 through 511; for y, 0 through 359.
Default: Omitted = 0,0

The PIXEL COPY command uses the ALU mode specified in the most recent BEGIN PIXEL OPERATIONS command. Copying pixels to the same location on the pixel writing surface can only occur in XOR mode to erase the pixels, or in other ALU modes to form mirror images (see following "Source Corners" discussion). Using other ALU modes to copy (without mirroring) to the same location will do nothing, and will not generate an error.

You can copy pixels using the off-screen raster memory with x values from 480 to 511, but a level 0 warning will be generated.

Source Corners. The two source corners need not be the lower-left and upper-right corners of the source region. However, if they are not, the pixels written to the destination region form a mirror (or inverted) image of the picture formed by the pixels in the source region. The pixel at the first source corner is copied onto the pixel at the lower-left corner of the destination region. The pixel at the second source corner is copied onto the pixel at the upper-right corner of the destination region.

PROMPT MODE

Puts the terminal in Prompt mode or terminates Prompt mode.

Host Syntax

ECNM prompt-mode

Setup Syntax

PROMPTMODE prompt-mode

prompt-mode: integer (keyword parameter in Setup syntax); 0 or **no** terminates Prompt mode, 1 or **yes** puts the terminal into Prompt mode.
Defaults: Factory = 0 (no)
Power-Up = 0 (no)
Omitted = 0 (yes)

Prompt mode is explained in the *Communications* section.

RASTER WRITE (Requires Optional Pixel ROMs)

Specifies individually the color indices of a specified number of pixels in the current pixel viewport starting at the current beam position.

Host Syntax

^{E}cRP number-of-pixels, color-index-codes

Setup Syntax

PXRASTERWRITE number-of-pixels, color-index-codes

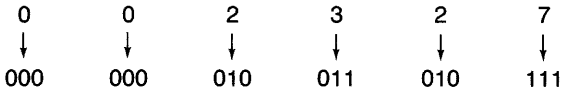
number-of-pixels: integer; specifies the number of pixels represented by the character array that follows this parameter. Must be in the range 0 through 65535.
Default: Omitted = Error

color-index-codes: character array; specifies in a packed format the color indices for the pixels specified by *number-of-pixels*. Each code is an ASCII character in the range ^{S}P through ' (ADE 32 through 96).
Default: Omitted = 0

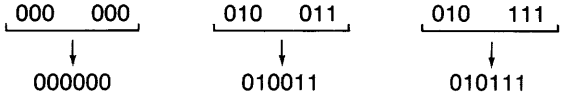
You can regard the data bits embedded within the code characters in the *color-index-codes* array as a sequential string of bits. The bits are grouped to form color indices for individual pixels according to the *bits-per-pixel* parameter in the most recent BEGIN PIXEL OPERATIONS command. Figures 5-6 and 5-7 show how to pack color indices into the *color-index-codes* parameter.

If *bits-per-pixel* is 3, then pack the color indices 0, 0, 2, 3, 2, 7 into a RASTER WRITE command as follows:

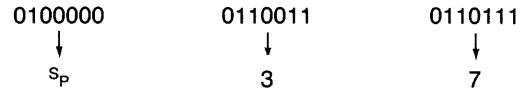
1. Express the color indices as three-bit binary numerals:



2. Group the binary bits into six-bit groups:



3. Add 32 (binary 100000) to these six-bit binary numerals to form seven-bit ASCII characters:



4. Issue a RASTER WRITE command. The command's first parameter is the integer 6, because the command holds six color indices. The second parameter is a character array holding the characters ^{S}P, **3**, and **7**.

RASTER WRITE = **^{E}cRP 6 3^{S}P37**

4526-39

Figure 5-6. Packing Color Index Codes Using Three Bits Per Pixel.

If you use 1 for the *bits-per-pixel* parameter in the BEGIN PIXEL OPERATIONS command, then six color indices (each consisting of a single bit) will fit into each code character. If *bits-per-pixel* is 2, then three color indices fit into each code character. If *bits-per-pixel* is 3, then two color indices fit into each code character, as shown in Figure 5-6.

If the *bits-per-pixel* parameter is set to 4, one-and-a-half color indices fit into each code character. That is, every pair of codes holds three color indices. Figure 5-7 shows the packing scheme. If the *bits-per-pixel* parameter is 6, the terminal interprets each code character as containing only one color index, and you can represent each color index in the range 0 through 7 with a single ASCII character.

The special code character ' (ADE 96), functions much like a $^{C}R^{L}F$ sequence; it moves the pixel beam position to the start of the following row of pixels. The ' code is not included in the pixel count.

If *bits-per-pixel* is 4, then pack the color indices 0, 0, 2, 3, 12, 15 into a RASTER WRITE command as follows:

1. Express the color indices as four-bit binary numerals:

0	0	2	3	12	15
↓	↓	↓	↓	↓	↓
0000	0000	0010	0011	1100	1111

2. Group the binary bits into six-bit groups:

0000 0000 0010	0011 1100 1111
↓ ↓	↓ ↓
000000 000010	001111 001111

3. Add 32 (binary 100000) to these six-bit binary numerals to form seven-bit ASCII characters:

0100000	0100010	0101111	0101111
↓	↓	↓	↓
^{S}P	"	/	/

4. Issue a RASTER WRITE command. The command's first parameter is the integer 6, because the command holds six color indices. The second parameter is a character array holding the characters ^{S}P, ", /, and /.

RASTER WRITE = ^{E}C**RP 6 4**^{S}P"//

4526-40

Figure 5-7. Packing Color Index Codes Using Four Bits Per Pixel.

RECTANGLE FILL (Requires Optional Pixel ROMs)

Sets all the pixels in a rectangle to the specified color.

Host Syntax

^{E}**cRR** lower-left-corner,
upper-right-corner,
fill-index

Setup Syntax

PXRECTANGLE lower-left-corner,
upper-right-corner,
fill-index

lower-left-corner: xy-coordinate; specifies one corner of a rectangle in raster memory space. The range of values for x is 0 to 511; for y, 0 to 359.
Default: Omitted = (00)

upper-right-corner: xy-coordinate; specifies the opposite corner of that rectangle. The range of values for x is 0 to 511; for y, 0 to 359.
Default: Omitted = (00)

fill-index: integer in the range 0 to 65535; the color index with which the rectangle is filled.
Default: Omitted = (00)

The color indices are written into raster memory using the ALU mode specified in the most recent BEGIN PIXEL OPERATIONS command.

If the lower-left and upper-right corners of the rectangle have the same x-value, then the rectangle filled is one-pixel wide. Likewise, if the lower-left and upper-right y-values are the same, then the rectangle filled is one-pixel high.

This command also functions in off-screen raster memory where the x-value is 480 through 511; however, a level 0 warning will be generated.

REPORT ERRORS

Causes the terminal to send an Error Message Report to the host.

Host Syntax

^{E}**cKQ**

This command is intended to be used only by the host.

When the host sends this command to the terminal, the terminal responds by sending to the host the eight most recently detected error codes — starting with the most recent one. Included in the report are the error codes, their severity levels, and how many times each error was detected. (Error report severity levels are explained in the *Error Codes* appendix.)

Error Message Report

The Error Message Report is a series of up to eight individual error messages. The error report message has the following format:

error-code-report
error-code-report
. . . (up to eight error-code-reports)
EOM-indicator

An example of an actual Error Message Report with two individual error messages is:

KE1121$^{C}_{R}$ LE0311$^{C}_{R}$

As shown in the *Error Codes* appendix, the first error message (*KE11*) means "invalid echo mode"; the numeral 2 indicates the severity level of the command; the numeral 1 indicates the number of occurences of this error; the $^{C}_{R}$ is the currently defined end-of-message string. The second error message consists of *LE03*, which means "no panel is currently being defined"; the 1 is the severity level of that command, and the next 1 indicates the number of occurences of that error; $^{C}_{R}$ is the currently defined end-of-message string.

The following paragraphs explain each part of the report in more detail.

Error-code-report. The error code report itself is a series of ASCII characters, as illustrated in the preceding example.

EOM-indicator. The EOM-indicator is sent to show between each error message in the report and at the end of the report to show that it is completed. If there are no errors to report, the terminal sends one EOM-indicator.

REPORT SYNTAX MODE

Sends a Terminal Settings Report that contains the syntax mode status to the host.

Host Syntax

^{E}c#!0

This command has the same effect as a REPORT TERMINAL SETTINGS command issued for the SELECT CODE command (as if **^{E}cIQ%!** was sent from the host). See REPORT TERMINAL SETTINGS.

This command is recognized in Ansi, Tek, and VT52 modes.

REPORT TERMINAL SETTINGS

Reports terminal status for specified setting.

Host Syntax

^{E}cIQ inquiry-code

inquiry-code: two characters; indicates the opcode of a command or a special two-character inquiry code.

If you specify a command opcode, the terminal reports the current values of the command's settings. Table 5-4 describes the report you receive if you specify a *special* inquiry code. The opcode is the two characters that follow ^{E}c in the Host syntax form of a command. The terminal responds to the REPORT TERMINAL SETTINGS by sending the status of the specified command to the host. For example, the following command inquires about the status of Prompt mode:

^{E}cIQNM

The terminal sends the Prompt mode status (that is, whether or not the terminal is in Prompt mode) to the host. More detail is included later in this explanation under the heading "Examples."

A *special inquiry code* instead of an opcode can be used to request information concerning the amount of terminal memory remaining, the terminal model number, and the firmware version installed in the terminal. Table 5-4 shows the inquiry code used in each case and the format of the reports returned.

Table 5-4

SPECIAL INQUIRY CODES AND REPORTS

Inquiry Code	Format of Report	Explanation
?M	*total-available-memory, largest-block-available*	Two encoded integers showing the total and largest blocks of memory available
?T	*terminal-model-number*	An integer showing the terminal model number, for example *4105*
00	*standard-firmware version-number*	An integer showing the version number of the firmware currently installed

The following command would be used to request information about the amount of terminal memory remaining (more detail is provided later under the heading "Examples"):

^{E}c IQ?M

The Terminal Settings Report

The terminal sends the terminal settings report in response to REPORT TERMINAL SETTINGS. It consists of the opcode report, the parameter report, and the EOM-indicator, in the following format:

inquiry-code
parameter-report
EOM-indicator

When you use a special inquiry code instead of an opcode, the report has the following format:

special-inquiry-code
special-inquiry-report
EOM-indicator

Inquiry-code or special-inquiry-code. This part of the report consists of two ASCII characters, which are:

- The same as the opcode specified in the corresponding REPORT TERMINAL SETTINGS command.
- The same as the special inquiry code specified in the corresponding REPORT TERMINAL SETTINGS command.
- Two ^{S}P characters if the specified opcode or special inquiry code does not exist.

Parameter-report or special-inquiry-report. This consists of one or more alphanumeric characters, depending on the particular parameter or special inquiry. See "Examples."

EOM-indicator. The EOM-indicator is sent to show that the message is complete.

The only exceptions to this format are the reports for SET SURFACE COLOR MAP and SET DIALOG AREA COLOR MAP. See the discussion following "Examples."

Examples

The following REPORT TERMINAL SETTINGS command requests information for parameters set by the command with the opcode *NR* (SET BAUD RATE):

^{E}clQNR

The following report indicates that the terminal has a transmit and receive baud rate of 1200:

NR! + 0! + 0^{C}R

Similarly, the following command asks how much memory is available:

^{E}clQ?M

The following report indicates that the total amount of available memory is 600 and the largest block available is 300 (a block has 16 bytes).

?M E8 2 < ^{C}R

The SET DIALOG AREA COLOR MAP Report

This report has the following format:

TF
color-info
EOM-indicator

The **TF** is the opcode for SET DIALOG AREA COLOR MAP.

The color-info consists of a series integers indicating the color mixtures for each color index. Each color is identified by a *triplet*. Each triplet reports the hue, lightness, and saturation for a color index. Eight triplets are sent; one for each color index. The last character in the report is the current EOM-indicator, which is sent to show that the report is over.

The SET SURFACE COLOR MAP Report

This report has the following format:

TG
1
color-info
EOM-indicator

The **TG** is the opcode for SET SURFACE COLOR MAP.

The integer **1** indicates that one graphics surface is defined (as is always the case for this terminal).

The color-info consists of a series of integers indicating the color mixtures for each color index. The first integer indicates that 25 integer reports follow — a triplet for the background index, followed by the surface number indicator, followed by seven more triplets. Each *triplet* consists of three integer reports that specify the hue, lightness, and saturation for a color index. One triplet is sent for each of the eight indices. The last character in the report is the current EOM-indicator, which is sent to show that the report is over.

REPORT 4010 STATUS

Causes the terminal to emulate a 4010-style terminal by sending a 4010 status report.

Host Syntax

$^{E}C\ ^{E}Q$

This command also terminates 4010 GIN mode and returns the terminal to Alpha mode.

4010 Status Report

This report is sent in response to REPORT 4010 STATUS. The report has two forms, depending on whether the terminal is in GIN mode when the command is sent.

If the terminal is *not* in GIN mode, the report has the following format:

terminal-status
alpha-cursor-position
EOM-indicator

If the terminal *is* in GIN mode, the report has the following format:

graphics-cursor-position
EOM-indicator

The following paragraphs detail the parts of the report.

Terminal-status. The status of the terminal is encoded into the seven bits of an ASCII character, as follows:

b7	b6	b5	b4	b3	b2	b1
0	1	HCU	NOLI	GRAPH	0	1

Bits 7 and 6 are always set to 0 and 1, respectively; Bits 2 and 1 are also set to 0 and 1, respectively.

The HCU (Bit 5) is set to 0 if a hardcopy unit is attached to the terminal and is ready to accept a copy request; otherwise this bit is set to 1.

Bits 4 and 5 indicate the No Linear Interpolation (NOLI) and Graph mode status as follows:

NOLI	GRAPH	
0	0	The terminal is in Marker mode
0	1	The terminal is in Alpha mode
1	0	The terminal is in Vector mode
1	1	This combination does not occur

For example, if the terminal (1) has a hardcopy unit attached, (2) is ready for a hardcopy command, and (3) is in Vector mode, the bits sent are:

0 1 0 1 0 0 1

The corresponding character on the ASCII chart is the closing parenthesis — **)** — which would be transmitted as the status byte.

Alpha-cursor-position and graphics-cursor-position. The terminal reports both cursor positions in the following format:

Hi-X
Lo-X
Hi-Y
Lo-Y

The Hi-X, Lo-X, Hi-Y, and Lo-Y characters follow the same general coding method described under the heading *XY-Coordinates in Host Syntax* earlier in this section, with two important differences:

- The first two bits of each charater are set to 01.
- Only the ten most significant bits are included in the report.

Figure 5-8 illustrates the binary process of decoding the ASCII characters that report that the cursor is at (1000,2000). To report these coordinates the terminal sends the following characters:

': / 4

The host program must convert each character to equivalent numbers (ASCII Decimal Equivalents). Figure 5-8 shows the binary representation of these numbers. The first two bits are always 01, confirming that these are report characters. To get rid of the 01 prefix, subtract decimal 32 from each number (Hi-X = 39-32 = 7). The remaining values combine to form the two decimal xy-coordinates:

- Multiply the Hi-X value by decimal 128 and save the product: (7 * 128 = 896)
- Multiply the Lo-X value by decimal 4 and save the product: (26 * 4 = 104)
- Add the two products, yielding the x coordinate: (896 + 104 = 1000)
- Repeat the previous three steps with the y-values to find the y-coordinate.

Notice that the last two bits of the cursor position are not reported and can be assumed to be (00).

EOM-indicator. The EOM (End Of Message) character may be sent to show that the 4010 Status Report message is complete. However, it is not an essential part of the message.

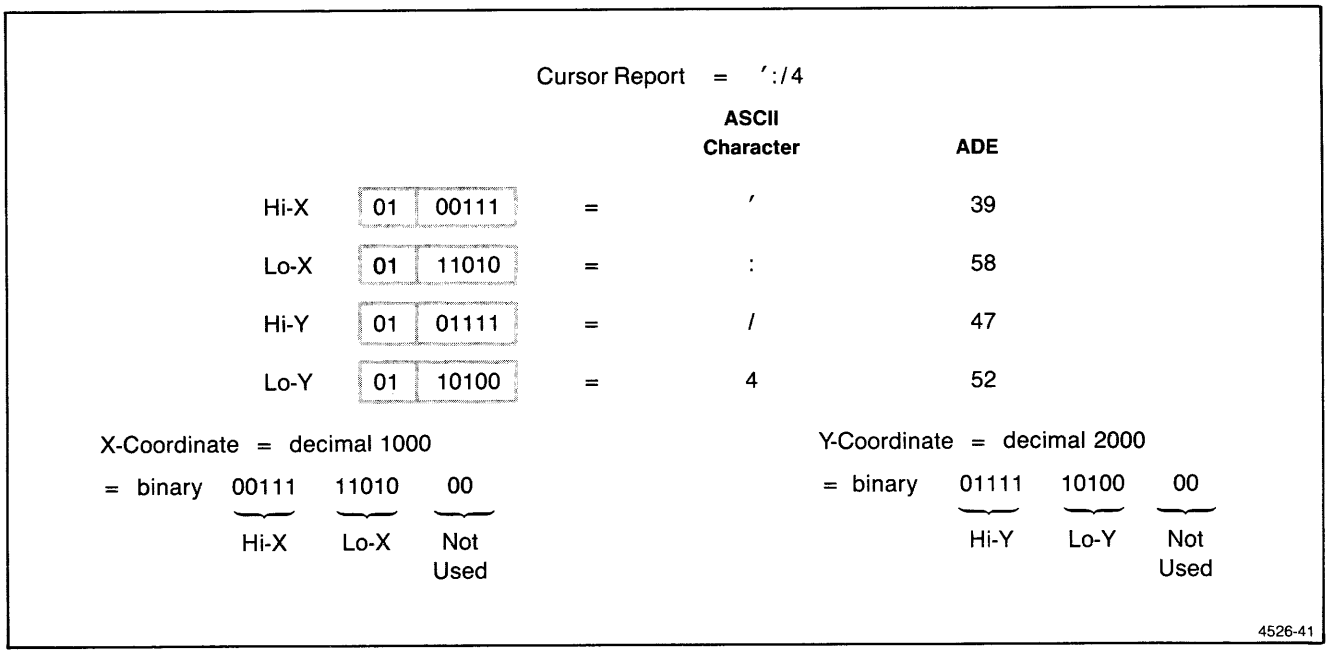


Figure 5-8. The Cursor Position Report.

RUNLENGTH WRITE (Requires Optional Pixel ROMs)

Loads color indices into the pixel viewport.

Host Syntax

^{E}**cRL** runcode-array

Setup Syntax

PXRUNLENGTHWRITE runcode-array

runcode-array: integer-array; specifies a color index and the number of pixels which are to be set to that color index. Can range from 0 to 65535.
Default: Omitted = Empty Array

Each runcode includes two numbers packed together: The runcodes are packed using the form

Runcode = number-of-pixels$*2^n$ + color-index

where n = number-of-bits-per-pixel

The *bits-per-pixel* parameter from the most recent BEGIN PIXEL OPERATIONS command supplies the value for n unless that parameter is 6 or 4; then the value of n is 3.

Starting at the current pixel beam position in the pixel viewport, the terminal sets the specified number of pixels to the specified color index for each runcode in the array. As each pixel is assigned a color index, the pixel beam position moves so that it points at the next pixel to the right on the same line. On encountering the right edge of the pixel viewport, the pixel beam position wraps around to point to the pixel at the left edge of the pixel viewport on the line below; however, if the pixel beam position is on the bottom line, the beam position wraps around to the left edge of the top line of the pixel viewport. When all the pixels for a given runcode have been loaded with the specified color index, the process is repeated for the next runcode in the integer array.

SAVE NONVOLATILE PARAMETERS

Saves nonvolatile parameters that have been altered.

Host Syntax

^{E}**cKU**

Setup Syntax

NVSAVE

This command causes the terminal to write to its nonvolatile memory all parameters that have been changed since the last SAVE NONVOLATILE PARAMETERS. The values of those parameters are restored at power-up and are referred to in this manual as the *Power-Up Defaults*.

Nonvolatile memory means memory that is retained after the terminal is turned off. Parameters that are saved in nonvolatile memory are referred to in this manual as *saved in memory*.

The appendix titled *Parameter Default Values* lists all parameters that are saved in memory.

This command writes only parameters that have changed since the last time this command was issued.

Each byte of nonvolatile memory has a lifetime of about 10,000 writes. If you attempt to write to nonvolatile memory after that, the terminal may display an error message (depending on the SET ERROR THRESHOLD LEVEL setting) that states that there is a "nonvolatile hardware error." When that occurs, parameters are reset to factory default the next time the terminal is powered up.

SELECT CODE

Causes the terminal to recognize Ansi, Tek, or VT52 mode command syntax. Also used to select Edit mode.

Host Syntax

Ec**%!** syntax

Setup Syntax

CODE syntax

syntax: integer (keyword in Setup syntax); **0** selects Tek mode syntax. **1** selects Ansi mode syntax for Ansi mode, **2** selects Ansi mode syntax for Edit mode, **3** selects VT52 mode syntax. In Setup mode, enter **TEK**, **ANSI**, **EDIT**, or **VT52**.
Default: Omitted = 0 (Tek mode)

The syntax of Tek, Ansi, and VT52 mode commands are not compatible. If you are using commands from one mode and want to execute one or more commands from another mode, you must issue the SELECT CODE command with the appropriate parameter.

This command is recognized in all major modes: Ansi, Setup, Tek, and VT52.

SELECT FILL PATTERN

Specifies the fill pattern for subsequent panels.

Host Syntax

Ec**MP** fill-pattern-number

Setup Syntax

FILLPATTERN fill-pattern-number

fill-pattern-number: integer; must be in the range –7 through 174:

- –7 through 0 cause a panel to be filled with a solid color indicated by the negative value of a color index (for example, –3 means fill with color index 3's color)
- 1 through 16 specify predefined patterns
- 17 through 49 are invalid and generate an error
- 50 through 174 specify predefined dither patterns

Defaults: Factory = –1
Power-Up = –1
Omitted = 0

For example, here's how you would select Fill Pattern 16 (encoded as **A0**) in Host syntax.

Ec**MPA0**

SELECT HARDCOPY INTERFACE

Selects the copier type to be used in the HARDCOPY command.

Host Syntax

Ec**QD** copier type

Setup Syntax

HCINTERFACE copier-type

copier-type: integer; 0 selects a monochrome copier, 1 and 2 select the TEKTRONIX 4695 Color Copier. The monochrome selection is for compatibility with black-and-white printers. Selecting a monochrome copy type allows only regular-sized dialog copies.
Defaults: Factory = 2
Power-Up = Saved in memory
Omitted = 0

SET ALPHA CURSOR INDEX

Assigns specified color indices to the alpha cursor.

Host Syntax

```
ESCTD first-index,
      second-index
```

Setup Syntax

```
ACURSOR first-index,
        second-index
```

first-index: integer; specifies the first color for the alpha cursor. Must be in the range 0 to 65535. The values 0 through 7 correspond to a color index (values greater than 7 set *first-index* to 7).
Defaults: Factory = 1
Power-Up = Saved in memory
Omitted = 0

second-index: integer; specifies the second color for the alpha cursor. Must be in the range 0 to 65535. The values 0 to 7 correspond to a color index (values greater than 7 set *second-index* to 7).
Defaults: Factory = 0
Power-Up = Saved in memory
Omitted = 0

The alpha cursor appears on the screen where the next alphanumeric character will be displayed. If *second-index* is a different color than *first-index,* the cursor blinks between the two colors. If the two indices are the same, the cursor does not blink.

The dialog area and the graphics area each have their own set of color indices. When the dialog area is enabled, the alpha cursor indices refer to dialog area indices. When the dialog area is disabled, the alpha cursor indices refer to graphics area indices.

SET ALPHA TEXT FONT

Selects font to be used for alphatext.

Host Syntax

```
ESCc font-code
```

font-code: small integer; the SI character selects the G0 character set, SO selects the G1 character set. Regardless of alphatext font selected, in Setup mode, the terminal uses the ASCII font.
Default: Power-Up = G0 character set

SET BAUD RATES

Sets the terminal's transmit and receive baud rates (in bits per second).

Host Syntax

```
ESCNR transmit-data-rate,
      receive-data-rate
```

Setup Syntax

```
BAUDRATE transmit-data-rate,
         receive-data-rate
```

transmit-data-rate: integer; the baud rate at which the terminal sends data to the host. Valid values are 1 (which means "external clock"), 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, and 38400.
Defaults: Factory = 2400
Power-Up = Saved in memory
Omitted = Error

receive-data-rate: integer; the baud rate at which the terminal expects to receive data from the host. Valid values are the same as for the *transmit-data-rate* with the addition of 0, which means "same as the transmit rate."
Defaults: Factory = 2400
Power-Up = Saved in memory
Omitted = Same value as *transmit-baud-rate* parameter

If you set the baud rate to 38400, the transmit and receive parameters must be the same.

SET BREAK TIME

Sets the duration (in milliseconds) of the break signal the terminal sends when you press the Break key.

Host Syntax

^{E}**cNK** break-time

Setup Syntax

BREAKTIME break-time

breaktime: integer; indicates the length of the break signal in milliseconds. Must be in the range 0 through 65535. A value of 0 causes no break signal to be sent.
Defaults: Factory = 200
Power-Up = Saved in memory
Omitted = 0

SET BYPASS CANCEL CHARACTER

Specifies the character that cancels Bypass mode.

Host Syntax

^{E}**cNU** bypass-cancel-character

Setup Syntax

BYPASSCANCEL bypass-cancel-character

bypass-cancel-character: small integer; indicates the ADE value of the character that cancels Bypass mode. In Setup syntax, you can specify an ADE value or enter the corresponding ASCII character.
Defaults: Factory = 10 (^{L}F)
Power-Up = Saved in memory
Omitted = 0 (^{N}U)

The bypass-cancel character should be the same as the last character that the host sends as it echoes a line of text to the terminal.

In Setup mode, to specify an ADE value in the range 0 through 9, precede the digit with 0. For example, indicate ^{H}T (ADE 9) as 09. The single digits 0 through 9 indicate the digit characters (ADE 48 through ADE 57).

If the bypass-cancel character is set to ^{N}U the terminal does not enter Bypass mode when reports are sent.

See the *Communications* section for an explanation of the terminal's Bypass mode.

SET CHARACTER PATH

Specifies the direction to move after writing each graphtext character.

Host Syntax

^{E}**cMN** direction

Setup Syntax

GTPATH direction

direction: integer (keyword in Setup syntax); indicates which direction graphtext characters are written. Table 5-5 defines each value's effect.
Default: Factory = 0 (right)
Power-Up = 0 (right)
Omitted = 0 (right)

Table 5-5

CHARACTER PATH SETTINGS

Integer	Setup Keyword	Meaning
0	right	Use path equal to rotation angle
1	left	Use path 180° greater than rotation angle
2	up	Use path 90° greater than rotation angle
3	down	Use path 90° less than rotation angle

Figure 5-9 shows how the string "ABC" is displayed using the four different directions for character path. Note that graphtext rotation is 0° for all examples in Figure 5-9.

The keywords (**right, up, left, and down**) refer to character path *after* orientation with the rotation angle specified in SET GRAPHTEXT ROTATION. Table 5-5 gives explicit definitions of the keywords.

Integer Setting	Setup Parameter	Example	Path Relation to Rotation Angle
0	RIGHT	xABC*	Same
1	LEFT	*CBAx	+180°
2	UP	* C B Ax	+90°
3	DOWN	x A B C*	–90°

Note: Graphtext is "ABC" with rotation = 0° in all examples.

x is graphics position before graphtext is displayed.

*** is updated graphics position after graphtext is displayed.**

4526-17B

Figure 5-9. Character Path Settings.

SET COPY SIZE

Sets the copy size.

Host Syntax

EcQA size

Setup Syntax

HCSIZE size

size: integer; 0 selects the default copy size (8½x11), 1 selects a smaller copy size.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

The smaller size produces a faster copy, but only in eight colors: black, white, red, green, blue, cyan, magenta, and yellow.

The smaller copy size also allows you to copy 132 columns on the same line. If, however, the default copy size is chosen and Column mode is set to 132, the extra 52 columns are wrapped to the next line.

If you are using a monochrome copier, selecting the smaller copy size does not produce a smaller copy.

SET DIALOG AREA BUFFER SIZE

Specifies the maximum number lines of text stored in the dialog area buffer.

Host Syntax

EcLB number-of-lines

Setup Syntax

DABUFFER number-of-lines

number-of-lines: integer; indicates the maximum number lines in the dialog buffer. Must be in the range 2 through 32767.
Defaults: Factory = 49
Power-Up = Saved in memory
Omitted = Error

This command takes effect as soon as it is given. The dialog area buffer is changed to the size specified, with the amount of free memory determining the maximum number of lines allowed in the buffer. If *number-of-lines* is greater than the amount of memory available, the terminal modifies the parameter amount to agree with the available memory. For a more detailed description of the dialog area, see *The Graphics Terminal* section of this manual.

If you set the dialog buffer size to less than the size of the dialog area, the size of the dialog area is automatically changed to match the size of the dialog buffer.

The Ansi mode commands TEKOM and TEKSTBM can set a nonscrolling dialog area. When this is the case, if you specify a dialog buffer larger than the screen size (30 lines), the top and bottom margins are reset to Lines 1 and 30. If Origin mode is Absolute and you specify a dialog buffer larger than the screen size, the Origin mode is set to Relative.

SET DIALOG AREA COLOR MAP

Specifies the color assigned to one or more color indices in the dialog area.

Host Syntax

EcTF color-mixture . . .

Setup Syntax

DACMAP color-mixture . . .

color-mixture: integer array; consists of groups of four integers (*quadruples*). Each quadruple specifies a color index (valid range 0 through 7) and the HLS coordinates of the color being assigned to that index: hue (a valid range of –32768 through 32767), lightness (a valid range of 0 through 100), and saturation (a valid range of 0 through 100).
Defaults: Factory = See Table 5-6
Power-Up = Saved in memory
Omitted = No change to color map

Table 5-6

FACTORY DEFAULT COLOR INDICES

Color Index	Color Mixture	Color Coordinates[a]		
		H	L	S
0	Black	0	0	0
1	White	0	100	0
2	Red	120	50	100
3	Green	240	50	100
4	Blue	0	50	100
5	Cyan	300	50	100
6	Magenta	60	50	100
7	Yellow	180	50	100

[a] H = hue, L = lightness, S = saturation

When specifying a color in most commands, you use a color index, which is an integer in the range 0 through 7. SET DIALOG AREA COLOR MAP defines the color for one or more dialog area color indices. The graphics area has its own set of color indices, which is defined with the SET GRAPHICS AREA COLOR MAP.

The color assigned to Index 0 applies only to alphatext characters. For the dialog area background and character backgrounds, Index 0 always means "transparent." The concept of transparency is explained in *The Graphics Terminal* section.

For example, the following command in Host syntax specifies white (HLS coordinates 0, 100, 0) for Index 3 in the dialog area:

^{E}cTF430F40

The first 4 indicates that four values follow.

In Setup syntax, this command is:

DACMAP 3 0 100 0

See *The Graphics Terminal* section and the *Tektronix Color Standard* appendix for more details of how the terminal displays colors.

SET DIALOG AREA INDEX

Specifies the color index for alphatext characters, character-cell background, and dialog area background.

Host Syntax

^{E}cLI character-index,
character-background-index,
dialog-background-index

Setup Syntax

DAINDEX character-index,
character-background-index,
dialog-background-index

character-index: integer; indicates the color index of the characters displayed in the dialog area. Must be in the range 0 through 65535.

Defaults: Factory = 1
Power-Up = Saved in memory
Omitted = 0

character-background-index: integer; indicates the color index used for each character-cell background. Must be in the range 0 through 65535. Index 0 indicates transparency.
Defaults: Factory = 0
Power-Up = Saved in memory
Omitted = 0

dialog-background-index: integer; indicates the color index of the dialog area background. Must be in the range 0 through 65535. This is the color of the dialog area before characters are written on it and after it is erased. Index 0 indicates transparency.
Defaults: Factory = 0
Power-Up = Saved in memory
Omitted = 0

Color Indices 0 through 7 represent colors identified by the dialog area's color indices. These colors are defined by SET DIALOG AREA COLOR MAP. The dialog area has its own indices separate from those used for graphics. When you specify a value greater than 7 for any color index, the terminal uses Index 7.

When the terminal displays a character in the dialog area, it also displays the character cell that encloses the character. The cell is displayed in the character background color. The color of a character cell is set by *character-background-index*.

When Index 0 is used as the *character-index*, it represents an opaque color just like Indices 1 through 7. However, when Index 0 is used as the *character-background-index* or the *dialog-background-index*, it means "make this area transparent." Graphics behind the dialog area show through a transparent area; if the character background is transparent, the dialog appears as if it were written on a piece of glass in front of the graphics.

This command does not affect lines in the dialog area that have already been displayed. Future lines are displayed with the new indices.

SET DIALOG AREA LINES

Specifies the maximum number of lines visible in the dialog area.

Host Syntax

EcLL number-of-lines

Setup Syntax

DALINES number-of-lines

number-of-lines: integer; sets the maximum size in lines for the dialog area. Must be in the range 2 through 30.
Defaults Factory = 30
Power-Up = Saved in memory
Omitted = Error

To be visible, the dialog area must be both enabled and visible. A host program enables the dialog area with ENABLE DIALOG AREA; the operator issues the Setup command DAENABLE. A host program sets visibility with SET DIALOG VISIBILITY; the operator uses DAVISIBILITY.

If you set the number of dialog area lines to be greater than the size of the dialog buffer, the size of the dialog buffer is changed to be the same size as the dialog area.

SET DIALOG AREA VISIBILITY

Specifies whether the dialog area is visible or invisible.

Host Syntax

^{E}**cLV** visibility-mode

Setup Syntax

DAVISIBILITY visibility-mode

visibility-mode: integer (keyword in Setup syntax); 0 or **no** makes the dialog area invisible, 1 or **yes** makes the dialog area visible.
Defaults: Factory = 1 (yes)
Power-Up = Saved in memory
Omitted = 1 (yes)

This command serves the same purpose as the Dialog key — it sets the dialog area to be is visible or invisible.

If ENABLE DIALOG AREA is set to send alphatext to the dialog buffer and the dialog area is enabled but not visible, alphatext is saved in the dialog buffer. Then, when SET DIALOG VISIBILITY is given with *1* or *yes* (making the dialog area visible), the alphatext in the dialog area becomes visible. The terminal will automatically scroll the dialog area, if necessary, to put the cursor in view.

SET DIALOG AREA WRITING MODE

Specifies whether the ^{S}P (Space) and _ (Underscore) characters overstrike or replace other characters in the dialog area.

Host Syntax

^{E}**cLM** writing-mode

Setup Syntax

DAMODE writing-mode

writing-mode: integer (keyword in Setup syntax); 0 or **replace** specifies replace, 1 or **overstrike** specifies overstrike.
Defaults: Factory = 0 (replace)
Power-Up = Saved in memory
Omitted = 0 (replace)

This command affects only how the space and underscore characters are displayed in the dialog area. If the dialog area is set to overstrike, you can underline alphanumeric characters.

Alphatext displayed in the graphics area is not affected by this command. (Alphatext is displayed in the graphics area when the dialog area is disabled.) SET GRAPHICS AREA WRITING MODE sets overstrike capability for graphics area alphatext.

SET DIALOG HARDCOPY ATTRIBUTES

Specifies the number of pages to be copied, the starting page, and how formfeed is interpreted.

Host Syntax

^{E}**cQL** number-of-pages, page-origin, ^{F}F-interpretation

Setup Syntax

HCDAATTRIBUTES number-of-pages, page-origin, ^{F}F-interpretation

number-of-pages: integer; determines the number of pages to be copied from the origin to the bottom of the dialog buffer. Must be in the range of 0 to 32767.
Defaults: Factory = 1
Power-Up = 1
Omitted = 1

page-origin: integer; determines the starting page for the copy. 0 specifies the first visible line in the scroll, 1 the top of the scroll, and 2 the bottom of the scroll.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

$^{F}_{F}$*-interpretation:* integer; determines how the terminal interprets ASCII formfeed in the dialog buffer. If $^{F}_{F}$*-interpretation* is 0, then terminal ignores $^{F}_{F}$ and divides the dialog buffer into pages containing 66 lines (60 lines of text and 3 blank lines at the top and bottom); if $^{F}_{F}$*-interpretation* is 1, the terminal starts a new page every 66 lines *or* when $^{F}_{F}$ appears in the text; if $^{F}_{F}$*-interpretation* is 2, the terminal starts a new page only when $^{F}_{F}$ appears in the text.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

If the origin is set to the bottom of the dialog buffer, the copy begins that many pages up from the bottom of the dialog buffer. If *number-of-pages* is 0, then there is no change from the last setting. If you specify a greater number of pages than the dialog buffer contains, only the contents of the dialog buffer are copied.

For pages that have lines longer than 80 characters, the extra characters are sent without an intervening $^{C}_{R}$. This allows printers with widths greater than 80 columns to be used. The TEKTRONIX 4695 Color Copier automatically generates a $^{C}_{R}{}^{L}_{F}$ and starts printing on the next line for lines containing more than 80 characters. If the SET COPY SIZE command has been executed with a parameter of 1 (to set the small copy size), then the 132 characters are printed on the same line.

When using this command, you need to be aware of how *number-of-pages* and $^{F}_{F}$*-interpretation* interact with each other. The amount of text that goes on each page can vary depending on how the terminal divides the dialog buffer into pages, and that depends on the setting of $^{F}_{F}$*-interpretation*. If you issue this command with its default parameters, it copies 60 lines of text, beginning with the first visible line on the screen. If, however, you set *number-of-pages* to 4, $^{F}_{F}$*-interpretation* to 2, and have four $^{F}_{F}$'s beginning the text on the screen, then the copier turns out four blank sheets of paper. Additionally, if *number-of-pages* is 1, *page-origin* is 1, $^{F}_{F}$*-interpretation* is 2, and there are no $^{F}_{F}$'s in the text, then the copier prints the entire dialog buffer with no page breaks (on the TEKTRONIX 4695 copier, this would be a continuous sheet of paper).

You can stop a hardcopy operation by pressing the Cancel key.

SET ECHO

Specifies whether the terminal echoes characters it transmits to the host.

Host Syntax

^{E}**cKE** echo-mode

Setup Syntax

ECHO echo-mode

echo-mode: integer (keyword in Setup syntax); 0 or **no** causes the terminal not to provide the echo, 1 or **yes** causes the terminal to provide the echo.
Defaults: Factory = 0 (no)
Power-Up = Saved in memory
Omitted = 1 (yes)

SET EDIT CHARS

Specifies the terminal's special editing characters used in Setup mode.

Host Syntax

^{E}**cKZ** character-delete, line-delete, literal

Setup Syntax

EDITCHARS character-delete, line-delete, literal

character-delete: integer (small integer in Setup syntax); specifies the key used in Setup mode to erase the character just left of the cursor.
Defaults: Factory = 127 ($^{D}_{T}$)
Power-Up = Saved in memory
Omitted = Unchanged

line-delete: integer (small integer in Setup syntax); specifies the key used in Setup mode to erase the current line.
Defaults: Factory = 24 (^{C}N)
Power-Up = Saved in memory
Omitted = Unchanged

literal: integer (small integer in Setup syntax); specifies the character used just before an editing character to suspend its control action and print it as text. Only the character immediately following the *literal* character is affected. Use the *literal* character before ^{C}R and the three special editing characters defined in this command.
Defaults: Factory = 126 (~)
Power-Up = Saved in memory
Omitted = Unchanged

In Setup syntax, precede ADEs less than ten with a 0. For example, 08 indicateds ADE 8, the Back Space character. The single digit 8 indicates the character 8, whose ADE is 56. Refer to the ASCII Code Chart appendix.

SET EOF STRING

Specifies the terminal's end-of-file string.

Host Syntax

^{E}**cNE** eof-string

Setup Syntax

EOFSTRING eof-string

eof-string: integer array (delimited string in Setup syntax); specifies end-of-file string sent by the host. Maximum length is ten characters. In Host syntax, each element of the integer array indicates a character's ADE value and must be a small integer (0 through 127). In Setup syntax, the ADE of each character in the string must be in this same range.
Defaults: Factory = Empty array
Power-Up = Saved in memory
Omitted = Empty array

After the host transmits all data in a COPY command, it sends the EOF string to terminate the command.

SET EOL STRING

Specifies the terminal's end-of-line string.

Host Syntax

^{E}**cNT** eol-string

Setup Syntax

EOLSTRING eol-string

eol-string: integer array (delimited string in Setup syntax); specifies the end-of-line string sent in reports to host. Maximum length is two characters. In Host syntax, each element of the integer array indicates a character's ADE value and must be a small integer (0 through 127). In Setup syntax, the ADE of each character in the string must be in this same range.
Defaults: Factory = 13 (^{C}R)
Power-Up = Saved in memory
Omitted = Empty array

This command sets the string that terminates each line of reports the terminal sends to the host. Typically this string is a ^{C}R (ADE 13). This command lets you change the end-of-line string.

SET EOM CHARACTERS

Specifies the character(s) used to terminate messages.

Host Syntax

^{E}**cNC** first-EOM-indicator,
second-EOM-indicator

Setup Syntax

EOMCHARS first-EOM-indicator,
second-EOM-indicator

first-EOM-indicator: small integer; indicates ADE of first end-of-message character. In Setup mode, you can just enter the character.
Defaults: Factory = 13 ($^{C}_{R}$)
Power-Up = Saved in memory
Omitted = 0 ($^{N}_{U}$)

second-EOM-indicator: small integer; indicates ADE of second end-of-message character. In Setup mode, you can just enter the character.
Defaults: Factory = 0 ($^{N}_{U}$)
Power-Up = Saved in memory
Omitted = 0 ($^{N}_{U}$)

SET ERROR THRESHOLD

Specifies the kinds of error messages the terminal displays.

Host Syntax

$^{E}_{c}$**KT** error-threshold-level

Setup Syntax

ERRORLEVEL error-threshold-level

error-threshold-level: integer; indicates minimum error level displayed. Must be in the range of 0 to 4:

0 All messages, warnings, errors, and terminal failure messages are displayed
1 Warnings, errors, and terminal failure are displayed
2 Errors and terminal failure messages are displayed
3 Terminal failure messages are displayed
4 No messages, warnings, errors, or terminal failure messages are displayed

Defaults: Factory = 2
Power-Up = 2
Omitted = 0

This command only determines the level of error messages that are displayed on the screen. The terminal records the eight most recent error messages and transmits them in response to REPORT ERRORS whether or not the error message was displayed on the screen.

SET FLAGGING MODE

Specifies the kind of flagging the terminal uses.

Host Syntax

$^{E}_{c}$**NF** flagging-mode

Setup Syntax

FLAGGING flagging-mode

flagging-mode: integer (keyword in Setup syntax); 0 or **none** means no flagging; 1 or **input** means the terminal uses $^{D}_{1}$ / $^{D}_{3}$ flagging when receiving from the host; 2 or **output** means the terminal uses $^{D}_{1}$ / $^{D}_{3}$ flagging when transmitting to the host; 3 or **in/out** means the terminal uses $^{D}_{1}$ / $^{D}_{3}$ flagging both when transmitting and receiving; 4 or **DTR/CTS** means that the DTR (Data Terminal Ready) and CTS (Clear To Send) signal lines at the RS-232 connector regulate the flow of data.
Defaults: Factory = 0 (none)
Power-Up = Saved in memory
Omitted = 0 (none)

SET GIN CURSOR COLOR

Specifies the color of the terminal's Graphics Input (GIN) cursor.

Host Syntax

$^{E}_{c}$**TC** hue, lightness, saturation

Setup Syntax

GCURSOR hue, lightness, saturation

hue: integer; specifies hue. Must be in the range –32767 to 32767. The terminal automatically converts this integer to an equivalent number of degrees in the range 0 through 359.
Defaults: Factory = 0
Power-Up = Saved in memory
Omitted = 0

lightness: integer; indicates lightness. Must be in the range 0 to 100.
Defaults: Factory = 100
Power-Up = Saved in memory
Omitted = 0

saturation: integer; indicates saturation. Must be in the range 0 to 100.
Defaults: Factory = 0
Power-Up = Saved in memory
Omitted = 0

The color of the GIN cursor is independent of the eight graphics area and eight dialog area colors.

See *The Graphics Terminal* section for a discussion of color. The operators manual discusses use of the color interface.

SET GIN CURSOR SPEED

Determines the speed at which the GIN cursor moves across the screen when the joydisk is pressed.

Host Syntax

E$_C$IJ normal-speed, shifted speed

Setup Syntax

GSPEED normal-speed, shifted speed

normal-speed: integer; determines the speed at which the gin cursor moves across the screen when the joydisk is pressed and the shift key *is not* held down. The range of values is 1 through 10, with 1 being the slowest speed and 10 the highest. The terminal interprets values lower than 1 as 1, and greater than 10 as 10.
Defaults: Factory = 10
Power-Up = 10
Omitted = 1

shifted-speed: integer; determines the speed at which the GIN cursor moves across the screen when the joydisk is pressed while the shift key *is* held down. The range of values is 1 through 10, with 1 being the slowest speed and 10 the highest. The terminal interprets values lower than 1 as 1, and greater than 10 as 10.
Defaults: Factory = 1
Power-Up = 1
Omitted = 1

SET GRAPHICS AREA WRITING MODE

Specifies whether the terminal overwrites or replaces alphatext in the graphics area of the display.

Host Syntax

E$_C$MG writing-mode

Setup Syntax

GAMODE writing-mode

writing-mode: integer (keyword in Setup syntax); 0 or **replace** specifies replace, 1 or **overstrike** specifies overstrike.
Defaults: Factory = 1 (overstrike)
Power-Up = Stored in memory
Omitted = 0 (replace)

This command affects only how alphatext is displayed in the graphics area. If *writing-mode* is set to overstrike, an alphatext character will be displayed on top of an existing character without erasing the existing character. You can use overstrike to underline characters with _ (the Underscore character).

Alphatext displayed in the dialog area is not affected by this command. (Alphatext is displayed in the dialog area when the dialog area is enabled.) SET DIALOG AREA WRITING MODE sets overstrike capability for dialog area alphatext.

SET GRAPHTEXT ROTATION

Sets the rotation angle for graphtext.

Host Syntax

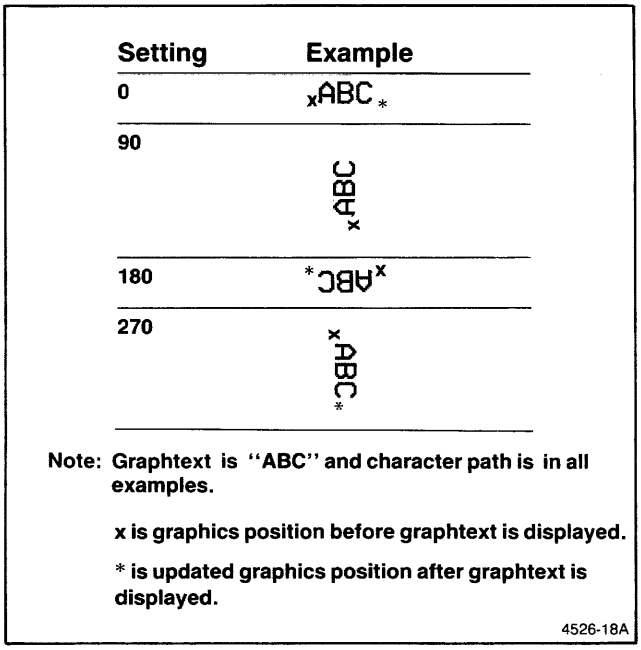
```
ESCMR mantissa,
      power-of-two
```

Setup Syntax

```
GTROTATION mantissa,
           power-of-two
```

mantissa: integer; indicates the graphtext rotation angle. Only the following values are valid: 0, 90, 180, and 270. However, if other angles are given, the terminal will round off to the nearest multiple of 90°.

Defaults: Factory = 0
Power-Up = 0
Omitted = 0



Note: Graphtext is "ABC" and character path is in all examples.

x is graphics position before graphtext is displayed.

*** is updated graphics position after graphtext is displayed.**

4526-18A

Figure 5-10. Graphtext Rotation Examples.

power-of-two: integer; gives the power of two by which the *mantissa* is multiplied. This value is usually 0.

This command sets the angle of rotation for subsequent graphtext. Figure 5-10 shows the four available choices. Positive angles represent counterclockwise rotations, while negative angles represent clockwise rotations.

Character path is relative to the rotation angle. See the description of SET CHARACTER PATH for an explanation of character path and how it relates to rotation angle.

SET GRAPHTEXT SIZE

Sets the size of graphtext.

Host Syntax

```
ESCMC unused,
      height,
      unused
```

Setup Syntax

```
GTSIZE unused,
       height,
       unused
```

unused: integer; provides compatility with other Tektronix terminals. You must include an integer even though it is not used by this terminal.

height: integer; indicates the height of subsequent graphtext in terminal space units (TSU).

Defaults: Factory = 61
Power-Up = 61
Omitted = Error

unused: integer; provides compatility with other Tektronix terminals. You must include an integer even though it is not used by this terminal.

The terminal can display graphtext in several fixed sizes. The smallest available size displays an uppercase letter as five pixels wide and seven pixels high. As shown in the examples in Table 5-7, the other available sizes are integer multiples of the smallest size. The terminal examines the height you specify and displays graphtext in the closest available size.

Changing the size of the window does not change the size of existing graphtext. However, if you first change window size and then display graphtext, the terminal displays the graphtext in the closest available size, based on the height you specify.

Table 5-7

GRAPHTEXT SIZE PARAMETERS EXAMPLES[a]

Specified Height (TSU)	Resulting Size (Pixels)
1 – 87	5 x 7
88 – 148	10 x 14
149 – 209	15 x 21

[a] **These examples assume the default window size is used.**

SET KEY EXECUTE CHARACTER

Specifies the character that directs the flow of macro expansion.

Host Syntax

^{E}**cKY** key-execute-character

Setup Syntax

KEYEXCHAR key-execute-character

key-execute-character: small integer; specifies the character that causes the terminal to direct the expansion of a macro to the host or the terminal.

Defaults: Factory = 16 (^{D}L)
Power-Up = Saved in memory
Omitted = 0 (^{N}U)

If the terminal is sending macros to the host, the key-execute character means "use what follows locally." If the terminal is currently using macros locally, the key-execute character means "send what follows to the host."

Always include the second key-execute character in the macro. If you omit the second key-execute character, subsequent macros are expanded at the terminal, even if they are intended for the host. This would continue until the terminal expands a macro that includes a key-execute character.

SET LINE INDEX

Specifies the color index for all subsequent lines, panel boundaries, and markers.

Host Syntax

^{E}**cML** line-index

Setup Syntax

LINEINDEX line-index

line-index: integer; indicates the color index for lines, panel boundaries, and markers. Must be in the range 0 to 32767 (values greater than 7 set *line-index* to 7).

Defaults: Factory = 1
Power-Up = 1
Omitted = 0

After this command is issued, all new lines, panel boundaries, and markers have the specified color index. This continues until the terminal receives another SET LINE INDEX that changes the color index.

A host program assigns a color to a color index with SET SURFACE COLOR MAP. An operator can use the color interface (see *4105 Graphics Terminal Operators Manual*) or the Setup command CMAP.

SET LINE STYLE

Specifies the line style for subsequent lines and panel boundaries.

Host Syntax

EcMV line-style

Setup Syntax

LINESTYLE line-style

line-style: integer; indicates a predefined line style. Must be in the range 0 through 7.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

Figure 5-11 illustrates the line styles. The terminal will display all new lines and panel boundaries with the line style specified in this command or with 4014 LINE STYLE, whichever was last executed. Existing items are not changed by this command.

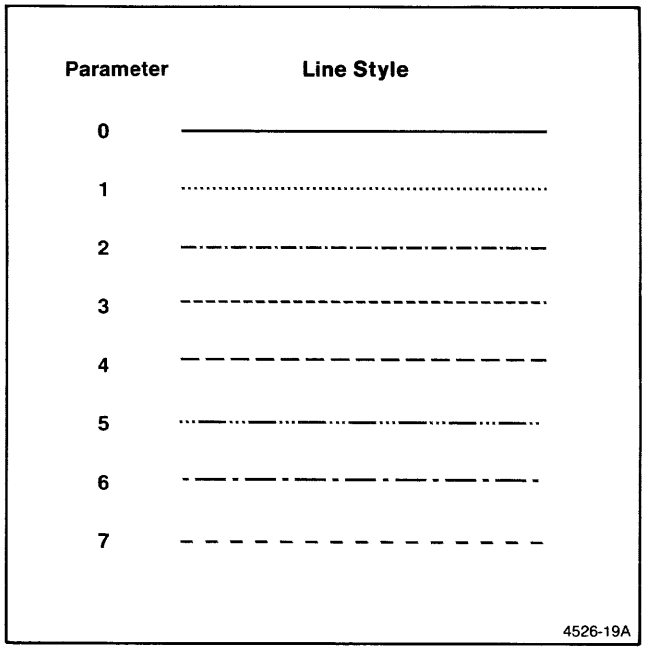


Figure 5-11. Line Styles.

SET MARKER TYPE

Specifies the kind of marker to be drawn.

Host Syntax

EcMM marker-number

Setup Syntax

MARKERTYPE marker-number

marker-number: integer; indicates a predefined marker type. Must be in the range 0 through 10.
Defaults: Factory = 0
Power-Up = 0
Omitted = 0

Figure 5-12 illustrates the marker types.

When you change marker types, markers already displayed remain the same; this command affects only subsequent markers.

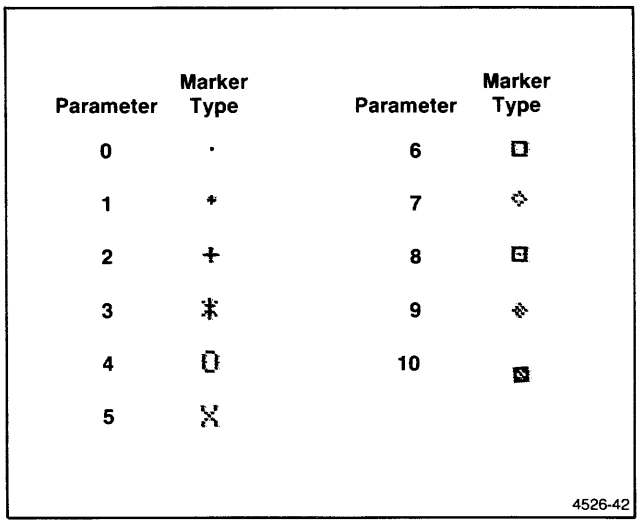


Figure 5-12. Marker Types.

SET PARITY

Specifies the kind of parity the terminal uses when it transmits to the host.

Host Syntax

^{E}cNP parity-mode

Setup Syntax

PARITY parity-mode

parity-mode: integer (keyword in Setup syntax); 0 or **none** sets parity to 0; 1 or **odd** causes the terminal to use odd parity; 2 or **even** causes the terminal to use even parity; 3 or **high** sets the parity bit to 1; 4 or **data** causes the terminal to use the eighth bit in each character as another data bit.
Defaults: Factory = 0 (none)
Power-Up = Saved in memory
Omitted = 0 (none)

The terminal always ignores the parity bit in characters it receives from the host.

SET PIXEL BEAM POSITION (Requires Optional Pixel ROMs)

Sets the position of the pixel beam in the pixel viewport for use by subsequent RASTER WRITE or RUNLENGTH WRITE commands.

Host Syntax

^{E}cRH beam-position

Setup Syntax

PXPOSITION beam-position

beam-position: xy-coordinate; specifies the position where the next RASTER WRITE or RUNLENGTH WRITE command will take effect. Values for x range from 0 to 511; for y, from 0 to 359; The position is relative to the lower-left corner of the pixel viewport.
Defaults: Factory = (0,359)
Power-Up = (0,359)
Omitted = (0,0)

SET PIXEL VIEWPORT (Requires Optional Pixel ROMs)

Sets the pixel viewport position on the pixel writing surface.

Host Syntax

^{E}cRS lower-left, upper-right

Setup Syntax

PXVIEWPORT lower-left, upper-right

lower-left: xy-coordinate; specifies (in raster memory space) one corner of the pixel viewport. Values for x range from 0 to 511; for y, from 0 to 359.
Defaults: Factory = (0,0)
Power-Up = Saved in memory
Omitted = (0,0)

upper-right: xy-coordinate; specifies the opposite corner of the pixel viewport. Values for x range from 0 to 511; for y, from 0 to 359.
Defaults: Factory = (479,359)
Power-Up = Saved in memory
Omitted = (0,0)

The SET PIXEL VIEWPORT command updates the current pixel beam position to the upper-left corner of the pixel viewport. The *lower-left* and *upper-right* coordinates may actually be the coordinates of any two diagonally opposite corners of the pixel viewport. The terminal will sort the x- and y-coordinate values to determine the correct upper and lower corners.

You can set the viewport's x-direction to include the pixels 480 to 511, which are off the screen. A level 0 warning will be generated, however.

SET PROMPT STRING

Specifies the string that initiates the terminal's Prompt mode.

Host Syntax

EcNS prompt-string

Setup Syntax

PROMPTSTRING prompt-string

prompt-string: integer array (delimited character string in Setup syntax); indicates the string that is a prompt sequence when received from the host. Maximum length of ten characters. The integer array consists of up to ten ADE values representing the characters used as the prompt string.

Defaults: Factory = Empty array
Power-Up = Saved in memory
Omitted = Empty array

See the *Communications* section for an explanation of Prompt mode.

SET QUEUE SIZE

Specifies the size (in bytes) of the terminal's input queue.

Host Syntax

EcNQ queue-size

Setup Syntax

QUEUESIZE queue-size

queue-size: integer; indicates the size in bytes of the input queue. Must be in the range 1 through 65535.

Defaults: Factory = 300
Power-Up = Saved in memory
Omitted = Error

When the terminal receives data from the host faster than it can be processed, it stores the data in the input queue. If the input queue fills up and the terminal is not using a flagging mechanism, the terminal ignores incoming data until there is more room in the queue. See the *Communications* section for more details.

Specifying an input queue that is too large can deplete the terminal's memory. This can affect the terminal's ability to display panels.

SET SEGMENT POSITION

Sets the position where the crosshair cursor will be enabled.

Host Syntax

EcSX segment-number, position

Setup Syntax

SGPOSITION segment-number, position

segment-number: integer; must be 0, which identifies the crosshair cursor.

position: xy-coordinate; specifies the position at which the terminal displays the crosshair cursor. Requesting a position outside the current window moves the cursor to the closest location within the current window but does not generate an error.

SET SNOOPY MODE

Specifies whether or not the terminal is in Snoopy mode.

Host Syntax

^{E}**cKS** snoopy-mode

Setup Syntax

SNOOPY snoopy-mode

snoopy-mode: integer (keyword in Setup syntax); the keyword **no** means that the terminal is not in Snoopy mode (there is no corresponding integer parameter for Host syntax); 1 or **yes** means that the terminal is in Snoopy mode.
Defaults: Factory = 0 (no)
Power-Up = 0 (no)
Omitted = 1 (yes)

When the terminal is in Snoopy mode, control characters are displayed instead of executed (except for ^{L}F, which is displayed and causes a new display line). You cannot execute commands from the host when the terminal is in Snoopy mode.

Snoopy mode cannot be terminated from the host; only the operator can do so. To terminate Snoopy mode, press the Cancel key or enter **SNOOPY no** in Setup mode.

If CRLF mode is not set, the terminal assumes that the host will send ^{C}R ^{L}F at the end of each line. When that is true, the terminal starts a new line on the screen after it displays the ^{L}F character. If CRLF mode is set (**yes** or 1), the terminal assumes that the host will send only ^{C}R at the end of a line. In that case the terminal starts a new line after displaying the ^{C}R character. In either case, the ^{L}F character is displayed and causes the terminal to start a new line.

SET STOP BITS

Specifies number of stopbits appended to each character the terminal transmits.

Host Syntax

^{E}**cNB** number-of-stopbits

Setup Syntax

STOPBITS number-of-stopbits

number-of-stopbits: integer; indicates the number of stopbits appended to each character the terminal transmits. Must have value of 1 or 2.
Defaults: Factory = 1
Power-Up = Saved in memory
Omitted = Error

See the *Communications* section for an explanation of stopbits.

SET SURFACE COLOR MAP

Specifies the color assigned to one or more color indices in the graphics area.

Host Syntax

^{E}**cTG** surface-number, color-mixture . . .

Setup Syntax

CMAP surface-number, color-mixture . . .

surface-number: integer; must have a value of 1.

color-mixture: integer array; consists of groups (quadruples) of four integers. Each quadruple specifies a color index (valid range 0 through 7) and the HLS coordinates of the color being assigned to that index: hue (a valid range of –32768 through 32767), lightness (a valid range of 0 to 100), and saturation (a valid range of 0 to 100). As in any integer array, the first element in the array specifies the number of following elements.
Defaults: Factory = See Table 5-8
Power-Up = Saved in memory
Omitted = Error

When you specify a color in most commands, you use a color index, which is an integer in the range 0 through 7. SET SURFACE COLOR MAP defines the color for one or more graphics area color indices. The dialog area has its own set of color indices, which is defined with SET DIALOG AREA COLOR MAP.

For example, the following command in Host syntax specifies white (HLS coordinates 0, 100, 0) for Index 3 in the graphics area:

^{E}cTG430F40

Note that the first 4 indicates that four values follow.

In Setup syntax, this command is:

CMAP 1 3 0 100 0

The Graphics Terminal section and the *Tektronix Color Standard* appendix have details on how the terminal displays colors.

Table 5-8 shows the default colors assigned to color indices when the terminal is manufactured.

Table 5-8

FACTORY DEFAULT COLOR INDICES

Color Index	Color Mixture	Color Coordinates[a]		
		H	L	S
0	Black	0	0	0
1	White	0	100	0
2	Red	120	50	100
3	Green	240	50	100
4	Blue	0	50	100
5	Cyan	300	50	100
6	Magenta	60	50	100
7	Yellow	180	50	100

[a] H = hue, L = lightness, S = saturation

SET TAB STOPS

Sets tab stop(s) at the specified position(s).

Host Syntax

^{E}cKB tab-positions

Setup Syntax

TABS tab-positions

tab-positions: integer array (can be a keyword in Setup syntax); specifies one or more tab stops. The keyword **all** can be used in Setup mode to set tab stops at every column. Specify 0 to clear all tab stops; specify -1 or **all** to set tab stops at every column; specify -2 to reset tab stops to factory default.

Defaults: Factory = 1, 9, 17, 25, 33, 41, 49, 57, 65, 73, 80
Power-Up = Saved in memory
Omitted = 0

SET TEXT INDEX

Specifies the color index for alphatext and graphtext in the graphics area.

Host Syntax

^{E}cMT text-index

Setup Syntax

GTINDEX text-index

text-index: integer; specifies a color index for text in the graphics area. Must have a value in the range of 0 through 7 (values greater than 7 specify Index 7).

Defaults: Factory = 1
Power-Up = 1
Omitted = 0

This command sets the color index for all new text displayed in the graphics region. This includes all graphtext and alphatext displayed when the dialog area is disabled. This command does not change the color index of existing text.

Alphatext displayed in the dialog area is not affected by this command. Use SET DIALOG AREA INDEX to set the color index of dialog area alphatext.

SET TRANSMIT DELAY

Specifies the period of time the terminal delays after transmitting one line of text before it transmits the next.

Host Syntax

```
ESCND transmit-delay
```

Setup Syntax

```
XMTDELAY transmit-delay
```

transmit-delay: integer; indicates the period of delay in milliseconds. Must have a value from 0 to 65535. (The actual period of time may be up to 33 milliseconds longer than that specified because of the resolution of the internal timer.)

Defaults: Factory = 100
Power-Up = Saved in memory
Omitted = 0

SET TRANSMIT RATE LIMIT

Specifies the effective transmit baud rate limit.

Host Syntax

```
ESCNL rate-limit
```

Setup Syntax

```
XMTLIMIT rate-limit
```

rate-limit: integer; specifies the terminal's transmit rate limit. Must be in the range 110 through 65535.

Defaults: Factory = 19200
Power-Up = Saved in memory
Omitted = Error

SET VIEW ATTRIBUTES

Sets the background color for the graphics area.

Host Syntax

```
ESCRA surface,
      background-index,
      unused
```

Setup Syntax

```
VIEWATTRIBUTES surface,
               background-index,
               unused
```

surface: integer; provides compatibility with other Tektronix terminals. Must be 1 or –1.

Defaults: Factory = 1
Power-Up = 1
Omitted = Error

background-index: integer; indicates the color index of the graphics area background. Must be in the range 0 through 7.

Defaults: Factory = 0
Power-Up = 0
Omitted = Error

unused: integer; any valid integer is allowed. Provides compatibility with other Tektronix terminals.

This command sets the background color for the graphics area. The graphics area is this color before anything is displayed in it and immediately after it is cleared. The color is specified as one of the eight graphics area color indices defined by SET SURFACE COLOR MAP. When you change the background color, you do not see the new color until the screen is cleared.

The first and third parameters are included in this command for compatibility with other Tektronix terminals.

SET WINDOW

Establishes the boundaries of the current window in terminal space.

Host Syntax

^{E}cRW lower-left-corner, upper-right-corner

Setup Syntax

WINDOW lower-left-corner, upper-right-corner

lower-left-corner: xy-coordinate; specifies the location of the lower-left corner of the window.

Defaults: Factory = (0,0)
Power-Up = Remembered
Omitted = (0,0)

upper-right-corner: xy-coordinates; specifies the location of the upper-right corner of the window.

Defaults: Factory = (4095,3132)
Power-Up = Remembered
Omitted = (4095,3132)

This command specifies a rectangular area in terminal space, called the *window*. The terminal then displays this window so that it fills the screen. The two corners you specify are placed at the corresponding corners of the screen. The dialog area might obscure part of the window since the dialog area is displayed on top of the graphics area.

If you do not specify two corners of a window with the same aspect ratio as the screen, the display will be linearly distorted. You can prevent a distorted display by specifying a window with 0 height or width. The terminal will then automatically replace the 0 value with a value that produces an undistorted display. To indicate 0 width, specify the same x-coordinate for both corners. To indicate 0 height, specify the same y-coordinate for both corners.

The factory default window produces a display that has a slight linear distortion, but is compatible with other Tektronix terminals. A window of (0,0), (4095,3071) produces an undistorted image and allows the maximum number of addressable locations.

The window (0,0), (479,359) lets you individually address each pixel.

SET 4014 LINE STYLE

Specifies line styles compatible with Tektronix 4010 and 4110 Series terminals.

Host Syntax

^{E}c line-style-code

line-style-code: single character; specifies a line style as shown in Table 5-9.

This command does the same thing as SET LINE STYLE. The line style for lines, markers, and panel boundaries is set by the last SET LINE STYLE or SET 4014 LINE STYLE, whichever occurred most recently.

SET 4014 LINE STYLE sets line styles that are available on other Tektronix terminals. This lets you emulate other terminal's displays.

Codes **h** through **o** indicate line styles that are displayed with a defocused beam on TEKTRONIX 4014,4016, and 4114 Terminals. The 4105 Terminal does not defocus these lines.

Table 5-9

LINE STYLE CODES

Character	Line Style	Emulated Terminals
`	Solid lines	4014/4016
a	Dots	4014/4016
b	Dot-dashes	4014/4016
c	Short-dashes	4014/4016
d	Long dash	4014/4016
e	Dash-dot-dot-dot	4112/4113/4114
f	Long dot-dashes	4112/4113/4114
g	Long dots	4112/4113/4114
h	Solid lines	4014/4016/4114
i	Dots	4014/4016/4114
j	Dot-dashes	4014/4016/4114
k	Short-dashes	4014/4016/4114
l	Long dash	4014/4016/4114
m	Dash-dot-dot-dot	4014/4016/4114
n	Long dot-dashes	4014/4016/4114
o	Long dots	4014/4016/4114

STATUS

Displays the current parameter values for a command or cluster of commands.

Setup Syntax

STATUS name

name: string; the Setup command name or command cluster name for which you want the current parameter values. While in Setup mode, type **STATUS** to display the valid command names and command cluster names.
Default: Omitted = All commands

Alternatively, you can use ^{E}C followed immediately by a two-character opcode for a Tek mode command. The characters of the opcode *must* be typed in capital letters.

4010 HARDCOPY

Generates a hardcopy of the entire screen.

Host Syntax

^{E}C ^{E}B

This command has the same effect as pressing the S Copy key.

If you copy the dialog area, during the time the copy is being made, the dialog area will not show your dialog with the host. To avoid copying the dialog area, press the Dialog key to make the dialog area invisible. After the copy starts, you can press the Dialog key to make the dialog area visible. This lets you use the dialog area while the copy is being made.

Appendix A

ASCII CHART

Table A-1

ASCII (ISO-7-US) CODE CHART

B7 B6 B5 / BITS B4 B3 B2 B1	Ø Ø Ø	Ø Ø 1	Ø 1 Ø	Ø 1 1	1 Ø Ø	1 Ø 1	1 1 Ø	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
Ø Ø Ø Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 Ø Ø 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 Ø 1 Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 Ø Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 Ø 1	CR 13	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1 1 1 Ø	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-21A

Appendix B

ALTERNATE CHARACTER SETS

This appendix lists the character sets available for use on the terminal. These character sets are defined in the ANSI X3.41 and ISO 2022 documents.

Each optional keyboard has a character set that corresponds to its special characters. When you plug a keyboard into the terminal, this character set is automatically selected as both the G0 and G1 character sets. For example, the ASCII character set is selected for the North American keyboard, while the German character set is selected for the Option 4G German keyboard.

You can designate any two of the character sets as the G0 and G1 sets and easily switch between them. The G0 set is selected by sending a ^{S}I (Shift In) character to the terminal, which causes the G0 character set to replace the current character set in the terminal. The G1 set is selected by sending a ^{S}O (Shift Out) character to the terminal, which causes the G1 character set to replace the current G0 character set in the terminal. Refer to the description of the SCS (Select Character Set), ^{S}I (Shift In), and ^{S}O (Shift Out) commands in the *Screen Editor Support* section of this manual for more information.

These alternate character sets are defined in the following tables.

Table B-1

ASCII CHARACTER SET

(Designated by Connecting the Standard North American Keyboard)

B7 B6 B5 / BITS B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0 0 0 0	NU 0	DL 16		0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 0 0 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 0 1 0	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 0 0	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 0 1	CR 13	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1 1 1 0	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	

4526-22

Table B-2
UNITED KINGDOM CHARACTER SET

(Designated by connecting the Option 4A keyboard)

BITS B4 B3 B2 B1	B7 B6 B5 = 0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0 0 0 0	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	ET 4	D4 20	£ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 0 0 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 0 1 0	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 0 0	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 0 1	CR 13	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1 1 1 0	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-23

Table B-3
FRENCH CHARACTER SET

(Designated by connecting the Option 4B keyboard)

BITS B4 B3 B2 B1	B7 B6 B5 = 0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0 0 0 0	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	EX 3	D3 19	£ 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 0 0 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 0 1 0	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	EC 27	+ 43	; 59	K 75	° 91	k 107	é 123
1 1 0 0	FF 12	FS 28	, 44	< 60	L 76	Ç 92	l 108	ù 124
1 1 0 1	CR 13	GS 29	- 45	= 61	M 77	§ 93	m 109	è 125
1 1 1 0	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	¨ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-24

Table B-4

SWEDISH CHARACTER SET

(Designated by connecting the Option 4C keyboard)

B7 B6 B5 / BITS B4 B3 B2 B1	000	001	010	011	100	101	110	111
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0000	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0001	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0010	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0011	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
0100	ET 4	D4 20	¤ 36	4 52	D 68	T 84	d 100	t 116
0101	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0110	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0111	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1000	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1001	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1010	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1011	VT 11	EC 27	+ 43	; 59	K 75	Ä 91	k 107	ä 123
1100	FF 12	FS 28	, 44	< 60	L 76	Ö 92	l 108	ö 124
1101	CR 13	GS 29	- 45	= 61	M 77	Å 93	m 109	å 125
1110	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1111	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-25

Table B-5

DANISH/NORWEGIAN CHARACTER SET

(Designated by connecting the Option 4F keyboard)

B7 B6 B5 / BITS B4 B3 B2 B1	000	001	010	011	100	101	110	111
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0000	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0001	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0010	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0011	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
0100	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0101	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0110	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0111	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1000	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1001	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1010	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1011	VT 11	EC 27	+ 43	; 59	K 75	Æ 91	k 107	æ 123
1100	FF 12	FS 28	, 44	< 60	L 76	Ø 92	l 108	ø 124
1101	CR 13	GS 29	- 45	= 61	M 77	Å 93	m 109	å 125
1110	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1111	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-26

Table B-6

GERMAN CHARACTER SET

(Designated by connecting the Option 4G keyboard)

B7 B6 B5 / BITS B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0 0 0 0	NU 0	DL 16	SP 32	0 48	§ 64	P 80	\ 96	p 112
0 0 0 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 0 0 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 0 1 0	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	EC 27	+ 43	; 59	K 75	Ä 91	k 107	ä 123
1 1 0 0	FF 12	FS 28	, 44	< 60	L 76	Ö 92	l 108	ö 124
1 1 0 1	CR 13	GS 29	- 45	= 61	M 77	Ü 93	m 109	ü 125
1 1 1 0	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	ß 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-27

Table B-7

SUPPLEMENTARY CHARACTER SET

B7 B6 B5 / BITS B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
0 0 0 0	NU 0	DL 16	SP 32	0 48	— 64	Ñ 80	♦ 96	112
0 0 0 1	SH 1	D1 17	Ä 33	1 49	¢ 65	ñ 81	■ 97	113
0 0 1 0	SX 2	D2 18	ä 34	2 50	¦ 66	¿ 82	HT 98	114
0 0 1 1	EX 3	D3 19	Å 35	3 51	† 67	¡ 83	FF 99	115
0 1 0 0	ET 4	D4 20	å 36	4 52	□ 68	α 84	CR 100	116
0 1 0 1	EQ 5	NK 21	Æ 37	5 53	■ 69	σ 85	LF 101	117
0 1 1 0	AK 6	SY 22	æ 38	6 54	● 70	τ 86	o 102	118
0 1 1 1	BL 7	EB 23	à 39	7 55	Δ 71	ρ 87	± 103	119
1 0 0 0	BS 8	CN 24	ç 40	8 56	δ 72	μ 88	NL 104	120
1 0 0 1	HT 9	EM 25	é 41	9 57	λ 73	Σ 89	VT 105	≤ 121
1 0 1 0	LF 10	SB 26	è 42	ù 58	¬ 74	Ω 90	106	≥ 122
1 0 1 1	VT 11	EC 27	Ö 43	β 59	└ 75	∫ 91	107	π 123
1 1 0 0	FF 12	FS 28	ö 44	<Θ 60	┌ 76	92	108	≠ 124
1 1 0 1	CR 13	GS 29	φ 45	¤ 61	┘ 77	÷ 93	109	£ 125
1 1 1 0	SO 14	RS 30	Ü 46	§ 62	¬ 78	≈ 94	110	• 126
1 1 1 1	SI 15	US 31	ü 47	•• 63	∞ 79	┌ 95	111	DT 127

4526-28B

Table B-8
RULINGS CHARACTER SET

B7 B6 B5 / BITS B4 B3 B2 B1	Ø Ø Ø	Ø Ø 1	Ø 1 Ø	Ø 1 1	1 Ø Ø	1 Ø 1	1 1 Ø	1 1 1
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
Ø Ø Ø Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	♦ 96	112
Ø Ø Ø 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	■ 97	113
Ø Ø 1 Ø	SX 2	D2 18	" 34	2 50	B 66	R 82	H_T 98	114
Ø Ø 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	F_F 99	115
Ø 1 Ø Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	C_R 100	116
Ø 1 Ø 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	L_F 101	117
Ø 1 1 Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	102	118
Ø 1 1 1	BL 7	EB 23	' 39	7 55	G 71	W 87	± 103	119
1 Ø Ø Ø	BS 8	CN 24	(40	8 56	H 72	X 88	N_L 104	120
1 Ø Ø 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	V_T 105	≤ 121
1 Ø 1 Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	┘ 106	≥ 122
1 Ø 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	┐ 107	π 123
1 1 Ø Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	└ 108	≠ 124
1 1 Ø 1	CR 13	GS 29	- 45	= 61	M 77	] 93	┌ 109	£ 125
1 1 1 Ø	SO 14	RS 30	. 46	> 62	N 78	^ 94	┼ 110	• 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	95	111	DT 127

4526-29B

Appendix C

ERROR CODES

INTRODUCTION

Each error that the terminal detects has an *error code* and a *severity level.*

When the terminal detects an error, it stores the error in a limited-size queue for later retrieval by the REPORT ERRORS command.

If the errors severity level is greater than or equal to the current error level, the terminal displays a message for the operator. When the terminal is shipped from the factory, its error threshold is set to 2; thus the only errors displayed are those with a severity level of 2 or 3. The error threshold can be changed with the SET ERROR THRESHOLD command (ERROR LEVEL in Setup Syntax). The error threshold is not remembered when the terminal is turned off.

ERROR CODES

The error codes are composed according to the following scheme:

- Each error code consists of four characters.
- In most error codes the first two characters are the *op-code*, an abbreviated code that identifies the command that caused the error.
- The third character is a digit. Digits from 1 to 9 name the parameter with which the error is associated; a 0 indicates that the error is associated with the command as a whole.
- The fourth character of the error code is also a digit:

  0 Indicates an *existence problem.* The object referred to does not exist when it should, or it does exist when it shouldn't.

  1 Indicates an *invalid value.*

  2 Indicates an *out-of-memory problem.*

  3 Indicates a *context problem.* The command is valid, but cannot be executed at this time.

For example, consider the MP10 error code. Here "MP" refers to the Select Fill Pattern command, which has the following syntax:

$^{E}_{C}$MP fill-pattern-number

The "1" refers to the first (and only) parameter of that command, which is the fill pattern number. The "0" indicates an existence problem; the fill pattern does not exist.

SEVERITY LEVELS

There are four error severity levels:

- **Level 0.** Errors of severity level 0 are minor errors. The corresponding message begins with, "Terminal issues message . . . ."
- **Level 1.** Level 1 errors are warnings. The corresponding message begins with, "Terminal issues warning . . . ." Typically these warnings occur when the command is inappropriate or not recognized.
- **Level 2.** Level 2 errors result from invalid commands. The corresponding message begins with, "Terminal detects error . . . ." For instance, a parameter may be outside the specified range.
- **Level 3.** Level 3 errors occur when the command is valid, but the terminal cannot execute the command. The corresponding message begins with, "Terminal system error . . . ." For instance, there may be insufficient memory to hold all the information being entered.

ERROR CODES

The rest of this appendix lists each error code alphabetically with its severity level and an explanation of the cause.

IU11 (Level 2). Invalid speed in parameter 1 (must be in range 0 through 65535).

IU21 (Level 2). Invalid speed in parameter 2 (must be in range 0 through 65535).

JC11 (Level 2). Invalid source device specifier (must be "HC:").

JC12 (Level 3). Out of memory while parsing the first parameter.

JC21 (Level 2). Invalid separator string (must be a <NU> character or the string "TO").

JC22 (Level 3). Out of memory while parsing the second parameter.

JC31 (Level 2). Invalid destination device specifier. Must be the string "HO:".

JC32 (Level 3). Out of memory while parsing the third parameter.

JC39 (Level 2). Hard copy device not ready. Check the hard-copy unit.

KA03 (Level 1). Context problem, cannot disable dialog area when in Ansi or Edit mode.

KA11 (Level 2). Parameter out of range (must be 0 or 1).

KB12 (Level 2). Out of memory while parsing the array.

KD11 (Level 2). Invalid macro number (must be in range −150 to 32767).

KD21 (Level 2). Invalid character-code in the ASCII-decimal-equivalents parameter. (Character codes must be in the range from 0 to 127. The array count must be in the range from 0 to 65535.)

KD22 (Level 3). Insufficient memory to define macro.

KE11 (Level 2). Invalid echo mode (must be 0 or 1).

KF11 (Level 2). Invalid LFCR mode (must be 0 or 1).

KH11 (Level 2). Invalid hardcopy code (must be 0, 1, 2, or 3).

KI11 (Level 2). Invalid ignore-deletes mode (must be 0 or 1).

KL11 (Level 2). Invalid keyboard-lock mode (must be 0 or 1).

KO11 (Level 2). Invalid macro number (must be in the range −150 to 32767).

KO21 (Level 2). Invalid character-code in the ADE parameter (character codes must be in the range 0 to 127; the array count must be in the range 0 to 65535).

KO22 (Level 3). Insufficient memory to define macro.

KR11 (Level 2). Invalid "CR-implies-LF" mode (must be 0 or 1).

KS11 (Level 2). Invalid parameter (must be 0 or 1)

KT11 (Level 2). Invalid error threshold (must be in the range 0 to 4).

KU02 (Level 2). Nonvolatile memory hardware error.

KW11 (Level 2). Invalid mode (must be 0 or 1).

KX01 (Level 2). The maximum nesting depth for the EXPAND-MACRO command has been exceeded. The nesting depth should not exceed five. Greater nesting depths may result in KX01 errors.

KX02 (Level 3). Out of memory while performing EXPAND-MACRO command.

KX11 (Level 2). Invalid macro identifier (must be in the range from −150 to 32767).

KY11 (Level 2). Invalid key-execute-delimiter code (must be in range 0 to 127).

KZ11 (Level 2). Invalid character-delete character (must be in range 0 through 127).

KZ21 (Level 2). Invalid line-delete character (must be in range 0 through 127).

KZ31 (Level 2). Invalid take-literally character (must be in range 0 through 127).

LB03 (Level 0). Dialog parameters modified.

LB11 (Level 2). Invalid number-of-lines parameter (must be in the range from 2 to 32767).

LE02 (Level 3). Out of memory while performing END-PANEL command.

LE03 (Level 1). No panel is currently being defined.

LI11 (Level 2). Invalid character index (must be in the range from 0 to 65535).

LI21 (Level 2). Invalid character background index (must be in the range from 0 to 65535).

LI31 (Level 2). Invalid dialog area wipe index (must be in the range from 0 to 65535).

LL11 (Level 2). Invalid parameter (must be in the range of 2 through 30.

LM11 (Level 2). Invalid writing mode (must be 0 or 1).

LP02 (Level 2). Out of memory while defining a panel.

LP03 (Level 2). Alphatext is not allowed within a PANEL-DEFINITION. When the terminal is in alpha mode, be sure the dialog area is enabled; otherwise, the terminal attempts to read alpha-text as xy parameters.

LP21 (Level 2). Invalid "draw boundary" mode (must be in the range of –32768 to 32767).

LT03 (Level 2). Command is invalid at this time. Graphtext is not allowed within a PANEL-DEFINITION. When this error is detected, the panel being defined is closed, as if an END-PANEL command were received.

LT11 (Level 2). Invalid graphtext string. Invalid array count (must be in range from 0 to 32767), or invalid character in the array (must be in the range from ^{S}P to (~) — decimal equivalents from 32 to 126).

LT12 (Level 3). Parameter 1 memory error (out of memory while parsing the string parameter).

LV11 (Level 2). Invalid dialog area visibility mode (must be 0 or 1).

MC21 (Level 2). Invalid value in parameter 2.

MG11 (Level 2). Invalid parameter (must be 0 or 1).

ML11 (Level 2). Invalid line index (must be in the range from 0 to 32767).

MM11 (level 2). Invalid marker type (must be in the range from 0 to 10)

MN11 (Level 2). Invalid rotation angle (must be either 0, 90, 180, or 270).

MO11 (Level 2). Invalid rotation angle (must be either 0, 90, 180, or 270).

MP10 (Level 2). Specified fill pattern does not exist.

MP11 (Level 2). Invalid fill pattern number (must be in the range from –32768 to 16 or 50 to 174).

MR11 (Level 2). Invalid rotation angle (must be in range –32767 through 32767).

MT11 (Level 2). Invalid text index (must be in the range from 0 to 65535).

MV11 (Level 2). Invalid line style (must be in the range from 0 to 7).

NB11 (Level 2). Invalid number of stop bits (must be 1 or 2).

NC11 (Level 2). Invalid value in parameter 1. Can be any ASCII character.

NC21 (Level 2). Invalid value in parameter 2. Can be any ASCII character.

ND11 (Level 2). Invalid transmit delay time (must be in the range from 0 to 65535 milliseconds).

NE11 (Level 2). Invalid EOF-string (must contain from 0 to 10 characters, with each character represented by an int in the range from 0 to 127).

NE12 (Level 3). Parameter 1 memory error (out of memory while parsing the end-of-file-string parameter).

NF11 (Level 2). Invalid flagging mode (must be in the range from 0 to 4).

NK11 (Level 2). Invalid parameter (must be in the range from 0 to 65535).

NL11 (Level 2). Invalid transmit rate limit (must be in the range 110 to 38400).

NM11 (Level 2). Invalid prompt mode parameter (must be 0, 1,or 2).

NP11 (Level 2). Invalid parity code (must be in the range from 0 to 4).

NQ02 (Level 3). Out of memory while performing SET-QUEUE-SIZE command.

NQ11 (Level 2). Invalid queue size (must be in the range from 1 to 65535).

NR11 (Level 2). Invalid transmit (terminal-to-host) data rate (must be either 1, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, or 38400).

NR21 (Level 2). Invalid receive (host-to-terminal) data rate (must be either 0, 1, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, or 38400).

NS11 (Level 2). Invalid prompt-string parameter (must be an array of up to 10 ASCII characters).

NS12 (Level 3). Parameter 1 memory error (out of memory while parsing the prompt-string parameter).

NT11 (Level 2). Invalid array count in EOL-string. The array must hold from 0 to 2 int parameters. Each int in the array must be in the range from 0 to 127.

NT12 (Level 3). Parameter 1 memory error (out of memory while parsing the EOL-string parameter).

NU11 (Level 2). Invalid numeric equivalent of bypass-cancel character (must be in the range from 0 to 255).

QA11 (Level 2). Invalid copy size (must be 0 or 1).

QD11 (Level 2). Invalid copier value.

QL11 (Level 2). Invalid page number.

QL21 (Level 2). Invalid page origin.

QL31 (Level 2). Invalid FF value.

RA11 (Level 2). Invalid surface number (must be –1 or 1).

RA21 (Level 2). Invalid background index (must be in range 0 through 65535).

RA31 (Level 2). Invalid value (must be in range 0 through 65535).

RP11 (Level 2). Invalid number of pixels (must be in the range 0 through 65535).

RP21 (Level 2). There are too many or too few pixels in the code array.

RP22 (Level 3). Parameter 2 memory error (out of memory while parsing the *character-array* parameter.

RR11 (Level 0). Invalid lower-left coordinates (x value is between range of 480 and 511).

RR11 (Level 2). Invalid lower-left coordinates (x must be in the range 0 through 511, and y from 0 through 359).

RR21 (Level 0). Invalid upper-right coordinates (x value is between range of 480 and 511).

RR21 (Level 2). Invalid upper-right coordinates (x must be in the range 0 through 511, and y from 0 through 359).

RR31 (Level 2). Invalid fill-index (must be in range 0 through 65535).

RS11 (Level 0). Invalid lower-left coordinate (x value is between range of 480 and 511).

RS11 (Level 2). Invalid lower-left coordinate (x must be in the range 0 through 511, and y from 0 through 359).

RS21 (Level 0). Invalid upper-right coordinate (x value is between range of 480 and 511).

RS21 (Level 2). Invalid upper-right coordinate (x must be in the range 0 through 511, and y from 0 through 359).

RU10 (Level 2). Surface does not exist.

RU11 (Level 2). Invalid surface number (must be in the range –1 through 4).

RU21 (Level 2). Invalid ALU mode (must be 0, 7, 11, 12, 15).

RU31 (Level 2). Invalid bits-per-pixel (must be 0, 1, 2, 3, 4, or 6).

RX11 (Level 2). Invalid destination surface (must be in the range –1 to 1).

RX21 (Level 0). Invalid destination lower-left corner (x must be in range 0 through 479 and y in range 0 through 359).

RX21 (Level 2). Invalid destination lower-left corner (x must be in range 0 through 511 and y in range 0 through 359).

RX31 (Level 0). Invalid first source-corner (x must be in range 0 through 479 and y in range 0 through 359).

RX31 (Level 2). Invalid first source-corner (x must be in range 0 through 511 and y in range 0 through 359).

RX41 (Level 0). Invalid second source-corner (x must be in range 0 through 479 and y in range 0 through 359).

RX41 (Level 2). Invalid second source-corner(x must be in range 0 through 511 and y in range 0 through 359).

SX10 (Level 2). Segment does not exist (must be 0).

TC11 (Level 2). Invalid hue value (must be between –32768 and 32767).

TC21 (Level 2). Invalid lightness value (must be between 0 and 100).

TC31 (Level 2). Invalid saturation value (must be between 0 and 100).

TD11 (Level 2). Invalid index value (must be between 0 and 65535).

TD21 (Level 2). Invalid index value (must be between 0 and 65535).

TF11 (Level 2). Invalid color-mixtures array (must be an array of quadruples consisting of a color index and the three HLS color coordinates).

TF12 (Level 3). Parameter 2 memory error (out of memory while parsing the color-mixtures array).

TF21 (Level 2). Invalid color-mixtures array (must be an array of quadruples consisting of a color index and the three HLS color coordinates).

TG11 (Level 2). Invalid surface number (must be 1 or –1).

TG21 (Level 2). Invalid color-mixtures array (must be an array of quadruples consisting of a color index and the three HLS color coordinates).

TG22 (Level 3). Parameter 2 memory error (out of memory while parsing the color-mixtures array).

[A11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[B11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[C11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[D11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[f11 (Level 2). Invalid value for parameter 1 (must be in the range of 0 to 32767).

[f21 (Level 2). Invalid value for parameter 2 (must be in the range of 0 to 32767).

[g11 (Level 2). Invalid value (must be 0, 2 or 3).

[h10 to **[h90 (Level 2).** Invalid mode value (must be either 2, 4, 12, 20, <1, ?1, ?3, ?5, ?6, ?7, or ?8).

[h11 to **[h91 (Level 2).** Invalid parameter syntax.

[H11 (Level 2). Invalid value for parameter 1 (must be in the range of 0 to 32767).

[H21 (Level 2). Invalid value for parameter 2 (must be in the range of 0 to 32767).

[I11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[J11 (Level 2). Invalid value (must be 0, 1 or 2).

[K11 (Level 2). Invalid value (must be 0, 1 or 2).

[l01 (Level 2). Invalid mode value (must be either 4, 20, <1, ?1, ?3, ?6, ?7, or ?8).

[l10 to **[l90 (Level 0).** Unrecognized mode value (treated as a no-op).

[l11 to **[l91 (Level 2).** Invalid parameter syntax.

[L11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[m11 (Level 2). Invalid rendition value (must be either 0, 1, 4, 5, 7, <, >, or =).

[M11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[n11 (Level 2). Invalid value (must be 0, 3, 5, or 6).

[P11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[r11 (Level 2). Invalid value for parameter 1 (must be in the range of 0 to 32767).

[r21 (Level 2). Invalid value for parameter 2 (must be in the range of 0 to 32767).

[S11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[T11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[X11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[Z11 (Level 2). Invalid value (must be in the range of 0 to 32767).

[@11 (Level 2). Invalid value (must be in the range of 0 to 32767).

%!11 (Level 2). Invalid parameter (must be 0, 1, 2, or 3).

#!11 (Level 2). Invalid parameter (must be 0).

$_{P}$A11 (Level 2). Invalid value (must be 0 to 32767).

$_{P}$@11 (Level 2). Invalid value (must be 0 to 32767).

Appendix D

PARAMETER DEFAULT VALUES

This appendix contains tables listing the terminal commands that take parameters. The commands are listed alphabetically in three tables:

- 4100-Style parameters
- ANSI-Style parameters
- Setup mode parameters

Commands that are not listed in the tables do not have parameters. Parameter default values are shown under the appropriate heading.

Table D-1

4100-STYLE PARAMETER DEFAULTS

Descriptive Name	Setup Name	Host Op-Code	Parameters	Defaults		Saved In Nonvolatile Memory
				Factory	Omitted	
BEGIN PANEL BOUNDARY	BEGINPANEL	LP	first-point		0,0	
			draw-boundary		0	
COPY	COPY	JC	source		error JC11	
			separator		error JC21	
			destination		error JC31	
CRLF	CRLF	KR	crlf-mode	0	1	YES
DEFINE MACRO	DEFINE	KD	macro-number		0	
			macro-contents		empty array	
DEFINE NONVOLATILE MACRO	NVDEFINE	KO	macro-number		0	YES
			macro-contents		empty array	YES
DRAW	DRAW	LG	position		0,0	
DRAW MARKER	MARKER	LH	marker-position		0,0	
ENABLE DIALOG AREA	DAENABLE	KA	enable-mode	1	1	YES
ENABLE KEY EXPANSION	KEYEXPAND	KW	switch	1	1	
EXPAND MACRO	EXPAND	KX	macro-number		0	
GRAPHIC TEXT	GTEXT	LT	text		empty string	
HARDCOPY		KH	hard-copy-code		0	
IGNORE DELETES	IGNOREDEL	KI	ignore-deletes-mode	0	1	YES
LFCR	LFCR	KF	lfcr-mode	0	1	YES
LOCK KEYBOARD		KL	lock-mode	0	1	
MOVE	MOVE	LF	position		0,0	
PROMPT MODE	PROMPTMODE	NM	prompt-mode	0	1	

Table D-1 (cont)

4100-STYLE PARAMETER DEFAULTS

Descriptive Name	Setup Name	Host Op-Code	Parameters	Defaults		Saved In Nonvolatile Memory
				Factory	Omitted	
REPORT TERMINAL SETTINGS		IQ	inquiry-code		0	
SELECT CODE	CODE	%!	syntax	0	0	YES
SELECT FILL PATTERN	FILLPATTERN	MP	fill-pattern-number −1	0		
SELECT HARDCOPY INTERFACE		QD	copier-type	0	0	YES
SELECT ALPHA CURSOR INDEX	ACURSOR	TD	first-index	1	0	YES
			second-index	0	0	YES
SET BAUD RATES	BAUDRATE	NR	transmit-data-rate	2400	error NR11	YES
			receive-data-rate	2400	error NR21	YES
SET BREAK TIME	BREAKTIME	NK	break-time	200	0	YES
SET BYPASS CANCEL CHARACTER	BYPASSCANCEL	NU	bypass-cancel-character	$^{L}_{F}$	$^{N}_{U}$	YES
SET CHARACTER PATH	GTPATH	MN	direction	0	0	
SET DIALOG AREA BUFFER	DABUFFER	LB	number-of-lines	49	error LB11	YES
SET DIALOG AREA COLOR MAP	DACMAP	TF	color-mixture	0 = black 1 = white 2 = red 3 = green 4 = blue 5 = cyan 6 = magenta 7 = yellow	empty array	YES
SET DIALOG AREA INDEX	DAINDEX	LI	character-index	1	0	YES
			char-background-index	0	0	YES
			dialog-background-index	0	0	YES
SET DIALOG AREA LINES	DALINES	LL	number-of-lines	30	error LL11	YES
SET DIALOG AREA VISIBILITY	DAVISIBILITY	LV	visibility-mode	1	1	YES
SET DIALOG AREA WRITING MODE	DAMODE	LM	writing-mode	0	0	YES
SET ECHO	ECHO	KE	echo-mode	0	1	YES
SET EDIT CHARS	EDITCHARS	KZ	character-delete	$^{D}_{L}$	unchanged	YES
			line-delete	$^{C}_{N}$	unchanged	YES
			take-literally	~	unchanged	YES
SET EOF STRING	EOFSTRING	NE	end-of-file-string	empty array	empty array	YES
SET EOL STRING	EOLSTRING	NT	end-of-line-string	$^{C}_{R}$	empty array	YES
SET EOM CHARACTERS	EOMCHARS	NC	first-eom-character	$^{C}_{R}$	$^{N}_{U}$	YES
			second-eom-character	$^{N}_{U}$	$^{N}_{U}$	YES
SET ERROR THRESHOLD	ERRORLEVEL	KT	error-threshold-level	2	0	
SET FLAGGING MODE	FLAGGING	NF	flagging-mode	0	0	YES

Table D-1 (cont)

4100-STYLE PARAMETER DEFAULTS

Descriptive Name	Setup Name	Host Op-Code	Parameters	Defaults		Saved In Nonvolatile Memory
				Factory	Omitted	
SET GIN CURSOR COLOR	GCURSOR	TC	hue	0	0	YES
			lightness	100	0	YES
			saturation	0	0	YES
SET GRAPHICS AREA WRITING MODE	GAMODE	MG	writing-mode	1	0	
SET GRAPHTEXT ROTATION		MR	angle	0,0	0,0	
SET GRAPHTEXT SIZE	GTSIZE	MC	height	61		
SET KEY EXECUTE CHARACTER	KEYEXCHAR	KY	key-execute-character	$^{D}_{L}$	$^{N}_{U}$	YES
SET LINE INDEX	LINEINDEX	ML	line-index	1	0	
SET LINE STYLE	LINESTYLE	MV	line-style	0	0	
SET MARKER TYPE	MARKERTYPE	MM	marker-number	0	0	
SET PARITY	PARITY	NP	parity-mode	0	0	YES
SET PROMPT STRING	PROMPTSTRING	NS	prompt-string	empty array	empty array	YES
SET QUEUE SIZE	QUEUESIZE	NQ	queue-size	300	error NQ11	YES
SET SNOOPY MODE	SNOOPY	KS	snoopy-mode	0	1	
SET STOP BITS	STOPBITS	NB	number-of-stopbits	1	error NB11	YES
SET SURFACE COLOR MAP	CMAP	TG	surface-number	1	error TG11	YES
			color-mixture	0 = black 1 = white 2 = red 3 = green 4 = blue 5 = cyan 6 = magenta 7 = yellow	empty array	YES
SET TAB STOPS	TABS	KB	tab-stops	1, 9, 17, 19, 33, 41, 49, 57, 65, 73		YES
SET TEXT INDEX	GTINDEX	MT	text-index	1	0	
SET TRANSMIT DELAY	XMTDELAY	ND	transmit-delay	100	0	YES
SET TRANSMIT RATE LIMIT	XMTLIMIT	NL	rate-limit	19200	error NL11	YES
SET VIEW ATTRIBUTES	VIEWATTRIBUTES	RA	surface	1	error RA11	
			background-index	0	error RA21	
SET WINDOW	WINDOW	RW	lower-left-corner	0,0	0,0	YES
			upper-right-corner	4095,3132	4095,3132	YES
SET 4010 LINE STYLE		code	line-style-code	solid		

Table D-2

ANSI-STYLE COMMAND PARAMETER

Descriptive Name	Setup Name	Host Op-Code	Parameters	Defaults		Saved In Nonvolatile Memory
				Factory	Omitted	
CBT		Z	number-of-tab-stops		1	
CHT		I	number-of-tab-stops		1	
CUB		D	number-of-columns		1	
CUD		B	number-of-lines		1	
CUF		C	number-of-columns		1	
CUP		H	row-number		1	
			column-number		1	
CUU		A	number-of-lines		1	
DCH		P	number-of-characters		1	
DSR		n			error [n11	
DL		M	number-of-lines		1	
ECH		X	number-of-characters		1	
ED		J	erase-extent		0	
EL		K	erase-extent		0	
HVP		f	row-number		1	
			column-number		1	
ICH		@	number-of-characters		1	
IL		L	number-of-lines		1	
RM SCS	SELECTCHARSET	I	modes character-set			
SD		T	number-of-lines		1	
SELECT CODE	CODE	%!	syntax	0	0	YES
SGR	TEXTRENDITION	m	graphic-rendition	0	0	
SM		h	modes			
SU		S	number-of-lines		1	
TBC		g	tab-clear-extent		0	
TEKSTBM	EDITMARGINS	r	top-margin	1	1	
			bottom-margin	30	1	

Table D-3

SETUP MODE ONLY COMMAND PARAMETER DEFAULTS

Setup Name	Parameters	Defaults		Saved In Nonvolatile Memory
		Factory	Omitted	
AUTOREPEAT	autorepeat-mode	YES		YES
AUTOWRAP	autowrap-mode	YES		YES
CURSORKEYMODE	pad-mode	application		
DAMODE	overstrike-replace-mode	REPLACE		YES
HELP	command-name		all commands	
INSERTREPLACE	insert-replace-mode	REPLACE		
KEYPADMODE	key-mode	YES		
LOCAL	local-mode	NO	YES	
MACROSTATUS	macro-number			
ORIGINMODE	origin-mode	RELATIVE		YES

Appendix E

GLOSSARY

4100 Commands

The set of terminal commands not included in the ANSI command set. These commands include terminal control and graphics commands.

Absolute Origin Mode

Refer to the Set Mode and Reset Mode commands in the *Screen Editor Support* section of this manual for more information.

Alpha Mode

In this mode, the terminal interprets any received ASCII characters as text to be displayed. If the dialog area is enabled, these characters are displayed in the dialog area; otherwise, the characters are displayed in the graphics area.

ANSI

The American National Standards Institute. This is the U.S.A member body of the International Standards Organization.

ANSI Command

A set of standardized terminal control functions that allow the editing of computer files on the terminal screen. This set of commands is not included in the 4100 command set. These commands are specified in ANSI Standard X3.64 and in ISO International Standard 6429.

ANSI Mode

The mode of operation in which the terminal can accept and execute the ANSI command set. In ANSI mode the terminal accepts the ANSI command syntax; in TEK mode, it accepts the 4100 command syntax.

ASCII Code

American Standard Code for Information Interchange. Each transmitted character has a unique 7-bit plus parity bit code established by ANSI (in ANSI Standard X3.4).

Baud

A unit of signalling speed corresponding to one signal event per second. This is usually the rate, in bits per second, at which the terminal communicates with a host.

Break

The effect of sending a continuous stream of binary 0's for a specified time — usually interpreted by the host as an interrupt.

Character Sets — Alternate

The optional character sets that are available on the terminal. These sets include foreign language, rulings and special character sets. The SCS (Select Character Set) command determines which of the optional sets are currently available.

Color Index

An integer in the range 0 through 7 that refers to a color. The color assigned to a color index can be defined by either the operator or the host program. The dialog area and the graphics area each have their own set of color indices.

Color Interface

An interactive program that lets the operator set the colors used in displays. The operator can change colors by choosing a color index or by choosing an item in the display.

Commands

The terminal recognizes 4 different types of commands: ANSI, 4010, 4100, and Setup mode commands. Each type has its own syntax differences. Refer to the *Screen Editor Support* and *4100-Style Commands, Reports, and Parameter Types* sections of this manual for more information.

Current Graphics Position

Location on the screen where the last graphics operation ended. Same location is starting point for next graphics operation.

Cursor

The movable screen marker that determines the screen location where commands will take effect. The Alpha cursor is an underscore that you can move with the cursor movement keys. The Graphics cursor is a crosshair cursor that you can move with the joydisk.

Default

The preset value that a parameter takes when a value is not supplied.

Deleting

The process of causing characters or lines and the space that they occupy to be removed from the display.

Delimiter

A character that marks the beginning and end of a string of characters. The delimiter must not be contained in the delimited string.

Dialog Area

The portion of the terminal's display in which text editing and host computer dialog take place.

Dialog Buffer

The 0 to 49 lines of text stored in the terminal's memory of which up to 30 lines are visible at any one time.

Echo

When a key is pressed on the terminal, the character displayed on the screen is the *echo*. This echo can be supplied either by the terminal or by the host.

Edit Margins

The margins set in the dialog area that define a scrolling region in which text editing is performed.

Edit Mode

A specific implementation of ANSI mode in which the terminal is configured to operate as a VT-100 terminal.

Erasing

The process of causing characters or lines to be removed from the display while filling the space that they occupy with the erase index.

Escape Sequence

A series of characters preceded by an ^{E}C character — used to enter commands from the keyboard and from the host computer.

Fixed Region

A user definable region of the dialog buffer that cannot be scrolled — typically used for status line information in screen editor programs.

Graphic Rendition

One of several styles in which text can be displayed in the dialog area. Choices include: Normal (default), Bold, Underscore, Slow Blink, and Reverse Video.

Graphics

Information displayed pictorially as a series of lines and panels.

Graphics Area

The portion of the terminal's display that is used to display graphics information.

Graphics Cursor

The displayed crosshairs that the operator uses to select a location in the graphics area.

Handshaking Protocol

A formalized sequence of operations between the terminal and the host in which each take turns transmitting data.

Home Position

The position of the cursor after a Page command has been executed when the dialog area is disabled; terminal coordinates 0,3071.

Input Queue

The portion of the terminal's memory in which incoming data is held until the terminal can process it.

Inserting

The process of adding new characters to the display by moving characters to the right of the cursor toward the right side of the screen (leaving a space) and displaying the new characters in that space.

Joydisk

Controls the position of the graphics cursor on the screen as well as scrolling text in the dialog area.

Nonvolatile Memory

The portion of terminal memory that keeps information even when the terminal is turned off. Many terminal parameters can be saved in this portion of memory.

Numeric Keypad

The group of keys on the right side of the keyboard labled for numeric input; can also be programmed for use with application programs.

Output Queue

The portion of the terminal's memory in which transmitted data is held until the host computer can process it.

Paging

The process of writing one screenful of text and then erasing the screen before more text is displayed.

Parameter Types

A class of parameters that has a specific format. Some examples of parameter types are integer array, delimited string, and xy-coordinate.

Parity

A simple form of error checking where a single bit is added to to a stream of bits so that the total number of bits is odd (odd parity) or even (even parity).

Peripheral Device

Auxiliary equipment of a computer system, usually performing input-output functions, such as disk drives or printers.

Pixel

One picture element in the graphics area. The terminal's graphics area is 480 pixels wide by 360 pixels high.

Saved Parameters

Terminal settings that remain in the terminal's memory even when the terminal is turned off.

Setup Command

A class of terminal command that is entered while in Setup mode.

Screen Editor

A text entry/editing program that displays a portion of a computer file on the screen of the terminal allowing you to change the text while viewing the changes. (As opposed to line editors which require line by line editing and redisplay of changed lines.)

Screen

The color picture tube on which the terminal's output is displayed.

Scrolling

The up or down movement of lines of text on the screen; as available lines of text appear at one end of the screen, other lines move off the screen at the opposite end.

Scrolling Region

A user definable region of the dialog area that can be scrolled — typically used for displaying the portion of the file being edited by a screen editor program.

Self Test

A program stored in the terminal's firmware that tests terminal functions and displays any errors it encounters. There are three types of self test: power-up, extended, and adjustment. See the *4105 Operators Manual* for more information.

Shift In

A control character used to change from the G1 character set to the G0 character set.

Shift Out

A control character used to change from the G0 character set to the G1 character set.

Software

Computer programs.

Stop Bit

A bit, always a 1, occurring at the end of a character. Used to hold the communication line in "mark" state until the receiving device is ready for the next character.

Surface

A named group of display items. The items assigned to a surface can be manipulated as a group and displayed simultaneously with other surfaces. The 4105 does not use surfaces, but some commands permit a surface specification for compatibility with other Tektronix terminals.

TEK Mode

The mode of operation in which the terminal can accept and execute the command set used in TEKTRONIX 4100 Series Terminals. In TEK mode the terminal accepts the 4100 command syntax; in ANSI mode, it accepts the ANSI command syntax.

Terminal Initialization File

The file required by many host computers to configure the terminal's operating characteristics.

Terminal Report

A report sent by the terminal to the host computer — usually contains some form of status information.

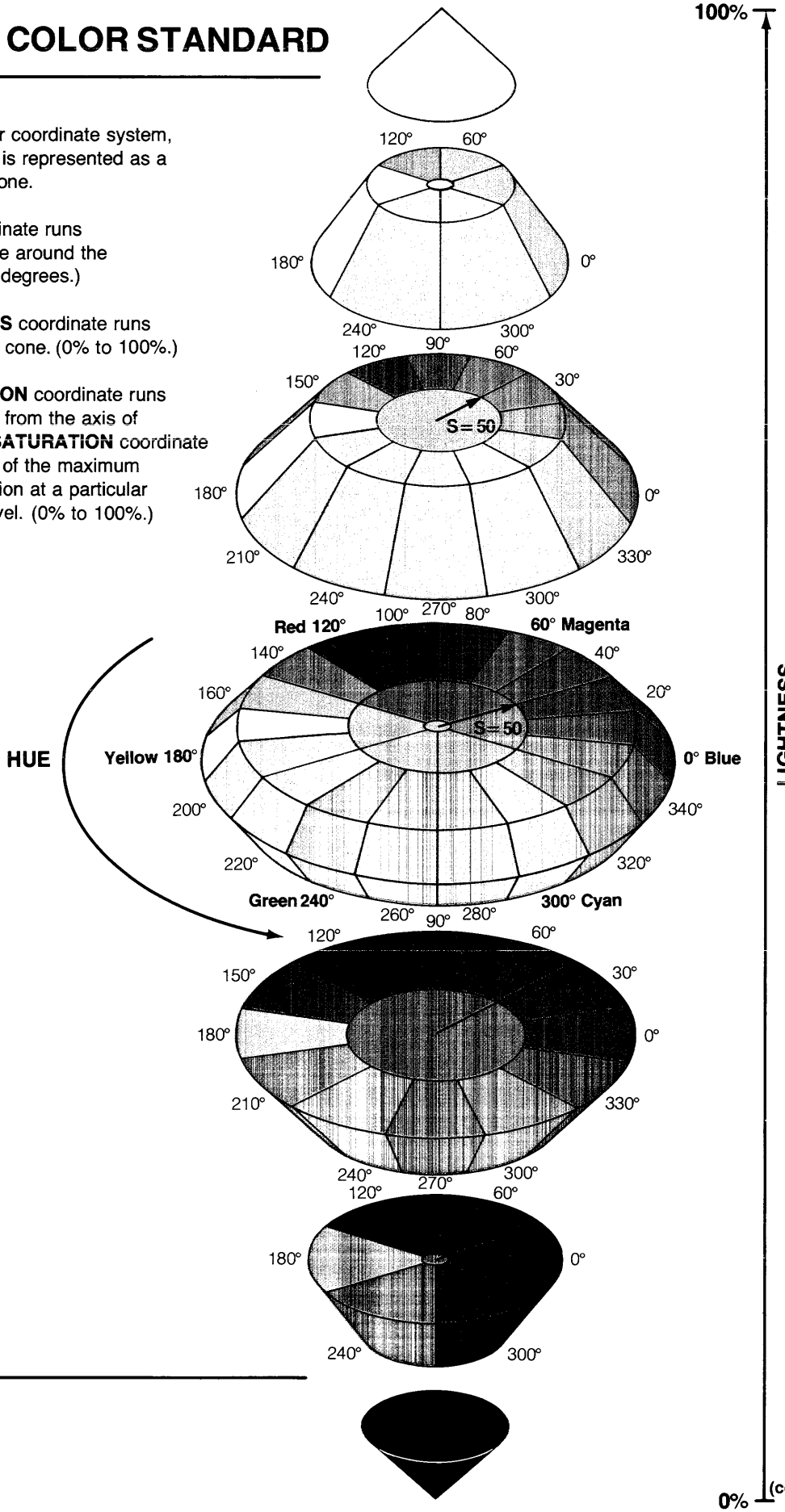
Appendix F
TEKTRONIX COLOR STANDARD

In the **HLS** color coordinate system, the color space is represented as a double-ended cone.

The **HUE** coordinate runs counterclockwise around the cone. (0 to 360 degrees.)

The **LIGHTNESS** coordinate runs vertically up the cone. (0% to 100%.)

The **SATURATION** coordinate runs radially outward from the axis of the cone. The **SATURATION** coordinate is a percentage of the maximum possible saturation at a particular **LIGHTNESS** level. (0% to 100%.)



(continued)

TEKTRONIX COLOR STANDARD

Overview:

The world of color is filled with ambiguous terminology, i.e. intensity, purity, value, etc. Many color users feel that "color theory" is a prerequisite to operating color systems; T.V., Videotaping, Photography, Computer Graphics.

In order to end this confusion, Tektronix has developed a color language and function based on human engineering, rather than machine engineering. Below is a description of this system, which will provide a clear and concise means for understanding how color is defined and how our syntax was derived.

Color Concepts:

Color selection is specified by hue, lightness and saturation which is the HLS method. The definitions are as follows:

Hue: The characteristic associated with a color name such as red, yellow, green, blue, etc. Hue is a gradation of color advanced by degrees, thus represented as an angle from 0 to 360.

Lightness: The characteristic that allows the color to be ranked on a scale from dark to light. Lightness is expressed as a parameter ranging from 0 to 100% with black being 0 (bottom of cone) and white being 100% (top of cone).

Saturation: The characteristic which describes the extent to which a color differs from a gray of the same lightness. Saturation is expressed as percentage, ranging from 0% (maximum white content at that lightness level) to 100% (full saturated).

Geometrically, colors can be described in terms of a double cone.
Variations in lightness are represented along the axis, with white at the apex of the cone and black at the opposite apex. Variations in saturation are represented by radial distances from the lightness axis, in constant lightness planes. Hue is represented as an angular quantity from a known reference point.

Appendix G

EXAMPLES OF INTEGER PARAMETERS

Table G-1 lists integer parameters between −1049 and +1049.

Table G-1

INTEGER PARAMETERS

+Int[a]	Param	−Int[b]	Param
0	0	−0	^{S}P
1	1	−1	!
2	2	−2	"
3	3	−3	#
4	4	−4	$
5	5	−5	%
6	6	−6	&
7	7	−7	'
8	8	−8	(
9	9	−9	)
10	:	−10	*
11	;	−11	+
12	<	−12	,
13	=	−13	−
14	>	−14	.
15	?	−15	/
16	A0	−16	$A^{S}P$
17	A1	−17	A!
18	A2	−18	A"
19	A3	−19	A#
20	A4	−20	A$
21	A5	−21	A%
22	A6	−22	A&
23	A7	−23	A'
24	A8	−24	A(
25	A9	−25	A)
26	A:	−26	A*
27	A;	−27	A+
28	A<	−28	A,
29	A=	−29	A−
30	A>	−30	A.
31	A?	−31	A/
32	B0	−32	$B^{S}P$
33	B1	−33	B!
34	B2	−34	B"
35	B3	−35	B#
36	B4	−36	B$
37	B5	−37	B%
38	B6	−38	B&
39	B7	−39	B'
40	B8	−40	B(
41	B9	−41	B)
42	B:	−42	B*
43	B;	−43	B+
44	B<	−44	B,
45	B=	−45	B−
46	B>	−46	B.
47	B?	−47	B/
48	C0	−48	$C^{S}P$
49	C1	−49	C!
50	C2	−50	C"
51	C3	−51	C#
52	C4	−52	C$
53	C5	−53	C%
54	C6	−54	C&
55	C7	−55	C'
56	C8	−56	C(
57	C9	−57	C)
58	C:	−58	C*
59	C;	−59	C+
60	C<	−60	C,
61	C=	−61	C−
62	C>	−62	C.
63	C?	−63	C/
64	D0	−64	$D^{S}P$
65	D1	−65	D!
66	D2	−66	D"
67	D3	−67	D#
68	D4	−68	D$
69	D5	−69	D%
70	D6	−70	D&
71	D7	−71	D'
72	D8	−72	D(
73	D9	−73	D)
74	D:	−74	D*
75	D;	−75	D+
76	D<	−76	D,
77	D=	−77	D−
78	D>	−78	D.
79	D?	−79	D/
80	E0	−80	$E^{S}P$
81	E1	−81	E!
82	E2	−82	E"
83	E3	−83	E#
84	E4	−84	E$
85	E5	−85	E%
86	E6	−86	E&
87	E7	−87	E'
88	E8	−88	E(
89	E9	−89	E)
90	E:	−90	E*
91	E;	−91	E+
92	E<	−92	E,
93	E=	−93	E−
94	E>	−94	E.
95	E?	−95	E/
96	F0	−96	$F^{S}P$
97	F1	−97	F!
98	F2	−98	F"
99	F3	−99	F#
100	F4	−100	F$
101	F5	−101	F%
102	F6	−102	F&
103	F7	−103	F'
104	F8	−104	F(
105	F9	−105	F)
106	F:	−106	F*
107	F;	−107	F+
108	F<	−108	F,
109	F=	−109	F−
110	F>	−110	F.
111	F?	−111	F/
112	G0	−112	$G^{S}P$
113	G1	−113	G!
114	G2	−114	G"
115	G3	−115	G#
116	G4	−116	G$
117	G5	−117	G%
118	G6	−118	G&
119	G7	−119	G'
120	G8	−120	G(
121	G9	−121	G)
122	G:	−122	G*
123	G;	−123	G+
124	G<	−124	G,
125	G=	−125	G−
126	G>	−126	G.
127	G?	−127	G/
128	H0	−128	$H^{S}P$
129	H1	−129	H!
130	H2	−130	H"
131	H3	−131	H#
132	H4	−132	H$
133	H5	−133	H%
134	H6	−134	H&
135	H7	−135	H'
136	H8	−136	H(
137	H9	−137	H)
138	H:	−138	H*
139	H;	−139	H+
140	H<	−140	H,
141	H=	−141	H−
142	H>	−142	H.
143	H?	−143	H/
144	I0	−144	$I^{S}P$
145	I1	−145	I!
146	I2	−146	I"
147	I3	−147	I#
148	I4	−148	I$
149	I5	−149	I%

[a] Positive integer

[b] Negative integer

For integers ± 144 through ± 159, the first character of the parameter is an uppercase I (ADE 73).

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param	+Int[a]	Param	-Int[b]	Param	+Int[a]	Param	-Int[b]	Param
150	I6	−150	I&	200	L8	−200	L(	250	O:	−250	O*
151	I7	−151	I'	201	L9	−201	L)	251	O;	−251	O+
152	I8	−152	I(	202	L:	−202	L*	252	O<	−252	O,
153	I9	−153	I)	203	L;	−203	L+	253	O=	−253	O−
154	I:	−154	I*	204	L<	−204	L,	254	O>	−254	O.
155	I;	−155	I+	205	L=	−205	L−	255	O?	−255	O/
156	I<	−156	I,	206	L>	−206	L.	256	P0	−256	P^{S_P}
157	I=	−157	I−	207	L?	−207	L/	257	P1	−257	P!
158	I>	−158	I.	208	M0	−208	M^{S_P}	258	P2	−258	P"
159	I?	−159	I/	209	M1	−209	M!	259	P3	−259	P#
160	J0	−160	J^{S_P}	210	M2	−210	M"	260	P4	−260	P$
161	J1	−161	J!	211	M3	−211	M#	261	P5	−261	P%
162	J2	−162	J"	212	M4	−212	M$	262	P6	−262	P&
163	J3	−163	J#	213	M5	−213	M%	263	P7	−263	P'
164	J4	−164	J$	214	M6	−214	M&	264	P8	−264	P(
165	J5	−165	J%	215	M7	−215	M'	265	P9	−265	P)
166	J6	−166	J&	216	M8	−216	M(	266	P:	−266	P*
167	J7	−167	J'	217	M9	−217	M)	267	P;	−267	P+
168	J8	−168	J(	218	M:	−218	M*	268	P<	−268	P,
169	J9	−169	J)	219	M;	−219	M+	269	P=	−269	P−
170	J:	−170	J*	220	M<	−220	M,	270	P>	−270	P.
171	J;	−171	J+	221	M=	−221	M−	271	P?	−271	P/
172	J<	−172	J,	222	M>	−222	M.	272	Q0	−272	Q^{S_P}
173	J=	−173	J−	223	M?	−223	M/	273	Q1	−273	Q!
174	J>	−174	J.	224	N0	−224	N^{S_P}	274	Q2	−274	Q"
175	J?	−175	J/	225	N1	−225	N!	275	Q3	−275	Q#
176	K0	−176	K^{S_P}	226	N2	−226	N"	276	Q4	−276	Q$
177	K1	−177	K!	227	N3	−227	N#	277	Q5	−277	Q%
178	K2	−178	K"	228	N4	−228	N$	278	Q6	−278	Q&
179	K3	−179	K#	229	N5	−229	N%	279	Q7	−279	Q'
180	K4	−180	K$	230	N6	−230	N&	280	Q8	−280	Q(
181	K5	−181	K%	231	N7	−231	N'	281	Q9	−281	Q)
182	K6	−182	K&	232	N8	−232	N(	282	Q:	−282	Q*
183	K7	−183	K'	233	N9	−233	N)	283	Q;	−283	Q+
184	K8	−184	K(	234	N:	−234	N*	284	Q<	−284	Q,
185	K9	−185	K)	235	N;	−235	N+	285	Q=	−285	Q−
186	K:	−186	K*	236	N<	−236	N,	286	Q>	−286	Q.
187	K;	−187	K+	237	N=	−237	N−	287	Q?	−287	Q/
188	K<	−188	K,	238	N>	−238	N.	288	R0	−288	R^{S_P}
189	K=	−189	K−	239	N?	−239	N/	289	R1	−289	R!
190	K>	−190	K.	240	O0	−240	O^{S_P}	290	R2	−290	R"
191	K?	−191	K/	241	O1	−241	O!	291	R3	−291	R#
192	L0	−192	L^{S_P}	242	O2	−242	O"	292	R4	−292	R$
193	L1	−193	L!	243	O3	−243	O#	293	R5	−293	R%
194	L2	−194	L"	244	O4	−244	O$	294	R6	−294	R&
195	L3	−195	L#	245	O5	−245	O%	295	R7	−295	R'
196	L4	−196	L$	246	O6	−246	O&	296	R8	−296	R(
197	L5	−197	L%	247	O7	−247	O'	297	R9	−297	R)
198	L6	−198	L&	248	O8	−248	O(	298	R:	−298	R*
199	L7	−199	L'	249	O9	−249	O)	299	R;	−299	R+

[a] Positive integer

[b] Negative integer

For integers ± 144 through ± 159, the first character of the parameter is an uppercase I (ADE 73).

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param
300	R<	-300	R,
301	R=	-301	R-
302	R>	-302	R.
303	R?	-303	R/
304	S0	-304	S^SP
305	S1	-305	S!
306	S2	-306	S"
307	S3	-307	S#
308	S4	-308	S$
309	S5	-309	S%
310	S6	-310	S&
311	S7	-311	S'
312	S8	-312	S(
313	S9	-313	S)
314	S:	-314	S*
315	S;	-315	S+
316	S<	-316	S,
317	S=	-317	S-
318	S>	-318	S.
319	S?	-319	S/
320	T0	-320	T^SP
321	T1	-321	T!
322	T2	-322	T"
323	T3	-323	T#
324	T4	-324	T$
325	T5	-325	T%
326	T6	-326	T&
327	T7	-327	T'
328	T8	-328	T(
329	T9	-329	T)
330	T:	-330	T*
331	T;	-331	T+
332	T<	-332	T,
333	T=	-333	T-
334	T>	-334	T.
335	T?	-335	T/
336	U0	-336	U^SP
337	U1	-337	U!
338	U2	-338	U"
339	U3	-339	U#
340	U4	-340	U$
341	U5	-341	U%
342	U6	-342	U&
343	U7	-343	U'
344	U8	-344	U(
345	U9	-345	U)
346	U:	-346	U*
347	U;	-347	U+
348	U<	-348	U,
349	U=	-349	U-
350	U>	-350	U.
351	U?	-351	U/
352	V0	-352	V^SP
353	V1	-353	V!
354	V2	-354	V"
355	V3	-355	V#
356	V4	-356	V$
357	V5	-357	V%
358	V6	-358	V&
359	V7	-359	V'
360	V8	-360	V(
361	V9	-361	V)
362	V:	-362	V*
363	V;	-363	V+
364	V<	-364	V,
365	V=	-365	V-
366	V>	-366	V.
367	V?	-367	V/
368	W0	-368	W^SP
369	W1	-369	W!
370	W2	-370	W"
371	W3	-371	W#
372	W4	-372	W$
373	W5	-373	W%
374	W6	-374	W&
375	W7	-375	W'
376	W8	-376	W(
377	W9	-377	W)
378	W:	-378	W*
379	W;	-379	W+
380	W<	-380	W,
381	W=	-381	W-
382	W>	-382	W.
383	W?	-383	W/
384	X0	-384	X^SP
385	X1	-385	X!
386	X2	-386	X"
387	X3	-387	X#
388	X4	-388	X$
389	X5	-389	X%
390	X6	-390	X&
391	X7	-391	X'
392	X8	-392	X(
393	X9	-393	X)
394	X:	-394	X*
395	X;	-395	X+
396	X<	-396	X,
397	X=	-397	X-
398	X>	-398	X.
399	X?	-399	X/
400	Y0	-400	Y^SP
401	Y1	-401	Y!
402	Y2	-402	Y"
403	Y3	-403	Y#
404	Y4	-404	Y$
405	Y5	-405	Y%
406	Y6	-406	Y&
407	Y7	-407	Y'
408	Y8	-408	Y(
409	Y9	-409	Y)
410	Y:	-410	Y*
411	Y;	-411	Y+
412	Y<	-412	Y,
413	Y=	-413	Y-
414	Y>	-414	Y.
415	Y?	-415	Y/
416	Z0	-416	Z^SP
417	Z1	-417	Z!
418	Z2	-418	Z"
419	Z3	-419	Z#
420	Z4	-420	Z$
421	Z5	-421	Z%
422	Z6	-422	Z&
423	Z7	-423	Z'
424	Z8	-424	Z(
425	Z9	-425	Z)
426	Z:	-426	Z*
427	Z;	-427	Z+
428	Z<	-428	Z,
429	Z=	-429	Z-
430	Z>	-430	Z.
431	Z?	-431	Z/
432	[0	-432	[^SP
433	[1	-433	[!
434	[2	-434	["
435	[3	-435	[#
436	[4	-436	[$
437	[5	-437	[%
438	[6	-438	[&
439	[7	-439	['
440	[8	-440	[(
441	[9	-441	[)
442	[:	-442	[*
443	[;	-443	[+
444	[<	-444	[,
445	[=	-445	[-
446	[>	-446	[.
447	[?	-447	[/
448	\0	-448	\^SP
449	\1	-449	\!

[a] Positive integer

[b] Negative integer

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param	+Int[a]	Param	-Int[b]	Param	+Int[a]	Param	-Int[b]	Param
450	\2	−450	\"	500	_4	−500	_$	550	b6	−550	b&
451	\3	−451	\#	501	_5	−501	_%	551	b7	−551	b'
452	\4	−452	\$	502	_6	−502	_&	552	b8	−552	b(
453	\5	−453	\%	503	_7	−503	_'	553	b9	−553	b)
454	\6	−454	\&	504	_8	−504	_(	554	b:	−554	b*
455	\7	−455	\'	505	_9	−505	_)	555	b;	−555	b+
456	\8	−456	\(	506	_:	−506	_*	556	b<	−556	b,
457	\9	−457	\)	507	_;	−507	_+	557	b=	−557	b−
458	\:	−458	*	508	_<	−508	_,	558	b>	−558	b.
459	\;	−459	\+	509	_=	−509	_−	559	b?	−559	b/
460	\<	−460	\,	510	_>	−510	_.	560	c0	−560	c^{SP}
461	\=	−461	\−	511	_?	−511	_/	561	c1	−561	c!
462	\>	−462	\.	512	'0	−512	'^{SP}	562	c2	−562	c"
463	\?	−463	\/	513	'1	−513	'!	563	c3	−563	c#
464	]0	−464	]^{SP}	514	'2	−514	'"	564	c4	−564	c$
465	]1	−465	]!	515	'3	−515	'#	565	c5	−565	c%
466	]2	−466	]"	516	'4	−516	'$	566	c6	−566	c&
467	]3	−467	]#	517	'5	−517	'%	567	c7	−567	c'
468	]4	−468	]$	518	'6	−518	'&	568	c8	−568	c(
469	]5	−469	]%	519	'7	−519	''	569	c9	−569	c)
470	]6	−470	]&	520	'8	−520	'(	570	c:	−570	c*
471	]7	−471	]'	521	'9	−521	')	571	c;	−571	c+
472	]8	−472	](	522	':	−522	'*	572	c<	−572	c,
473	]9	−473	])	523	';	−523	'+	573	c=	−573	c−
474	]:	−474	]*	524	'<	−524	',	574	c>	−574	c.
475	];	−475	]+	525	'=	−525	'−	575	c?	−575	c/
476	]<	−476	],	526	'>	−526	'.	576	d0	−576	d^{SP}
477	]=	−477	]−	527	'?	−527	'/	577	d1	−577	d!
478	]>	−478	].	528	a0	−528	a^{SP}	578	d2	−578	d"
479	]?	−479	]/	529	a1	−529	a!	579	d3	−579	d#
480	∧0	−480	∧^{SP}	530	a2	−530	a"	580	d4	−580	d$
481	∧1	−481	∧!	531	a3	−531	a#	581	d5	−581	d%
482	∧2	−482	∧"	532	a4	−532	a$	582	d6	−582	d&
483	∧3	−483	∧#	533	a5	−533	a%	583	d7	−583	d'
484	∧4	−484	∧$	534	a6	−534	a&	584	d8	−584	d(
485	∧5	−485	∧%	535	a7	−535	a'	585	d9	−585	d)
486	∧6	−486	∧&	536	a8	−536	a(	586	d:	−586	d*
487	∧7	−487	∧'	537	a9	−537	a)	587	d;	−587	d+
488	∧8	−488	∧(	538	a:	−538	a*	588	d<	−588	d,
489	∧9	−489	∧)	539	a;	−539	a+	589	d=	−589	d−
490	∧:	−490	∧*	540	a<	−540	a,	590	d>	−590	d.
491	∧;	−491	∧+	541	a=	−541	a−	591	d?	−591	d/
492	∧<	−492	∧,	542	a>	−542	a.	592	e0	−592	e^{SP}
493	∧=	−493	∧−	543	a?	−543	a/	593	e1	−593	e!
494	∧>	−494	∧.	544	b0	−544	b^{SP}	594	e2	−594	e"
495	∧?	−495	∧/	545	b1	−545	b!	595	e3	−595	e#
496	_0	−496	_^{SP}	546	b2	−546	b"	596	e4	−596	e$
497	_1	−497	_!	547	b3	−547	b#	597	e5	−597	e%
498	_2	−498	_"	548	b4	−548	b$	598	e6	−598	e&
499	_3	−499	_#	549	b5	−549	b%	599	e7	−599	e'

[a] Positive integer

[b] Negative integer

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param
600	e8	−600	e(
601	e9	−601	e)
602	e:	−602	e*
603	e;	−603	e+
604	e<	−604	e,
605	e=	−605	e−
606	e>	−606	e.
607	e?	−607	e/
608	f0	−608	f^{SP}
609	f1	−609	f!
610	f2	−610	f"
611	f3	−611	f#
612	f4	−612	f$
613	f5	−613	f%
614	f6	−614	f&
615	f7	−615	f'
616	f8	−616	f(
617	f9	−617	f)
618	f:	−618	f*
619	f;	−619	f+
620	f<	−620	f,
621	f=	−621	f−
622	f>	−622	f.
623	f?	−623	f/
624	g0	−624	g^{SP}
625	g1	−625	g!
626	g2	−626	g"
627	g3	−627	g#
628	g4	−628	g$
629	g5	−629	g%
630	g6	−630	g&
631	g7	−631	g'
632	g8	−632	g(
633	g9	−633	g)
634	g:	−634	g*
635	g;	−635	g+
636	g<	−636	g,
637	g=	−637	g−
638	g>	−638	g.
639	g?	−639	g/
640	h0	−640	h^{SP}
641	h1	−641	h!
642	h2	−642	h"
643	h3	−643	h#
644	h4	−644	h$
645	h5	−645	h%
646	h6	−646	h&
647	h7	−647	h'
648	h8	−648	h(
649	h9	−649	h)
650	h:	−650	h*
651	h;	−651	h+
652	h<	−652	h,
653	h=	−653	h−
654	h>	−654	h.
655	h?	−655	h/
656	i0	−656	i^{SP}
657	i1	−657	i!
658	i2	−658	i"
659	i3	−659	i#
660	i4	−660	i$
661	i5	−661	i%
662	i6	−662	i&
663	i7	−663	i'
664	i8	−664	i(
665	i9	−665	i)
666	i:	−666	i*
667	i;	−667	i+
668	i<	−668	i,
669	i=	−669	i−
670	i>	−670	i.
671	i?	−671	i/
672	j0	−672	j^{SP}
673	j1	−673	j!
674	j2	−674	j"
675	j3	−675	j#
676	j4	−676	j$
677	j5	−677	j%
678	j6	−678	j&
679	j7	−679	j'
680	j8	−680	j(
681	j9	−681	j)
682	j:	−682	j*
683	j;	−683	j+
684	j<	−684	j,
685	j=	−685	j−
686	j>	−686	j.
687	j?	−687	j/
688	k0	−688	k^{SP}
689	k1	−689	k!
690	k2	−690	k"
691	k3	−691	k#
692	k4	−692	k$
693	k5	−693	k%
694	k6	−694	k&
695	k7	−695	k'
696	k8	−696	k(
697	k9	−697	k)
698	k:	−698	k*
699	k;	−699	k+
700	k<	−700	k,
701	k=	−701	k−
702	k>	−702	k.
703	k?	−703	k/
704	l0	−704	l^{SP}
705	l1	−705	l!
706	l2	−706	l"
707	l3	−707	l#
708	l4	−708	l$
709	l5	−709	l%
710	l6	−710	l&
711	l7	−711	l'
712	l8	−712	l(
713	l9	−713	l)
714	l:	−714	l*
715	l;	−715	l+
716	l<	−716	l,
717	l=	−717	l−
718	l>	−718	l.
719	l?	−719	l/
720	m0	−720	m^{SP}
721	m1	−721	m!
722	m2	−722	m"
723	m3	−723	m#
724	m4	−724	m$
725	m5	−725	m%
726	m6	−726	m&
727	m7	−727	m'
728	m8	−728	m(
729	m9	−729	m)
730	m:	−730	m*
731	m;	−731	m+
732	m<	−732	m,
733	m=	−733	m−
734	m>	−734	m.
735	m?	−735	m/
736	n0	−736	n^{SP}
737	n1	−737	n!
738	n2	−738	n"
739	n3	−739	n#
740	n4	−740	n$
741	n5	−741	n%
742	n6	−742	n&
743	n7	−743	n'
744	n8	−744	n(
745	n9	−745	n)
746	n:	−746	n*
747	n;	−747	n+
748	n<	−748	n,
749	n=	−749	n−

[a] Positive integer

[b] Negative integer

For integers ± 704 through ± 719, the first character of the parameter is a lowercase l (ADE 108).

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param
750	n>	−750	n.
751	n?	−751	n/
752	o0	−752	o^{SP}
753	o1	−753	o!
754	o2	−754	o"
755	o3	−755	o#
756	o4	−756	o$
757	o5	−757	o%
758	o6	−758	o&
759	o7	−759	o'
760	o8	−760	o(
761	o9	−761	o)
762	o:	−762	o*
763	o;	−763	o+
764	o<	−764	o,
765	o=	−765	o−
766	o>	−766	o.
767	o?	−767	o/
768	p0	−768	p^{SP}
769	p1	−769	p!
770	p2	−770	p"
771	p3	−771	p#
772	p4	−772	p$
773	p5	−773	p%
774	p6	−774	p&
775	p7	−775	p'
776	p8	−776	p(
777	p9	−777	p)
778	p:	−778	p*
779	p;	−779	p+
780	p<	−780	p,
781	p=	−781	p−
782	p>	−782	p.
783	p?	−783	p/
784	q0	−784	q^{SP}
785	q1	−785	q!
786	q2	−786	q"
787	q3	−787	q#
788	q4	−788	q$
789	q5	−789	q%
790	q6	−790	q&
791	q7	−791	q'
792	q8	−792	q(
793	q9	−793	q)
794	q:	−794	q*
795	q;	−795	q+
796	q<	−796	q,
797	q=	−797	q−
798	q>	−798	q.
799	q?	−799	q/
800	r0	−800	r^{SP}
801	r1	−801	r!
802	r2	−802	r"
803	r3	−803	r#
804	r4	−804	r$
805	r5	−805	r%
806	r6	−806	r&
807	r7	−807	r'
808	r8	−808	r(
809	r9	−809	r)
810	r:	−810	r*
811	r;	−811	r+
812	r<	−812	r,
813	r=	−813	r−
814	r>	−814	r.
815	r?	−815	r/
816	s0	−816	s^{SP}
817	s1	−817	s!
818	s2	−818	s"
819	s3	−819	s#
820	s4	−820	s$
821	s5	−821	s%
822	s6	−822	s&
823	s7	−823	s'
824	s8	−824	s(
825	s9	−825	s)
826	s:	−826	s*
827	s;	−827	s+
828	s<	−828	s,
829	s=	−829	s−
830	s>	−830	s.
831	s?	−831	s/
832	t0	−832	t^{SP}
833	t1	−833	t!
834	t2	−834	t"
835	t3	−835	t#
836	t4	−836	t$
837	t5	−837	t%
838	t6	−838	t&
839	t7	−839	t'
840	t8	−840	t(
841	t9	−841	t)
842	t:	−842	t*
843	t;	−843	t+
844	t<	−844	t,
845	t=	−845	t−
846	t>	−846	t.
847	t?	−847	t/
848	u0	−848	u^{SP}
849	u1	−849	u!
850	u2	−850	u"
851	u3	−851	u#
852	u4	−852	u$
853	u5	−853	u%
854	u6	−854	u&
855	u7	−855	u'
856	u8	−856	u(
857	u9	−857	u)
858	u:	−858	u*
859	u;	−859	u+
860	u<	−860	u,
861	u=	−861	u−
862	u>	−862	u.
863	u?	−863	u/
864	v0	−864	v^{SP}
865	v1	−865	v!
866	v2	−866	v"
867	v3	−867	v#
868	v4	−868	v$
869	v5	−869	v%
870	v6	−870	v&
871	v7	−871	v'
872	v8	−872	v(
873	v9	−873	v)
874	v:	−874	v*
875	v;	−875	v+
876	v<	−876	v,
877	v=	−877	v−
878	v>	−878	v.
879	v?	−879	v/
880	w0	−880	w^{SP}
881	w1	−881	w!
882	w2	−882	w"
883	w3	−883	w#
884	w4	−884	w$
885	w5	−885	w%
886	w6	−886	w&
887	w7	−887	w'
888	w8	−888	w(
889	w9	−889	w)
890	w:	−890	w*
891	w;	−891	w+
892	w<	−892	w,
893	w=	−893	w−
894	w>	−894	w.
895	w?	−895	w/
896	x0	−896	x^{SP}
897	x1	−897	x!
898	x2	−898	x"
899	x3	−899	x#

[a] Positive integer

[b] Negative integer

Table G-1 (cont)

INTEGER PARAMETERS

+Int[a]	Param	-Int[b]	Param
900	x4	−900	x$
901	x5	−901	x%
902	x6	−902	x&
903	x7	−903	x'
904	x8	−904	x(
905	x9	−905	x)
906	x:	−906	x*
907	x;	−907	x+
908	x<	−908	x,
909	x=	−909	x−
910	x>	−910	x.
911	x?	−911	x/
912	y0	−912	y^{S_P}
913	y1	−913	y!
914	y2	−914	y"
915	y3	−915	y#
916	y4	−916	y$
917	y5	−917	y%
918	y6	−918	y&
919	y7	−919	y'
920	y8	−920	y(
921	y9	−921	y)
922	y:	−922	y*
923	y;	−923	y+
924	y<	−924	y,
925	y=	−925	y−
926	y>	−926	y.
927	y?	−927	y/
928	z0	−928	z^{S_P}
929	z1	−929	z!
930	z2	−930	z"
931	z3	−931	z#
932	z4	−932	z$
933	z5	−933	z%
934	z6	−934	z&
935	z7	−935	z'
936	z8	−936	z(
937	z9	−937	z)
938	z:	−938	z*
939	z;	−939	z+
940	z<	−940	z,
941	z=	−941	z−
942	z>	−942	z.
943	z?	−943	z/
944	{0	−944	$\{^{S_P}$
945	{1	−945	{!
946	{2	−946	{"
947	{3	−947	{#
948	{4	−948	{$
949	{5	−949	{%

[a] Positive integer

[b] Negative integer

+Int[a]	Param	-Int[b]	Param
950	{6	−950	{&
951	{7	−951	{'
952	{8	−952	{(
953	{9	−953	{)
954	{:	−954	{*
955	{;	−955	{+
956	{<	−956	{,
957	{=	−957	{−
958	{>	−958	{.
959	{?	−959	{/
960	|0	−960	$\|^{S_P}$
961	|1	−961	|!
962	|2	−962	|"
963	|3	−963	|#
964	|4	−964	|$
965	|5	−965	|%
966	|6	−966	|&
967	|7	−967	|'
968	|8	−968	|(
969	|9	−969	|)
970	|:	−970	|*
971	|;	−971	|+
972	|<	−972	|,
973	|=	−973	|−
974	|>	−974	|.
975	|?	−975	|/
976	}0	−976	$\}^{S_P}$
977	}1	−977	}!
978	}2	−978	}"
979	}3	−979	}#
980	}4	−980	}$
981	}5	−981	}%
982	}6	−982	}&
983	}7	−983	}'
984	}8	−984	}(
985	}9	−985	})
986	}:	−986	}*
987	};	−987	}+
988	}<	−988	},
989	}=	−989	}−
990	}>	−990	}.
991	}?	−991	}/
992	~0	−992	$\sim^{S_P}$
993	~1	−993	~!
994	~2	−994	~"
995	~3	−995	~#
996	~4	−996	~$
997	~5	−997	~%
998	~6	−998	~&
999	~7	−999	~'

For integers ±960 through ±975, the first character of the parameter is a vertical bar (ADE 124.).

+Int[a]	Param	-Int[b]	Param
1000	~8	−1000	~(
1001	~9	−1001	~)
1002	~:	−1002	~*
1003	~;	−1003	~+
1004	~<	−1004	~,
1005	~=	−1005	~−
1006	~>	−1006	~.
1007	~?	−1008	~/
1008	$^{D_T}0$	−1008	$^{D_T}S_P$
1009	$^{D_T}1$	−1009	^{D_T}!
1010	$^{D_T}2$	−1010	^{D_T}"
1011	$^{D_T}3$	−1011	^{D_T}#
1012	$^{D_T}4$	−1012	^{D_T}$
1013	$^{D_T}5$	−1013	^{D_T}%
1014	$^{D_T}6$	−1014	^{D_T}&
1015	$^{D_T}7$	−1015	^{D_T}'
1016	$^{D_T}8$	−1016	^{D_T}(
1017	$^{D_T}9$	−1017	^{D_T})
1018	^{D_T}:	−1018	^{D_T}*
1019	^{D_T};	−1019	^{D_T}+
1020	^{D_T}<	−1020	^{D_T},
1021	^{D_T}=	−1021	^{D_T}−
1022	^{D_T}>	−1022	^{D_T}.
1023	^{D_T}?	−1023	^{D_T}/
1024	A@0	−1024	$A@^{S_P}$
1025	A@1	−1025	A@!
1026	A@2	−1026	A@"
1027	A@3	−1027	A@#
1028	A@4	−1028	A@$
1029	A@5	−1029	A@%
1030	A@6	−1030	A@&
1031	A@7	−1031	A@'
1032	A@8	−1032	A@(
1033	A@9	−1033	A@)
1034	A@:	−1034	A@*
1035	A@;	−1035	A@+
1036	A@<	−1036	A@,
1037	A@=	−1037	A@−
1038	A@>	−1038	A@.
1039	A@?	−1039	A@/
1040	AA0	−1040	AA^{S_P}
1041	AA1	−1041	AA!
1042	AA2	−1042	AA"
1043	AA3	−1043	AA#
1044	AA4	−1044	AA$
1045	AA5	−1045	AA%
1046	AA6	−1046	AA&
1047	AA7	−1047	AA'
1048	AA8	−1048	AA(
1049	AA9	−1049	AA)

INDEX

Note: Entries shown in all uppercase letters are command names.

[1] **ANSI command**
[2] **VT52 command**
[3] **4100 command**

[1] ANSI command
[2] VT52 command
[3] 4100 command

[1] **ANSI command**
[2] **VT52 command**
[3] **4100 command**

[1] ANSI command
[2] VT52 command
[3] 4100 command

[1] **ANSI command**
[2] **VT52 command**
[3] **4100 command**

[1] ANSI command
[2] VT52 command
[3] 4100 command

[1] **ANSI command**
[2] **VT52 command**
[3] **4100 command**

[1] **ANSI command**
[2] **VT52 command**
[3] **4100 command**