TEK OPERATOR'S MANUAL

Part No. 070-3683-00
Product Group 16

4112
COMPUTER DISPLAY TERMINAL

Please Check for CHANGE INFORMATION at the Rear of this Manual

First Printing JUN 1981
Revised APR 1983

Tektronix
COMMITTED TO EXCELLENCE

WARNING

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instruction manual, may cause interference to radio communications. It has been tested and found to comply with the limits for Class A computing devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the users at their own expense will be required to take whatever measures may be required to correct the interference.

This instrument, in whole or in part, may be protected by one or more U.S. or foreign patents or patent applications. Information provided on request by Tektronix, Inc., P.O. Box 500, Beaverton, Oregon 97077.

MANUAL REVISION STATUS

PRODUCT: 4112 Computer Display Terminal

This manual supports the following versions of this product: Serial Numbers B010100 and up.

REV DATE	DESCRIPTION
JUN 1981	Original Issue
AUG 1981	Extensively revised.
NOV 1981	Revised: page vi. Added: Appendix E.
MAR 1982	Revised: pages 6-53, 6-54, A-4, A-5, and C-4.
JUN 1982	Revised: pages 6-5 and 6-39 through 6-41.
AUG 1982	Revised: pages 1-4 and 1-5.
AUG 1982	Added: pages 1-6, 3-12, and 3-13.
NOV 1982	Revised: page 6-39.

CONTENTS

Section 6 (continued) **Page**

ILLUSTRATIONS

TABLES

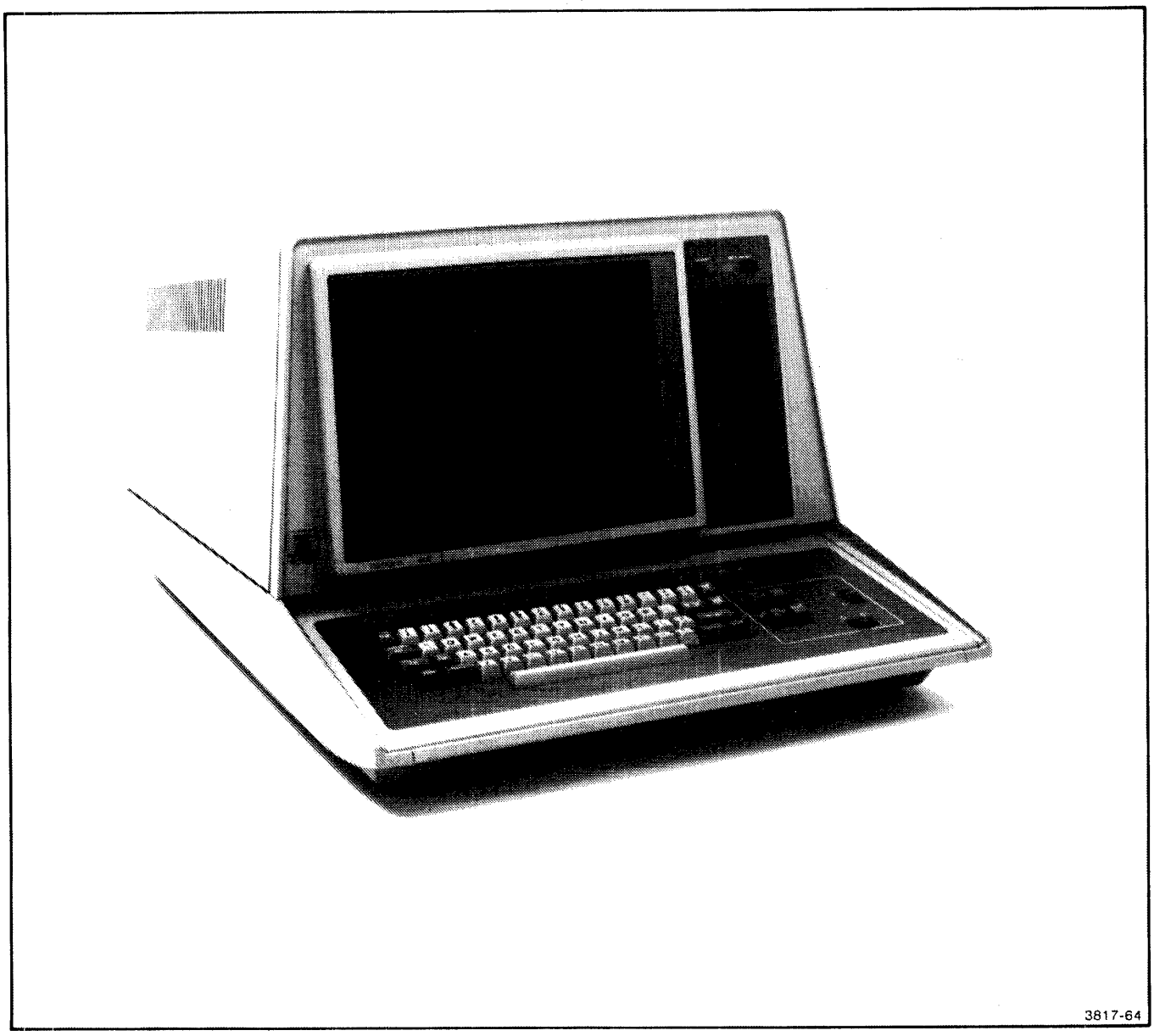


Figure 1-1. The 4112 Computer Display Terminal.

Section 1

INTRODUCTION

The TEKTRONIX 4112 Computer Display Terminal (Figure 1-1) is a graphics and alphanumerics terminal. The terminal's features are briefly described in this section and discussed in detail in subsequent sections of this manual.

ABOUT THIS MANUAL

This is the 4112 Computer Display Terminal Operator's Manual. It explains in detail the features and capabilities of the terminal from the operator's point of view.

Before reading this manual, you should read "Introducing Your 4112." This short brochure provides initial information and examples to help you use the 4112.

RELATED DOCUMENTATION

Additional documentation is included in the following manuals:

- 4112 Computer Display Terminal Host Programmer's Manual
- 4110 Series Command Reference Manual
- 4112 Computer Display Terminal Service Manual Volumes 1 and 2 (Optional)

SOFTWARE SUPPORT

The new features of the 4112 don't make your existing 4010-series applications programs obsolete.

The 4112 is compatible with 4010-style graphics. In most cases, a program written and executed on a 4010-family Tektronix terminal can be executed on your 4112 with few if any changes. The only modifications necessary are those which take advantage of features that are new to the 4112. These include the dialog area (see Section 2), new features of graphics input (GIN) mode (see Section 4), and such graphics features as segments, panels and gray-index levels.

The 4112 can be driven by TEKTRONIX PLOT 10 software products:

- Interactive Graphics Library
- Terminal Control System (and associated packages)
- Easy Graphing

See the user's manuals for those products for information on how to generate graphics output.

If you choose to write your own software to drive the terminal, see the Host Programmer's Manual and the 4110 Series Command Reference Manual for important information.

ABOUT THE 4112

The rest of this section briefly introduces the 4112's major features. References are made to sections in this manual where the features are discussed in detail.

SELF DIAGNOSTICS

Every time you turn the terminal on, it tests certain parts of its circuitry. If there is a problem which would prevent use of the terminal or one of its options, a light pattern flashes on the indicator lamps on the keyboard. In some cases an error message is also displayed on the screen.

You can record this light pattern and report it to a service technician. This makes it much easier and faster for the service person to correct the difficulty.

See Appendix B and the discussion of the "Power Up Sequence" and "Self Test" in Section 2 for details.

THE GRAPHICS AREA

Pictures (graphics) and text are displayed on the 15-inch screen (Figure 1-2). The brightness of the display is controlled by the INTENSITY control knob.

Segments

A "graphics segment" is a picture or a part of a picture which can be manipulated by terminal commands.

The 4112 stores segments locally, allowing a graphics application program to be more flexible and allowing a greater interaction with the terminal operator. Local storage of segments also reduces terminal/host communications needs and allows the terminal to handle some of the computational and storage tasks for which the host would otherwise be responsible.

Multiple Viewports

The screen can be divided into as many as 64 individual viewports. Creating viewports is normally done by a host resident software progam. See the 4112 Host Programmer's Reference Manual for details. The segments in the viewports can be manipulated locally by using the viewing keys.

Graphics Input

Graphics input (GIN) mode allows you to send graphics information to a host resident program, either from the terminal, a plotter, or a graphics tablet. GIN mode functions in much the same way on a 4112 as it does on a 4010-family terminal, but with some added features. See Section 4 for details.

THE VIEWING KEYS

The 4112's four viewing keys allow you to manipulate segments by:

- putting the terminal into Framing mode so you can "zoom" in on a segment for a closer view
- putting the terminal into Framing mode so you can "pan" across a segment to view part of it in greater detail
- restoring the original view
- restoring the original aspect ratio for the viewport (if you changed it by zooming and panning)
- getting an overview of the entire viewport (after you have zoomed or panned for a close-up of part of it)
- creating a border around a viewport

There is also a key which allows you to select the viewport in which the viewing keys are active.

The viewing keys are described in detail in Section 2.

DIALOG AREA

The dialog area is a part of the terminal's memory that stores non-graphics text which is the dialog between the terminal operator and the host or the terminal itself. The dialog may be invisible, entirely visible or a part of the dialog area may be visible.

In most programs, text sent to and received from the host is dialog between the terminal operator and the application program or host monitor. You can define any portion of the screen as the area in which all such dialog appears. This keeps it from becoming mixed in with vectors and text which are intended as part of the graphics output.

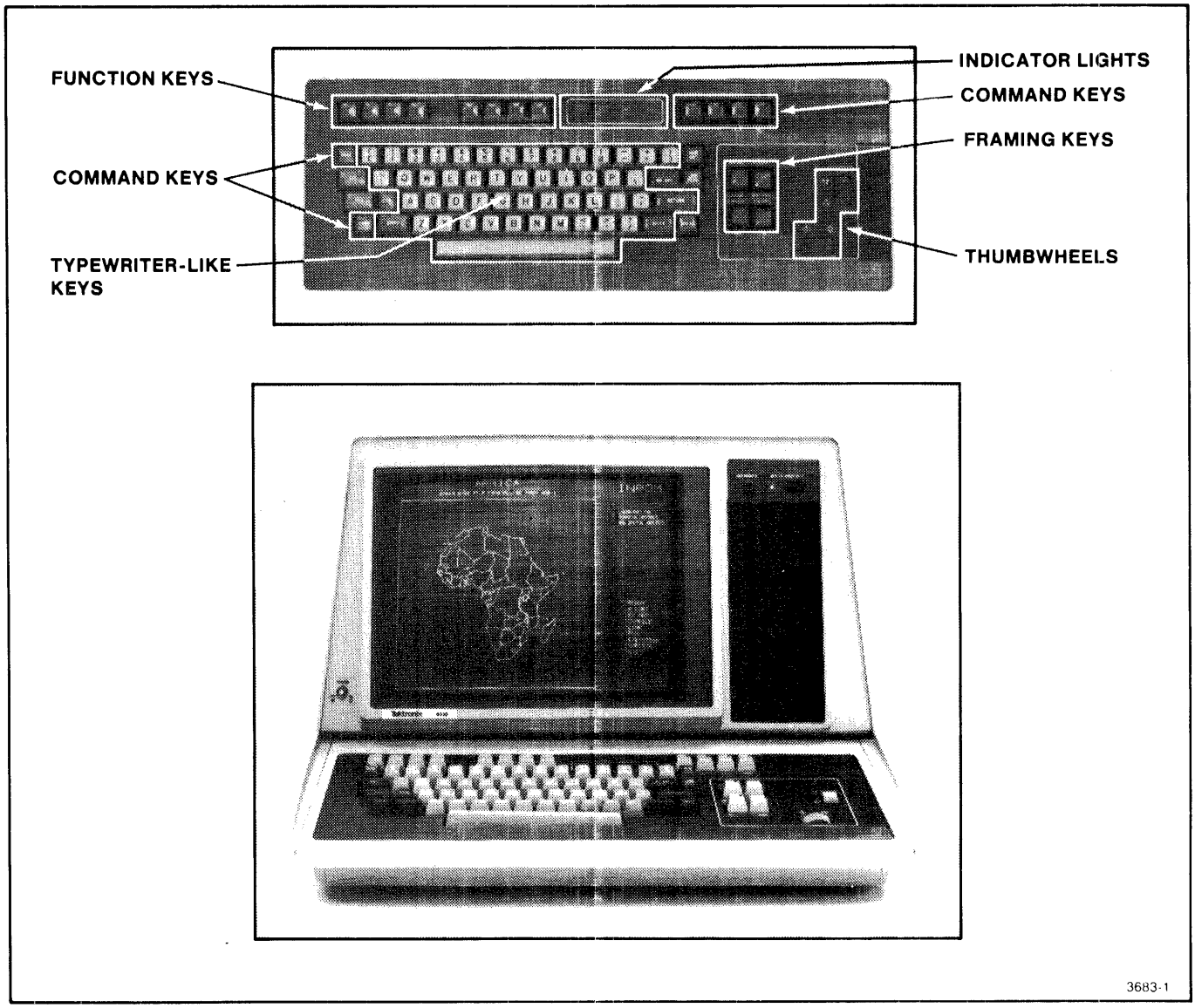


Figure 1-2. 4112 Display and Keyboard.

SETUP MODE

There are many English-like commands (called "setup commands") which can be entered directly from the keyboard. These commands are used to prepare the terminal for communications with a host, describe and position the dialog area, specify what kinds of errors the terminal should report, control peripheral devices, and so on.

You can enter setup commands when the terminal is in Setup mode. When the red light in the SET UP key is on, the terminal is in Setup mode. The setup commands are discussed in Section 6.

The terminal "remembers" most setup commands after it is turned off and then turned back on. That means that you don't need to enter the same setup commands every time you use the terminal unless you want to change some of the setup parameters.

There is also a set of commands called "escape sequences" which you can enter in Setup mode.

LOCAL MODE

The LOCAL key initiates or terminates Local mode. When the terminal is in Local mode, it is as if it were not connected to a host computer. Data which would normally be sent to the host is interpreted as if it came from the host. Host input that comes to the terminal while it is in Local mode is stored in a queue and is executed when Local mode is terminated. If the dialog area is in use, text entered in Local mode appears in the dialog area.

ERROR REPORTING

When the 4112 detects an error condition, it is recorded in an internal error queue and may be displayed on the screen. There are four levels of errors. You can establish the error threshold, which determines whether or not an error message is displayed on the terminal (see ERRORLEVEL in Section 6).

The terminal's error messages are listed in Appendix A.

THE DISK DRIVE

The terminal's optional flexible disk drive can be used for local mass storage of graphics and text data. It is easily operated by a group of setup commands. See Sections 3 and 6 for details.

THE KEYBOARD

The 4112's keyboard (Figure 1-2) is described in detail in Section 2. It's features include:

- A typewriter-like keyboard for typed input
- Pre-programmed command keys for commonly used features
- Programmable keys
- Indicator lights to inform you when the terminal is in a particular mode, or transmitting or receiving data
- Viewing keys to manipulate segments
- Thumbwheels to scroll the dialog area, control the graphics input, and define a new view when the terminal is in Framing mode.

REAR PANEL FEATURES

The following connectors, switches, and features are located on the 4112 rear panel and are shown in Figure 1-3.

- AC Power Cord Connector — This connector accepts the female end of a standard 110V/220V AC 60-Hz power cord. This cord is provided with the terminal as a standard accessory.

- Fuse holder — The main power fuse is located just under the power cord connector. SEE CAUTION BELOW. To remove the bad fuse, unscrew the cover on the fuse holder with a flat blade screwdriver; the fuse pops out when the cover is removed.

CAUTION

If a fuse burns out this may indicate a fault in the equipment. First, find and remedy the problem (call a qualified service person). Then, replace the fuse with the one of proper value (3AG 6.25A, for 125 V operation).

- Configuration/Options label — This label lists all of the options installed in the terminal at the factory. If you are not sure whether a particular option is installed in your terminal check this label.

- To Modem RS-232 (connector) — This connector accepts a standard RS-232 cable and connects to the host computor via a direct line or a modem/TIA[1] and telephone line.

[1] **MODEM means data line MOdulator/DEModulator. TIA means Terminal Interface Adapter.**

- SELF TEST and RESET buttons — These switches are for test purposes only and should not be used by the operator. See the 4112 Service Manual for complete service information related to these controls.

- Hard Copy Connector — This connector provides a hard copy output and accepts a cable from a TEKTRONIX hard copy unit.

Optional connectors:

- VIDEO OUT[2]— This connector supplies a standard video signal to an external monitor (on Option 11 equipped terminal).

- EXT SYNC — The connector accepts an input from an external video synchronization source (only on Option 11 equipped terminals).

- 3-PORT PERIPHERAL INTERFACE — Consists of three RS-232 type connectors; only on Option 10 (3PPI) equipped terminals.

- GRAPHIC TABLET connector — Provides connection to TEKTRONIX graphic tablets; only on Option 13/14 equipped terminals.

[2]This connector is only installed on terminals equipped with Option 12.

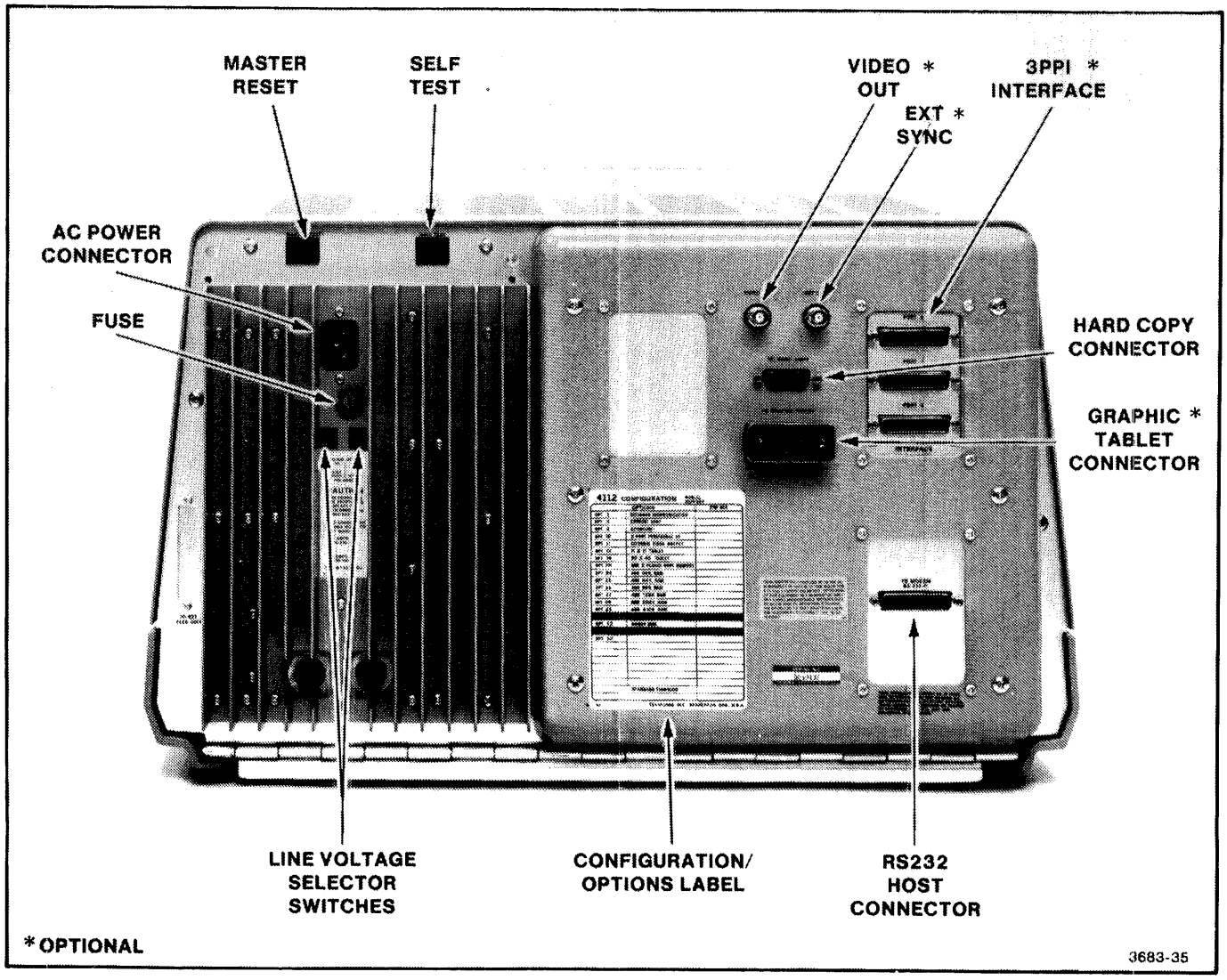


Figure 1-3. Rear Panel Features.

HOST COMMUNICATIONS

The 4112 communicates with the host computer via a direct line or a modem/TIA and telephone line. An RS-232 host connector is located on the rear panel of the 4112. See Figure 1-3.

First, connect the AC power cord to the mating connector on the rear panel of the terminal. Plug cord into power source. Then connect the RS-232 cable to the connector labeled: TO MODEM RS-232 (also on rear panel of terminal). Connect the free end of this cable to the modem. Power up the 4112 and verify proper host communications by logging on to your host system. Log off when finished.

OPTIONAL COMMUNICATIONS

Option 01 provides block mode and half-duplex communications. Section 5 provides a brief overview of communications concepts. Section 6 includes explanations of commands which control communications. The Host 4112 Programmer's Manual and the 4110 Series Command Reference Manual have more detailed communications information.

PERIPHERALS

The 4112 is compatible with most other Tektronix peripheral devices, such as plotters, printers, and hard copy units. See Section 3 for details.

Section 2

CONTROLS, INDICATORS, AND KEYBOARD

This section is divided into two sub-sections. The first explains the terminal's controls and indicators. The second sub-section is a description of the terminal's keyboard.

CONTROLS AND INDICATORS

The terminal's controls and indicators include the following:

- power switch
- MASTER RESET button
- SELF TEST button
- WRITE PROTECT switch (optional)
- WRITE PROTECT indicator (optional)
- disk drive indicator lamp (optional)
- REFRESH INTENSITY knob
- KBD LOCK lamp
- PAGE FULL lamp
- XMT lamp
- RCV lamp
- thumbwheels

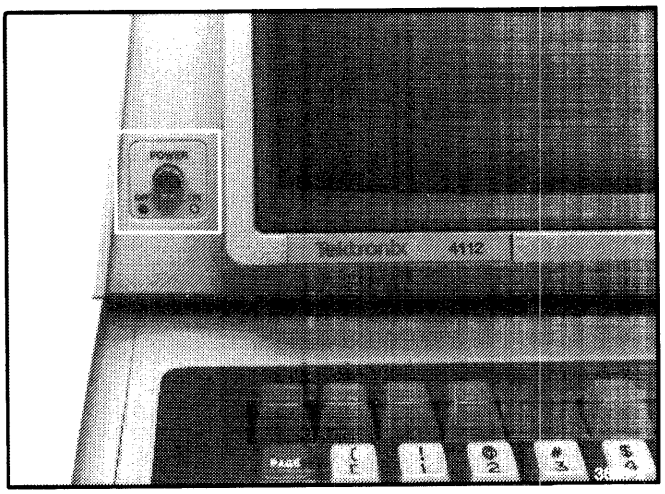


Figure 2-1. The Power Switch.

TURNING THE TERMINAL ON

To turn the terminal on, push the power switch (Figure 2-1). Release it when it clicks. A green indicator appears in the switch. You should also hear the ventilation fan inside the terminal turn on.

CAUTION

Do not block the air vents in the top of the terminal's cabinet (Figure 2-2). If you do, the terminal could overheat and damage its circuitry. These air vents allow proper ventilation.

The Power up Sequence

Each time you turn the terminal on, it automatically tests parts of its memory and circuitry. This is called the "power up sequence." It takes from 15-60 seconds depending on what options your terminal has, and requires no operator interaction.

During the power up sequence the lights in the keyboard (Figure 2-3) go through the following sequence:

- All lights turn on.
- The CAPS LOCK light turns off.
- All other lights turn off.
- The XMT and RCV lights flash once simultaneously.
- All lights turn off.

If the terminal bell does not ring during the power up sequence (within 60 seconds of turning the terminal on), the cursor flashes in the upper left corner of the screen. If the dialog area is enabled (see "DIALOG" later in this section) the cursor appears on the first available line of the dialog area. As soon as the cursor appears on the screen, the terminal is ready for use.

If the power up sequence detects an error the terminal bell rings one, three, or four times, depending on the severity of the problem. There may also be a message on the screen. If this occurs, see Appendix B.

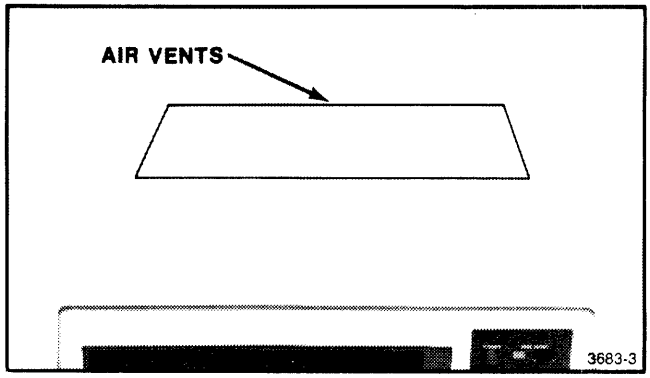


Figure 2-2. The Air Vents.

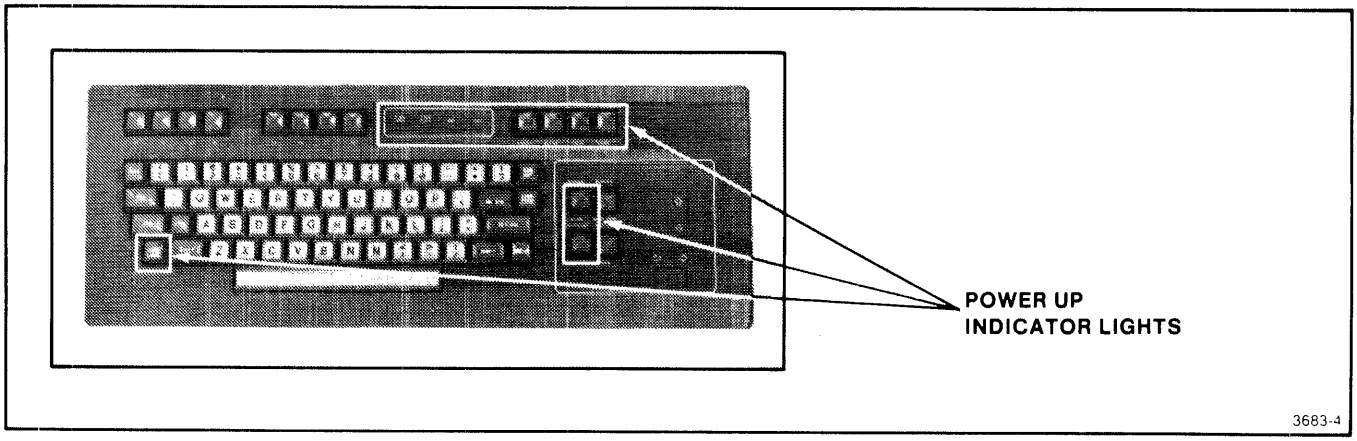


Figure 2-3. Keyboard Indicators in Power Up Sequence.

Power Up Defaults

Much of the terminal's operating environment when it is turned on (power up defaults) depends on its environment when it is turned off.

Many elements of the operating environment can be altered under program control or by "setup commands." The results of many of these changes are stored in "setup memory" and are remembered the next time the terminal is turned on. Table 6-1 (in Section 6) indicates the parameters which are saved in setup memory.

Setup commands and Setup mode are discussed in this section under the heading "SET UP" and in Section 6.

Table 6-1 also indicates the "factory default" operating environment of the terminal. "Factory default" means the setting which the terminal has when it is shipped from the factory. The status of your terminal is different from what is shown in that table if the parameters were altered by setup commands. See the STATUS command in Section 6 for information on how to determine the environment of your terminal.

If the setup memory fails, the operating environment is reset to the factory default. If this has happened, you are informed by a message during the power up sequence (see Appendix B).

At times you may want to reset the terminal to all of its default parameters yourself. For this procedure, see Appendix B.

MASTER RESET

NOTE

Do not press the MASTER RESET button if there is important information stored in the terminal's memory. Resetting the terminal deletes all graphics data as well as macros which have been programmed into keys.

Use the MASTER RESET button (Figure 2-4) when you want to return the terminal to its power up condition.

Pressing the MASTER RESET button has the same effect on the circuitry as if the terminal were turned off and back on:

- Macros are deleted from programmed keys.
- Graphics data is deleted from the terminal's memory.
- The terminal runs its power up sequence.
- The screen is erased.

You can save information before resetting the terminal by making a hardcopy of the screen (the HARD COPY key is explained later in this section). If your terminal has a flexible disk (Option 42) you can copy data to a disk or to a host file (see Section 3).

When you press the MASTER RESET button in conjunction with the SELF TEST button, an extensive self testing procedure is initiated.

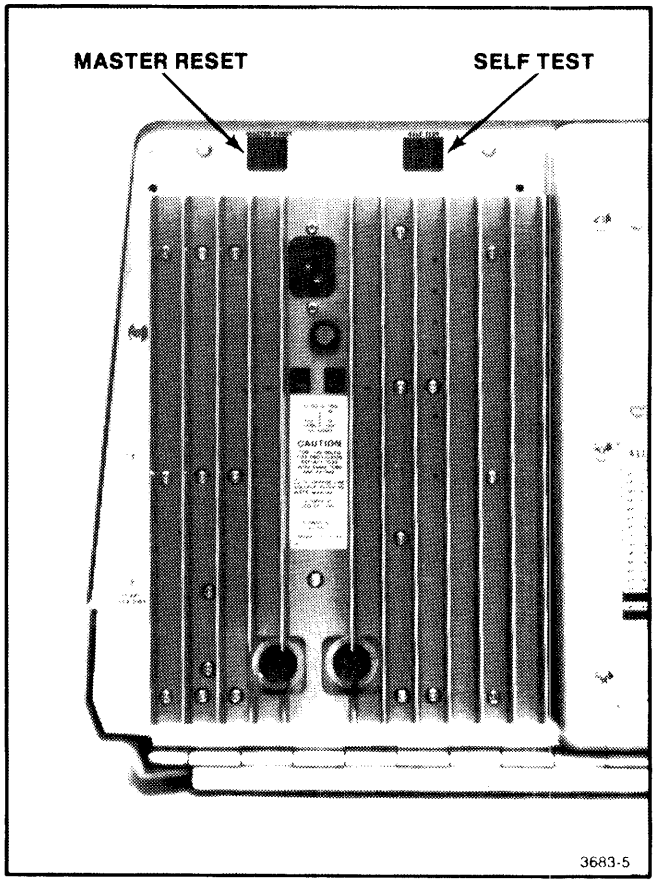


Figure 2-4. The MASTER RESET and SELF TEST Buttons.

SELF TEST

The power up sequence is an abbreviated test of the terminal's memory and circuitry.

A more extensive test of the terminal's operating conditions is initiated by the SELF TEST button (Figure 2-4). The SELF TEST thoroughly tests the terminal and allows a service technician to make any necessary adjustments. Self test is more fully explained in Appendix B.

The SELF TEST can also be used to reset all terminal parameters to their factory default values. See Appendix B for details.

WRITE PROTECT SWITCH AND INDICATOR

Your terminal may have a flexible disk drive (Option 42). If it does, the drive is located to the right of the display. There is a WRITE PROTECT switch for the disk drive. The switch is located in the panel above the drive (Figure 2-5).

If there is no disk in the drive, the red indicator light is always on. It cannot be turned off.

When there is a disk in the drive and you press the WRITE PROTECT switch toward the disk drive, the light next to the switch turns on. This means that the disk in the drive is "write protected." When a disk is write protected, the terminal will not write data on it. This prevents you from writing over (and thus destroying) data already on a disk.

Press the switch to the right to terminate the write protection feature (and turn off the light). You can also protect a disk by the PROTECT setup command (see Section 6) or by covering the "write protect" notch in the disk itself (see Section 3).

For additional details on the use of the WRITE PROTECT switch and information on use of the disk drive, see Section 3.

DISK DRIVE INDICATOR LAMP

When the red light in the disk drive door (Figure 2-5) is on, the drive is either reading from or writing to the disk. When the light is off, the drive is not operating. The use of the disk drive is explained in Section 3.

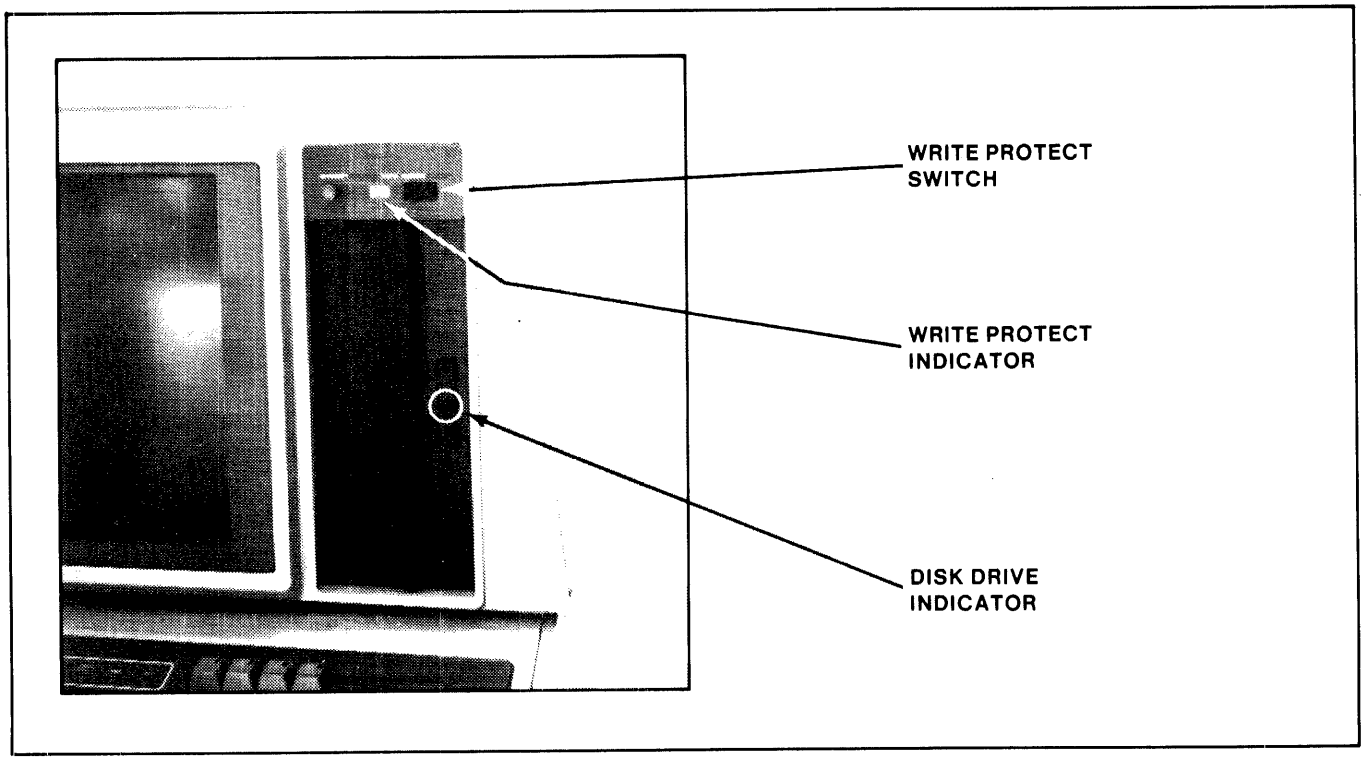


Figure 2-5. The WRITE PROTECT Switch, Indicator and Disk Drive Indicator.

CAUTION

Do not open the disk drive's door when the drive is operating (the red light is on). To do so interrupts the READ or WRITE in progress. Wait until the READ or WRITE is finished, then remove the disk.

INTENSITY

The brightness of the display is controlled by the INTENSITY knob (Figure 2-6). Turn the knob to the right to make the display brighter. Turn it to the left to make the display less bright; if you turn the knob too far to the left, text or graphics on the display becomes invisible.

KBD LOCK

When the KBD LOCK light (Figure 2-7) is on, the keyboard is "locked." A program can lock the keyboard to prevent you from typing anything at a time which would interfere with data being sent from the host.

When the keyboard is locked, you can still press the keys, but data is not transmitted, echoed, or queued. When you press any key (except BREAK or CANCEL) the terminal's bell rings to remind you that no data is being input.

The program should unlock the keyboard when the transmission of the critical data is complete.

You can also unlock the keyboard by pressing the MASTER RESET button or the CANCEL or BREAK keys. However, you should use MASTER RESET, CANCEL, and BREAK with caution. The MASTER RESET button deletes information in the terminal's memory. The CANCEL key terminates several terminal functions besides the locked keyboard. And the BREAK key may have undesirable effects on some host computers.

For further details, see the explanations of MASTER RESET, BREAK, and and CANCEL in this section and the LOCKKEYBOARD command in Section 6.

PAGE FULL

When the PAGE FULL light (Figure 2-7) is on, the terminal will not display any more text or graphics until the "page full" condition is cleared. The "page full" is cleared when you press any key except the SHIFT.

The terminal can be set up to make a hard copy of the screen, erase the screen, and continue automatically. See the PAGEFULL setup command in Section 6 for details.

XMT

The abbreviation "XMT" stands for "transmit." The transmit light (Figure 2-7) is on when data is being sent from the terminal to the host computer.

RCV

The abbreviation "RCV" stands for "receive." The receive light (Figure 2-7) is on when data is being received by the terminal from the host computer.

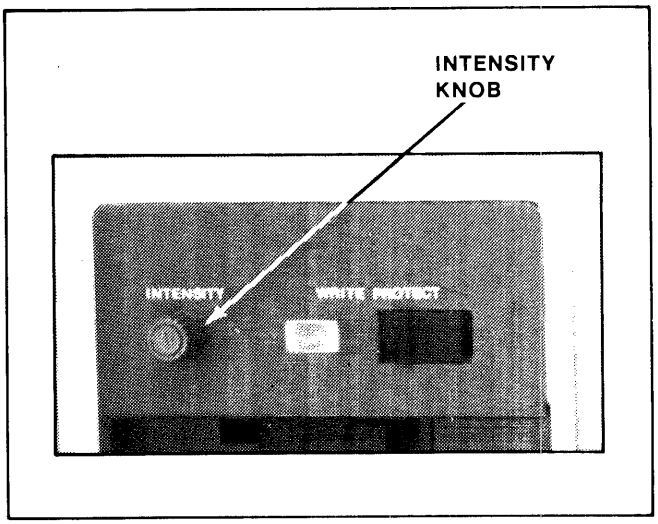


Figure 2-6. The INTENSITY Knob.

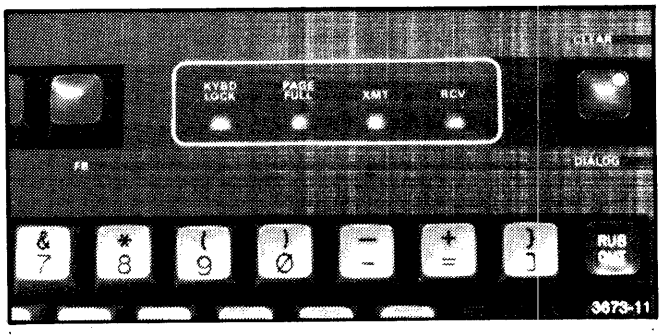


Figure 2-7. The KBD LOCK, PAGE FULL, XMT and RCV Indicator Lights.

THE THUMBWHEELS

The thumbwheels (Figure 2-8) have three uses:

- to scroll the dialog area.
- to locate a point on the screen to be sent to a graphics program being run from a host computer.
- to manipulate the framing box when the terminal is in Framing mode.

The dialog area is discussed later in this section, under the heading "DIALOG." Using the thumbwheels when the terminal is in graphics input (GIN) mode and to manipulate the framing box are discussed later in this section.

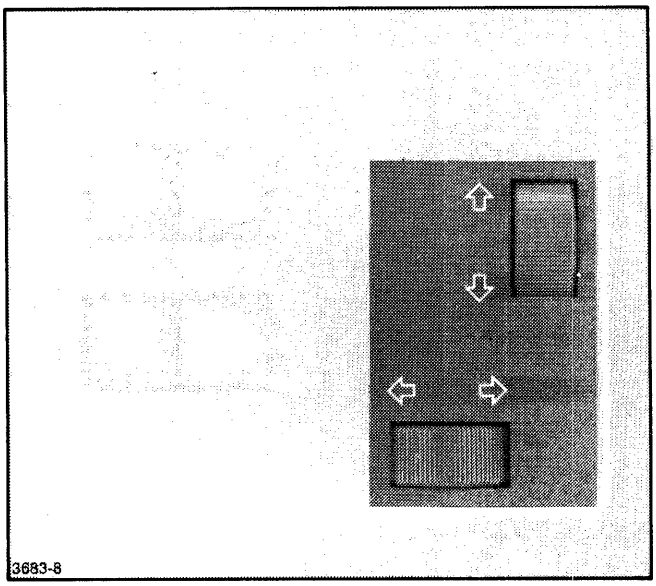


Figure 2-8. The Thumbwheels.

THE KEYBOARD

The keyboard can be grouped into six categories:

- Alphanumeric Keys
- Control Keys
- The BREAK Key
- Command Keys
- Function Keys
- Viewing Keys

ALPHANUMERIC KEYS

Most of the keyboard looks like a typewriter (Figure 2-9). The alphanumeric keys transmit the characters shown on their keycaps to the terminal or to a host computer. These are also called "printing" characters since they have symbolic representations which are "printed" (echoed) on the screen or printed by a line printer.

The "space" is a printing character.

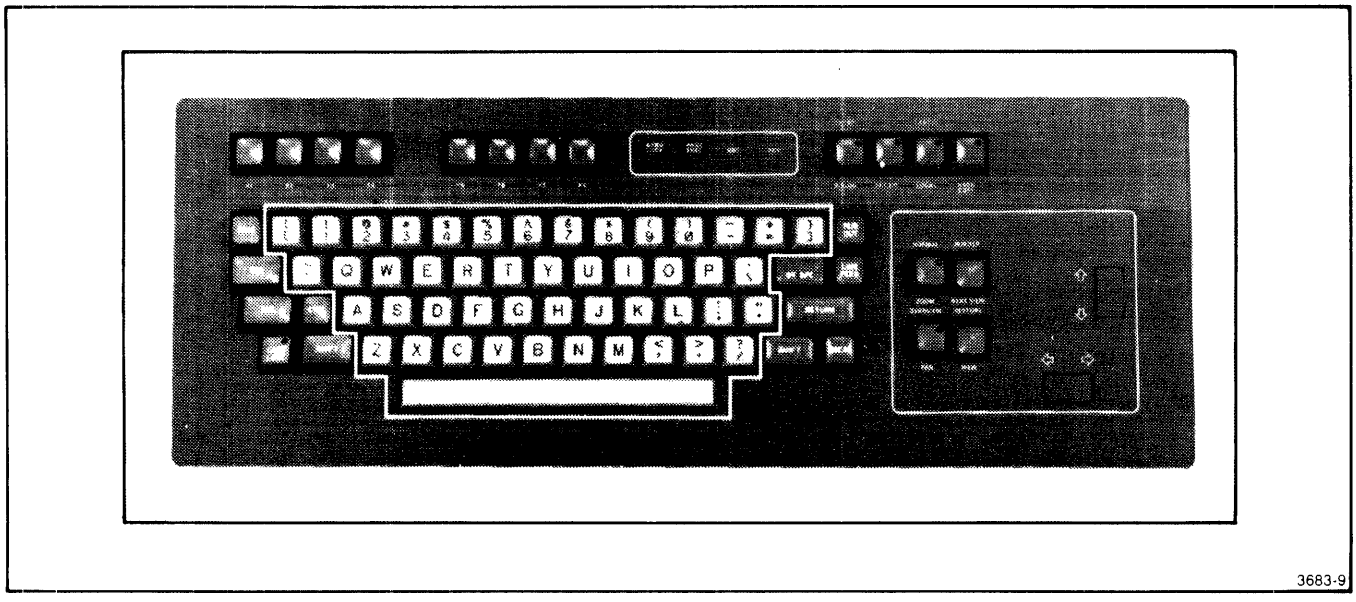


Figure 2-9. The Alphanumeric Keys.

NOTE

Four optional keyboards are available for international use. Descriptions in this section apply to the standard keyboard. There are variations if your terminal has an optional keyboard.

If you hold down an alphanumeric key for more than half a second, the character repeats at a rate of 10 times per second until you release the key or press another key.

When the terminal is in Local or Setup mode and you press an alphanumeric key, it is displayed (echoed) on the screen. Whether or not the characters are echoed when you are logged onto a host computer depends on whether the terminal or the host is providing the echo. See the ECHO command in Section 6 for details.

The character normally transmitted by an alphanumeric key can be replaced by a "macro" (a character or character string programmed into the key). This is explained later in this section under the heading "Function Keys."

CONTROL KEYS

The terminal can send and receive all standard control characters. A "control character" is a character that has a special meaning for the terminal or host computer. When some control characters are transmitted, you may not see them echoed or see their result on the display.

Several control characters have a specific meaning to the terminal. For example, the "carriage return" control character causes the cursor to return to the left margin; the "line feed" control character causes the cursor to move down one line.

There are six control characters which have their own keys (Figure 2-10):

- RUB OUT (delete)
- BK SPC (back space)
- LINE FEED (line feed)
- RETURN (carriage return)
- ESC (escape)
- TAB (horizontal tab)

The CTRL and SHIFT keys are included in the following description of the control keys because they are used to specify control characters which are not represented on the keyboard.

RUB OUT

The RUB OUT key sends the "rub out" (or "delete") character.

The RUB OUT is the factory default "character delete" key in Setup mode. See the explanation of the EDITCHARS command in Section 6.

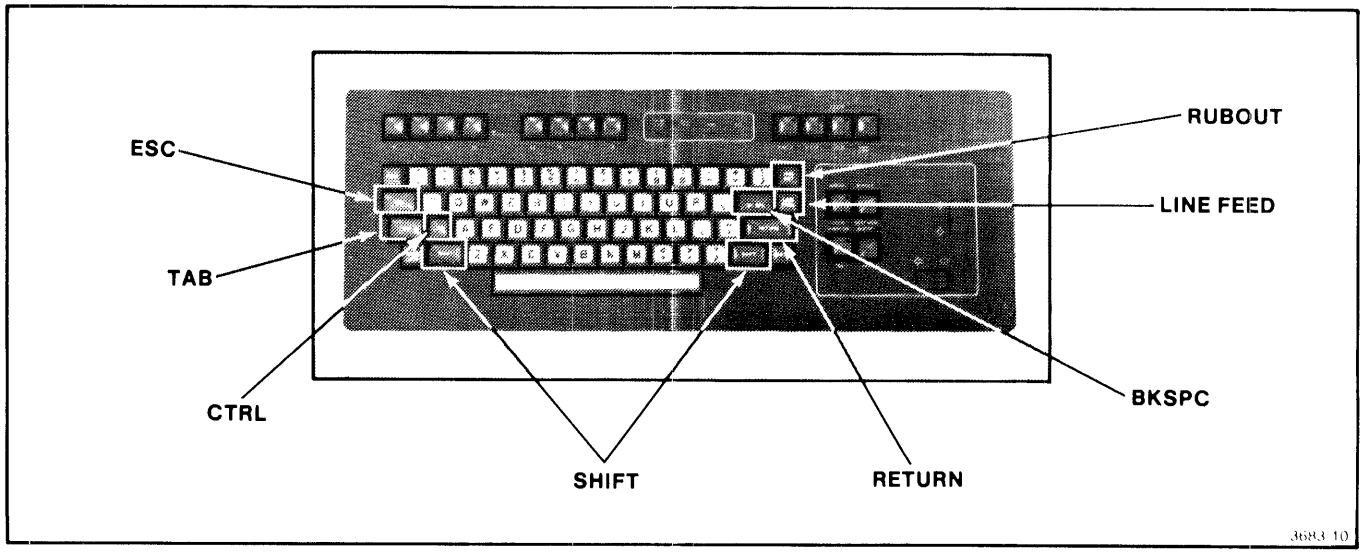


Figure 2-10. The Control Keys.

BK SPC

The BK SPC key sends the "back space" character, which moves the cursor one character position to the left.

If the cursor is already at the left margin when you press the BK SPC key, it moves to the right margin of the same line.

LINE FEED

This key sends the "line feed" character, which causes the cursor to move down to the next line without returning it to the left margin.

If the cursor is at the bottom of the screen and is not in the dialog area, a line feed moves it to the top of the screen or causes a "page full" condition (see PAGE-FULL in Section 6).

You can use the setup command LFCR to cause a line feed received from the host computer (or from the terminal when it is in Local mode) to cause the terminal to generate a carriage return (see Section 6).

RETURN

The RETURN key sends the "carriage return" control character, which returns the cursor to the left margin of the current line.

Some host computers automatically include a "line feed" when they echo or send the "return" character, so that pressing the RETURN key moves the cursor down one line and to the left margin. If your host does not, you can use the setup command CRLF (explained in Section 6) to cause the terminal to include a line feed with the return character.

When in Setup mode, the terminal automatically provides a carriage return-line feed when you press the RETURN key.

ESC

The terminal sends the escape character when it is communicating with a host computer and you press this key.

The "escape" is the first character in a special sequence called the "escape sequence." These are commands issued to the terminal in a special format in Local and Setup modes. See Appendix C for details.

For detailed explanations of all escape sequence commands, see the 4112 Host Programmer's Reference Manual.

TAB

The TAB key transmits the horizonal tab character.

Depending on how your host defines a tab, this character can cause the cursor to move a number of spaces to the right.

When the terminal is in Local mode, the cursor moves one character to the right each time you press the TAB key.

CTRL

Some control characters are used often and have their own keys, such as RETURN and LINE FEED.

However, you can also transmit these characters and all other control characters by holding down the CTRL key while you press another key.

For example, press the CTRL key and the "G" key (CTRL-G) at the same time to send the "bell" character and cause the terminal bell to sound; CTRL-M sends a "carriage return" character and causes the cursor to return to the left margin; CTRL-J sends the "line feed" character and causes the cursor to go down one line.

SHIFT

Most keys have two meanings: "unSHIFTed" and "SHIFTed."

When you press the key labeled "W" while holding the SHIFT key, the terminal sends the upper case character (W); pressing the same key without the SHIFT (if the CAPS LOCK key light is not on) sends the lower case letter (w). Some keys are labeled with two non-alphabetic characters, one above the other. For example, press the key labeled "5" to transmit a 5; press the same key and the SHIFT key to send the percent (%) character.

Programmable function keys can also have shifted and unshifted meanings, as explained elsewhere in this section under the heading "Function Keys." The command keys, (see "Command Keys" elsewhere in this section) have shifted and unshifted functions programmed into them as labeled on the keyboard.

The SHIFT key is also used at the same time as the CTRL key to send some control characters.

THE BREAK KEY

The BREAK key (Figure 2-11) can be used to "interrupt" communications from some host computers to the terminal. It is in a category by itself because it does not actually send a character to the host. It alters the state of the communications line such that the computer stops the sending of information to the terminal or stops the execution of a program.

The length of time that the interrupt lasts is important to some host computers. You can use the setup command BREAKTIME to specify the length of the interrupt sent by the terminal to the host, or to disable the BREAK key, as explained in Section 6.

It is also important that various host computers react differently when they receive the BREAK. One host might stop the execution of a program while another may log you off the host and break the communications link to the computer. Therefore, before using the BREAK key or BREAKTIME command, you should find out how the interrupt is handled by your host.

Pressing the BREAK key also unlocks the keyboard (see "KBD LOCK" earlier in this section.)

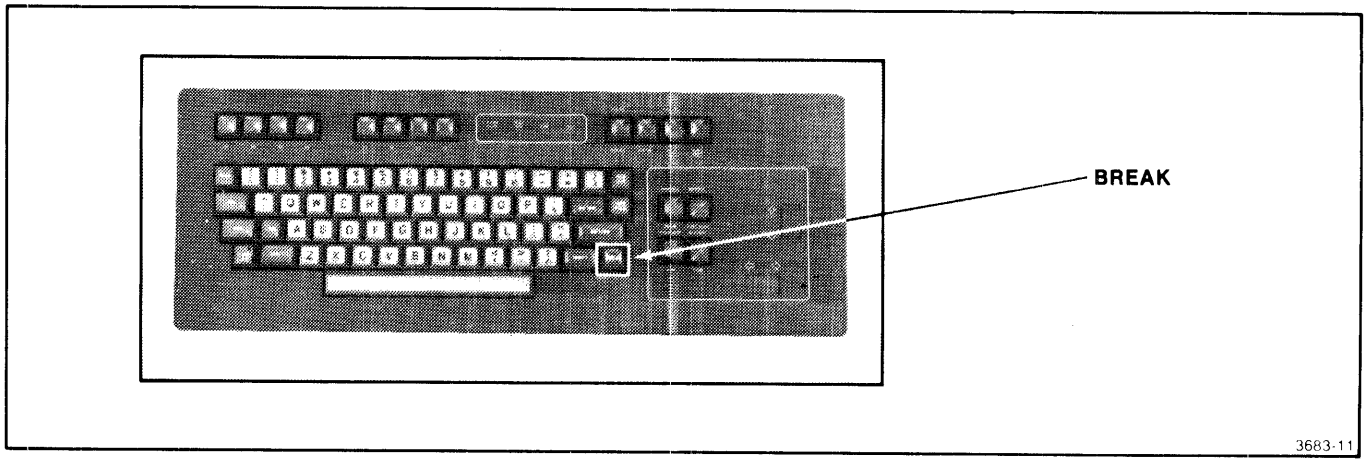


Figure 2-11. The BREAK Key.

COMMAND KEYS

There are six command keys (Figure 2-12) which perform special functions or put the terminal into a particular mode of operation. These keys cannot be programmed to do any other function. Their purpose is to allow you to easily access frequently used terminal features.

The functions performed by the command keys are discussed in the following order:

- PAGE
- CAPS LOCK
- SET UP
- DIALOG
- CLEAR
- LOCAL
- CANCEL
- COPY

PAGE

When you press the PAGE key, the following items are erased in the current view:

- graphics that are not part of a graphics segment (a "graphics segment" is a picture or part of a picture which can be manipulated by 4112 commands).
- text that is not part of a graphics segment and not in the dialog area.

The following are erased momentarily and then redrawn when you press the PAGE key:

- the dialog area.
- segments visible in the current view.
- the framing box.
- border around a viewport.

A segment is permanently erased by being deleted from the terminal's memory. A segment can also be made "invisible" (not erased from the terminal's memory, but not visible either). After a segment has been deleted or made invisible, you may need to press the PAGE key to erase its image from the screen, depending on how the terminal's FIXUP parameter is set (see Section 6).

If the dialog area is not enabled (see "DIALOG" elsewhere in this section) the following actions occur when you press the PAGE key. Because of these actions, it is generally best not to press PAGE while a program is running and the dialog area is not enabled. (These actions do not take place if the dialog area is enabled.)

- the cursor returns to the "home" position (the upper left corner of the screen).
- the terminal goes into alpha mode (that is, it interprets all incoming data as alphanumeric data, not graphic data).
- the linestyle is set to solid.
- 4010-style graphics input (GIN) mode is terminated.

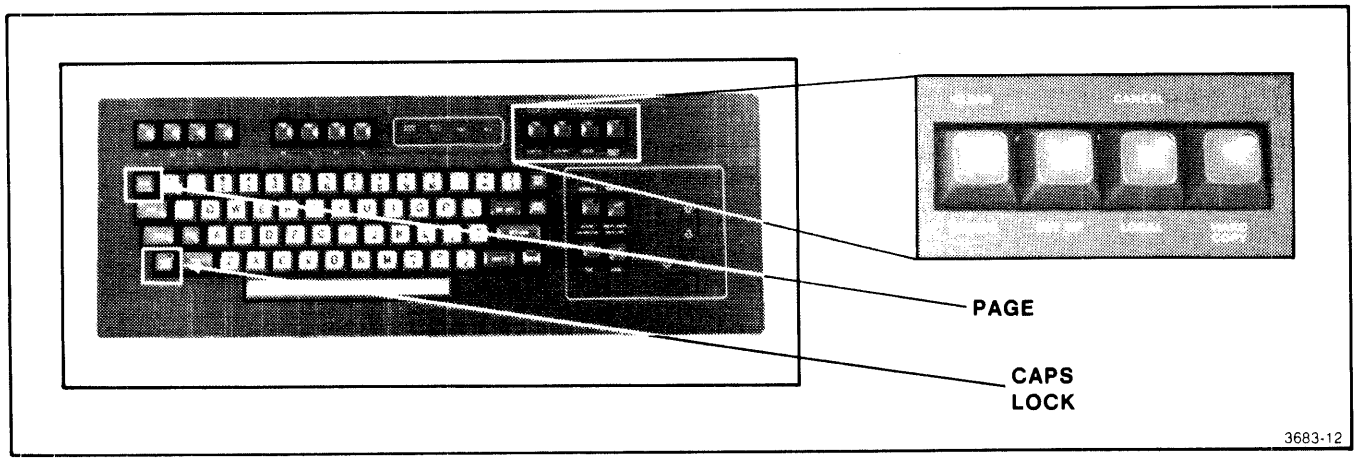


Figure 2-12. The Command Keys.

CAPS LOCK

The CAPS LOCK key is similar to the "Shift Lock" key on a typewriter.

When you press CAPS LOCK, the red light in the key comes on and alphabetic characters are transmitted in upper case. The CAPS LOCK does not cause the upper case version of numeric, command, or function keys to be transmitted.

The CAPS LOCK allows you to easily send only uppercase alphabetic characters to the host, but still allows you to conveniently send numeric characters rather than special symbols. This is especially useful on host systems which do not accept lowercase alphabetic characters.

SET UP

When you press the SET UP key, the light in the key turns on and the terminal enters Setup mode. (If the dialog area is enabled, the light in the DIALOG key also turns on. See the discussion of the dialog area later in this section.) With the terminal in Setup mode you can type English-like commands (called "setup" commands) directly to the terminal.

Setup commands allow you to establish the terminal's operating environment and to use peripherals attached to the terminal. For example, setup commands are used for such things as establishing communications rates, inquiring about the terminal's status, determining what kinds of errors the terminal should display, and programming special meanings (macros) into keys.

The results of many setup commands are stored in setup memory. Specifications stored in the setup memory are remembered even if the terminal is turned off, unplugged, plugged in, and turned on again.

See Section 6 for a description of Setup mode and all of the setup commands.

DIALOG

Part of the screen can be used to display your "conversation" (dialog) with the terminal and the host computer. This portion of the screen is called the "dialog area."

The text which can be written in the dialog area includes:

- the terminal's error messages.
- commands entered in Setup mode.
- text entered in Local mode (most commands entered in Local mode are not echoed on the screen).
- non graphics (alpha) text included in a graphics program.
- host generated messages, such as:
  - prompts
  - warnings
  - error messages
- normal terminal/host communications.

See the explanation of the CLEAR key later in this section for details on how to erase text from the dialog area.

Enabling the Dialog Area. Alpha text is directed to the dialog area when the dialog area is enabled. When the dialog area is disabled, alpha text is displayed with graphics. The DAENABLE command is used to enable or disable the dialog area. As illustrated in Figure 2-13, if the dialog area is enabled, the alpha text is directed to the dialog buffer. The dialog buffer can be thought of as a scroll, part of which is displayed at a time, at a specified screen location.

Whether or not the dialog area is enabled is stored in setup memory. For example, if you enable the dialog area and turn the terminal off, it will be enabled the next time you turn the terminal on.

4010-Compatibility. If a program is written to be run on the 4112, the terminal can distinguish between text which is part of the graphics ("graphics text" such as titles and labels) from text which is not part of graphics (non-graphics, or "alpha" text). Graphics text is displayed with the graphics. Alpha text is sent to the dialog area. Programs written for 4010 Tektronix terminals can also be run on the 4112, but special consideration must be made for text since those terminals do not have a dialog area.

For example, you can run a program on the 4112 which was originally written for a 4010 terminal. If the program includes text (such as a title or labels) and the dialog area is active, the 4112 would strip the title and labels out of the picture and display that text in the dialog area.

This is the reason for the DAENABLE command. When the dialog area is "disabled," the 4112 does not intercept text and display it in the dialog area, allowing a 4010-style program with text to execute on a 4112 just as on a 4010-family terminal.

Dialog Area Visibility. Even if the dialog area is enabled, it must also be made visible so you can see the text which is sent to it. If the dialog area is enabled but not visible, dialog is written in the dialog area but neither the text nor the cursor is visible.

The DIALOG key is used to make the dialog area visible or not visible. When the red light in the key is on, the dialog area is visible. When the red light in the key is off, the dialog area is not visible. (You can also use a setup command called DAVIS to make the dialog area visible or not visible, see Section 6.)

If the dialog area is enabled when you turn on the terminal, the light in the DIALOG key turns on, the dialog area is visible, and the cursor is displayed in the dialog area. If the dialog area is not enabled when you turn on the terminal, the dialog area is not visible and the cursor appears in the upper left corner of the screen.

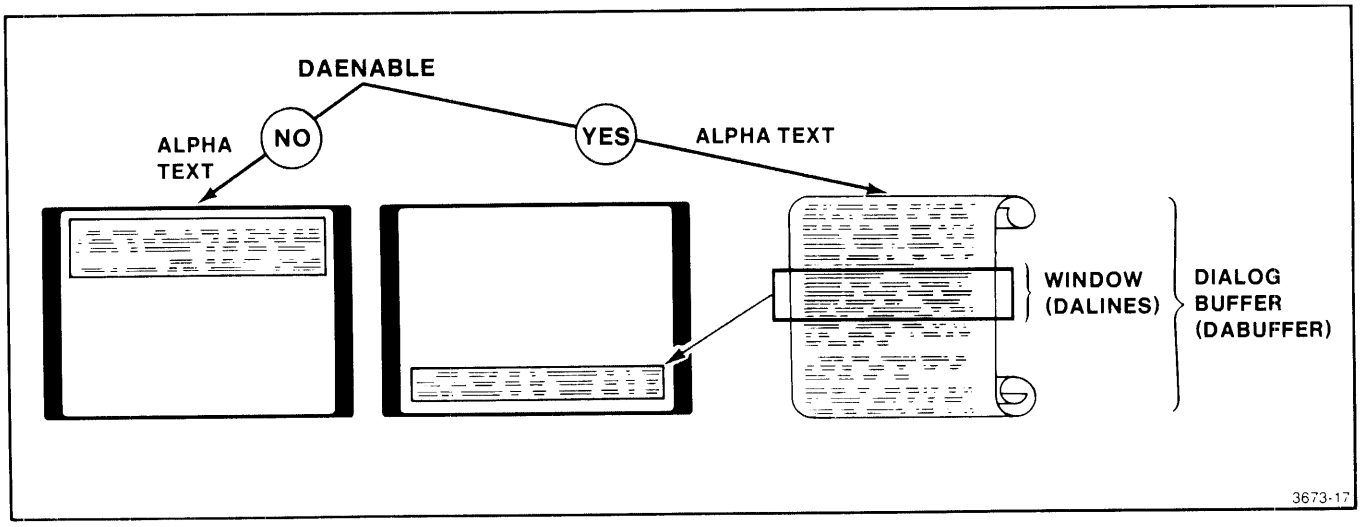


Figure 2-13. Enabling the Dialog Area.

Dialog Area Buffer. You can specify the total size of the dialog buffer with the DABUFFER command. This command allocates a certain amount of the terminal's memory for a buffer in which the dialog is stored. The factory default is 34 lines.

When the dialog buffer is full, the first line is deleted so another line of dialog can be added to the end.

The maximum length of each line is specified by the DACHARS command.

Visible Portion of the Dialog Area. It is not usually necessary or desirable to have the entire dialog buffer visible. The DALINES setup command specifies the number of visible lines in the dialog buffer. The factory default is five lines.

When the dialog area is visible you can use the top thumbwheel (Figure 2-8) to move (scroll) through the dialog area to observe its entire contents. As you "scroll," you are moving the "window" shown in Figure 2-13, so different lines in the dialog area are visible. The cursor stays on the line below the last line of dialog. This assures that information sent to the dialog area while you are examining it does not overwrite information already in the dialog area.

Line Length. The maximum length of each line in the dialog area is specified by the DACHARS setup command. The factory default is 80.

Dialog Area Writing Mode. The DAMODE setup command specifies whether a character in the dialog area is overwritten or replaced when you backspace over it. The factory default condition is "replace." That means that when the dialog area is enabled and visible and you backspace over a character, it is erased. The other setting is "overstrike." That means that when the dialog area is enabled and visible and you backspace over a character, the terminal writes over the character without erasing it.

A special "character delete" key is used to backspace in Setup mode. See the EDITCHARS command in Section 6.

Dialog Area Position. You can position the dialog area anywhere on the screen. The DAPOSITION command specifies the location of the lower left corner of the dialog area. The terminal positions the last visible line of the dialog area as close as possible to that location.

For example, the factory default position is the lower left corner of the screen (0,0). This means that the dialog area is located such that the last visible line (the fifth line, by factory default) begins at the lower left corner of the screen.

If you alter the configuration of the dialog area and move it, the location may not be exactly the one you specify. This is because the number of visible lines and length of each line may not permit the dialog area to be displayed at the location you specified.

Dialog Area Gray Scales. The DAINDEX setup command specifies the three indices (gray scales) in the dialog area:

- the "character index" specifies the gray index used for text in the dialog area.
- the "background index" specifies the gray index for the background of the dialog area.
- the "wipe index" specifies the gray index to which the dialog area is set when it is erased (wiped).

CLEAR

To erase the dialog area, press the SHIFT and DIALOG/CLEAR keys at the same time. This clears all text from the dialog buffer. The result is that the dialog area is erased. It does not erase any graphics on the screen (unless it is on the same surface) or affect whether or not the dialog area is visible.

LOCAL

When you press LOCAL, the red light in the key turns on and the terminal enters Local mode. If the terminal is already in Local mode, pressing the key turns the light off and terminates Local mode.

When in Local mode, communications to and from the host computer are temporarily suspended. Everything typed at the keyboard is interpreted by the terminal as if it came from the host. Information or data received from the computer is not processed; it is put into a queue and processed when you exit from Local mode.

Figure 2-14 illustrates the flow of communications. Notice that when the terminal is in Local mode, data transmission is routed back to the terminal instead of being sent to the host.

In Local mode you can enter commands ("escape sequences") from the keyboard in the same format that a host uses and the terminal executes them as if they came from the host.

If both Setup and Local modes are active, Setup takes priority.

CANCEL

Press the SHIFT-LOCAL keys at the same time to initiate a CANCEL operation. The CANCEL:

- halts all disk, host, and 3PPI file copy operations (except spooling, see Section 6).
- cancels "bypass" mode (see BYPASSCANCEL in Section 6).
- terminates graphics input (GIN) mode (see Section 4).
- unlocks the keyboard (LOCKKEYBOARD in Section 6).

If a file transfer is cancelled (see COPY in Section 6) the file is closed. All information in that file operation up to the time of the CANCEL command is preserved.

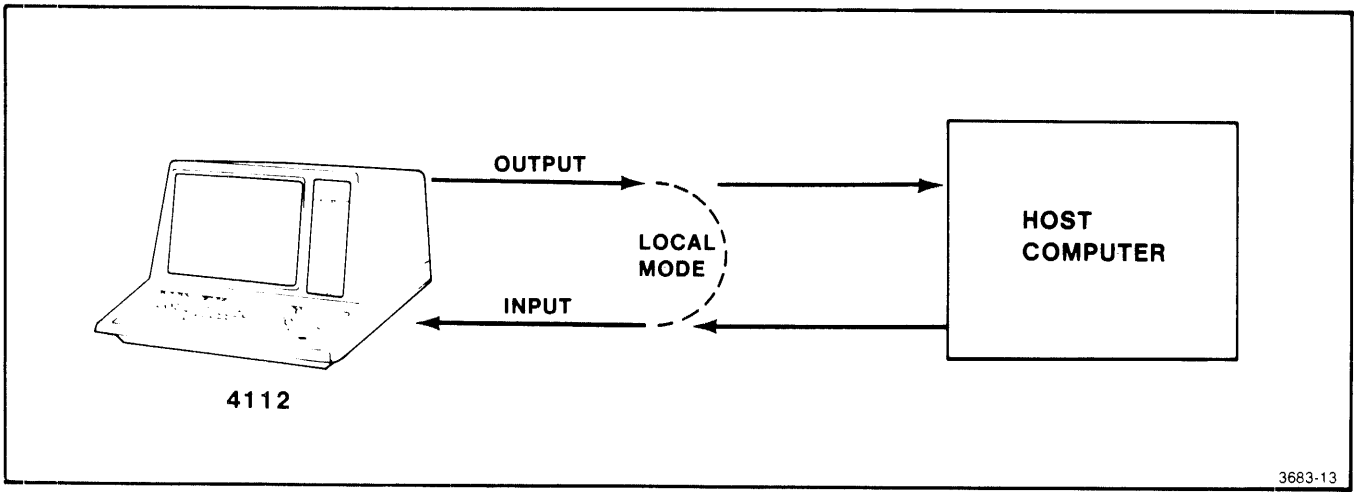


Figure 2-14. Communications Flow in Local Mode.

COPY

If you have a hard copy unit attached to the terminal, you can make hard copies of the display. You can copy the display either normally or in "inverse video."

A compatible hard copy unit must be attached to the terminal (see Section 3) and the terminal and hard copy unit must both be turned on. The hard copy unit may need to be on for a few minutes to warm up before it can make a good quality copy.

To copy normally, simply press the COPY key. If you are using the factory default gray indices, this copies black text and graphics on a white background.

To make an inverse video copy, press the SHIFT and COPY keys at the same time. If you are using the factory default gray indices, this copies white text and graphics on a black background.

The hard copy operation takes about ten seconds. Data sent to the terminal during the hard copy operation is queued until the hard copy is finished. Data transfer operations and other non-display terminal functions continue while the hard copy is being made.

FUNCTION KEYS

Most of the terminal's keys are programmable. This means that you can define the character or character string that is transmitted when you press a particular key. The character string or individual character assigned to a key is called a "macro."

The eight programmable function keys (Figure 2-15) transmit nothing unless they are programmed. The alphanumeric and control keys have characters assigned to them, but they can be assigned a macro instead.

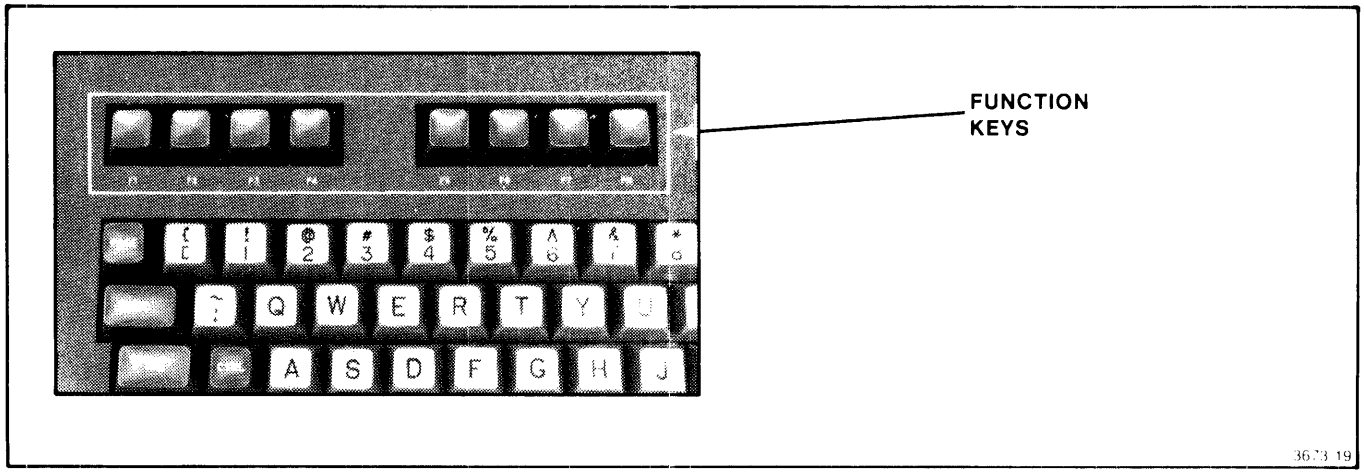


Figure 2-15. Programmable Function Keys.

Most keys can have two macros (shifted and unshifted). There is no shifted version of some keys, so they can have just one macro. A few keys are reserved for specific commands and cannot be programmed. Figure 2-16 illustrates keys which are not programmable.

The DEFINE setup command (see Section 6) is used to program a macro into a key. When you press a key that has a macro programmed into it, the key is "expanded" into the macro.

When the terminal is reset or turned off, macros are deleted from the terminal's memory. The DEFINE command can also be used to delete macro or return a key to its original meaning.

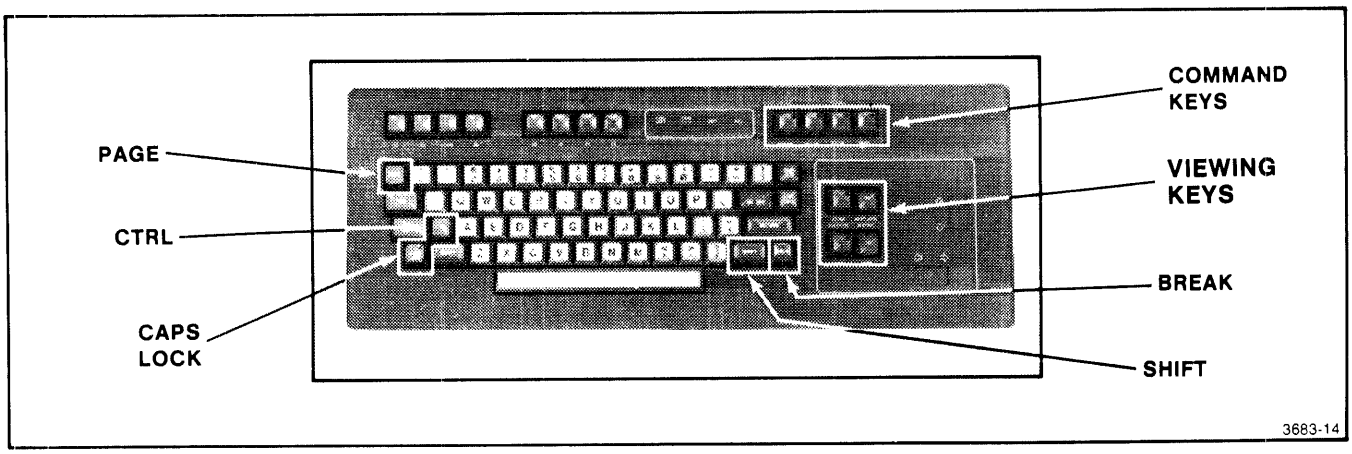


Figure 2-16. Non-programmable Keys.

THE VIEWING KEYS

The terminal's four viewing keys (Figure 2-17) allow you to examine the graphics display in more detail.

The PAN and ZOOM keys put the terminal into Framing mode. There are several additional functions, as indicated by the labels on the other viewing keys: VIEW, OVERVIEW, NORMAL, NEXT VIEW, RESTORE, and BORDER.

The terminal's display can be divided into as many as 64 separate viewports. When there is more than one viewport, graphics actions (such as graphics input and Framing mode) take place in the viewport which is "currently active." The currently active viewport can be selected by program control or by pressing the terminal's NEXT VIEW key.

NOTE

If the bell rings when you press any of the viewing keys, they have been disabled by the terminal's "lock viewing keys" comand. To "unlock" the viewing keys, put the terminal into Setup mode and enter the following:

<EC>RJ 0 <CR>

Framing Mode

When you press the ZOOM or PAN keys, the terminal enters Framing mode. In Framing mode, a framing box appears at the edge of the currently active viewport. You can change the location or shape of the framing box with the thumbwheels. The box frames the portion of the view you want to see in more detail.

The graphics displayed on a 4112 is defined in an area 4096 horizontal units by 4096 vertical units. The screen normally displays 4096 x 3127 units (a 4:3 aspect ratio).

The screen does not display the full resolution of the 4096 x 3127 space. Therefore, you can "zoom" into a picture and see more detail. As you zoom into a picture, you can also move across or up and down to identify a smaller portion of it (the pan function).

The ZOOM Function

If the light in the ZOOM key is off, when you press the ZOOM key the light turns on and the terminal enters Framing mode with the zoom function active. When the zoom function is active, the lower left and upper right corners of a rectangle appear in the framing box.

When you rotate the top thumbwheel up or the bottom thumbwheel to the right, the height and width of the box are increased. When you rotate the top thumbwheel down or the bottom thumbwheel to the left, the height and width of the box are decreased. In both cases, the framing box maintains its current aspect ratio.

The framing box represents the proposed new view (window). When you make the box smaller, you are zooming in for a closer look. When you make the box bigger you are zooming out for an enlarged view.

When you have positioned the box where you want it, press the VIEW key to update the display. The viewport is erased and the portion of the window within the framing box is displayed in the viewport. The framing box again appears at the edge of the viewport.

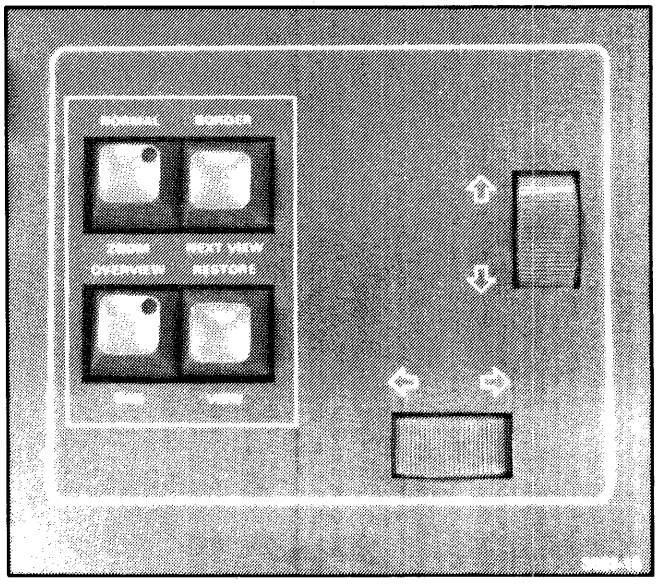


Figure 2-17. The Viewing Keys.

You can zoom in to 1/16th the size of the current window. If you want to zoom in more than that, press the VIEW key to update the viewport and then zoom in on that view.

You can zoom out on a view until it is about eight times its current size. If you want to zoom out more, press the VIEW key to update the viewport and then zoom out on that view.

If you press the SHIFT key while you rotate the thumbwheels, the box changes shape more slowly, allowing you to more precisely change the window.

See the explanation of the RESTORE key for information on restoring a previous view.

If you press the CTRL key while you rotate a thumbwheel, you alter the aspect ratio of the framing box. Press the CTRL key while you rotate the top thumbwheel to change the height of the box while not affecting its width. Press the CTRL key while you rotate the bottom thumbwheel to change the framing box's width while not affecting its height. Press the SHIFT and CTRL keys while you rotate a thumbwheel to change the aspect ratio more slowly.

See the explanation of the NORMAL key for information on restoring the original aspect ratio.

Zooming occurs in the currently active viewport. To zoom in on another viewport, use the NEXT VIEW key as explained later in this section.

Terminate the zoom function by exiting from Framing mode or by making the pan function active. When the light is on in the ZOOM key, you exit from Framing mode by pressing the ZOOM key. The light in the key turns off as Framing mode is terminated. Make the pan function active by pressing the PAN key. The light in the ZOOM key turns off and the light in the PAN key turns on.

You can use both the zoom and pan functions before you update a view. For example, you could press the ZOOM key and zoom in, then press the PAN key to pan across a view, then press ZOOM again, and so on until you have the framing box located correctly. Then press the VIEW key.

The PAN Function

If the light in the PAN key is off, when you press it the light turns on and the terminal enters Framing mode with the pan function active. A dashed-line box appears at the edge of the currently active viewport, with a cross in the center of the box.

When you rotate the top thumbwheel up or down, the framing box moves up or down. When you rotate the bottom thumbwheel left or right, the framing box moves to the left or right. The position of the box represents the proposed new view (window).

When you have positioned the box where you want it, press the VIEW key to update the display. The viewport is erased and the new view, as specified by the framing box, is displayed. The framing box again appears at the edge of the viewport.

You can pan until an edge of the framing box reaches the edge of the terminal (4096 x 4096) space or until only about 1/8 of the framing box is still in the current view, whichever happens first. If you have not reached the edge of the viewport and you want to pan more, press the VIEW key to update the current view and pan across that view.

If you press the SHIFT key while you rotate a thumbwheel, the framing box moves more slowly so you can locate it more precisely.

You can pan in the currently active viewport. To pan in another viewport, use the NEXT VIEW key as explained later in this section.

Terminate the pan function by exiting from Framing mode or by making the zoom function active. When the light is on in the PAN key, you exit from Framing mode by pressing the PAN key. The light in the key turns off as Framing mode is terminated. Make the zoom function active by pressing the ZOOM key. The light in the PAN key turns off and the light in the ZOOM key turns on.

You can use both the Zoom and Pan sub-modes before you update a view. For example, you could press the ZOOM key and zoom in, then press the PAN key to pan across a view, then press ZOOM again, and so on until you have the framing box located correctly. Then press the VIEW key.

The VIEW Key

As described with the ZOOM and PAN keys, the VIEW key is used to update the view described by the framing box.

If you press the CTRL and VIEW keys at the same time, the view described by the framing box is displayed in the viewport with the next higher identifier (see the NEXT VIEW key). However, the framing box remains active in the current viewport.

For example, you may have two viewports defined on the screen. One large one containing a large picture, and a smaller viewport. You can zoom into a small portion of the large picture and press the CTRL-VIEW keys. The specified detailed view is displayed in the smaller viewport. This provides a detailed view as specified, but also leaves an overview of the larger picture visible. The framing box continues to be active in the current (larger) viewport.

The OVERVIEW Key

When you press the OVERVIEW key (SHIFT and PAN keys at the same time) the entire amount of terminal space normally displayed (4095 x 3127) is made visible. If the pan or zoom function is active, it remains active and the framing box is visible.

If you press the CTRL and OVERVIEW keys (CTRL-SHIFT-PAN) at the same time, the entire 4095 x 4095 terminal space is displayed. (Notice that this changes the aspect ratio from its normal 4:3 to a 4:4 ratio.) If the pan or zoom function is active, it remains active and the framing box is still visible. To restore the normal (4095 x 3127) view and aspect ratio, press the OVERVIEW key.

See the explanation of the RESTORE key for information on how to restore the original view.

The NORMAL Key

As included in the explanation of the Zoom sub-mode, you can change the aspect ratio of the framing box by pressing the CTRL key while rotating a thumbwheel.

If you then press the VIEW key, the view is updated with the same aspect ratio as the framing box. To cause the framing box to return to its normal (4:3) aspect ratio, press the NORMAL key (the SHIFT and ZOOM keys at the same time).

The NEXT VIEW Key

When the display contains more than one viewport, the terminal internally identifies each viewport with an integer value. You can make the view with the next larger identifier active by pressing the NEXT VIEW key.

You can make the view with the next smaller identifier active by pressing the CTRL key at the same time as the NEXT VIEW key. When a viewport becomes visible, a border around that viewport blinks once to indicate it is active.

You can then use the Framing mode and viewing key functions in that viewport.

The RESTORE Key

The terminal remembers the last three views for each viewport, as well as the view before you change it with any of the viewing keys (that is, the view as it was defined by the last "window" command as explained in the 4110 Series Command Reference Manual). To restore the most recent view, press the RESTORE key. To restore the second most recent, press it again. Press it a third time to restore the third most recent view. Since the three most recent views are saved, this list of views changes every time you press the VIEW key. For example, when you press the VIEW key:

- The third most recent view is deleted.
- The second most recent view becomes the third most recent.
- The most recent view becomes the second most recent.
- The current view becomes the most recent.
- The contents of the framing box becomes the current view.

If you press RESTORE a fourth time, the terminal restores the view as it was before you altered it with a viewing function. This original version of the view is not replaced when you press the VIEW key, as the three most recent views are. This means that you are always able to return the view to its original condition.

You can immediately restore the view as it was before you altered it with a viewing function by pressing the CTRL and RESTORE keys at the same time.

If you press the RESTORE key a fifth time, you see the most recent view again. If you press it a sixth time you see the second most recent view again, and so on.

The BORDER Key

You can cause the terminal to draw a border around the current viewport by pressing the BORDER key. If that viewport already has a border, pressing the key deletes the border.

Section 3

HARDWARE OPTIONS AND PERIPHERALS

This section explains the use of the disk drive and three port peripheral interface (3PPI) options to the 4112. It also explains how to attach the optional graphics tablets to the terminal. Since the tablets are always used for graphics input, their use is explained in Section 4.

This section also includes instructions on the use of the following TEKTRONIX peripheral devices with the terminal:

- 4632 and 4612 Hardcopy Units
- 4634 Imaging Hardcopy Unit
- 4662 and 4663 Plotters
- 4923 Tape Drive

THE DISK DRIVE

Figure 3-1 shows a 4112 terminal with a flexible disk drive built into it (Option 42). The disk drive is used for local storage of segments, macros, text fonts, and general text data.

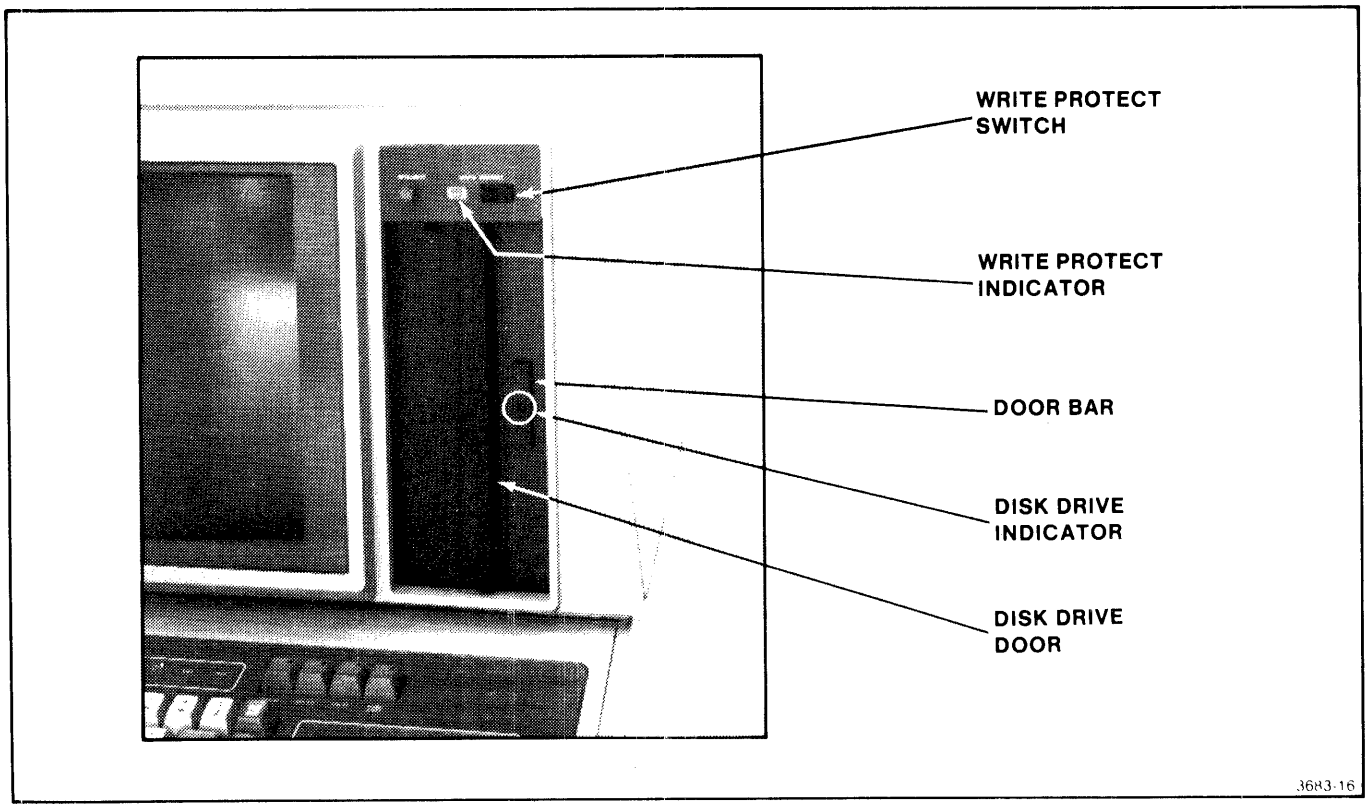


Figure 3-1. The Flexible Disk Drive.

HOW TO INSERT A DISK

Figure 3-2 illustrates the front and back sides of a flexible disk.

To insert a flexible disk, open the door to the drive by pushing the disk drive's door bar to the left (Figure 3-1). Insert the disk notched edge first, with the label to the left (Figure 3-3). Slide the disk in enough that it stays, but not hard enough to bend the disk. Push the door to the right to close it.

Remove the disk when you are through using it by pushing to the left on the door bar. The door will open and your disk will pop out.

CAUTION

Do not remove a disk while the light in the disk drive's door is lit; it indicates that the disk is in operation. Removing an operating disk halts the operation in progress, and may damage or destroy your data.

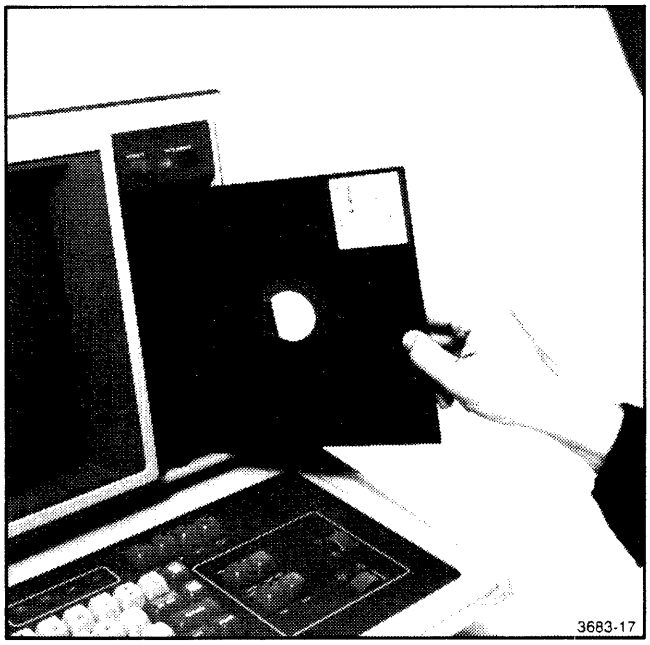


Figure 3-3. Inserting a Flexible Disk.

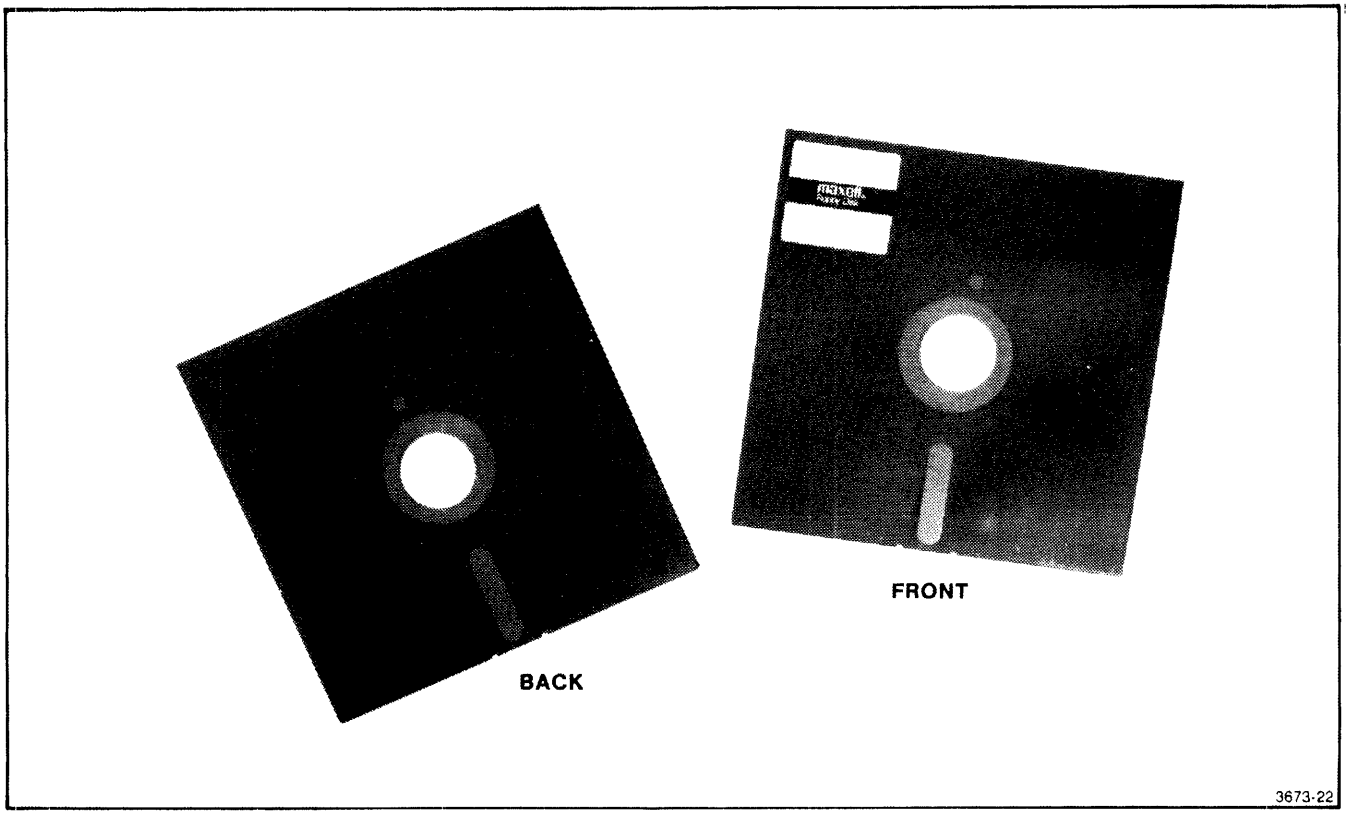


Figure 3-2. Flexible Disks.

HANDLE THE DISKS WITH CARE

Flexible disks are easy to store and handle. It is important to take a few simple precautions to insure the integrity of your data.

Do not bend flexible disks intentionally since they crease and tear, which destroys the data stored on them. Oil, dirt, and dust on the disk cause problems when the disk head tries to read the disk, so touch only the paper sleeve when handling the disk. Always keep the disks in jackets when not in use. Disks should be stored in a place protected from dust, extreme heat (above 122°F, 50°C) and extreme cold (below −40°F, −40°C).

If you are storing critical data on the disk, it is always safest to back up the data with a second copy.

The disk should last for about 3,500,000 read/write operations per track or 30,000 insertions, whichever comes first. When the disk begins to wear out you will see translucent rings around its edges. You will also notice repeated errors when you try to read from or write to the disk. When this occurs, discard the disk and use the backup.

WRITE PROTECTING A DISK

There are three ways to protect a disk or files on the disk from being inadvertently deleted or overwritten by different data.

The WRITE PROTECT switch (Figure 3-1) on the disk drive protects the entire disk. When the write protect feature is active, the red light in the switch is on and you cannot format, delete a file from or add a file to a disk in that drive. An error results whenever you attempt to do so.

You can also use the setup command PROTECT to protect an entire disk or individual files on the disk. Using the PROTECT command does not prevent you from reformatting a disk; it only prevents you from writing to or copying from a disk. See the explanation of the PROTECT command in Section 6.

Some disks also have a "write protect" notch in them. If the disk you use has a write protect notch, you cannot write to the disk when that notch is exposed. Before writing to the disk, you must cover the notch with a piece of opaque tape.

USING THE DISK DRIVE COMMANDS

The following brief example illustrates the use of a disk drive. It includes the use of the FORMAT, SAVE, DIRECTORY, LOAD, PROTECT, and DELETE commands. This example also uses the DEFINE command to create three macros, which are then saved to a disk file. You may need to read the explanation of DEFINE in Section 6 prior to doing the example.

All of the commands related to the disk drive are briefly presented in Table 3-1, following this example.

Insert a disk, following the directions under the heading "How to Insert a Disk" earlier in this section. Be sure that the disk does not contain information that should be saved.

Put the terminal in Setup mode. If the light in the SET UP key is not on, press the key. If it is already on, the terminal is already in Setup mode.

Be sure the disk is not write protected. Toggle the WRITE PROTECT switch to the left position (the red light in the switch should be off). If the disk has a write protect notch, be sure it is covered with a piece of opaque tape. Enter the following setup command to assure that the disk has not been protected by the terminal:

PROTECT F0: no $^{C}_{R}$

FORMAT F0: <name> <size>
No space

Enter the following command to "format" the disk — prepare it for the creation of files:

FORMAT PRACTICE 368 ^{C_R}

A disk must be formatted the first time it is used, to prepare the disk for storing files. Do not format a disk that has already been formatted unless you want to destroy all of the files on that disk and start over.

While the disk is being formatted, the red light in the disk drive's door comes on indicating that the disk drive is operating. It takes about 60 seconds to format the disk. Continue with the next step in the example when the red light goes out. (Do not open the disk drive's door while the disk is being formatted.)

The parameters indicate that the disk in drive 0 should be named "PRACTICE" and that the maximum number of files allowed on the disk is 368.

Program macros into keys F1, F2, and F3 with the following DEFINE command. Remember that the "literal" character must precede the carriage return which is included in the macro. In this example the tilda (~) is the literal character. It is shown above the ^{C_R} because when you use it the cursor backspaces and the ^{C_R} types over it. See the DEFINE and EDITCHARS commands in Section 6 for details.

DEFINE F1 /STATUS $\widetilde{C_R}$ / ^{C_R}

DEFINE F2 /STATUS DIALOG $\widetilde{C_R}$ / ^{C_R}

DEFINE F3 /STATUS BAUDRATE $\widetilde{C_R}$ / ^{C_R}

PAGEFULL STOP ^{C_R}

Press F1, F2, and F3 to be sure they are programmed. A general status report, the status of the dialog area and of the BAUDRATE command should be displayed. When the "page full" condition occurs, press any alphanumeric key or PAGE to continue.

Save the macros onto the disk with the following command:

SAVE MAC ALL TO KEYS ^{C_R}

This command saves all of the macros currently defined. The file on the disk in drive 0 that the segments are saved to is named "keys."

Display a directory of the disk with the following command to be sure that the file was created. A directory similar to the one in Figure 3-4 is displayed.

DIRECTORY ^{C_R}

Press the terminal's RESET button (located on the back of the terminal). The terminal goes through the power up procedure. Among other things, this deletes the macros currently programmed into the keys.

Press function keys F1, F2, and F3. The terminal takes no action because the macros have been deleted.

Press the SET UP key to put the terminal back into Setup mode.

Load the macros into the terminal from the disk by entering the following command:

LOAD KEYS ^{C_R}

The red light in the disk drive door turns on indicating that the disk is operating. When it finishes loading the specified file, the light in the door turns off.

Press F1, F2, and F3. The macros which you saved have been programmed back into the function keys by the LOAD command.

```
*DIRECTORY
-NAME          BLOCKS BYTES PROTECT
F0:PRACTICE      2002           NO
   KEYS             1     90    NO
-
-ENTRIES USED:      1
-ENTRIES FREE:    367
-BLOCKS USED:      27
-BLOCKS FREE:    1975
-LARGEST FREE:   1975
*_
```

3683-18

Figure 3-4. A Disk Directory.

Delete "KEYS" from the disk by entering the following command:

DELETE KEYS ^{C_R}

Enter the DIRECTORY command to be sure that the file was deleted:

DIRECTORY ^{C_R}

The directory should no longer include the file named "KEYS."

Enter the following command to restore the default PAGEFULL condition:

PAGEFULL NONE ^{C_R}

DISK DRIVE COMMANDS

Table 3-1 briefly introduces all of the terminal's setup commands relating to the operation of the disk drives. The abbreviation (3PPI) appears next to those commands which are also used by the three port peripheral interface, which is explained later in this section. See Section 6 for detailed explanations of all setup commands.

Table 3-1

THE DISK DRIVE COMMANDS

Command	Explanation
COPY (3PPI)	Copies data from one device to another.
DELETE	Deletes a specified file from a disk.
DIRECTORY	Displays a directory of an entire disk or individual files.
FORMAT	Initializes a disk so data can be stored on it in files.
LOAD	Reads data from a disk and executes it as if the data were a command file.
PLOT (3PPI)	Reads information from the terminal's screen and sends it to a specified output device.
PROTECT	Write protects or unprotects files on a disk.
RENAME	Assigns a new name to a file on a disk.
SAVE	Creates a file on a disk containing a graphics segment or special text font.
SPOOL (3PPI)	Copies data from one device to another, while still allowing the terminal to proceed as usual.
STOP	Halts the spooling operation.

THE THREE PORT PERIPHERAL INTERFACE

The 3PPI, Option 10, allows the terminal to communicate with peripheral devices. With the 3PPI installed (Figure 3-5), the terminal supports most devices which communicate through an RS-232 interface.

The TEKTRONIX 4662 and 4663 plotters are explicitly supported by the 3PPI.

Table 3-2 briefly introduces the commands associated with the terminal's 3PPI.

USING THE 3PPI

To use the 3PPI, plug the RS-232 cable from the device into one of the ports in the interface. You then use a setup command to assign an appropriate device driver to that port.

For example, you could connect a 4663 plotter to port 0 of the 3PPI. You would put the terminal into Setup mode and enter the following command:

PASSIGN P0: 4663 $^{C}_{R}$

Table 3-2

THE THREE PORT PERIPHERAL INTERFACE COMMANDS

Command	Explanation
PASSIGN	Assigns a device driver to a port.
PBAUD	Specifies the baud rate for a port.
PBITS	Specifies the stop bits and data bits for a port.
PCOPY	Establishes a two-way communication path between two devices.
PEOF	Specifies the end of file string for a port.
PEOL	Specifies the end of line string for a port.
PFLAG	Establishes the flagging convention used by a port.
PORTSTATUS	Displays a report on the specified port.
PPARITY	Establishes the parity convention used by a port.

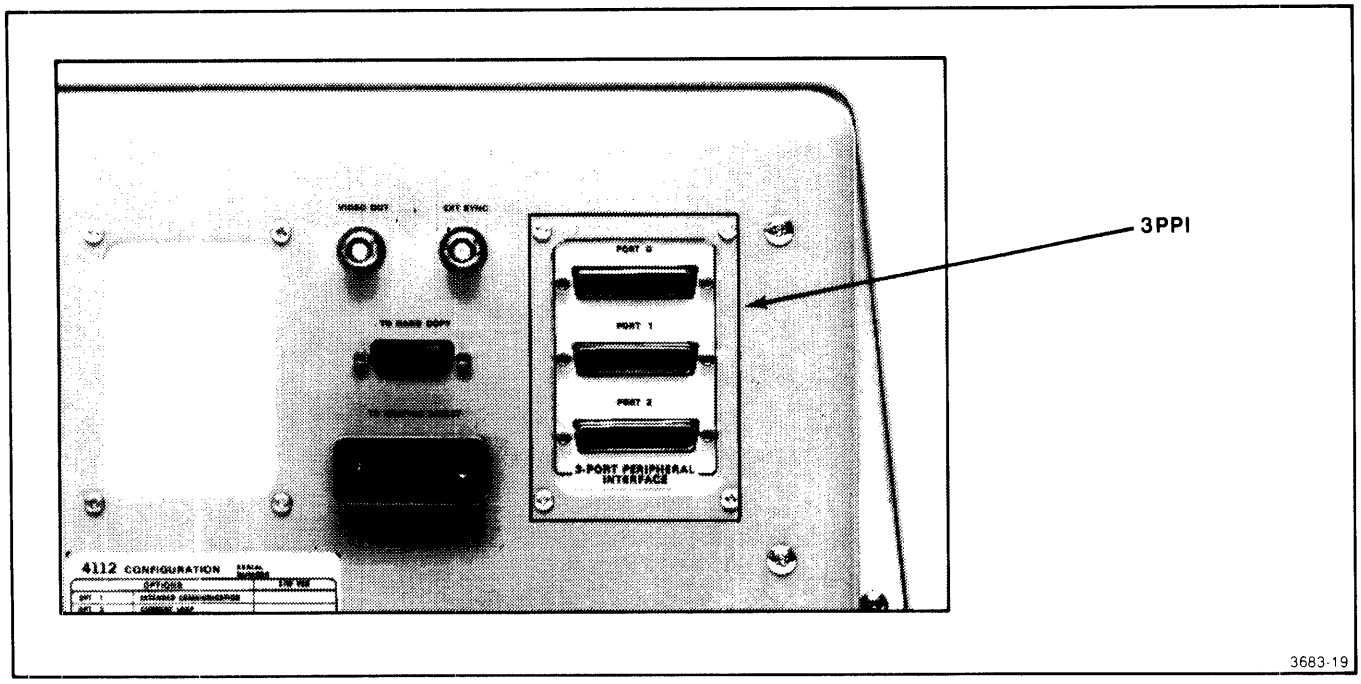


Figure 3-5. The Three Port Peripheral Interface (3PPI).

This command assigns the terminal's 4663 "device driver" to peripheral port 0. The device driver translates terminal commands into 4663 commands when the terminal sends data to the plotter, and 4663 commands into terminal commands when the plotter sends data to the terminal. See the PASSIGN command in Section 6 for more complete information.

The 3PPI supports any device with an RS-232 connector with its "general device driver." The general device driver does not translate commands from one format to another.

In addition to assigning a device driver to a port, you must also be sure that communications parameters are set correctly. It is important for correct data transmission that the parameters assigned to the port agree with the parameters of the device attached to the port. The parameters and the setup commands used to set them are:

- baudrate (PBAUD)
- flagging (PFLAG)
- stop bits (PBITS)
- data bits (PBITS)
- parity (PPARITY)
- end of line string (PEOL)
- end of file string (PEOF)

For full information on the above commands please see Section 6, the Command Reference section.

ATTACHING THE 4662 AND 4663 PLOTTERS

Plug the male end of an RS-232 cable into a 3PPI port and the female end into the MODEM port on the back of the plotter.

The plotter is often connected between the host port and the terminal. However, if you wish to use all the capabilities of the terminal, connect the plotter to the terminal's 3PPI port and let the terminal drive the plotter.

When the terminal and plotter are connected, you must make sure that the parameters are set properly. Use the 4662 and 4663 Plotter Operator Manuals for information on how to set the communications parameters.

ATTACHING THE 4923 OPTION 1 TAPE DRIVE

You can attach a TEKTORNIX 4923 Option 1 Tape Drive to a 4112 peripheral port, using the terminal's general device driver.

Plug the RS-232 cable into the MODEM port on the 4923 and into one of the peripheral ports on the terminal. If it is plugged into peripheral port 0, for example, you would use the following command:

PASSIGN P0: PPORT $^{C}_{R}$

Use the PBAUD command to match the port's baud rate to the tape drive's, as indicated by the 4923's baud rate control knob on the back of the tape drive.

Use the terminal's PFLAG command to set character flagging to DC1/DC3. All other communicatons parameters should be set to match the conventions used in the files on the tape.

When the terminal is receiving data from a non-file-structured device, such as the 4923, you can indicate the "end-of-file" by pressing the terminal's CANCEL key after all data has been transferred. This erases the last received data which would otherwise be left inside the 3PPI system.

THE GRAPHICS TABLETS

The two graphics tablets are Options 13 and 14 to the terminal.

Option 13 is 11" x 11" and Option 14 is 30" x 40". Except for their size, the two tablets are identical. In this manual they are both referred to as "the graphics tablets" or "the tablets" since operating instructions are the same for both.

This manual assumes that the hardware for the tablet is installed and that it is ready for use. If that is not the case, a qualified technician should follow the installation instructions included in the 4110 Series F13/14 Graphics Tablet Instruction Manual.

Figure 3-6 shows the smaller graphics tablet. Both tablets include:

- the tablet (with cables connected to it)
- the tablet interface
- the input devices (one or more of the following):
  - a stylus (standard)
  - a one button cursor (optional)
  - a four button cursor (optional)
- a bar magnet for restoring the tablet's bias
- a precision grid (optional)

Table 3-3 lists the terminal's setup commands which deal specifically with the operation of the tablet. See Section 6 for more details.

Table 3-3

TABLET SETUP COMMANDS

Command	Explanation
TBFILTER	Specifies the time and distance filtering the tablet is to use for strokes in Graphics Input mode
TBHEADERCHARS	Specifies the header characters used by the tablet when it sends data to the host program via 4010-style GIN.
TBSTATUS	Specifies whether a STATUS byte is included in GIN reports when using 4010-style GIN.

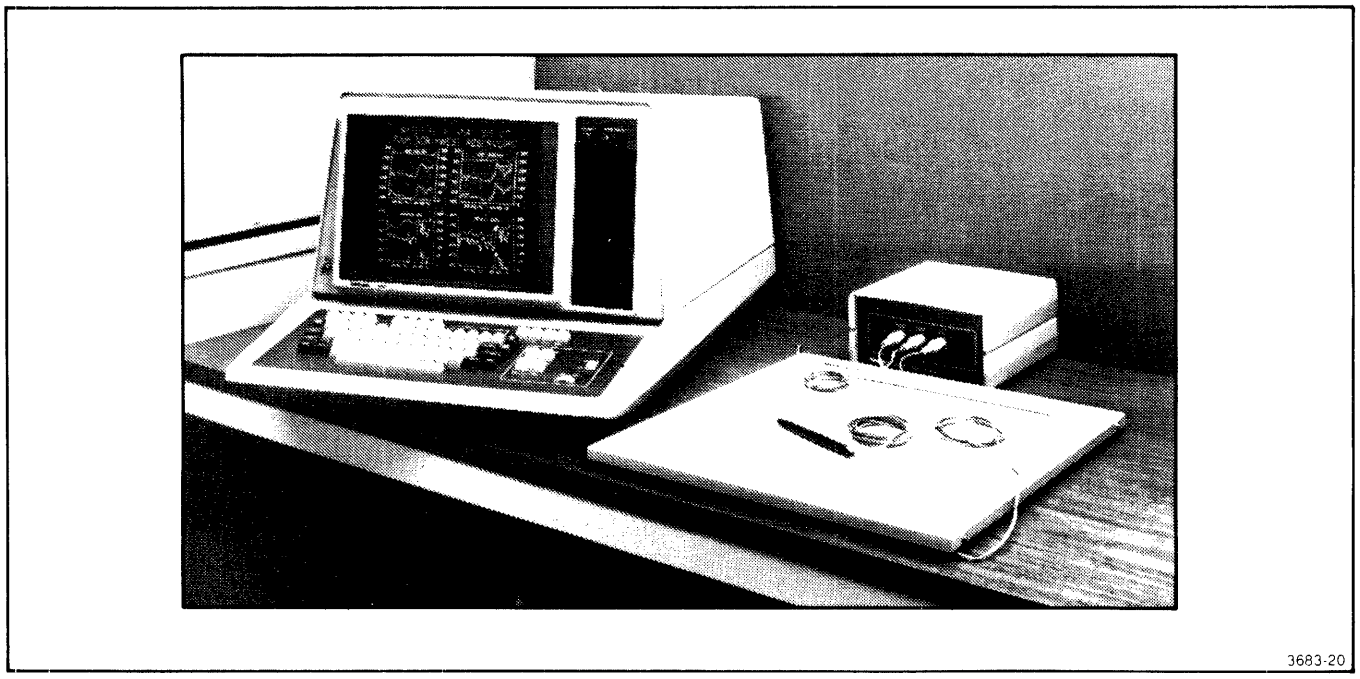


Figure 3-6. The 4112 and (Option 13) Graphics Tablet.

ATTACHING THE TABLET

Figure 3-7 illustrates how to plug the cables into the graphics tablet interface. The cable from input device (stylus, one-button, or four-button cursor) goes into the connector on the right; the cable from the upper left corner of the tablet goes to the middle connector; and the cable from the lower right corner goes to the left connector.

It is important to plug the correct cable into the correct outlet. If they are crossed, the tablet will not function properly.

The cable coming from the back of the interface plugs into the connector on the back of the terminal labeled "TO GRAPHIC TABLET."

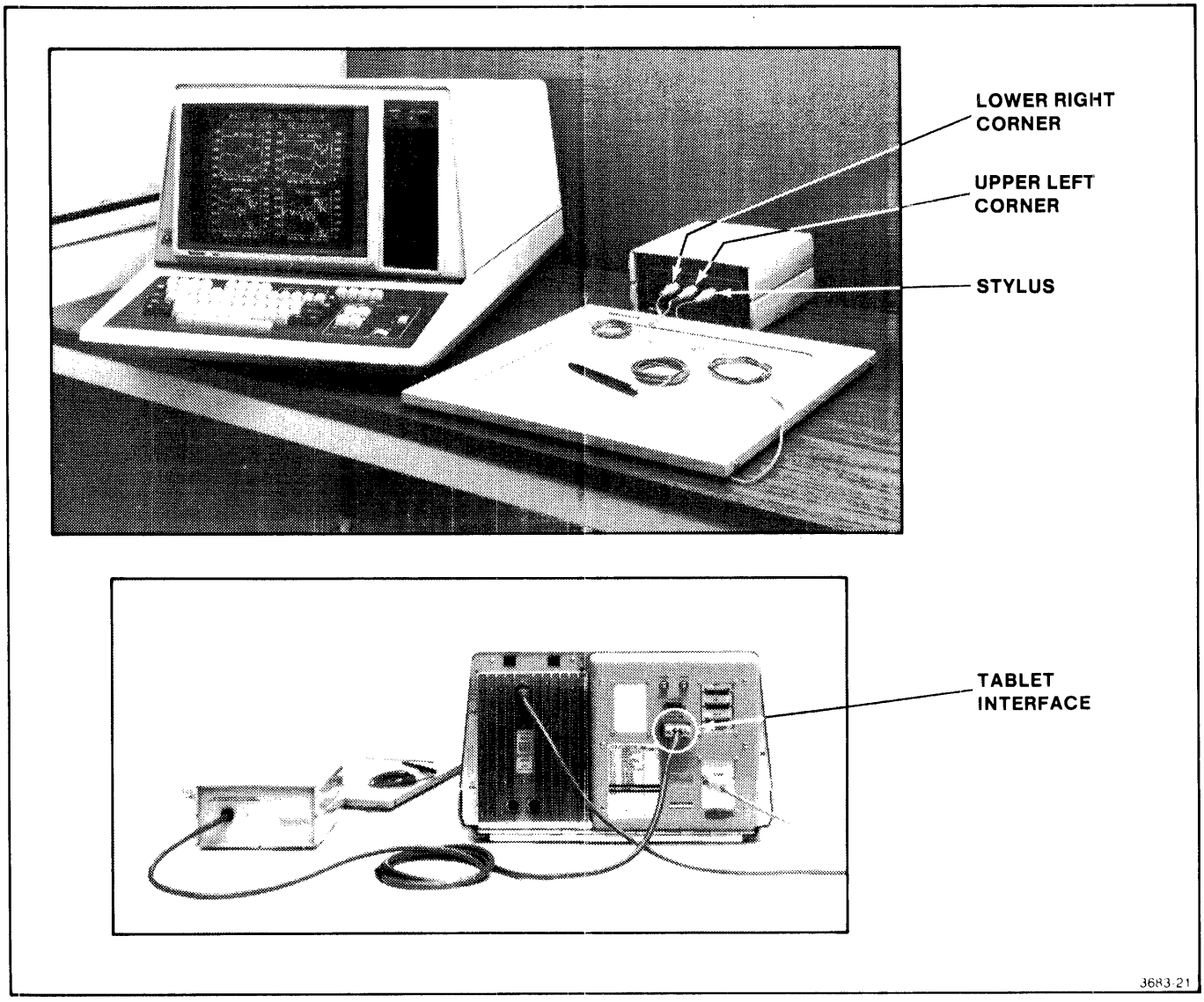


Figure 3-7. How to Attach the Tablet.

THE STYLUS

There are two different kinds of fillers for the stylus. One is a ballpoint filler, the other filler is a "blank" so it won't draw lines. You can use either of these fillers, depending on your needs.

To change the filler, unscrew the bottom part of the stylus just as you would a ballpoint pen. Gently pull the filler straight down. Be careful not to break either of the two probes (Figure 3-8). Insert the other filler and screw the stylus back together.

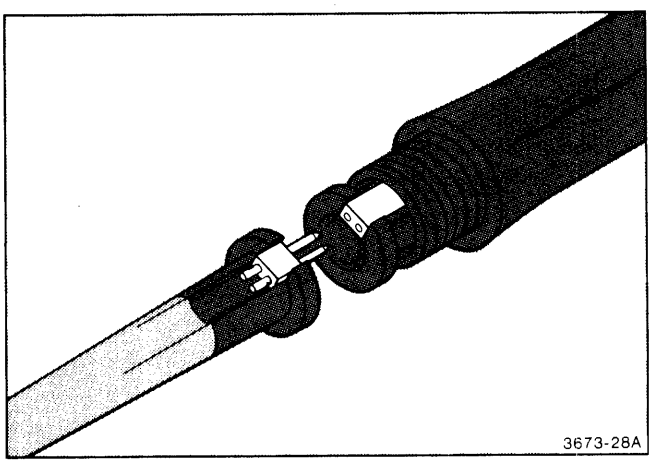


Figure 3-8. Taking the Stylus Apart.

POSITIONING THE PRECISION GRID

The precision grid is an optional accessory to the graphics tablet. If you have one, the grid can be used to help you enter data more precisely.

The procedure for positioning the grid on the tablet is included in the 4110 Series F13/14 Graphic Tablet Instruction Manual.

CLEANING THE TABLET SURFACE

Clean the tablet as often as necessary, using a damp cloth and a mild detergent, such as window or glass cleaner. Do not immerse the tablet in water or get it too wet. Never use a cleaner that contains hydrocarbons, since they will disolve the tablet surface.

RESTORING THE TABLET BIAS

About once a week you should use the following procedure to restore the tablet's magnetic bias. You can use this procedure more often if the tablet begins to relay incorrect data or if the graphics cursor moves erratically.

1. Turn the terminal off and unplug it.
2. Disconnect the tablet from the interface.
3. Pull the magnet across the tablet from the upper left corner towards the lower right corner in a single, smooth motion (Figure 3-9). Wipe the entire surface in about two or three seconds.
4. Connect the tablet again, plug the terminal back in, and turn the terminal back on.

You should store the magnet away from the tablet and other magnetically sensitive items (such as flexible disks and magnetic tapes).

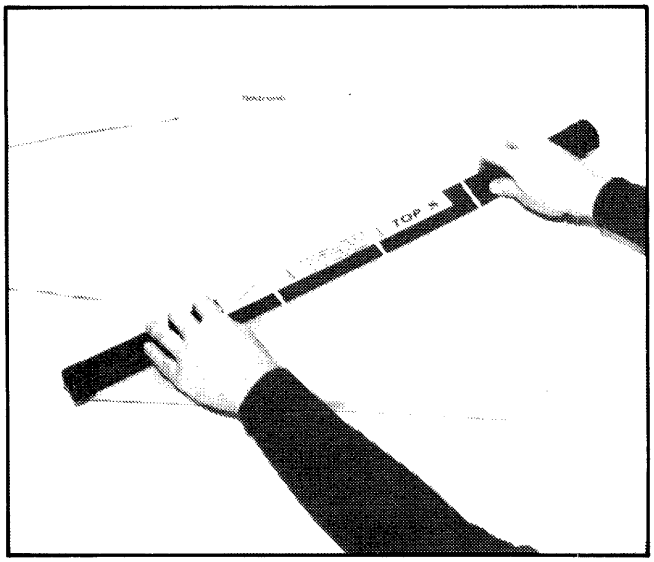


Figure 3-9. Restoring the Tablet's Bias.

HARDCOPY UNITS

The 4112 terminal is compatible with the TEKTRONIX 4612 and 4632 Hardcopy Units and the 4634 Imaging Hardcopy Unit. (Attach them by running the hardcopy cable from the copier to the interface in the back of the terminal (Figure 3-10).

The hardcopier should be running for about 10 minutes before you use it. After this period of time, it can make a good quality copy. To make a copy, press the HARD COPY key on the terminal or the COPY button on the copier.

NOTE

If the copies made by your 4612, 4632, or 4634 are distorted on the left side, your hardcopy unit needs to be adjusted. Contact a service technician or see the 4612 Service Manual, 4632 Service Manual, or 4634 Instruction Manual for details.

For more details on the HARD COPY key see Section 2 of this manual. See the operator's manual for the 4612 and 4632 hardcopy units for further operating instructions, or the Instruction manual for the 4634 Imaging Hardcopy Unit.

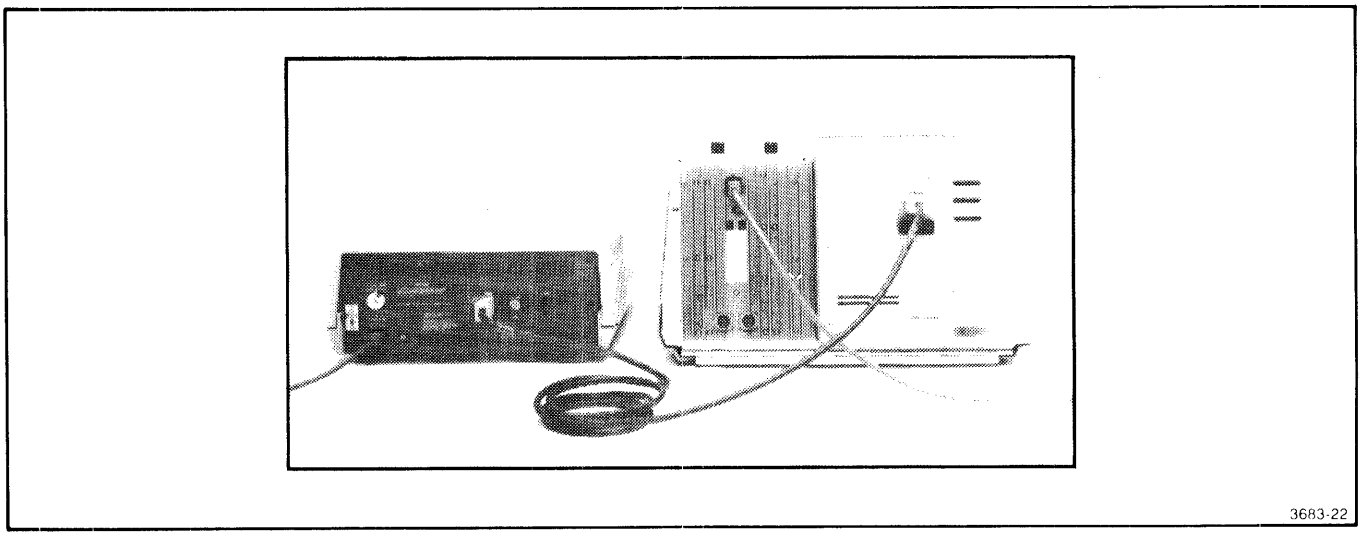


Figure 3-10. How to Attach a Hardcopy Unit.

EXTERNAL VIDEO OPTION

The External Video option (Opt. 11) provides a 30-Hz RS-170/330 video output so the terminal's graphics data may be displayed on an external video monitor. This option produces a standard studio video signal, with or without serration and equalization pulses (See NOTE below). The external video output signal terminates on a standard BNC type connector on the terminal's rear panel, see Figure 3-11. Use a standard cable to connect between the terminal's video output connector and the monitor's video input connector.

If all terminals and monitors in your system are synchronized to a master sync signal, connect a BNC cable from the sync source to the terminal connector labeled EXT SYNC.

After the video signal cable (and sync cable, if any) are connected between the terminal and the monitor, turn on the power to both units. The terminal's display image will appear on the terminal screen and the external monitor screen. If the monitor image is not stable (rolls, etc.) adjust the sync controls as on any video monitor or television set.

NOTE

Serration and equalization pulses aid in vertical and horizontal synchronization of the picture, much like an ordinary television set. If your system requires any changes to the video output signal, call your TEKTRONIX field service representative or refer to strapping and service information in the TEKTRONIX 4112 Service Manual.

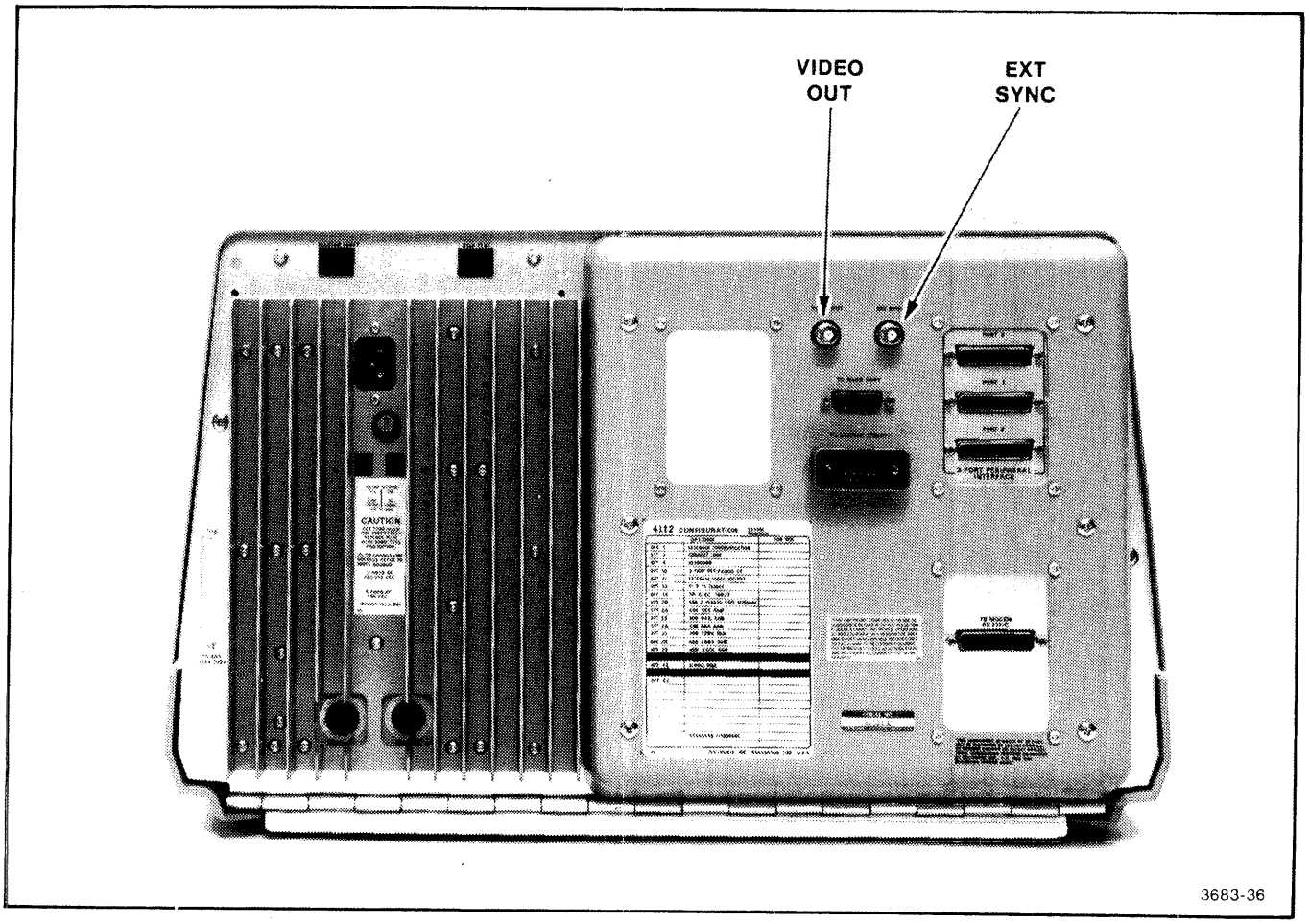


Figure 3-11. External Video Connectors.

Section 4

GRAPHICS INPUT MODE

Graphics input (GIN) mode allows you to send graphics information to a program.

Usually a host resident program will put the terminal into GIN mode and prompt you for certain information. You can use escape sequence commands in Setup mode to put the terminal in GIN mode.

The program should also prompt you concerning what kind of data you should enter. For example, it might tell you to locate a certain number of points on the screen (the "locate" function), to "pick" a certain "menu item" or part of the picture (the picking function), or to "stroke" (draw) something on the screen.

The devices that can be used to send graphics information to a program (GIN devices) are:

- the terminal's thumbwheels
- a 4662 or 4663 plotter
- a graphics tablet

An explanation of how to position the graphics cursor with each input device and how to use each of the functions is included in this section.

THE PICK FUNCTION

A program may be written such that you are to choose between several actions. This list of actions is called the "menu." The program would instruct you to "pick" the desired action. To do so, position the graphics cursor near the description of the appropriate action. The way that you signal the program that the cursor is at the position you want to "pick" depends on the input device you are using. They are each described later in this section.

You may also be asked to pick a part of the picture to be moved, enlarged, reoriented, or changed in some way. Again, you would position the graphics cursor near the object and signal the program that the cursor is at the location you want to pick.

THE LOCATE FUNCTION

Move the graphics cursor to the appropriate point on the screen. When it is properly located, signal the program that you have located the cursor. The input device sends the coordinates of that point to the program, and it is acted on accordingly.

THE STROKE FUNCTION

The graphics tablet allows you to draw lines as you move the stylus along the surface of the tablet. (The terminal draws lines between points when it is using inking but the lines appear only on the screen; they are not saved.)

INKING

As you locate points on the screen, lines between these points may be drawn. If this occurs, the terminal is using the "inking function."

RUBBERBANDING

When you move the cursor a line may "follow" the graphics cursor. One end of the line is anchored at the last point and the other end of the line is attached to the graphics cursor. This function is called "rubber banding." It allows you to draw a line more precisely from one location to another. Rubberbanding works with the locate and pick functions.

GRIDDING

For the locate and pick functions, the program may also specify that an invisible "grid" be in effect on the screen. When a grid is active, you can only locate the cursor on points where the vertical and horizontal lines of the grid intersect. This allows you to precisely position the cursor when it is necessary to draw lines that intersect at a 90 degree angle.

There is a setup command called GRIDDING which allows you to easily establish a grid for the input device. See Section 6.

When you move the graphics cursor it may "jump" from one point to the next. If this happens, you will know that gridding is in effect and the cursor is moving between points on the grid.

FILTERING

For the stroke function you can set time and distance values to specify the number of points the terminal will send to the program.

USING THE TERMINAL'S THUMBWHEELS

When the terminal is in GIN mode, a graphics cursor appears on the screen. The default is the "crosshair cursor" (Figure 4-1). However, the program can select another graphics entity to be the graphics cursor.

Position the graphics cursor with the thumbwheels (Figure 4-2). As the arrows indicate, the top thumbwheel moves the graphics cursor up and down and the bottom thumbwheel moves it back and forth across the screen.

If you hold the SHIFT key at the same time as you rotate a thumbwheel, the cursor moves more slowly so you can position it more precisely.

When you have positioned the graphics cursor where you want it, you send the location of that point to the program by pressing any alphanumeric key. When you press the key, the crosshair cursor blinks momentarily as the point is transmitted to the program. Do not press the RETURN key, as that causes invalid data to be transmitted.

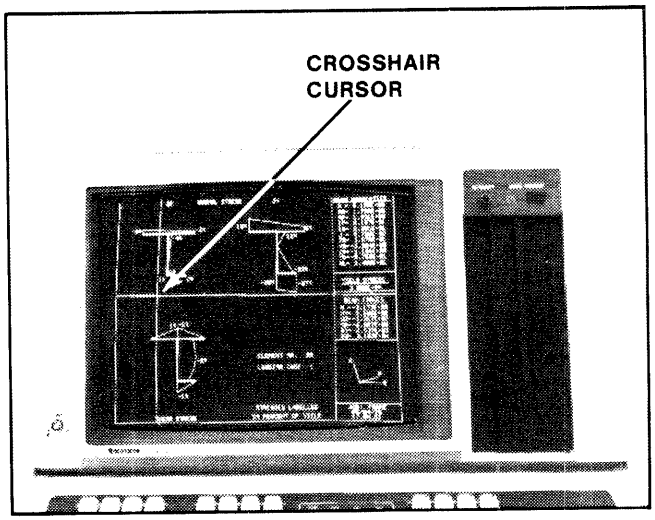


Figure 4-1. The Crosshair Cursor.

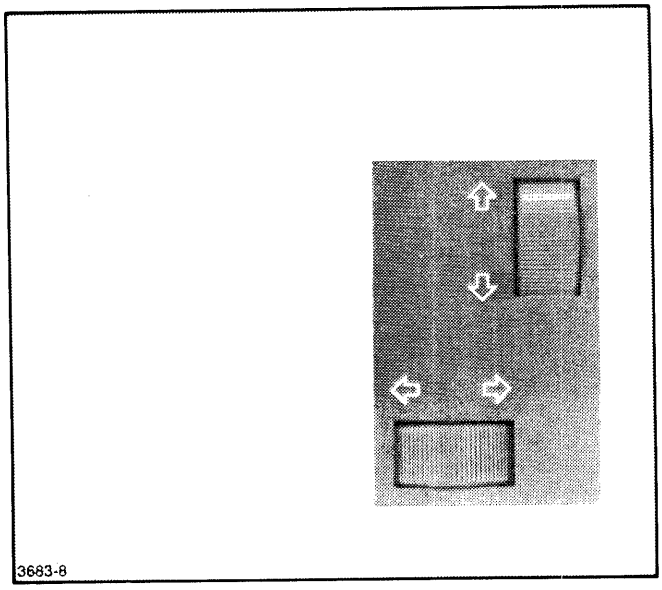


Figure 4-2. The Thumbwheels.

USING THE TABLET

When you use a graphics tablet for a GIN device the graphics cursor will be visible on the screen. The crosshair cursor (Figure 4-1) is the default graphics cursor, but the program can select any graphic entity as the cursor. The applications program should inform you what is the graphics cursor if it is not evident.

When the tablet is active, you can use one of three input devices: the stylus, a single button cursor, and a four button cursor (Figure 4-3).

You can locate points in an 11" by 11" square on the smaller tablet (Option 13) and a 30" by 40" area on the larger tablet (Option 14). This is called the "active writing area." In both cases the center of the active writing area is the center of the tablet.

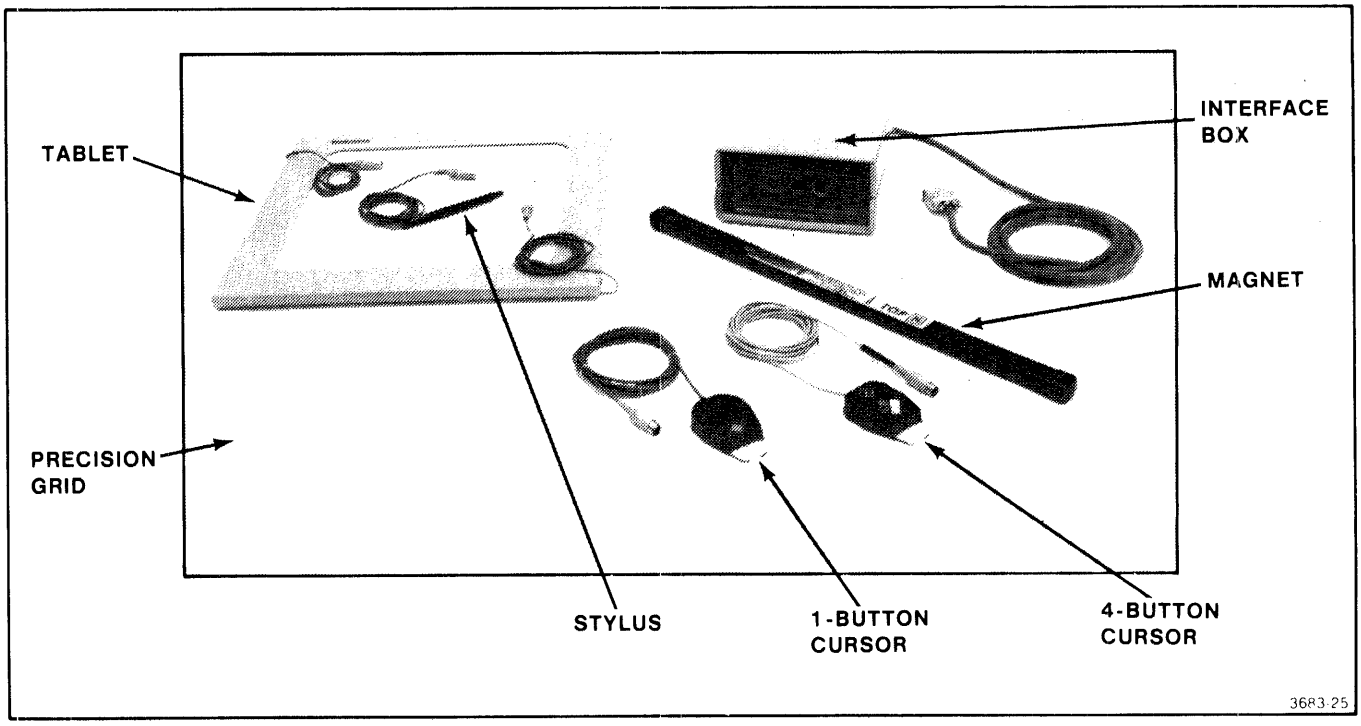


Figure 4-3. The Option 13 Graphics Tablet.

THE STYLUS AND ONE-BUTTON CURSOR

Hold the stylus as you would a ballpoint pen, with the tip resting on the tablet. The cursor on the terminal screen tracks with the stylus as you move it. The tablet can keep track of the stylus as long as it is with 5/32 of an inch of the tablet surface.

Lay the cursor flat on the tablet surface with the button facing up. As you move the cursor along the surface of the tablet, the graphics cursor tracks with it.

For the pick and locate functions, when you locate the appropriate point on the tablet or terminal display, press down on the stylus. The stylus' tip moves slightly and the graphics cursor blinks momentarily as the point is transmitted to the applications program. For the cursor, press the button to pick or locate a point.

For the stroke function, press the stylus or cursor button down and move the stylus or cursor across the tablet surface as if you were drawing on it. The number of points transmitted to the program depends on the setting of the TBFILTER command (see Section 6).

THE FOUR-BUTTON CURSOR

Lay the cursor flat on the tablet surface with the buttons facing up. As you move it along the surface of the tablet, the cursor tracks with it.

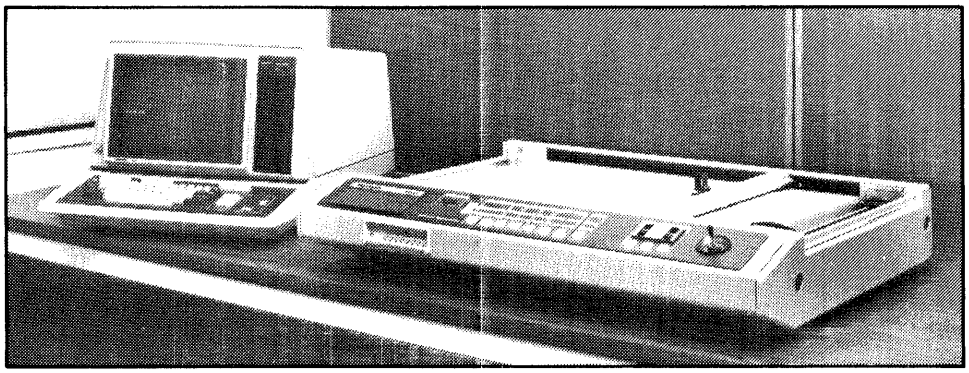
The functions of the four buttons can be defined by the applications program. For example, one button might cause inking to be used, another button may erase the line between the next two points, the third could implement the stroking function, and so on. The applicatons program should inform you how to use the buttons.

If the buttons are not programmed, you can press any of the four buttons to transmit the current location of the graphics cursor to the applications program.

USING THE 4662 OR 4663 PLOTTER

The 4662 and 4663 plotters can also be used as graphics input devices (Figure 4-4). Use the plotter's joystick to position the pen at the appropriate location. To indicate to the program that you have located the point you want and you are using a 4662, press the the plotter's CALL button. If you are using a 4663 plotter, press the plotter's DRAW POINT button.

For additional operating instructions, see the 4662 or 4663 Operator's Manual.



3683-34

Figure 4-4. The 4112 and 4663 Plotter.

Section 5

COMMUNICATIONS OVERVIEW

This section provides an overview of the terminal's communications modes. For detailed information on these subjects, see the 4112 Host Programmer's Manual, a system manual for your host, or a systems programmer for your host.

In most cases communications parameters are set by program control and not by an operator. However, since there are setup commands which allow these conditions to be set by an operator, the concepts are introduced in this manual. And, there are some commands (such as BAUDRATE) which would usually be set by the operator.

Some communications parameters are crucial. For example, if the host and terminal have conflicting baud rates, there can be no communications. Similarly, if the host checks for even parity and the terminal is set up for odd parity, there will be communications problems. It is important for the operator to have some understanding of basic communications considerations to avoid some obvious problems.

Table 5-1 lists all of the setup commands which deal with communications. Commands under the heading "Standard Commands" are standard for the terminal. Those under the heading "Optional Commands" are part of the communications option (Option 01).

The concepts which these commands introduce are discussed in the rest of this section.

Table 5-1

COMMUNICATIONS COMMANDS

Standard Commands	Explanation
BAUDRATE	Specifies transmit and receive baud rates
BREAKTIME	Specifies length of break transmission
BYPASSCANCEL	Specifies bypass cancel character
EOFSTRING	Specifies end of file string
EOLSTRING	Specifies end of line string
EOMCHARS	Specifies end of message characters
FLAGGING	Specifies whether flagging is used and what kind
PARITY	Specifies how parity is checked
PROMPTMODE	Specifies whether or not terminal is in Prompt mode
PROMPTSTRING	Specifies prompt string
QUEUESIZE	Specifies size of terminal's input queue
STOPBITS	Specifies number of stop bits used
XMTDELAY	Specifies time terminal waits after transmitting an end of message character before resuming transmission
XMTLIMIT	Sets upper limit of terminal's tranmission rate

Optional Commands	Explanation
BCONTINUECHARS	Specifies block continue characters
BENDCHARS	Specifies block end characters
BHEADERS	Specifies block header sequences
BLENGTH	Specifies block length
BLINELENGTH	Specifies maximum line length for Block mode
BLOCKMODE	"Arms" or "disarms" the terminal for Block mode
BMASTERCHARS	Specifies block master characters
BNONXMTCHARS	Specifies block non-transmittable characters
BPACKING	Specifies manner in which blocks are packed for transmission and receiving
BTIMEOUT	Specifies how long terminal waits before retransmitting previous block
DUPLEX	Specifies Duplex mode

STANDARD COMMUNICATIONS

The terminal's standard communications feature all facilities necessary to support I/O with most host computers. These include:

- full duplex
- RS-232-C compatibility
- Prompt mode
- flagging

"Full duplex" implies that the terminal can send and receive data at the same time. Whether or not this actually occurs depends on whether the host is capable of sending and receiving data simultaneously.

Each character is transmitted as soon as you press the key. Likewise, each character is processed as soon as it is received from the host.

"Prompt mode" and "flagging" are explained later in this section.

COMMUNICATIONS CONCEPTS

The following concepts and terminology apply to standard communications, as well as communications in Block mode, and half duplex.

BAUDRATE

The baud rate is the rate at which data is transmitted between the host and the terminal. As used in this manual, the term "baud rate" means "bits per second." Transmission at a rate of 9600 baud means 9600 bits per second, or approximately 960 characters per second.

The factory default baud rate is 2400. You can use the setup command BAUDRATE command to modify it. The transmit and receive baud rates can be set independently.

It is important that the terminal and the host be set to communicate at the same baud rates. If not, data will be misinterpreted or not communicated at all. You can use the STATUS command, as explained in Section 6 of this manual, to determine the terminal's current transmit and receive baud rates.

PARITY

The accuracy of the transmission of data can be verified in many different ways. One way is to check the "parity" of each character after it is sent.

Each character is coded into eight binary digits (bits) with a value of 0 or 1. The first seven bits define the character. The eighth character is often used for a parity check.

"Even" parity means that the sum of the bits is always even. That is, when the sum of the bits of a character add to an even number, the eighth bit is set to 0; if the sum is an odd number, the eighth bit is set to 1.

Similarly, "odd" parity means the sum of the bits must always be an odd number.

"None" means that the parity is always 0. "High" means that the parity bit is always set to 1. "Data" means that the eighth bit is part of the character being transmitted instead of being used as a parity check.

The terminal's parity must always be set to match the host computer's needs. The terminal ignores the parity bit on characters the host transmits to it.

FLAGGING

It may be necessary for the terminal and the host to be able to inform each other when they are ready to receive data and when transmission should be temporarily suspended.

For example, the terminal has an input queue into which data is collected until it can be processed. When the input queue is full, any additional data which the host tries to send to the terminal would be lost.

However, the terminal can send a character (called a "flag") to stop data transmission from the host when its input queue is full. When there is more room in the queue, another flag is sent to tell the host to resume transmission.

Since this is not useful on all hosts and since some hosts require different kinds of flagging, the setup command FLAGGING allows the operator or a programmer to define the flagging parameters.

QUEUESIZE

The terminal processes data as soon as it receives it from the host. However, if data is coming to the terminal at a high baud rate, it may not be able to process it as quickly as it is received.

When the terminal is in Block mode, it waits until it has received a complete block of data before it begins to process it.

In both of these cases, the terminal stores data in an input queue until it can process it.

The size of the queue can be specified by the QUEUESIZE command. Its size depends on the host computer, the rate of transmission, the kind of communications link used, and other installation dependent factors. The factory default is 300 bytes.

XMTLIMIT

Sometimes the rate at which a host computer can effectively receive data is less than the maximum possible for the baud rate at which the communictions port is specified.

For example, a time-share host may not be able to receive data at 9600 baud, even though you are logged onto a 9600 baud port. In such a case, you could leave the terminal's transmit baud rate setting at 9600 but set the transmit limit parameter (established by the XMTLIMIT command) to a level with which the host can cope — perhaps 8000. The terminal then sees to it that data is not transmitted at a rate faster than 8000 baud.

The baud rate is not actually changed, but the characters transmitted are spaced apart so that the number transmitted per second is the same as if the baud rate were the transmit limit rate.

The factory default transmit limit is 19200 baud. If your host can receive data at speeds up to 19200 baud, you do not have to be concerned with the transmit limit.

BREAKTIME

There will be occasions when you need to interrupt the transmission of data from the host or halt a host program. One way to do so is with the BREAK key.

The BREAK key sends an interrupt transmission to the host. The length of the interrupt is set by a command called BREAKTIME. The factory default BREAKTIME parameter is set to 200 ms. However, it may be need to be longer or shorter — and on some hosts you may not want to use the BREAK key at all. In that case, BREAKTIME can be set to 0 ms.

EOLSTRING

The "end of line string" indicates the end of the transmission of a line of data. It is sent by the terminal at the end of (or during) transmissions generated by the terminal.

The factory default termination string is the "carriage return." This works for all hosts which interpret the carriage return as the end of a particular line of data.

Since you explicitly inform the host of the end of a line by typing a termination string (carriage return) when entering data from the keyboard, no special action must be taken by the terminal.

However, sometimes the host requests information from the terminal which is sent in a different format and requires no operator interaction. For example, the program may request a report of error messages or some graphics input data. In such cases, the terminal sends an end of line character to the host to indicate the end of a transmission.

The EOLSTRING command (see Section 6) is used to change the end of string.

XMTDELAY

The terminal can be set to wait for an interval of time after it receives data from the host before it begins to transmit. Whether or not this is desirable or necessary depends on your host.

The interval for the terminal is set by the XMTDELAY command. The delay can be set from 0 to 65535 ms (which is about one minute). The factory default is 0. In some cases, for example, a delay of 50 or 100 ms is necessary to allow a modem or other transmission equipment to "turnaround."

EOFSTRING

When you are transferring files between the host and terminal, it is necessary to know when the file transfer is complete. The "end of file" indicator is a string of from one to ten ASCII characters.

When a data file is transferred from the host to the terminal using a COPY or SPOOL command, the host must send the end of file string when it reaches the end of the file. The terminal sends the same string to the host when it reaches the end of a file it is transferring to the host.

This string is defined by the EOFSTRING command. It should be set to whatever the host computer sends when it reaches the end of a file. It is best to set this to the null string except when you are actually transmitting files, since the string is ignored by the terminal whenever it is detected in a host transmission.

STOPBITS

Each character sent between the host and terminal is coded into eight binary digits (bits) of data. Each of these eight bit codes is called a byte.

After sending a byte of data, the host or terminal sends one or two extra bits to indicate the end of a byte. These extra bits are called "stop bits."

The terminal includes a STOPBITS command which allows you to specify whether the terminal sends one or two stop bits at the end of each byte. The factory default is 1.

EOMCHARS

The "end-of-message" characters are communications "turnaround" characters.

When the terminal is transmitting to the host and it encounters an end-of-message character, it transmits that character and then pauses for a period of time before resuming transmission.

The period of time the terminal waits is specified by the XMTDELAY command.

If the terminal is in Prompt mode, when it sees the end of message character it knows to wait for a prompt string from the host before resuming transmission. When the terminal is in Block mode, it first transmits the block, then it waits for XMTDELAY and for the next prompt.

The end-of-message characters are set by the EOMCHARS command. The factory default characters are CARRIAGE RETURN and NUL. Since $^{N}_{L}$ (null) is ignored, setting both end of message characters to $^{N}_{L}$ disables this feature.

BYPASSCANCEL

There are many characters which are transmitted between the host and the terminal which you do not need to or even want to see. For example, graphics input characters and other report characters are not important to an operator, yet they must be processed by the terminal.

Instead of being displayed, these characters are automatically "bypassed." The "bypass cancel character" is used by the terminal to assure that the correct characters are bypassed and that the display of characters is resumed appropriately.

This character is specified by the BYPASSCANCEL command. The factory default is the linefeed character.

PROMPT MODE

The PROMPTMODE command can be used to initiate and terminate Prompt mode.

All of the communications parameters detailed under the "Standard Communications" heading are also valid when the terminal is in Prompt mode.

In Prompt mode, all data to be transmitted to the host is kept in an output queue. It is sent when a prompt is received from the host. When the prompt is received (and the transmit delay is satisfied), data continues to be transmitted until an "end of message" character (specified by the EOMCHARS command) is sent.

The "end of message" character halts transmission of data and is also an indication to the host that transmission is over until it sends another prompt.

The PROMPTSTRING command is used to specify the prompting string. The prompt string can be from one to ten characters, and it must conform to the actual prompt used by the host.

BLOCK MODE (OPTION 01)

In Block mode, data is packed into blocks and transmitted as a unit. After each block is transmitted, there is a response from the device which received it before another block is sent. This process verifies the accuracy of the transmission of each block. If there were any errors in the transmission, the same block is retransmitted until it arrives in a correct form.

Because of this format, Block mode allows easier transmission of full ASCII or binary code. It also provides error detection and automatic retransmission of bad data blocks.

Block mode can be used at the same time as Prompt mode and in Full Duplex or Half Duplex mode.

Specific details concerning how Block mode works are very complex. The 4112 Host Programmer's Reference Manual contains more information on Block mode.

The BLOCKMODE command arms the terminal for Block mode. The terminal actually enters Block mode when it receives the first block header.

The following definitions are an introduction to the concepts of block mode communications. They are included because there are set up commands which specify these parameters. The syntax of each block-mode communications command is detailed in Secton 6.

BLENGTH

The maximum length of each block, for both transmitting and receiving, is specified by the BLENGTH command.

The best size for a block length depends on several factors. For example, after each block is transmitted by either the terminal or the host, a response is required before the next block is sent. Therefore, the shorter the block, the more time is spent responding to blocks sent and received. However, for a long block, more time is spent retransmitting a block if that is necessary.

The factory default is 256 bytes.

BHEADERS

The start of each line of data in Block mode is indicated by a header sequence. The header sequence is a string of from one to ten ASCII characters. The first character in the sequence cannot be used elsewhere in the sequence; and the sequence cannot include any character which is part of the prompt string.

The header sequence is specified by the BHEADERS command. The factory defaults are HEADTX, HEADRX.

BCONTCHARS

A block of data may be one or more lines long. If a block extends to more than one line, each line ends with a block continue character, indicating that more lines follow. There can be different block continue characters for transmitting and receiving.

The block continue character is be specified by the BCONTCHARS command. The choice of characters depends on what characters the host expects.

The default block continue characters are ampersand (&) for both transmit and receive.

BENDCHARS

A "block end" character specifies the end of a block. The block end character can be specified with the BENDCHAR command. The block end character must be compatible with the host software.

BLINELENGTH

The block line length specification is the longest line that can be transmitted from the terminal that will not overrun the host's input buffer. Therefore, it should not be set to a value which exceeds the input buffer of the host. The default value is 70.

There is no maximum line length for the terminal, since it can receive a line of any length.

The block line length parameter is set by the BLINELENGTH command.

BPACKING

There are various data packing schemes available for host systems with limited transmission capabilities. That is, if your host cannot transmit full 7-bit ASCII or 8-bit binary data, the data can be coded (packed) into a reduced character set.

The data packing scheme is specified by the BPACKING command.

BTIMEOUT

When the terminal sends a block of data, it waits for a response from the host to acknowledge that it has received that block intact. If the terminal does not receive the appropriate response within a period of time or if it receives an erroneous response the terminal retransmits that block. It pauses again, and will repeat the sending of the block until it is acknowledged.

The period of time the terminal waits before sending the block again is specified by the BTIMEOUT command.

Be sure that the specified timeout period is longer than may be encountered due simply to a busy host.

BNONXMTCHARS AND BMASTERCHARS

There are several charactes which cannot be transmitted in Block mode.

For example, if "A" is the block end character, it cannot be transmitted as an "A," since whenever the terminal or host sees "A," it assumes it indicates the end of a block.

Therefore, whenever "A" or any other restricted character must be sent, the terminal or host instead sends a master character, followed by a substitute character. When that two-character combination is received, it is decoded into the intended character.

This is done automatically by the terminal and host. However, all non-transmittable characters must be specified by the BNONXMTCHARS command. The character used as the master character is specified by the BMASTERCHARS command.

HALF DUPLEX MODE

The standard mode of communication is full duplex, including when the terminal is in Prompt mode or Block mode. In full duplex, the terminal and host can both send and receive data at the same time.

In half duplex, either the terminal and host send data, but not at the same time.

All of the standard communications commands are meaningful in half duplex mode.

In half duplex with Supervisor Communicatons mode, the host always controls the flow of data. That is, the host decides when the terminal or host sends or receives data.

Section 6

SETUP COMMAND DICTIONARY

This section contains a detailed explanation of Setup mode and of each setup command. Table 6-1 provides an overview of the setup commands in a functional organization. The rest of this section is a detailed explanation of each command, in alphabetical order.

To put the terminal into Setup mode, press the key labeled SET UP (Figure 6-1). The terminal must be in Setup mode to execute setup commands.

Setup commands allow you to establish the terminal's operating environment and to use peripherals attached to the terminal. For example, setup commands are used for establishing communicatons rates, inquiring about the terminal's status, determining what levels of errors the terminal should display, and programming special meanings (macros) into keys.

SETUP COMMAND FORMAT

A setup command starts with a command word, usually has one or more parameters, and ends with a carriage return.

The "command word" is a descriptive English-like word indicating what the command does. A command word can be entered in either upper or lower case; it can be spelled out completely or abbreviated to as few characters as are necessary to distinguish it from other command words — usually two or three characters. Whatever amount of the command word is typed must be spelled correctly or an error results.

In most cases a command includes one or more of six different parameter types. (Parameters are explained in more detail with the individual commands.) The parameter types are:

- single characters
- keywords
- delimited character strings
- un-delimited character strings
- key specifiers
- integers

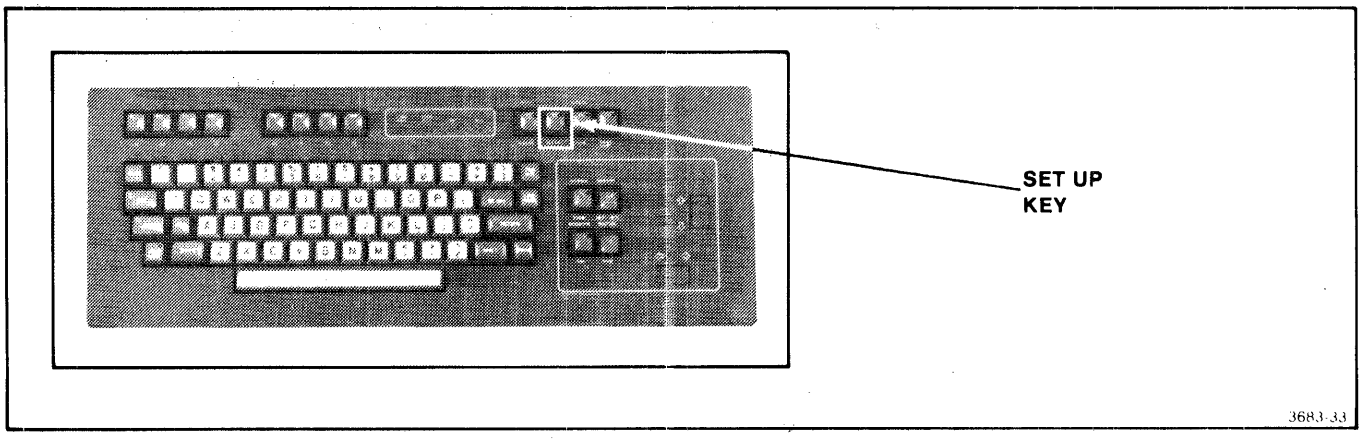


Figure 6-1. The SET UP Key.

Keyword parameters can be abbreviated to as few characters as are required to distinguish them from other keywords in the same command.

A "key specifier" parameter can be entered by pressing the appropriate alphanumeric key or by entering its ADE (ASCII Decimal Equivalent) value.

A carriage return terminates a line and causes the command on that line to execute. One setup command can be entered per line.

A "line" can be up to 130 characters long. If you continue typing when the cursor is at the right edge of the screen or dialog area, the cursor wraps around to the left margin of the next line on the screen. It is still the same line until you enter a carriage return. If you continue to type after you reach the maximum of 130 characters, the terminal bell rings each time you press any key except RETURN, the "character delete" key, or the "line delete" key (see the EDITCHARS command later in this section).

CORRECTING A MISTAKE IN SETUP MODE

You can backspace to edit one or more characters on a particular line by pressing the "character delete" key. You can discontinue the current line and start a new line by pressing the "line edit" key (see the EDITCHARS command later in this section for details).

If you enter a carriage return to terminate a command and there is a mistake in the command just entered, an error message is displayed on the next line. When this happens, you can look up the meaning of the error message in Appendix A and reenter the command correctly.

SETUP MEMORY

The terminal stores many setup command parameters in "setup memory." When the terminal is turned on, it remembers the parameters which are stored in setup memory.

In Table 6-1, command names preceded by a superscript are stored in setup memory. In the explanations of the commands, "(Memory)" is printed immediately following the statement of the purpose of the command, to indicate that the command's parameters are saved in setup memory.

You can use the STATUS command to determine current status of the terminal or a particular parameter, as explained later in this section.

HOW SETUP INTERACTS WITH OTHER FUNCTIONS AND MODES

Host Communications. Communication to the host is suspended when the terminal is in Setup mode. Data sent from the host to the terminal is stored in a queue in the terminal during Setup mode and processed when the terminal exits from Setup mode. (The size of the input queue is determined by the QUEUESIZE setup command.)

GIN Mode. When the terminal enters Setup mode, GIN mode is suspended. The GIN cursor disappears and graphics input data is not communicated. When you exit from Setup mode, the cursor reappears and GIN mode continues. Graphics input is explained in Section 4. Also see the 4112 Host Programmer's Reference Manual and the 4110 Series Command Reference Manual.

Control Characters. If you enter a control character when the terminal is in Setup mode, the control character is displayed rather than executed. Control characters are displayed in a coded two character abbreviation called "snoopy" characters.

Programmed Keys. If the terminal is in Setup mode and you press a key which has a macro programmed into it (see DEFINE, later in this section), the macro is expanded as usual.

Local Mode. When the terminal is in both Setup and Local modes at the same time, Setup mode takes priority.

ESCAPE SEQUENCE COMMANDS

Setup commands are a subset of the terminal's command repetoire. Setup commands are those which are used most often and which an operator will find most useful. Setup mode also allows you to execute commands for which there is not an English-like command word. These are called "escape sequences," since the first character of these commands is the "escape" character. (See Appendix C.)

COMMAND SYNTAX CONVENTIONS

Figure 6-2 illustrates the format used in presenting the syntax of the terminal's setup commands.

The beginning of the explanation of each command is at the top of a column. The column's heading is the command name. Immediately after the command name is a brief explanation of the purpose of that command. Following that, the word "(Memory)" appears if the command's parameters are saved in setup memory, and the option number if this command is part of a terminal option.

The next line shows the command in its correct syntax, according to the following conventions:

- The command name is shown first, in all capital letters, and bold faced type. You can enter the command name in lower case if you prefer, and can abbreviate the command name to as few characters as are necessary to distinguish it from other commands.
- Words that must be entered as shown, such as command names and keywords, are shown in bold faced type.
- Parameters in regular faced type are descriptive of the kind of information that parameter specifies.
- Each parameter is separated by one or more spaces or a comma. You can separate by one comma and a space or by multiple spaces. Multiple commas have a special meaning.
- Parameters in square brackets are optional.
- You must choose one parameter when two or more are stacked and in curly brackets.
- You may choose one parameter when two or more are stacked within square brackets.
- When you can choose between a number of different variations, the presentation of the syntax is simplified by showing it in two or more statements.

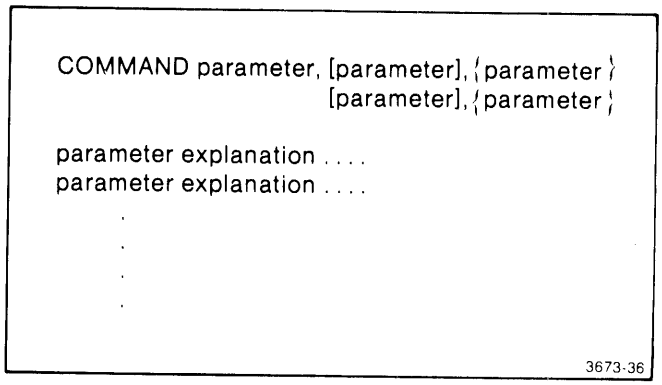


Figure 6-2. Format of Setup Command Syntax.

SETUP COMMAND SUMMARY

Table 6-1 lists and briefly describes all setup commands. It also indicates the factory default parameters, which commands are part of options to the terminal, and those commands whose parameters are stored in setup memory. The factory default is shown for each parameter because that is what the terminal defaults to if the setup memory fails, or is changed by the addition or deletion of a circuit board.

NOTE

In this table, the information in parenthesis AFTER the commands indicates the following:

- *01= Extended Communications option*
- *10= Three Port Peripheral Interface option*
- *13/14= Graphics Tablet options*
- *42= Flexible Disk option*

Table 6-1

SETUP COMMANDS

Function	Command	Factory Default	Purpose
General Terminal Operation	[a]CRLF	No	Sets or terminates "carriage return implies line feed."
	[a]ECHO	No	Sets local or remote echo.
	[a]EDITCHARS	RUB-OUT, ^{C}N, ~	Defines character delete, line edit, and "literal" characters.
	[a]ERRORLEVEL	2	Sets error reporting threshold.
	[a]FIXUP	6	Determines how thoroughly the viewport is updated when a change is made that affects it.
	[a]GAMODE	Replace	Determines if background of character is visible when text is displayed in graphics area.
	[a]GRIDDING		Establishes an invisible grid on the screen for GIN mode.
	[a]IGNOREDEL	No	Specifies whether or not the "delete" character is ignored.
	[a]LFCR	No	Sets or terminates "line feed implies carriage return."
	[a]PAGEFULL	None	Determines action to be taken when page full condition occurs.
	[a]PICKAPERTURE	8	Determines size of aperture used for picking function.
	[a]REOM	1	Determines use of end of message character for GIN mode.
	[a]RLINELENGTH	0	Determines GIN report line length limit.
	SNOOPY		Sets or terminates "snoopy" mode.
	STATUS		Displays the terminal's status.

[a]Saved in Setup Memory.

Table 6-1 (cont)
SETUP COMMANDS

Function	Command	Factory Default	Purpose
Dialog area control	[a]DABUFFER	15	Sets maximum size of dialog buffer in lines.
	[a]DACHARS	80	Sets maximum number of characters per line in the dialog area.
	[a]DAENABLE	No	Specifies whether or not alpha text is sent to the dialog area.
	[a]DAINDEX	7, 0, 0	Specifies the gray-scale index in the dialog area.
	[a]DALINES	5	Specifies the visible portion of the dialog buffer.
	[a]DAMODE	Replace	Sets Overwrite or Replace mode in dialog area.
	[a]DAPOSITION	0, 0	Sets position of dialog area.
	[a]DASURFACE	1	Sets dialog area surface.
	DAVIS	No	Sets visibility of the dialog area.
Keyboard control	DEFINE		Assigns macro to key or integer.
	[a]KEYEXCHAR	$^{D}_{E}$	Specifies the key execute toggle character.
	LOCKKEYBOARD	No	Locks the keyboard.
Data transfer control	COPY		Copies data from one device to another.
	LOAD		Reads a file and executes it as a command file.
	MAPINDEX		Assigns a gray-scale index to a plotter pen.
	PCOPY(10)		Copies between peripheral devices and the host.
	PLOT(10)		Plots currently visible segments to specified device.
	SAVE		Saves macro, segment, text font, or pixels to a disk file.
	SPOOL		Copies data from one device to another, leaving the terminal free to do other jobs at the same time.
	STOP		Halts the spooling operation.
Flexible disk control	DELETE(42)		Deletes a file from the disk.
	DIRECTORY(42)		Displays a directory of the disk.
	FORMAT(42)		Prepares a disk so files can be created on it.
	PROTECT(42)		Prevents the the terminal from writing to a disk or to a specific disk file.
	RENAME(42)		Renames a disk file.

[a]**Saved in Setup Memory.**

Table 6-1 (cont)
SETUP COMMANDS

Function	Command	Factory Default	Purpose
Three port peripheral interface control	PASSIGN(10)		Assigns a device to a port.
	[a]PBAUD(10)	2400	Sets baud rate for a port.
	[a]PBITS(10)		Sets the stop and data bits for a port.
	[a]PEOF(10)		Defines the end of file string for a port.
	[a]PEOL(10)		Defines the end of line string for a port.
	[a]PFLAG(10)		Establishes type of flagging for a port.
	STATUS(10)		Returns status on specified port.
	[a]PPARITY(10)		Establishes the parity checking used on a port.
Standard Communications	[a]BAUDRATE	2400, 2400	Sets transmit and receive baud rates.
	[a]BREAKTIME	200	Defines length of break transmission in milliseconds.
	[a]BYPASSCANCEL	^{L}F	Sets bypass cancel character.
	[a]EOFSTRING	''	Defines end of file string.
	[a]EOLSTRING	$'^{C}R$	Defines end of line string.
	[a]EOMCHARS	$'^{C}R\ ^{L}F$	Defines end of message characters.
	[a]FLAGGING	None	Specifies kind of flagging done by the terminal.
	[a]PARITY	None	Specifies the kind of parity sent from the terminal.
	[a]PROMPTMODE	No	Specifies whether or not the terminal is in Prompt mode.
	[a]PROMPTSTRING	' '	Specifies prompt used when terminal is in Prompt mode.
	[a]QUEUESIZE	300	Sets size of terminal input queue.
	[a]STOPBITS	1	Specifies number of stop bits the terminal uses.
	[a]XMTDELAY	100	Specifies length of time terminal waits after receiving end of message character.
	[a]XMTLIMIT	19200	Specifies maximum transmit rate for the terminal.

[a]**Saved in Setup Memory.**

Table 6-1 (cont)
SETUP COMMANDS

Function	Command	Factory Default	Purpose
Optional Communications	[a]BCONTINUECHARS(01)	& &	Specifies block continue characters.
	[a]BENDCHARS(01)	$ $	Specifies end of block characters.
	[a]BHEADERS(01)	'HEADTX' 'HEADRX'	Specifies block header sequences.
	[a]BLENGTH(01)	256 256	Specifies block length.
	[a]BLINELENGTH(01)	70	Specifies line length for Block mode transmission.
	[a]BLOCKMODE(01)	No	Specifies whether or not terminal is armed for Block mode.
	[a]BMASTERCHARS(01)	# #	Specifies master characters for Block mode.
	[a]BNONXMTCHARS(01)	'#$&' '#$&'	Specifies non transmittable characters for Block mode.
	[a]BPACKING(01)	7, 6, 7, 6	Specifies manner in which blocks are packed.
	[a]BTIMEOUT(01)	0	Specifies how long terminal waits before retransmitting previous block of data.
	[a]DUPLEX(01)	Full	Specifies which Duplex mode the terminal is in.
Graphic Tablet	[a]TBFILTER(13/14)		Specifies tablet data filtering.
	[a]TBHEADERCHARS(13/14)	LETTERS	Specifies tablet stroke header characters.
	TBSTATUS(13/14)	IN	Specifies if status byte is sent when input device is lifted from tablet surface.

[a]Saved in Setup Memory.

BAUDRATE

Specifies the transmitting and receiving baudrates for communications between the terminal and the host computer.
(Memory)

BAUDRATE trans, rec

trans is an integer parameter specifying the baud rate at which the terminal transmits data. Valid baud rates are the same as those listed for "rec." A value of 1 indicates the use of an external device to control the transmission rate. The factory default baud rate is 2400.

rec is an integer parameter specifying the baud rate at which the terminal expects to receive data from the host. A value of 1 indicates external clocking. Valid specifications for "trans" and "rec" are listed. If you enter no value for "rec," it is set to the same value as "trans." The factory default is 2400.

Valid baud rates are:

50
75
110
134 (sets baud rate to 134.5)
150
300
600
1200
1800
2000
2400
4800
9600

NOTE

The terminal's baud rate settings must be compatible with the transmit and receive baud rates used by the host computer.

The transmit and receive rates do not have to be set to the same values. Also, either the transmit or receive-rate can be specified by an external clock while the other is specified by the terminal firmware.

The following command sets the transmit rate at 9600 baud and the receive rate at 2400 baud.

BAUDRATE 9600, 2400 ^{C_R}

The following command sets the transmit baud rate to 2400 while the receive baud rate is controlled by an external clock.

BAUDRATE 2400 1 ^{C_R}

The baud rate parameter may be set for you by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BCONTINUECHARS (Option 01)

Sets the line continue characters for block mode. (Memory)

BCONTINUECHARS xmt, rec

xmt is a single character representing the transmit continue character. If you do not enter a value for "xmt" it is set to null. Any ASCII character is valid; the factory default is an ampersand (&).

rcv is an single character representing the receive block continue character. If you do not enter a value for "rcv" it is set to null. Any ASCII character is valid; the factory default is an ampersand (&).

This command cannot be used if the terminal is already in or armed for Block mode. See the BLOCKMODE command.

If you enter just the command word and no parameters, both "xmt" and "rcv" are set to null.

The terminal sends the "xmt" character to the host to indicate when a block continues to the next line. When the terminal receives the "rcv" block continue character, the terminal knows that the block it is receiving continues to the next line.

The following command specifies the percent (%) as the transmit block continue character and the asterisk (*) as the receive block continue character:

BCONTINUECHARS % * $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BENDCHARS (Option 01)

Sets the block end characters for Block mode. (Memory)

BENDCHARS xmt, rcv

xmt is a single character representing the transmit block end character. If you do not enter a value for "xmt" it is set to null. Any ASCII character is valid; the factory default is the dollar sign ($).

rcv is a single character representing the receive block end character. If you do not enter a value for "rcv" it is set to null. Any ASCII character is valid; the factory default is the dollar sign ($).

This command cannot be used if the terminal is already armed for Block mode. See the BLOCKMODE command.

If you enter just the command word and no parameters, both "xmt" and "rcv" are set to null.

The terminal transmits the "xmt" character to the host to indicate the last line of a transmission in Block mode. When the terminal receives the "rcv" character, the terminal knows that it is the last line of a block transmission.

The following command sets the block end transmit character to the pound sign (#) and the block end receive character to the ampersand (&):

BENDCHARS # & $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BHEADERS (Option 01)

Sets the transmit and receive header sequences for Block mode.
(Memory)

BHEADERS $^{D}_{L}$xmt$^{D}_{L}$, $^{D}_{L}$rcv$^{D}_{L}$

$^{D}_{L}$ is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.

xmt is a delimited character string representing the transmit block header sequence. Any string of up to ten ASCII characters can be used (restrictions are listed under "rcv"). The factory default "xmt" header is HEADTX.

rcv is a delimited character string representing the receive block header sequence. The factory default "rcv" header is HEADRX; "rcv" is set to null if no string is specified. Any string of up to ten ASCII characters can be used with the following restrictions:

- the first character cannot be repeated in the sequence.
- the sequence cannot include any characters which are part of the prompt string (see PROMPTSTRING).
- the sequence should be one that would never appear in normal text.

This command cannot be used if the terminal is already in or armed for Block mode. See the BLOCKMODE command.

If you enter just the command word and no parameters, both "xmt" and "rcv" are set to null.

When the terminal sends the "xmt" sequence, it signals the host that a block of packed data follows immediately. When the terminal receives the "rcv" sequence, it signals the terminal that a block of packed data follows immediately.

The following command sets the transmit block header to "+ send" and the receive block header to "+ recve":

BHEADERS /+send/ /+recve/ $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BLENGTH (Option 01)

Sets the block length for Block mode.
(Memory)

BLENGTH xmt, rcv

xmt is an integer parameter representing the transmit block length. The valid range of values is from five to 65535 lines. The factory default is 256.

rcv is an integer parameter which is the receive block length value. The values range from five to 65535 lines. The factory default is 256.

This command cannot be used if the terminal is already armed for block mode. See the BLOCKMODE command.

The following command sets the block length to 100:

BLENGTH 100 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BLINELENGTH (Option 01)

Specifies the maximum length line that the terminal will transmit in Block mode.
(Memory)

BLINELENGTH length

> **length** is an integer parameter specifying the maximum number of characters the terminal can transmit in one line in Block mode. The valid range is from 12 to 65535 characters. The factory default is 70 characters.

This command cannot be used if the terminal is already armed for Block mode. See the BLOCKMODE command.

The specified line length has no effect on input to the terminal for communications in non-Block mode.

The following command sets the line length to 55:

BLINELENGTH 55 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BLOCKMODE (Option 01)

Specifies that the terminal will go into Block mode as soon as it receives a block header.
(Memory)

BLOCKMODE {**yes**}
{**no**}

> **yes** is a keyword parameter specifying that the terminal should enter Block mode when it receives the next block header (see BHEADER). If you enter just the command word and no parameter, the terminal assumes "yes" and arms itself for Block mode.
>
> **no** is a keyword parameter specifying that the terminal is not in Block mode. If the terminal is in Block mode when you enter this command, it exits from Block mode immediately. This is the factory default.

Attempting to arm the terminal for Block mode when it is already in Block mode will cause communications difficulties with the host computer.

The following command causes the terminal to enter Block mode:

BLOCKMODE yes $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BMASTERCHARS (Option 01)

Sets the master characters for Block mode. (Memory)

BMASTERCHARS xmt, rec

xmt is a single character the transmit "master character" for block mode. If you do not enter a value for "xmt" it is set to null. Any ASCII character is valid; the factory default is the pound sign (#).

rcv is a single character representing the receive "master character" for Block mode. If you do not enter a value for "rcv" it is set to null. Any ASCII character is valid; the factory default is the pound sign (#).

This command cannot be used if the terminal is already armed for Block mode. See the BLOCKMODE command.

If you enter just the command word and no parameters, both "xmt" and "rcv" are set to null.

The following command sets the transmit master character to an asterisk (*) and the receive master characer to a dollar sign ($):

BMASTERCHARS * $ ^{C_R}

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BNONXMTCHARS (Option 01)

Sets the non-transmittable characters for Block mode. (Memory)

BNONXMTCHARS ^{D_L}xmt^{D_L}, ^{D_L}rcv^{D_L}

^{D_L} is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.

xmt is a string of up to 20 ASCII characters representing the non-transmittable characters in Block mode. The factory default string is the pound sign, dollar sign and ampersand ('#$&').

rcv is a string of up to 20 ASCII characters representing the non-receivable characters in Block mode. The factory default string is the pound sign, dollar sign, and ampersand ('#$&').

This command cannot be used if the terminal is already armed for Block mode. See the BLOCKMODE command.

If you enter just the command word with no string for either "xmt" or "rcv," both are set to null.

A character is to be "non-transmittable" because it has a special meaning to the terminal (such as the block continue, block end, or block master characters) or host.

The following command sets the plus sign (+) and per cent sign (%) as non-transmittable characters for both transmitting and receiving:

BNONXMTCHARS '+&' '+&' ^{C_R}

These characters may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BPACKING (Option 01)

Sets the transmitting and receiving packing values for Block mode.
(Memory)

BPACKING xmt, xmtpack, rcv, rcvpack

xmt must be set to either 7 or 8 bits. It specifies the size of each byte that is transmitted. The factory default is 7 bits.

xmtpack must be set to 6, 7, or 8. It specifies the number of effective data bits per word that is transmitted. The factory default is 6.

rcv must be set to either 7 or 8. It specifies the size of each byte received from the host. The factory default is 7.

rcvpack must be set to 6, 7, or 8. It specifies the number of effective data bits per word that is received. The factory default is 6.

This command cannot be used if the terminal is already armed for Block mode. See the BLOCKMODE command.

The following command sets the "xmt" value to 8, the "xmtpack" value to 7, the "rcv" value to 8, and the "rcvpack" value to 7:

BPACKING 8 7 8 7 $^{C}_{R}$

These parameters may be set by an applications program on the host computer or by an initialization file on a flexible disk (see Section 3). If it is not, you may need to consult with a system programmer for your host computer before using this command.

BREAKTIME

Specifies the length of the break transmission.
(Memory)

BREAKTIME ms

ms is an integer parameter specifying the length of the interrupt, in milliseconds. The valid range is from 0 to 65535. The factory default is 200 ms. Setting "ms" to 0 means that no break transmission is sent. If you do not enter any value for the parameter, it is set to zero.

When you press the BREAK key, an interrupt is sent by the terminal to the host. The purpose of the interupt is to halt execution of a host resident program or transmission of data from the host.

Different host computers must receive an interrupt for different amounts of time to recognize it and break off the transmission. The length of the interrupt is determined by the BREAKTIME command.

The following command sets the interrupt time to 400 ms.

BREAKTIME 400 $^{C}_{R}$

Since the interrupt is handled differently by different hosts, you should check with a system programmer for your host before using the BREAKTIME command or the BREAK key.

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk (see Section 3). If it is not, you may need to consult with a system programmer for your host computer before using this command.

BTIMEOUT (Option 01)

Determines how long the terminal will wait for a response from the host before retransmitting the previous block.
(Memory)

BTIMEOUT seconds

> **seconds** is an integer parameter specifying the number of seconds the terminal will wait before it retransmits a block of data. Valid values are from 0 to 65535; a setting of 0 disables the feature. If you do not enter a value for the parameter, it is set to zero. The factor default is zero.

If the setting is too long, you would have to wait a long time to become aware that there was a problem such as a block being lost. If it is too short, the terminal might try to send a block again just because the host was slow to respond. Setting it to zero disables the timeout feature.

The following command sets the "time out" to 2 seconds:

BTIMEOUT 2 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

BYPASSCANCEL

Specifies the bypass cancel character.
(Memory)

BYPASSCANCEL char

> **char** is an un-delimited single ASCII character. This parameter is usually set to either a $^{L}_{F}$ (line feed) or a $^{C}_{R}$ (carriage return). The factory default is the $^{L}_{F}$. If no character is specified the parameter is set to null and the bypass feature is turned off.

Remote echo computers use Bypass mode to suppress the echoing of unnecessary control characters. The terminal uses the bypass cancel character to get out of Bypass mode.

The following command sets the bypass cancel character to carriage return. The "tilda" above the first carriage return indicates that the carriage return was preceded by the literal character (see EDITCHARS).

BYPASSCANCEL $\tilde{C}_{R}$ C_{R}

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk (see Section 3). If it is not, you may need to consult with a system programmer for your host computer before using this command.

COPY

Copies data from one device to another.

COPY {source} {filename} **to** {destination} {filename}

source is a device identifier indicating the location of the file to be copied. If no device is specified, the terminal defaults to the disk drive. Valid devices are:

HO: host computer
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

filename is an un-delimited character string. On the "source" side of the command (before the word "to"), it identifies the name of the file to be copied. On the "destination" side of the command (after the word "to"), it identifies the name of the file into which the copy is to be made. Only use a filename when the file (either source or destination) is on a disk drive.

to is a keyword parameter which helps you remember the flow flow of the copy — **from** the source **to** the destination.

destination is a device identifier indicating the location to which a copy is to be made. The terminal defaults the disk drive if no device is specified. Valid device indentifiers are the same as for "source."

The copy operation continues until:

- an "end of file" string (EOF) is encountered
- you press the CANCEL key
- an error is detected

An error can be caused by many factors. For example, if the destination device is a disk drive an error can occur if:

- there is no disk in the destination drive
- the disk in the destination drive is write protected
- the destination drive is trying to write onto a bad disk area
- there isn't enough room on the destination disk to hold the entire output file

If a COPY is invoked in Setup mode and you enter anything from the terminal's keyboard during the copy, the first 130 characters are queued and executed when the COPY is finished. After you enter 130 characters, the bell rings each time you press any key (except CANCEL).

Before you copy a file to a disk you should make sure that there is enough room for the file. You can determine the length of an individual file as well as the room available on the destination disk with the DIRECTORY command.

You cannot copy to a file or disk that is write protected. A disk can be write protected by the WRITE PROTECT switch (see Section 2) or by uncovering the notch in the disk (see Section 3). A file or disk can be write protected by the PROTECT setup command (described later in this section).

If there is already a file on the destination disk by the name you specify and it is not write protected, the existing file will be replaced. If the existing file is write protected, the copy fails and an error message is displayed.

The following command copies data from file A on the disk to file B on the disk:

COPY A0: to B1: ^{C_R}

This command copies "stuff" from the disk to the output device attached to peripheral port 2:

COPY stuff to P2: ^{C_R}

Copying To and From the Host

If the source device is the host (HO:), you must begin the data transmission before you enter the COPY command from Setup mode. Once you enter the command, the terminal will wait for a data stream from the host until an end-of-file string comes from the host or until the CANCEL key is pressed.

One way to arrange the data stream is:

- Type the command to the host to begin the data stream without the end-of-command character from outside of Setup mode.
- Enter Setup mode.
- Type in the COPY command with the desired destination.

- Exit Setup mode (after you press the SET UP key, the light in the key stays on, but data is being transmitted to the host).
- Type the end-of-command character. If your host echos the character typed on the terminal, the above results in the end-of-command character being the first character in the data stream.

Another way to do so is:

- Program the host so that there is a delay between the time you enter the end-of-command character for the host-data-send command and the time the data stream begins to flow. If you use this method you should give yourself enough delay-time to enter Setup mode and type the COPY command with the desired destination. You may also want to use the QUEUESIZE command to create a large input queue.

If you specify the host as destination, then be sure that the host is ready to accept data, or you may lose your data and/or cause problems with the host.

CRLF

Specifies whether or not a carriage return character is interpreted as a carriage return or a carriage return-line feed.
(Memory)

CRLF {**yes**}
{**no**}

> **yes** is a keyword indicating that the terminal should attach a line feed character to each carriage return character.
>
> **no** is a keyword indicating that the terminal should not attach a line feed character to the carriage return. This is the factory default.

A carriage return moves the cursor back to the left margin of the current line; a line feed moves the cursor down one line from its current position.

The following command causes the terminal to attach the line feed character to the carriage return:

CRLF yes $^{C}_{R}$

If you set the parameter to "yes" and the host automatically sends a line feed character with the carriage return, the terminal "double spaces" each time you press the RETURN key because it is getting a line feed from the host and from the terminal.

If the terminal is in Local mode, pressing the RETURN key only returns the cursor to the left margin of the current line. You must press the LINE FEED key to move to the next line or set CRLF to "yes" so that the RETURN key echoes a carriage return line feed combination.

When the terminal is in Setup mode, pressing the RETURN key always gives the carriage return-line feed combination regardless of the CRLF specification.

DABUFFER

Specifies the total size of the dialog buffer.
(Memory)

DABUFFER lines

> **lines** is an integer parameter which specifies the size of the dialog buffer. The minimum dialog buffer size is two; the factory default is 15.

A change in the dialog's buffer size becomes effective the next time the dialog area becomes visible.

The maximum dialog buffer depends on the amount of available memory. If you specify a dialog buffer larger than the currently available memory, it is set to the maximum currently available memory. If more memory becomes available later, the size of the dialog buffer is increased accordingly.

If the buffer is full when a line of text is received, the first line is deleted to make room for a new line at the end.

The following command increases the dialog buffer so it will hold 30 lines of data:

DABUFFER 30 $^{C}_{R}$

If you change the size of the scroll buffer, the information in the dialog area is not lost or changed, unless the size is reduced. In that case, the most recent dialog is maintained in the buffer. The new size takes affect the next time the dialog area becomes visible.

DACHARS

Specifies the maximum number of characters per line in the dialog buffer.
(Memory)

DACHARS width

> **width** is an integer parameter indicating the maximum number of characters per line. The factory default is 80. The minimum width allowed is 5; the maximum number of characters allowed per line is 80.

If you specify more than 80 characters per line, the terminal reports an error. If you specify more characters than fit due to the current dialog area position, the dialog area is moved to allow the specified width. The following "warning" is displayed the next time the dialog area becomes visible (you will not see the warning if the terminal's error reporting mechanism is set to suppress warnings):

```
>> Terminal Issues Message LV03:
>>      Dialog Parameters Modified.
```

The terminal remembers the width you specified and if the dialog area position is changed, the width parameter is automatically changed accordingly.

If you change the width, information in the dialog area is not lost. The new width takes effect the next time the dialog area is visible, and when the next text is received in the dialog area.

The following command changes the maximum width of each line in the dialog area to 35:

DACHARS 35 $^{C}_{R}$

DAENABLE

Enables or disables the dialog area.
(Memory)

DAENABLE {**yes**}
{**no**}

yes is a keyword parameter which enables the dialog area. If you do not specify "yes" or "no," the parameter is set to "yes."

no is a keyword parameter which disables the dialog area. This is the factory default.

When the dialog area is enabled, all alpha (non graphics) text is directed to it. The dialog area itself is made visible by a separate command (see the DIALOG key in Section 2 and the DAVIS command in this section). If the dialog area is enabled when you turn the terminal on, the dialog area will automatically be visible.

When the dialog area is not enabled, alpha text is directed to the graphics area of the screen. If the dialog area is visible but not enabled, you can see text written in the dialog area before it was disabled but the subsequent text will not go into the dialog area.

When the dialog area is enabled, pressing the RETURN key does not cause the terminal to exit from Vector, Marker, or 4010-style GIN mode. If you are in one of those modes at the end of a program, press the CANCEL key to return the terminal to Alpha mode.

The following command enables the dialog area:

DAENABLE $^{C}_{R}$

DAINDEX

Specifies gray indices in the dialog area.
(Memory)

DAINDEX character, background, wipe

character is an integer parameter specifying the gray index of text in the dialog area. For all three parameters, the lower the integer, the darker the gray index. The factory default for the "character" parameter is white.

background is an integer parameter specifying the gray index of the part of the character cell in which the character is not drawn. The factory default is black.

wipe is an integer parameter specifying the gray index of the dialog area when it is wiped (erased). The factory default is black.

The indices you can use depend on the number of planes your terminal has access to. Without any option, your 4112 has one plane (0 is black and 1 is white). If your terminal has the three-bit plane option there are three planes (0 is black and 7 is white).

When you change a dialog area index, it is effective the next time the dialog area is made visible.

Note that if you were to set all three indices to 0, the dialog area would be invisible because you would have black characters and black character cells on a black background. Text in the dialog area would also be invisible if you set all three indices to 7 since you would have white characters and character cells on a white background.

If you enter just the command and no parameters, all three indices are set to 0 causing the dialog area to become invisible the next time it is turned on.

DALINES

Specifies the number of lines of the dialog buffer displayed on the screen.
(Memory)

DALINES lines

> **lines** is an integer parameter, specifying the number of lines in the dialog buffer which are to be visible. The factory default is five. The valid range is from 2 to 34.

If you specify more visible lines than the current dialog area position allows, the terminal automatically changes the position to allow the specified number of lines. In that case the following warning is displayed the next time the dialog area becomes visible (you will not see this warning if the terminal's error reporting mechanism is set to suppress warnings):

```
>> Terminal Issues Message LV03:
>>     Dialog Parameters Modified.
```

If you change the number of visible lines, the information in the dialog buffer is not changed or lost. The change is effective the next time the dialog area is made visible (see DAVIS).

The following command causes 15 lines of the dialog buffer to be displayed at a time:

DALINES 15 $^{C}_{R}$

DAMODE

Specifies the manner in which characters in the dialog area are replaced.
(Memory)

DAMODE {**replace**}
{**overstrike**}

> **replace** is a keyword meaning that the character is erased as you backspace over it. This is the factory default.
>
> **overstrike** is a keyword meaning that the character is not erased as you backspace over it.

If you change the setting of DAMODE, it is effective the next time the dialog area becomes visible.

The following command changes the DAMODE setting to overstrike:

DAMODE overstrike $^{C}_{R}$

DAPOSITION

Specifies the location of the lower left corner of the dialog area.
(Memory)

DAPOSITION x, y

x is an integer parameter specifying the X-axis coordinate of the lower left corner of the dialog area.

y is an integer parameter specifying the the Y-axis coordinate of the lower left corner of the dialog area.

Figure 6-3 illustrates the limits of the screen coordinate system for this command.

The terminal will not allow you to position the dialog area off the screen. An error results if you enter "x" or "y" coordinates that are not within the extents shown in Figure 6-3. The position is automatically adjusted if the current DACHARS and DALINES settings would not fit in the specified position. The terminal remembers the specified location and if the line width or number of visible lines is changed, the location of the dialog area is automatically changed appropriately.

If the dialog area position is modified, the following warning is displayed the next time the dialog area becomes visible (you will not see the warning if the terminal's error reporting mechanism is set to suppress warnings):

```
>> Terminal Issues Message LV03:
>>      Dialog Parameters Modified.
```

If you change the position of the dialog area information already in the dialog area is not lost. It appears at its new position the next time the dialog area is made visible (see DAVIS).

The following command moves the lower left corner of the dialog area near the center of the screen (if DACHARS is set to 40 or more or if DALINES is set to 17 or more, the terminal adjusts the actual position):

DAPOSITION 2049, 1561 ^{CR}

DASURFACE

Specifies screen surface on which dialog area is to appear.
(Memory)

DASURFACE surface

surface specifies the screen surface on which the dialog should appear. Valid surface values are 1 (the front surface), 2, and 3 (the back surface). The factory default is 1. Your terminal should have the three bit plane memory (Option 20) to use surfaces 2 and 3.

Putting the dialog area and graphics on different surface allows you to have a visible dialog area and still use the full screen for graphics. However, graphics on a surface in front of the dialog area obscures the portion of the dialog area where the two overlap.

Before DASURFACE is efective, surfaces must be defined for the terminal. There is no setup command for doing so. See the <**set-surfaces-definition**> command in the 4110 Series Command Reference Manual for information.

DAVIS

Specifies whether or not the dialog area is visible.

DAVIS {**yes**}
{**no**}

> **yes** is a keyword specifying that the dialog area should be visible. This is the power up and reset default if the dialog area is enabled. If you do not enter "yes" or "no," the terminal defaults to "yes."
>
> **no** is a keyword specifying that the dialog area should not be visible. This is the power up and reset default if the dialog area is not enabled.

Entering this command has the same effect as pressing the DIALOG key. When you use DAVIS to make the dialog area visible, the light in the DIALOG key turns on. The light turns off when you use DAVIS to make the dialog area not visible.

If the dialog area is enabled when the terminal is turned on, the dialog area is automatically visible.

When you enter the command, it takes effect immediately.

You can use the DAVIS command to update the dialog area if you change any of its attributes. For example, if you change the dialog area's position and the number of visible lines, you can enter the following command to make the changes effective:

DAVIS $^{C}_{R}$

Even though the dialog area may already be visible, this command updates it as though the dialog area were turned off and back on again.

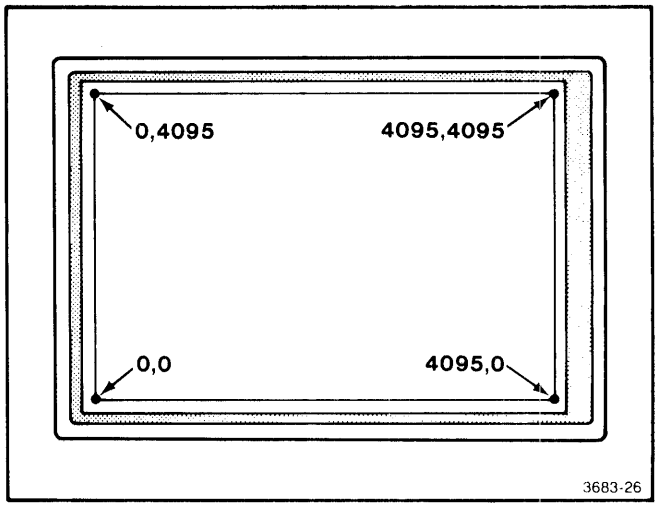


Figure 6-3. DAPOSITION Extents.

DEFINE

Defines a macro and associates it with a key or an integer.

DEFINE [key]
[**all**]

DEFINE key, ^{D_L}macro^{D_L}

DEFINE key, ^{D_L}macro [^{K_E} macro ^{K_E}] ^{D_L}

key specifies the key or integer with which this macro is to be associated. If the command is terminated after this parameter, the macro associated with the specified key or integer is deleted.

all is a keyword indicating that all existing macros should be deleted.

^{D_L} is the delimiter character. You can use any ASCII character to delimit the macro except:

- a comma
- a space
- the "character delete" character (see EDITCHARS)
- the "line delete" character (see EDITCHARS)
- the "literal" character (see EDITCHARS)

The first character after the "key" parameter that is not a space, comma, or edit character is the delimiter for that macro. The delimiter should not occur in the macro itself, since the second use of the delimiter terminates the macro. The same delimiter must be used to terminate the macro.

macro is the character or character string transmitted when you press the specified key or when the host executes the integer associated with the macro. The total length of the command (including the command word and macro) can be a maximum of 130 characters. If you want to include a "carriage return" or one of the edit characters in a macro, it must be preceded by the "literal" character — see the EDITCHARS command.

^{K_E} is the "key execute delimiter" character. When this character appears in a macro, it changes the direction of the macro expansion. If it was being expanded to the host, the key execute delimiter causes it instead to be expanded to the terminal; if the terminal had been receiving the macro it is subsequently sent to the host. The expansion of a subsequent macro starts in the same direction as when the last macro ended. The "key execute delimiter" is defined by the KEYEXCHAR command.

Specifying a Key or Integer

Indicate an alphanumeric key you want to associate with a macro by pressing the key or by entering its ADE value.

For example, the following commands both assign a macro to the upper case A character. After this command, whenever you press the upper case A character "a macro or string" would be transmitted.

DEFINE A /a macro or string/ ^{C_R}

DEFINE 65 /a macro or string/ ^{C_R}

Each function key can be identified by a mnemonic abbreviation or by an integer as in Table 6-2.

Table 6-2
FUNCTION KEY IDENTIFIERS

Mnemonic	Integer	Key
F1	(128)	function key 1
F2	(129)	function key 2
F3	(130)	function key 3
F4	(131)	function key 4
F5	(132)	function key 5
F6	(133)	function key 6
F7	(134)	function key 7
F8	(135)	function key 9
S1	(136)	shifted function key 1
S2	(137)	shifted function key 2
S3	(138)	shifted function key 3
S4	(139)	shifted function key 4
S5	(140)	shifted function key 5
S6	(141)	shifted function key 6
S7	(142)	shifted function key 7
S8	(143)	shifted function key 8

The following commands show the two ways to assign "a macro or string" to function key 1.

DEFINE F1 /a macro or string/ ^{C_R}

DEFINE 128 /a macro or string/ ^{C_R}

Associating a Macro with an Integer

You can assign an integer value for the "key" to cause the macro to be associated with the specified integer. The valid range of integers is 144 to 34767. A macro associated with an integer can only be executed by a command from the host (or by using a special form of commands called "escape sequences").

The following command shows how to associate "a macro or string" with the integer 1000. An actual macro would be accessed by the host issuing a command to the terminal:

DEFINE 1000 /a macro or string/ ^{C_R}

Carriage Return and Edit Characters

Since the carrige return control character is executed in Setup mode when you press the RETURN key, it must be treated specially to be part of a macro. The same is true for the keys reserved as edit characters (see EDITCHARS).

The third character defined by EDITCHARS is the "literal" character. To include a carriage return or edit character in a macro, precede it with the literal character.

The factory default literal character is the tilda (~). In the following example, the tilda key is pressed before the RETURN key, causing the $^{C}_{R}$ to be displayed as a snoopy character (instead of being executed) and a carriage return becomes part of the macro. (The tilda is shown above the $^{C}_{R}$ because the character entered after the literal character replaces it.)

If you enter a return before the literal character, the carriage return is processed and an error message is displayed because you improperly terminated the DEFINE command. If you press the "character delete" before the tilda, the terminal backspaces and deletes the last character entered. If you press the line edit key before you press the literal key, the current line is deleted.

Using the Key Execute Delimiter

When a macro is expanded by pressing the key, every time the terminal encounters the key execute delimiter it reverses the direction of the macro expansion. (When a macro is expanded by the "expand" command from the host or from the terminal in the escape sequence format, the key execute delimiter character is ignored.)

In the following example, the macro starts by being expanded to the host. When it encounters the key execute toggle delimiter, it changes direction and is expanded to the terminal. The second key execute delimiter directs the flow back to the host again.

DEFINE 128 /FORT GRAF $^{C}_{R}$ $^{K}_{E}$ ^{E}**cSO1** $^{K}_{E}$ **EX GRAF** $^{C}_{R}$/ $^{C}_{R}$

The initial direction of a macro depends on how it was set in the last macro that was expanded. In the previous example, the second key execute toggle character returns the flow toward the host. If it had not been included, the next macro would have been expanded by the terminal until another key execute toggle character was entered.

DELETE

(Option 42)

Deletes the specified file.

DELETE {filename}

filename is an un-delimited character string identifying the name of the file to be deleted.

An error results if the disk is write protected (by the WRITE PROTECT switch or the write protect notch in the disk) or if the file you try to delete is write protected by the PROTECT setup command.

The following command would delete the file named "info" from the disk:

DELETE 1 info

```
*DIRECTORY STUFF
-NAME          BLOCKS  BYTES  PROTECT
F0:PRACTICE      2002             NO
   STUFF            4     99     NO
-
-ENTRIES USED:      4
-ENTRIES FREE:    364
-BLOCKS USED:      42
-BLOCKS FREE:    1960
-LARGEST FREE:   1960
*
```

3683-27

Figure 6-4. The Directory of a File.

DIRECTORY (Option 42)

Displays or copies the file directory of the specified disk.

DIRECTORY [filename] **to** [filename]
[destination]
[desintation-filename]

if no parameter is specified, the directory of the disk is displayed on the screen.

filename identifies the specific file on the disk for which a directory is to be displayed.

to helps identify the direction if the directory is being copied.

destination indicates the location to which a directory is to be copied. The "source" and "destination" can be the disk. If no device name is specified, the terminal defaults to the screen. Valid device names are:

none	the screen
HO:	the host port
P0:	peripheral port 0
P1:	peripheral port 1
P2:	peripheral port 2
Filename:	file on the disk

destination-filename indicates the specific file to which a directory is to be copied.

The following command causes the directory of a file named "stuff" to be displayed on the screen (Figure 6-4).

DIRECTORY 0 stuff ^{C_R}

The directory entry for each file includes the file's name, size in blocks, the number of bytes in its last block, and whether or not it is write protected.

The following command displays the directory for the entire disk in drive 0 (Figure 6-5).

DIRECTORY ^{C_R}

If you specify the host as destination for the directory, then be sure that the host is ready to accept data. If you do not, you may lose your data stream causing communications problems with the host.

DUPLEX (Option 01)

Specifies what kind of duplex mode the terminal is in. (Memory)

DUPLEX {**full**}
{**normal**}
{**auto**}
{**super**}

full specifies that the terminal is in full duplex. This is the factory default communications mode for a terminal with Option 01 installed. If you enter the command word and no parameters, the terminal defaults to "full."

normal specifies that the terminal is to be in "Half Duplex Normal" mode.

auto puts the terminal into "Half Duplex with Automatic Request to Send" mode.

super puts the terminal into "Half Duplex with Supervisor" mode.

Section 5 in this manual provides a very brief overview of the terminal's communications modes. See the 4112 Host Programmer's Manual for information.

```
*DIRECTORY
-NAME          BLOCKS BYTES PROTECT
F0:PRACTICE      2002           NO
   STUFF            4    99     NO
   HOUSE            4    40     NO
   DATA             4    40     NO
   INFO             4    40     NO
-
-ENTRIES USED:      4
-ENTRIES FREE:    364
-BLOCKS USED:      42
-BLOCKS FREE:    1960
-LARGEST FREE:   1960
*
```

3683-28

Figure 6-5. Directory for a Disk.

ECHO

Specifies whether or not the terminal provides an "echo" of characters typed on the keyboard and sent to the host computer.

ECHO {**yes**}
{**no**}

yes is a keyword meaning that the terminal will provide the echo. If you enter the command word but do not specify "yes" or "no," the terminal defaults to "yes."

no is a keyword meaning that the terminal will not provide the echo. This is the factory default.

The "echo" is the display of characters on the terminal screen as they are typed at the keyboard.

When the terminal is in Setup or Local modes, the echo is provided by the terminal regardless of the status of ECHO.

If the computer you log onto is a remote-echo host, it provides the echo for you automatically. If you set the terminal to provide the echo, too, each character is echoed twice when you are in communications with the host — once by the terminal and once by the host.

If the computer you log onto is not a remote-echo host, enter the following command so the terminal will provide the echo when you are communicating with the host:

ECHO yes ^{C}R

EDITCHARS

Specifies "delete" characters for Setup mode and a "literal" character to precede control characters in macros.

EDITCHARS chardel, linedel, literal

chardel can be any ASCII character except the "linedel" or "literal" characters. Specify a character by pressing the appropriate key or entering its ADE (ASCII Decimal Equivalent) value. The "chardel" character is used in Setup mode to backspace along a line to correct a typing error. The factory default is RUB OUT (ADE 127).

linedel can be any ASCII character except the "chardel" or "literal" characters. Specify a character by pressing the approriate key or entering its ADE value. The "linedel" character is used in Setup mode to abort a line without executing it, and start a new line. The factory default is "cancel" (ADE 24). Type the cancel character from the keyboard by pressing the CTRL and X keys at the same time.

literal can be any ASCII character except the "chardel" or "linedel" characters. Specify a character by pressing the appropriate key or entering its ADE value. The factory default literal character is the "tilda" (~) (ADE 126). The "literal" must precede the following characters if they are included in a macro:

a carriage return
the character delete
the line delete
the literal character

The following command specifies the "chardel," "linedel," and "literal" characters as the percent sign (%), pound sign (#), and dollar sign ($), respectively.

EDITCHARS % # $ ^{C}R

If you are in Setup mode and you need to delete one or more characters, press the "%" key to back up along that line to correct or change it. Press the "#" key to back up to the beginning of the line.

If the dialog area is not used, the terminal will always overwrite characters when you use the "chardel" key — it will not replace the characters. If the dialog area is not used, the "linedel" key moves down to the next line instead of to the beginning of the current line.

Use of the literal character ($ in this example) is illustrated with the DEFINE command.

You don't have to enter all three parameters if you only want to change the first or second edit characters. For example, if you only want to change the "character delete" character, you could enter the following command:

EDITCHARS & $^{C}_{R}$

That command changes the character delete character to & and leaves the line delete and literal characters at their current specifications.

If you specify the same character for all three parameters, it will be used as the character delete character.

EOFSTRING

Specifies the string which is automatically sent to the host by the terminal when it detects an end of file and which the terminal uses to end a file being sent to it during a SPOOL, COPY, or PCOPY.
(Memory)

EOFSTRING $^{D}_{L}$string$^{D}_{L}$

> $^{D}_{L}$ is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.
>
> **string** is a delimited combination of up to ten ASCII characters. The first character cannot be repeated elsewhere in the string. The default is the null string. If you enter just the command word and no string, the terminal defaults to the null string.

When the terminal is copying a file from the host, it terminates the copy when it encounters the end of file string.

The end of file string should be set to whatever the host computer actually sends at the end of a file. If you do not, you will have to press the CANCEL key to close disk files.

In the following command, the slash (/) is the delimiter and the end of file string is set to an ampersand (&).

EOFSTRING /&/ $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk (see Section 3). If it is not, you may need to consult with a system programmer for your host computer before using this command.

EOLSTRING

Assigns the end of line string used by the terminal. (Memory)

EOLSTRING $^{D}_{L}$ string $^{D}_{L}$

$^{D}_{L}$ is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, character (see EDITCHARS) is the delimiter character for this command.

string is the character string the terminal uses to signal the end of the transmission of a line of data. The factory default is the carriage return.

The "end of line string" is sent by the terminal at the end of (or during) transmissions generated by the terminal.

When a host program requests information from the terminal, it may be sent automatically, not requiring operator interaction. For example, the program may request a report of error messages or some graphics input data. In such cases, the terminal sends an end of line character to the host to indicate the end of a transmission.

The following command changes the end of line string to the $^{L}_{F}$ (line feed):

EOLSTRING '$^{L}_{F}$' $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

EOMCHARS

Specifies the end-of-message (turnaround) characters. (Memory)

EOMCHARS [char1]
[char1, char2]

char1 and **char2** are un-delimited ASCII characters. If you specify just one character, the second character is set to null. If you do not specify either character, both are set to null and no end of message checking is done at all. The default end of message characters are $^{C}_{R}$ (carriage return) and $^{N}_{L}$ (null).

If the terminal is in Prompt mode, Block mode or Half Duplex mode, it stops transmitting when it encounters either "end of message" character. (See the PROMPTMODE, BLOCKMODE, and DUPLEX commands).

In Prompt mode, for example, when an end-of-message character is transmitted the terminal terminates transmission until it receives a prompt.

The following command would set the end of message characters to $^{L}_{F}$ (line feed) and $^{E}_{C}$ (escape).

EOMCHARS $^{L}_{F}$ $^{E}_{C}$ $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

ERRORLEVEL

Specifies how severe an error must be before it is displayed by the terminal.

ERRORLEVEL {**0**}
{**1**}
{**2**}
{**3**}
{**4**}

0 specifies that all messages, warnings, errors, and terminal failure messages should be displayed. If you enter just the command word and no parameter, the error level is set to 0.

1 specifies that warnings, errors and terminal failure messages should be displayed.

2 specifies that errors and terminal failure messages should be displayed. This is the default.

3 specifies that only terminal failure messages should be displayed.

4 suppresses all messages, warnings, errors, and terminal failure messages.

The ERRORLEVEL command does not affect error messages that are generated by host resident software programs or the host computer.

The terminal's error messages are stored in an error queue whether or not they are displayed. The application program can examine that queue for an error report.

The following command would set the terminal so that only terminal failure messages are displayed:

ERRORLEVEL 3 $^{C}_{R}$

Appendix A lists the terminal's warnings, error messages and terminal failure messages.

FIXUP

Specifies when and how thoroughly a viewport is updated when a change is made that affects it. (Memory)

FIXUP level

level is an integer parameter with a value of 0, 2, 4 or 6. Each value specifies when changes are made in a viewport due to changes made that affect it:

0 the screen is updated when you press the PAGE key.

2 the actions for fixup level 0 are taken and the display is also updated as changes are made to the current view.

4 the actions for levels 0 and 2 are taken and the current viewport is updated when a segment is moved.

6 the actions for levels 0, 2 and 4 are taken and segments are immediately erased from the viewport if they are deleted or made invisible. This is the factory default.

The higher the value of the "level" parameter, the more the current view is updated as it is changed.

If you enter just the command word with no parameter, the level is set to 0.

Regardless of the fixup level, the current view is always completely updated when you press the PAGE key; the specified view is updated whenever you issue the RENEW setup command.

FLAGGING

Specifies whether or not the terminal is to use flagging, and if so, what kind of flagging.
(Memory)

FLAGGING {**none**}
{**input**}
{**output**}
{**inout**}
{**dtr**}

none is a keyword indicating that flagging is not to be done. This is the factory default.

input is a keyword indicating that when the terminal's input buffer is full, it sends a DC3 character character to stop the host; the terminal sends a DC1 character when it is ready to receive data again

output is a keyword indicating that the DC1 and DC3 characters are used for flagging.

inout is a keyword indicating that the terminal and host both use the DC1 and DC3 characters to flag each other as to when to start and halt communications.

dtr is a keyword indicating that when the terminal's input queue is nearly full it lowers the DTR (DATA TERMINAL READY) line on the RS-232 connector; when the terminal's input queue is ready for more data, the terminal raises the line.

For output, when the host lowers the CTS (CLEAR TO SEND) line, the terminal stops sending data; the terminal resumes transmission when CTS goes high again.

Flagging allows the terminal and host computer to indicate to each other when they are ready to send or receive data. This prevents their input buffers from overflowing, which would cause data to be lost.

The following command would cause the terminal to use input flagging:

FLAGGING input $^{C}_{R}$

Whether or not the terminal should use flagging, and what kind of flagging mechanism it does use depends on your host computer.

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

FORMAT (Option 42)

Formats a flexible disk so it can be used to store data.

FORMAT diskname [files]

diskname is an un-delimited string of up to nine alphanumeric characters identifying the disk; a space cannot be included in the "diskname." If no "diskname" is given but a device is specified, no "diskname" is assigned.

files is an integer specifying the maximum number of files that can be created on the disk. The default is 368; the valid range is 368 to 1872.

A disk must be formatted before it can be used. If you format a disk which already has files on it, the files are inaccessible.

If you specify a number less than 368, the disk is formatted for 368 directory entries. If you specify a number greater than 1872 the disk is formatted for 1872. If the number you specify is not divisible by 16, it is rounded up to the nearest multiple of 16.

A disk cannot be formatted if it is write protected by the WRITE PROTECT switch (see Section 2) or when the write protect notch in the disk is not covered (see Section 3).

The following command formats the disk for 384 files and names the disk SALESDATA.

FORMAT SALESDATA 384 $^{C}_{R}$

CAUTION

Do not open the door to the disk drive while the disk is being formatted. If you do, you must close the door, press the RESET button to reset the terminal, and then reformat the disk.

GAMODE

Specifies background for alpha text in the graphics area.
(Memory)

GAMODE {**replace**}
{**overstrike**}

replace specifies that only the character itself is displayed; the rest of the character cell is not displayed. This is the factory default.

overstrike specifies that the character cell not used by the character is displayed in the same gray level of the current background gray-level (see DAINDEX).

This command is similar to DAMODE except that it affects the display of alpha text in the graphics area instead of in the dialog area.

If you enter just the command word and no parameter, the parameter is set to overstrike.

GRIDDING

Establishes a grid on the terminal's screen for use in graphics input (GIN) mode.

GRIDDING device-function, [xgrid, ygrid]

device-function indicates the graphics input device used and what function is being performed by the input device. Valid specifications are:

0	thumbwheels for locator function
1	thumbwheels for pick function
8	tablet for locator function
9	tablet for pick function
24	port 0 for locator function
25	port 0 for pick function
32	port 1 for locator function
33	port 1 for pick function
40	port 2 for locator function
41	port 2 for pick function

xgrid specifies the distance between the grid lines parallel to the x-axis. The default when the terminal is turned on is zero; if you do not enter a value for "xgrid" it is set to zero. The valid range is from 0 to 4095.

ygrid specifies the distance between the grid lines parallel to the y-axis. The default when the terminal is turned on is zero; if you do not enter a value for "ygrid" it is set to zero. The valid range is from 0 to 4095.

Gridding is usually set by a host program when it puts the terminal into graphics input (GIN) mode. It can help you draw lines or locate a point more precisely.

The following command sets up a 100 x 100 grid for the terminal's thumbwheels for the locator function:

GRIDDING 0 100 100 $^{C}_{R}$

IGNOREDEL

Specifies whether or not the terminal ignores the delete (RUBOUT) character.
(Memory)

IGNOREDEL {**yes**}
{**no**}

yes is a keyword indicating that the delete character should be ignored.

no is a keyword indicating that the delete character should not be ignored; this is the factory default.

You should only set this parameter to "yes" if your program uses ^{E}C < ?> as a substitute for the ^{D}L in graphics.

The following command changes the setting of IGNOREDEL to "yes":

IGNOREDEL ^{C}R

KEYEXCHAR

Specifies the "key execute delimiter" character.
(Memory)

KEYEXCHAR char

char specifies the "key execute delimiter." Any ASCII character can be used, but it is best to choose one that is not used in a typical text string. The factory default is ^{D}E (CTRL-P).

The "key execute delimiter" determines whether a macro is expanded by the terminal or the host. When a macro is being expanded by the host and the terminal encounters the key execute delimiter, the direction is reversed so the terminal is executing the macro. When it encounters another key execute delimiter, it causes the host to begin to expand the macro again.

If the terminal is in Local or Setup mode, all macros are expanded locally and the key execute delimiter is ignored.

The following command would define ^{C}N as the key execute delimiter:

KEYEXCHAR 24 ^{C}R

For an example of the use of the key execute delimiter, see the DEFINE command earlier in this section.

LFCR

Specifies whether the line feed character is interpreted as a line feed or a line feed-carriage return. (Memory)

LFCR {**yes**}
{**no**}

yes is a keyword indicating that the terminal should attach a carriage return character to each $^{L}_{F}$ (line feed).

no is a keyword indicating that the terminal should not attach a carriage return character to each $^{L}_{F}$ (line feed).

A carriage return causes the cursor to move to the left margin of the current line; a line feed moves the cursor down one line from its current position. A carriage return-line feed combination moves the cursor to the left margin of the next line.

The following command would cause the terminal to attach the carriage return character to the line feed:

LFCR $^{C}_{R}$

When the terminal is in Local mode, it is necessary to change the parameter to "yes" if you want to get the carriage return-line feed combination by pressing just the LINE FEED key.

LOAD

Reads a file and executes it as a command file. (Option 42)

LOAD [filename]
[source-filename]

filename specifies the name of the file to be read. If no source is specified, the terminal defaults to the disk drive.

source-filename specifies the source of the file and the name of the specific file to be read. Valid sources are:

HO: host computer port
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

Pressing the CANCEL key aborts the loading of a file. All other input to the terminal is queued until the file is loaded or aborted.

Nesting of LOAD commands is allowed. That is, a file being loaded can in turn contain a LOAD command. That file can also contain a LOAD command, and so on. The limitation of nesting varies, depending on how busy the terminal is with other tasks.

This command loads the file named INITIALIZ from the disk:

LOAD INITIALIZ $^{C}_{R}$

LOCKKEYBOARD

Locks the terminal's keyboard.

LOCKKEYBOARD

No parameters need to be entered. As soon as you enter the carriage return, the keyboard is locked.

When the keyboard is locked you can still press the keys, but the characters are not echoed on the screen. The terminal's bell rings when you press any key except the BREAK or CANCEL.

The keyboard should usually be locked and unlocked by the application program.

CAUTION

Pressing the CANCEL key, or the RESET button may cause you to lose information on the screen or halt other terminal operations in process. Do not do so unless you are sure you will not lose important data.

You can unlock the keyboard by pressing the CANCEL key, the BREAK key, or the RESET button.

PAGEFULL

Specifies action to be taken when the "page full" condition occurs.
(Memory)

PAGEFULL {**none**}
{**stop**}
{**autocopy**}
{**break**}

none is a keyword meaning that no action is taken; when the cursor reaches the last line of the display, it returns to the upper left corner and continues, writing over any text already there. This is the factory default. If you enter the command word with no parameter, it is set to "none."

stop is a keyword meaning that when a page full condition occurs, the terminal stops displaying text. Incoming text is stored in the input queue and displayed when the page full condition is cleared.

autocopy is a keyword meaning that when a page full condition occurs, the terminal makes a hard copy of the screen, erases the screen, and continues. While the hard copy is being made, incoming text is stored in the terminal's input queue.

break is a keyword meaning that when a page full condition occurs, the terminal sends an interrupt to the host to discontinue the sending of information (the equivalent to pressing the BREAK key).

If flagging is not being used (see the FLAGGING command in this section, the communications overview in Section 5, and the 4112 Host Programmer's Manual) the input queue may be overrun when you use "autocopy" due to the time it takes to make the hard copy. If this is the case, you may want to use a flagging mechanism or the QUEUESIZE command to be sure that the input queue is large enough so that it is not overrun.

The following command sets the terminal so that when a page full condition occurs, it makes a hard copy, erases the screen, and then continues.

PAGEFULL autocopy $^{C}_{R}$

PARITY

Specifies the kind of parity used by the terminal for checking input and when transmitting to a host. (Memory)

PARITY {**none**}
{**high**}
{**even**}
{**odd**}
{**data**}

none means that the parity bit is set to 0 for transmission of data.

high means that the parity bit is set to 1 for transmission of data.

even means that the sum of the bits in the byte must be even for output.

odd means that the sum of the bits in the byte must be odd for output.

data means that all eight bits are used to send data for output.

Parity is a means for verifying the accuracy of communication between the terminal and a host computer. The terminal does not check parity on input, but it can be set to transmit data according to the host computer's parity convention.

The following command sets the terminal to transmit odd parity:

PARITY odd $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

PASSIGN (Option 10)

Assigns a device driver and/or an identification code to the specified port.
(Memory)

PASSIGN port, device

port identifies the port to which a device driver and/or an identification code is assigned. Valid port specifications are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

device is an integer that identifies the device driver is assigned to the specified port. Valid device drivers are:

Device	Meaning
4643	Tektronix 4643 line printer.
4662	Translates terminal commands into 4662 commands and 4662 commands into terminal commands; the plotter must be used in Block mode.
4662/NT	Does not translate commands between the terminal and 4662.
4662/MP	Same as "4662" driver, but includes pen changing support added by the plotter's multiple-pen option.
4663	Translates terminal commands into 4663 commands and 4663 commands into terminal commands; the plotter must be used in Block mode.
4663/NB	Translates terminal commands into 4663 commands and 4663 commands into terminal commands; the plotter must be used on Continuous Communications mode (use DC1/DC3 or CTS/DTR flagging).
4663/NT	Does not translate commands between the terminal and the 4663; plotter must be used on Continuous Communications mode.
PPORT	General purpose; supports any Tektronix device with an RS-232 connector.

If you attach a device to a port, the device must be compatible with the assigned driver.

The following command assigns the 4663 device driver to peripheral port 0:

PASSIGN P0: 4663 ^{C_R}

This command assigns the general purpose device driver to peripheral port 2:

PASSIGN P2: PPORT ^{C_R}

PBAUD (Option 10)

Assigns a rate for transmitting and receiving data for the specified port.
(Memory)

PBAUD port, baudrate

port indicates which port is receiving a baud rate assignment; valid port assignments are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

baudrate is an integer which specifies the rate at which data will pass through the assigned port. The factory default is 2400.

The baud rate that you assign to each port must be compatible with the baud rate of the device you attach to the port. If it is not, there will be communications problems.

Valid baud rates are:

50
75
110
134(sets baud rate to 134.5)
150
300
600
1200
1800
2000
2400
3600
4800
7200
9600

The following command sets the baud rate at 1200 for peripheral port 2:

PBAUD P2: 1200 $^{C}_{R}$

PBITS (Option 10)

Specifies the stop bit and data bit convention for the indicated peripheral port.
(Memory)

PBITS port, stopbits, databits

port indicates the port for which the stop-bits and data bits conventions are specified. Valid ports are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

stopbits is an integer specifying the minimum number of bits following the data bytes in the communications with the device attached to the port. The integers 1 and 2 are valid; the factory default is 2.

databits indicates the length in bits of the data bytes used to communicate with the device attached to the port. The integers 5, 6, 7, and 8 (representing 5-bit, 6-bit, 7-bit, and 8-bit bytes) are valid. The factory default is 8.

The standard ASCII character has 7 data bits and one parity bit (see PPARITY later in this section.)

Both "stopbits" and "databits" for a port must be compatible with the device that is attached to that port. If they are not, there will be communications problems.

The following command sets the data bit size to 7 and stop bits to 1 for peripheral port 0:

PBITS P0: 1 7 $^{C}_{R}$

PCOPY (Option 10)

Establishes a two-way data path between two devices.

PCOPY source **to** destination

source indicates the device which initiates and terminates the communication. Valid devices are:

HO: the host port
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

to is a keyword that helps you remember the flow of the copy — from the source **to** the destination.

destination specifies the device which is originally on receiving end of the communication path. Valid devices are:

HO: the host port
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

Since the data path is two-way, the two devices can carry on communication without involving the terminal. It is as if the terminal were not present and the two devices were connected directly to each other.

The connection between the devices is established by the PCOPY command. Communications must be started by the "source" device. The connection is broken when the terminal detects an end-of-file string (see PEOF) from the source device, when you press the CANCEL key, or if there is an error.

The Setup mode prompt reappears on the screen when the PCOPY is finished. The first 130 characters you type during a PCOPY are stored in a queue. They are echoed and executed when the PCOPY is finished.

The following command allows the host and the device attached to peripheral port 2 to transfer data without involving the terminal:

PCOPY HO: to P2: $^{C}_{R}$

If the source device is the host (HO:), then you must arrange to begin the data stream before you enter the command in Setup mode, because once the command is entered the terminal will wait for a data stream from the host until an end-of-file comes from the host or the CANCEL key is pressed.

Two ways to start the data stream appropriately are:

1. Type the command to the host to start the data stream without the end-of-command character from outside of Setup mode, enter Setup mode, type in the PCOPY command with the desired destination, exit Setup mode, and type the end-of-command character. This should start the data stream from the host. If your host echos the characters typed on the terminal this method results in the end-of-command character being the first character in the data stream.
2. Program the host so that there is a delay between the time you enter the end-of-command character for the host-data-send command and the time the data stream begins to flow. If you use this method you should give yourself enough time to enter Setup mode and type the PCOPY command with the desired destination.

PEOF (Option 10)

Specifies the string to be sent to the peripheral port when an end-of-file is detected by the terminal.
(Memory) ı)

PEOF port, ^{D_L}string^{D_L}

port indicates the port for which the specified string denotes end-of-file. Valid ports are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

^{D_L} is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.

string is a delimited string of up to ten ASCII characters. The factory default is the null string.

The port-EOF string that you assign to a port should be the same as the end of file string for the device attached to that port.

In the following command the slash (/) is used as the delimiter and the "peripheral end of file" string is changed to "????."

PEOF P1: /????/ ^{C_R}

PEOL (Option 10)

Assigns the end-of-line string to be used by the specified port.
(Memory)

PEOL port, ^{D_L}string^{D_L}

port indicates the port for which the specified string denotes end-of-line. Valid ports are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

^{D_L} is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.

string is a delimited string of one or two ASCII characters. The factory default is a carriage return.

The port-EOL string that you assign to a port should be the same as the end of file string for the device attached to that port.

In the following command, the end of line string is set to be a ^{L_F} (line feed) for peripheral port 0.

PEOL P0: $'^{L_F}'$ ^{C_R}

PFLAG (Option 10)

Sets the Flagging mode for the specified peripheral port.
(Memory)

PFLAG port, flag, [go], [stop]

port indicates the port for which Flagging mode is being defined. Valid ports are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

flag indicates which of the three types of flagging is to be assigned.

NONE = no flagging
CHAR = character flagging (factory default)
DTR = DTR/CTS (hardware) flagging

go is a single character parameter which can be specified when "character flagging" is used. The terminal uses the "go" character to indicate that data transmission can proceed. Any ASCII character is valid; DC1 (ADE 17) is the factory default. If no character is specified, "go" defaults to DC1.

stop is a character parameter which can be specified when "character flagging" is used. The terminal uses the "stop" character to indicate that data transmission should stop. Any ASCII character is valid; DC3 (ADE 19) is the factory default. If no character is specified, "stop" defaults to DC3.

The following example specifies that flagging is not used at peripheral port 0:

PFLAG P0: NONE $^{C}_{R}$

This command causes peripheral port 1 to use character flagging with "?" as the "go" character and the default DC3 as the "stop" character:

PFLAG P1: CHAR ? $^{C}_{R}$

PICKAPERTURE

Specifies the "a" square within which the graphics input device can "pick" an object.
(Memory)

PICKAPERTURE [size]

size specifies the side of the square within which the graphics input device is centered. The default when the terminal is turned on or reset is 8. If you do not specify a size, the terminal defaults to 0.

The following command specifies a pick aperture of 4. This creates a 4 x 4 square with the graphics input locator centered in the square. If you attempt to "pick" an object, it must have its X and Y coordinates within two units (plus or minus) of the locator device in order to be detected.

PICKAPERTURE 4 $^{C}_{R}$

See the 4112 Host Programmer's Manual and the 4110 Series Command Reference for details on "picking" objects on the screen.

PLOT (Option 10)

Copies all currently visible segments to the specified output device.

PLOT to [destination]
[destination-filename]

to is a keyword to help you remember the flow of the PLOT

destination indicates the device to which the segments will be copied and the specific file to be copied. If no destination is specified, the terminal defaults the disk drive. Valid ports are:

HO: the host port
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

destination-filename specifies the destination of the plot and the specific file to which the plot is to be sent.

If the "destination" parameter is a port that has been assigned a 4662 or 4663 driver (see the PASSIGN command), terminal commands are converted into plotter commands. Otherwise, plots are sent as to the specified device as terminal commands.

If you specify the host as destination, then be sure that the host is ready to accept the data, Otherwise you may lose your data or cause problems with the host.

The following command copies the currently visible segments to the file named "picture1" on the disk:

PLOT to picture1 ^{C_R}

This command copies the segments to the device on peripheral port 2:

PLOT to P2: ^{C_R}

PMAP (Option 10)

Assigns one or more gray indices to a plotter pen number.

PMAP port, index, pen

port indicates the port for which the index-to-pen assignment is valid.

index specifies the gray index assigned to a pen. A pen can have more than one index assigned to it. Valid index numbers are integers from −2 through 255.

−2 means to "unmap" (disassociate) all indices.

−1 means to map all gray indices to the given pen.

0−255 means to map the given gray index to the given pen.

pen specifies the pen to which a gray index is mapped. Valid pen numbers are integers from −1 to the number of pens on the plotter:

−1 means that all plotter pens are to be "unmapped" (disassociated) from all gray indices; this can only be used if "index" is set to −2.

0 "unmap" the pen.

1−N specifies the pen number.

Each time the terminal is turned on, all gray-scale indices are initialized to pen 0.

When data is drawn on the plotter, all lines with a mapped gray index are drawn by the specified pen. Any index without a pen assignment is not be drawn by the plotter.

Mapping a gray-scale index to one pen automatically "unmaps" that index from any other pen. This means that while one pen can have several indices assigned to it, an index can be assigned to only one pen at a time.

The following command maps gray index 6 to plotter pen 1, attached at peripheral port 2:

PMAP P2: 6 1 ^{C_R}

This command "unmaps" all indices previously assigned to pen 1:

PMAP P1: −2 1 ^{C_R}

PPARITY (Option 10)

Assigns a Parity mode to the specified port. (Memory)

PPARITY port, {**low**}
{**odd**}
{**even**}
{**high**}
{**none**}

port indicates the port for which parity is being assigned. Valid ports are:

P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2

low is a keyword indicating that the parity bit is given a value of 0.

odd is a keyword indicating that the value of the parity bit is assigned such that the sum of all bits in the byte is odd.

even is a keyword indicating that the value of the parity bit is assigned such that the sum of all bits in the byte is even.

high is a keyword indicating that the parity bit is given a value of 1.

none is a keyword indicating that there is no parity check.

A standard ASCII character with parity would have seven data bits (see PBITS) and one parity bit.

The following command sets the terminal for even parity checking on peripheral port 2:

PPARITY P2:even ^{C_R}

PROMPTMODE

Specifies whether or not the terminal is in Prompt mode.
(Memory)

PROMPTMODE {**yes**}
{**no**}

yes is a keyword which causes the terminal to enter Prompt mode.

no is a keyword which causes the terminal to exit Prompt mode. This is the factory default.

When the terminal is in Prompt mode, data from the terminal is stored in a queue and transmitted to the host when it is prompted to do so. If you press the CANCEL key, the terminal flushes the queue. If you turn Prompt mode off, all the characters in the queue are sent to the host.

The characters used to prompt the terminal are defined by the PROMPTSTRING command. After the terminal enters Prompt mode, the first prompt string is displayed. Subsequent prompt strings are executed, but not displayed. If you press the CANCEL key, Prompt mode starts over — that is, the first prompt after pressing CANCEL is displayed and subsequent prompts are executed.

The following command would cause the terminal to enter Prompt mode:

PROMPTMODE yes $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

PROMPTSTRING

Specifies the prompt sequence.
(Memory)

PROMPTSTRING $^{D}_{L}$string$^{D}_{L}$

$^{D}_{L}$ is the delimiter character, which must precede and follow each parameter. The first character after the command word that is not a comma, space, or edit character (see EDITCHARS) is the delimiter character for this command.

string is a delimited string of up to ten ASCII characters. The factory default is the null string.

The prompt string you specify must conform to the prompt string actually sent by your host computer.

The following command is an example of defining a prompt string:

PROMPTSTRING /+&%/ $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

PROTECT (Option 42)

Specifies the write protect status for a disk or file.

PROTECT [device]
[device-filename] **[yes]**
[filename] **[no]**

device indicates the disk drive containing the disk that is to be protected or unprotected. The valud device name for the disk drive is:

F0:

filename specifies the individual file that is to be protected or unprotected. Valid filenames are alphanumeric strings of up to nine characters. If no device is specified, the terminal defaults to the disk drive.

yes is a keyword meaning that the file or disk should be protected. "Yes" is the default; if you do not specify "no" then the file or disk is protected.

no is a keyword meaning that the file or disk should be unprotected.

A protected file must be unprotected before it can be deleted or overwritten. A protected disk cannot be copied, however it can be reformatted. You cannot use the PROTECT command if the disk is already protected by the disk drive's WRITE PROTECT switch.

The following command protects the disk in drive 0:

PROTECT F0: $^{C}_{R}$

This command protects the file named "important" on disk drive 1:

PROTECT F0:important $^{C}_{R}$

QUEUESIZE

Specifies the size of the terminal's input queue. (Memory)

QUEUESIZE bytes

bytes is an integer, specifying the size of the input queue. The factory default is 300.

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 flagging (see FLAGGING) is not being used, the terminal ignores incoming data or text until there is more room in the queue. If a flagging mechanism is being used and the queue fills up, the terminal flags the host to stop sending data. When there is more room in the queue for more data, the terminal flags the host to resume sending data.

The following command sets the input queue to 500 bytes:

QUEUESIZE 500 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

RENAME (Option 42)

Changes the name of a file on the specified disk.

RENAME oldname **to** newname

oldname is an undelimited alphanumeric string of up to nine characters indicating the current name of the file.

to is a keyword which helps distinguish between current name of the file and its new name.

newname is an undelimited alphanumeric string of up to nine characters indicating the new name of the file.

An error results if a file already exists with the newly specified name. An error also occurs if no file with the specified "oldname" exists, or if the file you are attempting to rename is write protected (by the WRITE PROTECT switch, the write protect notch in the disk, or by the PROTECT command).

The following command changes the file named "wrong" to "right":

RENAME wrong to right ^{C_R}

RENEW

Erases the specified viewport and redraws all visible segments.

RENEW view

view is an integer parameter in the range from −1 to 64. A value of −1 renews all viewports; 1−64 renew the specified viewport.

When a viewport is renewed, it is completely updated to reflect all changes made to it since it was updated last. See the FIXUP command for more details.

REOM

Specifies when the "end of message" string is sent in a series of graphics input (GIN) reports.
(Memory)

REOM {**1**}
{**0**}

> **1** specifies that the end of line string is sent after each report element and at the end of the report. This is the factory default.
>
> **0** specifies that the end of line string is sent only at the end of the report.

This command should be set by the applications program. If it has not been, you may need to see a system programmer for your host system for more information.

RLINELENGTH

Specifies the maximum length of a terminal report sent to the host computer.

RLINELENGTH length

> **length** specifies the maximum length in characters for reports the terminal sends to the host computer. The factory default is 0. A specification of 0 sets the maximum length to infinity.

See the 4110 Series Command Reference Manual for details.

SAVE

Saves the specified item(s) to the specified file or device.

SAVE {**macro**} {**id**} **to** [destination-filename]
{**segment**} [filename]
{**ras**}
{**run**}
{**numpix**}

macro is a keyword indicating that a macro is to be saved.

segment is a keyword indicating that a segment is to be saved.

ras saves pixels using "raster write" commands.

run saves pixels using "run length write" commands.

id is an integer parameter identifying the macro or segment to be saved.

Valid macro identifiers for a macro are −1 and 1 to 32767. If you indicate "all" or −1, all currently defined macros are saved. The values 1 to 127 save the macro associated with the corresponding ASCII key, 128 to 143 saves macro with the the corresponding function keys, and 144 to 32767 save macros associated with integers. See the ASCII chart in Appendix D and the DEFINE command for more details.

Valid segment identifiers are −3, −1, and 1 to 32767. −3 saves all segments in the current matching class, −1 or "all" saves all existing segments, and 1 to 32767 save specific segments. See the Host Programmer's Reference Manual for details.

numpix is an integer parameter used when "ras" or "run" is the first parameter. It is within the range from −1 to 32767, specifying the number of pixels to be saved. −1 means to save all pixels in the viewport. 1−32767 means to save the specified number of pixels starting at the current beam position.

to is a keyword which indicates the flow of the data: it is saved **from** the terminal memory (macro or segment) **to** the specified destination.

(continued)

destination indicates to which device the data should be saved.

- H0: host port
- P0: peripheral port 0
- P1: peripheral port 1
- P2: peripheral port 2

filename is an undelimited string of up to nine alphanumeric characters identifying the file into which the macro or segment is to be saved. If the filename is specified and a destination is not specified, a file by that name is created on the disk.

You can save files to and from the host without the disk drive and three port peripheral options. You can save files to the disk(s) if you have the disk option (Option 42). You can use the peripheral ports if you have the three port peripheral option (Option 10).

Graphics and macros that are saved can be put back into terminal memory with the LOAD command.

If you specify the host as destination, then be sure that the host is ready to accept the data stream, or you may lose your data or cause communications problems with the host.

The following command would save all current segments to the device attached to peripheral port 2:

SAVE segment all to P2: $^{C}_{R}$

This command saves macro number 579 in the file named "macfile":

SAVE mac 579 to macfile $^{C}_{R}$

For more detailed information on pixel saving operations, see the 4112 Host Programmer's Manual and the 4110 Series Command Reference Manual.

SNOOPY

Specifies whether the terminal should display control characters instead of executing any of them.

SNOOPY {**yes**}
{**no**}

yes is a keyword indicating that Snoopy mode should be initiated.

no is a keyword indicating that Snoopy mode should be terminated. This is the default. The "no" specification puts the terminal in Alpha mode.

When the terminal is in Snoopy mode (but not in Setup mode) it displays control characters instead of executing them. When the terminal is in Setup mode and Snoopy mode at the same time, it displays the control character and then executes the control character.

Since Snoopy mode displays characters which would otherwise be executed, it is a handy tool for debugging code.

When in Snoopy mode, characters which are normally filtered out during communications to the host are not displayed. That means, for example, that even though the terminal is in Snoopy mode, the prompt string, block headers, and flagging characters are not displayed.

The terminal displays control characters using the two-character abbreviation of control characters shown in the ASCII code chart (Appendix D).

Snoopy mode can also be used to examine the contents of macros without executing any commands in them. To do so, first enter Setup mode and give the following command:

SNOOPY yes $^{C}_{R}$

After entering this command, exit from Setup mode and enter Local mode. Now when you press a key which has a macro assigned to it, that macro is displayed but not expanded.

To de-activate snoopy, enter Setup mode and give the following command:

SNOOPY no $^{C}_{R}$

SPOOL

Copies data from one device to another, allowing the terminal to be used for other functions at the same time.

SPOOL [source] **to** [destination]
[filename] [filename]
[source-filename]

source indicates the source device. If no source is specified, the terminal defaults to disk drive 0 (F0:). Valid devices are:

HO: host port
F0: disk drive 0 (requires a filename)

filename is an undelimited string of up to nine alphanumeric characters identifying the file to be copied. This filename must be specified if the "source" or "destination" is a disk. If a filename is specified as the destination, a file by that name is created on the disk.

to is a keyword to indicate the flow of the status report -- from the "source" **to** the "destination."

destination indicates the destination device for the data being spooled. If no device is specified, the terminal defaults the disk drive. Valid devices are:

H0: host port
P0: peripheral port 0
P1: peripheral port 1
P2: peripheral port 2
F0: disk drive 0 (requires a filename)
F1: disk drive 1 (requires a filename)

Once initiated, the SPOOL operation proceeds in the background, allowing the terminal to be used for other operations. Data is transferred from the source to the destination without destroying the source copy.

Spooling continues until:

- the terminal receives the end-of-file character
- the STOP command is issued
- there is a source or destination error

If the "destination" is a disk drive, an output error can result because:

- There is no disk in the drive.
- The disk in the drive is write protected.
- The disk does not have enough room on it for the output file.

If the last of these errors occurs, you should use a different disk.

If the source is the host, you should either arrange for the data to be transmitted for the SPOOL before you enter Setup mode, or be aware that after you issue the command everything that comes from the host is considered data, including the remote character echo.

All characters echoed from the host either before the data stream begins or while the data stream is flowing are sent to the destination device.

This command copies "file1" on disk drive 0 to the output device attached to peripheral port 2:

SPOOL F0:file1 to P2: $^{C}_{R}$

STATUS

Reports the terminal's general status, the status of a group of related parameters or the setting of an individual terminal parameter.

STATUS [cluster]
[command]
[command-abbreviation]

If no parameter is entered, the terminal displays a general report of the terminal's status.

cluster is an undelimited alphabetic string specifying a status report of a group of related parameters. Valid clusters are:

report/input
general
communications
optional (optional communications)
dialog (dialog area)
3PPI

command is an undelimited alphabetic string specifying a command for which the terminal should report the status. Valid command names include all commands whose parameters are stored in the auxiliary backup memory.

command-abbreviation is an undelimited alphabetic string specifying one command or a group of commands all of which start with the given abbreviation.

The following command generates a general report of the status of the terminal (Figure 6-6):

STATUS $^{C}_{R}$

```
*STATUS

    TERMINAL STATUS

        REPORT/INPUT
TBSTATUS..... IN
TBHEADERCHARS..... LETTERS
REOM..... 1
PICKAPERTURE..... 8
RLINELENGTH..... 0


        GENERAL
EDITCHARS..... ▮ ␆ ~
PAGEFULL..... STOP
ECHO..... NO
LOCKKEYBOARD..... NO
KEYEXCHAR..... ␄
DAENABLE..... NO
IGNOREDEL..... NO
LFCR..... NO
CRLF..... NO
SNOOPY..... NO
ERRORLEVEL..... 2


        DIALOG
DABUFFER..... 34
DACHARS..... 80
DAINDEX..... 7 0 0
DALINES..... 5
DAMODE..... REPLACE
DAPOSITION..... 0,0
- - - - - - - - - - - - - - - - - - - - - - - - -
DASURFACE..... 1
DAVIS..... NO


        COMMUNICATIONS
BAUDRATE..... 2400 2400
XMTLIMIT..... 19200
STOPBITS..... 1
XMTDELAY..... 100
PARITY..... NONE
PROMPTSTRING..... ''
PROMPTMODE..... NO
QUEUESIZE..... 300
FLAGGING..... NONE
EOMCHARS..... ␆ ␄
EOLSTRING..... '␆'
EOFSTRING..... ''
BREAKTIME..... 200
BYPASSCANCEL..... ␄


        OPTIONAL COMM
DUPLEX..... FULL
BLOCKMODE..... NO
BHEADERS..... 'HEADTX' 'HEADRX'
BLENGTH..... 256 256
BCONTINUECHARS..... & &
BENDCHARS..... $ $
BMASTERCHARS..... ▮ ▮
BLINELENGTH..... 70
BPACKING..... 7 6 7 6
BTIMEOUT..... 0
BNONXMTCHARS..... '▮$&' '▮$&'

                                    3683 29
```

Figure 6-6. The STATUS Report.

This command generates a report of the status of commands relating to the dialog area (Figure 6-7):

STATUS DIALOG ^{C_R}

This command causes the terminal to report the status of the BAUDRATE command (Figure 6-8):

STATUS BAUDRATE ^{C_R}

When you abbreviate the name of a command, the terminal reports the status of all commands with the same spelling up to the letters you specified. For example, if you enter the following command the terminal reports the status of all commands beginning with the character "E" (Figure 6-9):

STATUS E ^{C_R}

If you request a report for a command that is not stored in the terminal's setup memory, there is no response. The cursor moves to the next line and displays the Setup mode prompt (*).

```
*STATUS DIALOG
       DIALOG
DABUFFER..... 34
DACHARS...... 80
DAINDEX...... 7 0 0
DALINES...... 5
DAMODE....... REPLACE
DAPOSITION..... 0,0
DASURFACE..... 1
DAVIS........ NO


*
```

3683-30

Figure 6-7. The Dialog Area STATUS Report.

```
*STATUS BAUDRATE
BAUDRATE..... 2400 2400
*
```

3683-31

Figure 6-8. The STATUS BAUDRATE Report.

```
*STATUS E
EDITCHARS..... ▮ ␑ ~
ECHO.......... NO
ERRORLEVEL..... 2
EOMCHARS...... ␑ ␄
EOLSTRING..... '␑'
EOFSTRING..... ''
*
```

3683-32

Figure 6-9. The STATUS E Report.

STOP

Halts the current "spooling" operation.

STOP

When a spool is halted, the output file is closed and all data that has already been transferred is saved.

The command is ignored if no spool is in progress.

STOPBITS

Specifies the number of stop bits used at the end of each character transmited from the terminal.

STOPBITS {**1**}
{**2**}

1 indicates that the terminal appends one stop bit. This is the factory default.
2 indicates that terminal appends two stop bits.

The purpose of the "stop bit" is to indicate the completion of the transmission of a data byte.

The following command sets the terminal to transmit two stop bits at the end of each byte:

STOPBITS 2 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

TBFILTER (Options 13/14)

Specifies the time or distance between points entered per stroke when the graphics tablet is in use.

TBFILTER device, distance, time

device is an integer parameter indicating the graphic input device to which this filter specification applies. Valid graphics input (GIN) devices are:

8 tablet for "locator" function
9 tablet for "pick" function
10 tablet for "stroke" function

distance is an integer parameter specifying the minimum distance the graphics cursor or pen must travel before another point is transmitted to the program. The valid range is from 0 to 4096. The default is 0.

time is an integer parameter specifying the minimum number of milliseconds that elapse between successive points transmitted to the program. The valid range is from 0 to 65535. The default is 0.

The following command sets the graphics tablet to locate every 100th point every 100 milliseconds:

TBFILTER 10 100 100 ^{C_R}

TBHEADERCHARS (Options 13/14)

Specifies the header and key characters used in graphics input reports.

TBHEADERCHARS {**letters**}
{**control**}

letters specifies the characters M, J, and O as the header characters. This is the default.

control specifies the control characters (GS), (SUB), and (US) to be the header characters.

This emulates the HEADER strap on the TEKTRONIX 4953 Graphics Tablet control board found in 4010-family terminals. For more details see the description of the <**set-tablet-status-strap**> command in the 4110 Series Command Reference Manual.

TBSIZE (Options 13/14)

The graphic tablets included with Options 13 and 14 have a slightly larger active area than the TEKTRONIX 4953 and 4954 tablets used with earlier 4010-series Tektronix terminals. To emulate the earlier tablets more closely, the <**set-tablet-size**> command is included in versions 2 and above of the firmware for Options 13 and 14.

TBSIZE {**large**}
{**small**}
{**automatic**}

large has the tablet operating as an option to a 4110-series terminal. The active area of the Option 13 tablet is 121 square inches (11 in x 11 in, or 279 mm x 279 mm). The active area of the Option 14 tablet is 1200 square inches (30 in x 40 in, or 762 mm x 1016 mm).

small is used when a smaller active area is required, to emulate more closely the 4953 and 4954 tablets used with earlier Tektronix 4010-series terminals. The Option 13 tablet emulates the 4953 tablet, with an active area of 105 square inches (10.24 in x 10.24 in, or 260 mm x260 mm). The Option 14 tablet emulates the 4954 tablet, with an active area of 1180 square inches (30.72 in x 38.4 in, or 780 mm x 975 mm).

automatic is used when the tablet is enabled with an <**enable-GIN**> command from the host, the entire tablet area is used. When the tablet is enabled with the <**enable-4953-tablet-GIN**> command from the host, the smaller active area of the tablet is used, emulating the smaller active area of the 4953 and 4954 tablets.

The following command sets the terminal to use the smaller area of the 4953/4954 tablets.

TBSIZE SMALL $^{C}_{R}$

See the 4110 Host Programmer's manual and the 4110 Series Command Reference manual for detailed information concerning TBSIZE.

TBSTATUS (Options 13/14)

Specifies whether or not the terminal status byte is sent when the graphics tablet pen is lifted away from the tablet surface.

TBSTATUS {**out**}
{**in**}

out specifies that the status byte is not sent when the tablet pen is lifted away from the tablet surface. This is the default.

in specifies that the status byte is sent when the tablet pen is lifted away from the tablet surface.

This emulates the STATUS strap in the TEKTRONIX 4953 Graphics Tablet board in 4010-style terminals. For more information, see the description of the <**set-tablet-status-strap**> command in the 4110 Series Command Reference Manual.

XMTDELAY

Specifies the length of time for which transmission of data to the host is halted after the terminal encounters an end of message character (see EOMCHAR).

XMTDELAY ms

> **ms** is an integer parameter representing the number of milliseconds delay in transmission. The acceptable range of values is from 0 to 65535; the factory default is 100.

The following command sets the transmit delay to 300 milliseconds:

XMTDELAY 300 $^{C}_{R}$

This parameter may be set by an applications program on the host computer or by an initialization file on a flexible disk. If it is not, you may need to consult with a system programmer for your host computer before using this command.

XMTLIMIT

Specifies the upper limit for data transmission from the terminal to a host computer.
(Memory)

XMTLIMIT limit

> **limit** is an integer parameter indicating the fastest rate at which the terminal is to send data to a host. The range of valid values is from 110 to 65535; the factory default is 19200.

If the host computer cannot effectively receive data at the rate at which the communications port is specified, you can use XMTLIMIT to cause the terminal to pace its transmission of data at the given limit. This lowers the effective baud rate without altering the baud rate itself.

The following command sets the transmit limit to 1800 bits per second.

XMTLIMIT 1800 $^{C}_{R}$

This terminal operating parameter is host dependent. It should usually be set by an application program when it is executed from the host. If it is not, see a system programmer for your host computer for more information. See the 4110 Series Command Reference Manual for additional details.

Appendix A

ERROR CODES

INTRODUCTION

Each error condition which the terminal can detect has an "error code" and a "severity level."

When the terminal detects an error condition, it stores the error code in a limited-size queue for later retrieval by a <report-errors> command.

If the error's severity level is greater than or equal to the current error level, then the terminal displays a message for the operator. (The error level is determined by the most recent ERRORLEVEL command.)

SEVERITY LEVELS

There are four severity levels, numbered from zero to three:

- **Level 0.** Errors of severity level zero are hardly errors at all. The associated message begins with the words "Terminal issues message . . . ." Typically, these errors occur for commands which are not installed. For instance, when 4112 commands are sent to a 4114, the terminal detects level zero errors.
- **Level 1.** Level one errors are "warnings." The corresponding messages begin with the words "Terminal issues warning . . . ." Typically these occur when the command is inappropriate: deleting a segment that does not exist, for example.
- **Level 2.** Level two errors result from invalid commands. For instance, a command's parameter may be outside the specified range. The corresponding message begins with the words, "Terminal detects error . . . ."
- **Level 3.** Level three errors occur when the command is valid, but for some reason the terminal cannot execute the command. (For instance, there may be insufficient memory to hold all the information being included in a segment definition.) For these errors, the message starts with the words, "Terminal system error . . . ."

COMMANDS NOT INSTALLED IN THE TERMINAL

When the terminal receives an escape-sequence command which it does not recognize, it detects the appropriate error message (of severity level zero). After detecting the error, the terminal then ignores all subsequent characters until it receives an ^{E}C, ^{G}S, ^{F}S, or ^{U}S character. (It does this so as to skip over any parameters for the unrecognized command.)

For instance, suppose the terminal does not have Option 1 installed, and the host sends it the following character sequence:

^{E}C(O)(D)(0)^{E}C(K)(A)(1)

Since Option 1 is not installed, the terminal does not recognize the <set-duplex-mode: 0> command, ^{E}C(O)(D)(0). On receiving the ^{E}C(O)(D) op code, the terminal detects a type OD00 error (and displays the OD00 error message if the error threshold is set to zero). It ignores the following character, (0). On receiving the following ^{E}C, the terminal resumes processing of the characters received, so that it correctly interprets and executes the <enable-dialog-area: 1> command, ^{E}C(K)(A)(1).

If your host program sends commands which may not be installed in all 4110-series terminals, then it should follow those commands with other commands which **are** recognized by all terminals in the series. For instance, after issuing commands to change 4112-only settings, the host could send a ^{U}S character (the <enter-alpha-mode> command) before sending any alphatext to the terminal. That way, if the terminal is a 4114 rather than a 4112, the ^{U}S character causes the terminal to resume normal processing of the characters it receives.

ERROR CODES

The error codes are each 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 for some command: the command being executed when the error is detected. For example, error IA11 is associated with the PICKAPERTURE command.

  Some errors, however, are associated with no particular command. For these errors, the first two characters are a letter and a digit. For instance, error I011 (invalid device-function code) can occur with many graphic input commands. Again, error J109 (disk hardware initialization error) can occur only when the terminal is turned on, before any commands have been sent to it.
- The third character in an error code is a numeric digit. Digits from 1 to 9 name the parameter with which the error is associated. Digit 0 indicates that the error is associated with the command as a whole: the op code itself is regarded as the "zeroeth parameter."
- The fourth character in an error code is also a digit. The most frequently used digits here are 0, 1, 2, and 3:

  0 : Indicates an "existence problem." The object referred to does not exist when it ought to exist, or does exist when it ought not to exist.

  1 : Indicates an "invalid value."

  2 : Indicates an "out of memory problem."

  3 : A "context error." The command is valid, but cannot be executed at this time. (For instance, trying to end a segment when no segment is currently being defined.)

For example, consider the "JD10" error code. Here, "JD" means the DIRECTORY command. The "1" refers to the first parameter of that command, which is the device. The "0" indicates an "existence problem;" there is no such device.

DISK HARDWARE ERRORS

For some disk system errors ("type 9" errors such as JC19, JC39, JD19, etc.), the error code reports a "disk hardware error number." Table A-1 lists the disk hardware errors:

Table A-1

DISK HARDWARE ERRORS

Error Number	Explanation
1	Cannot complete result phase.
2	Cannot sense drive status.
3	Cannot sense interrupt status.
4	RQM wrong state.
5	(Reserved.)
6	(Reserved.)
7	Invalid command.
8	Bad track.
9	Control mark.
10	CRC (cyclic reduncy check) error.
11	Missing address data.
12	Missing address mark.
13	No data.
14	Wrong cylinder.
15	Overrun.
16	(Reserved.)
17	Not two-sided.
18	Failed command phase (bytes not output).

DISK SYSTEM CONTEXT ERRORS

For some disk system errors ("type 3" errors such as JC13, JC33, JD33, etc.), a supplemental error message may be displayed. This supplemental message describes the type of "context error" which has occurred. Table A-2 lists these context error types:

Table A-2

DISK SYSTEM CONTEXT ERROR

Error	Explanation	Error	Explanation
File Already Open	Tried to open for input a file which is already open for output; or, tried to open for output a file which is already open for input.	Directory Full	This diskette already contains the maximum number of files for which it was formatted.
		File Not Open	Tried to close a file which is not currently open.
No FCB	No File Control Block is available when opening a file; or, the File Control Block has been destroyed.	File Trailer Error	Unsuccessful attempt to terminate the writing of a file. (The file being written becomes inaccessible.)
Secure Volume	A "secure volume" is a diskette (usually holding proprietary software) which cannot be copied. (However, files can be <load>ed from a secure volume, and a directory can be obtained with the <directory> command.	Directory Update Error	Error in attempting to update the disk directory on the diskette.
		Transfer Direction Error	Tried to read from a file which was opened for output; or, tried to write to a file which was opened for input.

ERROR CODES

The remainder of this appendix lists the specific error codes. They are listed alphabetically, according to the op code associated with each command.

If the command has an English-like Setup mode equivalent, it is indicated. If there is not, only the escape sequence format for the command is shown.

NOTE

^{E}C may appear as (ESC) in other manuals. However, in this manual, it is shown as ^{E}C.

I0 (For several GIN commands.)

I002 (Level 2): Insufficient memory available for GIN functions.

I011 (Level 2): Invalid device-function code. (See the description of the <enable-GIN> command for a table of device-function codes.)

I1 <Enable-4953-Tablet-GIN>
<Disable-4953-Tablet-GIN> = ^{E}C(I)(!)<ASCII-char>

I100 (Level 2): Unrecognized command (Option 13 or 14 not installed).

IA PICKAPERTURE=<Set-Pick-Aperture> = ^{E}C(I)(A)<int>

IA11 (Level 1): Invalid aperture width (must not be in the range from 0 to 4095).

IC **<Set-GIN-Cursor> =^{E}c(I)(C)<int><int>**

IC13 (Level 2): Graphic input has already been enabled for the specified device-function code.

IC20 (Level 2): Segment does not exist, or is currently being defined.

IC21 (Level 2): Invalid segment number (must be in the range 0 to 32767).

IE **<Enable-GIN> =^{E}c(I)(E)<int><int+>**

IE00 (Level 2): the cursor segment for the specified device-function code does not exist. (It has been deleted since the <set-GIN-cursor> command which assigned it to that device-function code.)

IE03 (Level 2): Command is invalid at this time. (The segment being used as the cursor for the specified device-function code is a segment which is currently being defined.)

IE10 (Level 2): The specified GIN device is not installed in the terminal.

IE13 (Level 2): The specified device is already enabled.

IE21 (Level 2): Invalid number of GIN events (must be in the range from 1 to 65535).

IF **TBFILTER=<set-GIN-Stroke-Filtering> =^{E}c(I)(F)<int><int><int>**

IF00 (Level 2): Unrecognized command. (The tablet option is not installed.)

IF10 (Level 2): Stroke filtering does not apply to the specified device-function code. (Stroke filtering applies only to the stroke function on the tablet device.)

IF21 (Level 2): Invalid filtering distance (must be in the range from 0 to 4095).

IF31 (Level 2): Invalid filtering time (must be in the range from 0 to 32767).

IG **GRIDDING=<Set-GIN-Gridding> =^{E}c(I)(G)<int><int><int>**

IG10 (Level 2): Gridding does not apply to the specified device-function code. (Gridding is not allowed for the stroke function.)

IG21 (Level 2): Invalid x-spacing (must be in the range from 0 to 4095).

IG31 (Level 2): Invalid y-spacing (must be in the range from 0 to 4095).

IH **TBHEADERCHARS=<Set-Tablet-Header-Characters> =^{E}c(I)(H)**<int><int>

IH00 (Level 0): Unrecognized command. (Tablet option is not installed.)

IH11 (Level 2): Invalid parameter (must be 0 or 1; in Setup mode, must be CONTROL or LETTERS).

II **<Set-GIN-Inking> =^{E}c(I)(I)<int><int>**

II11 (Level 2): Inking does not apply to the specified device-function code. (Inking is not allowed for the pick function.)

II21 (Level 2): Invalid Inking mode (must be 0 or 1).

IL **<Set-Report-Max-Line-Length> =^{E}c(I)(L)<int>**

IL11 (Level 2): Invalid maximum report line length (must be in the range from 0 to 65535).

IM **REOM=<Set-Report-EOM-Frequency> =^{E}c(I)(M)<int>**

IM11 (Level 2): Invalid report-EOM-frequency setting (must be 0 or 1).

IN **<Set-Tablet-Mode> = ^{E}c (IN)<int-array>**

IN00 (Level 0): The <set-tablet-mode> command is not installed. (This command requires version 2 or above of the Option 13 or 14 firmware.)

IN11 (Level 2): Invalid parameter. (Must be 0, 1, or 2.)

IP **<Report-GIN-Point> =^{E}c(I)(P)<int>**

IP13 (Level 2): The device-function code names a device which has already been enabled for a different graphic input function.

IQ **<Report-Terminal-Settings> =^{E}c(I)(Q)<char><char>**

No errors are detected for this command.

IR **<Set-GIN-Rubberbanding> =^{E}c(I)(R)<int><int>**

IR10 (Level 2): Rubberbanding does not apply to the specified device-function code. (Rubberbanding is only allowed for the locator function. It is forbidden for the pick and stroke functions.)

IR21 (Level 2): Invalid Rubberbanding mode (must be 0 or 1).

IS **<Set-Report-Sig-Chars> =^{E}c(I)(S)<int><char><char>**

IS11 (Level 2): Invalid report type code (must be valid device-function code, or in the range from −1 to −3).

IS21 (Level 2): Invalid second parameter (must be in the range from 0 to 127).

IS31 (Level 2): Invalid third parameter (must be in the range from 0 to 127).

IT **TBSTATUS=<Set-Tablet-Status-Strap> =^{E}c(I)(T)<int>**

IT00 (Level 0): Unrecognized command. (The tablet option is not installed.)

IT11 (Level 2): Invalid strap setting (must be 0 or 1).

J0 and J1: Disk System Errors on Power-Up

J002 (Level 3): Memory error detected by standard firmware.

J102 (Level 3): Memory error detected by optional peripheral firmware.

J109 (Level 3): Hardware initialization error in Disk Controller board.

JC **COPY=<Copy> =^{E}c(J)(C)<string><string><string>**

JC03 (Level 2): Attempt to copy an entire disk volume onto itself (e.g., a copy from F0: to F0:).

JC10 (Level 2): Specified source device does not exist.

JC11 (Level 2): Invalid source device specifier.

JC12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the first parameter, or while executing the command.)

JC13 (Level 2): Parameter 1 context error (not an input device).

JC19 (Level 2): Disk hardware error or drive not ready on the source disk drive.

JC21 (Level 2): Invalid separator string (must be empty string or "TO").

JC22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the separator string.)

JC30 (Level 2): Specified destination device does not exist.

JC31 (Level 2): Invalid destination specifier.

JC32 (Level 3): Parameter 3 memory error. (Out of memory while parsing the third parameter, or while executing the command.)

JC33 (Level 2): Parameter 3 context error (not a valid destination device).

JC39 (Level 2): Disk hardware error or drive not ready on the destination diskdrive.

JE **STOP=<Stop-Spooling> =^{E}c(I)(T)**

No errors are detected for this command.

JD DIRECTORY=<Directory>=^{E}c(J)(D)<string><string><string>

JD00 (Level 2): Unrecognized command. (Disk drive option is not installed.)

JD10 (Level 2): The device for which the directory is requested does not exist.

JD11 (Level 2): Invalid input device specifier in parameter 1.

JD12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the source string, or while executing the command.)

JD13 (Level 2): Context error in parameter 1. (The specified device is not a disk drive, or failed reading bit map.)

JD19 (Level 2): Disk hardware error (or drive not ready) for the disk drive whose directory is being requested.

JD21 (Level 2): Invalid separator string (must be empty string or "TO").

JD22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the separator string.)

JD30 (Level 2): The specified destination device does not exist.

JD31 (Level 2): Invalid destination device specifier.

JD32 (Level 3): Parameter 3 memory error. (Out of memory while parsing the destination string, or while executing the command.)

JD33 (Level 2): Parameter 3 context error. (The device specified is not a valid destination device, or is write-protected.)

JD39 (Level 2): Disk hardware error for the destination device. (I/O error, write-protect error, or disk drive not ready.)

JF FORMAT=<Format-Volume>=^{E}c(J)(F)<string><int>

JF00 (Level 2): Unrecognized command. (Disk drive option not installed.)

JF10 (Level 2): Device does not exist or is not installed.

JF11 (Level 2): Invalid device specifier.

JF12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the device specifier string.)

JF13 (Level 2): The device specified in parameter 1 (a) is not a disk drive, (b) is write protected, (c) is busy, (d) detects a verify error, or (e) detects a bit map error.

JF19 (Level 2): Hardware error at the specified disk drive. (Format error, drive not ready, or write-protect switch or notch error.)

JK DELETE=<Delete-File>=^{E}c(J)(K)<string>

JK00 (Level 2): Unrecognized command. (Disk drive option is not installed.)

JK10 (Level 2): The specified file or disk drive does not exist.

JK11 (Level 2): Invalid device specifier.

JK12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1.)

JK13 (Level 2): The specified device (a) is not a disk drive, (b)is write-protected, or (c) detects a bit map error.

JK19 (Level 2): Disk hardware error. (I/O error, drive not ready, or hardware write-protect error.)

JL LOAD=<Load>=Ec(J)(L)<string>

JL02 (Level 3): Out of memory while performing <load> command.

JL03 (Level 2): Nesting error. (<Load> commands are nested too deeply.)

JL10 (Level 2): File or device does not exist.

JL11 (Level 2): Invalid file specifier.

JL12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1, or while executing the command.)

JL13 (Level 2): Context error in parameter 1. (Not a valid source device.)

JL19 (Level 2): Disk hardware error or drive not ready.

JP PROTECT=<Protect-File>=Ec(J)(P)<string><int>

JP00 (Level 2): Unrecognized command. (Disk drive option is not installed.)

JP10 (Level 2): The specified file or disk drive does not exist.

JP11 (Level 2): Invalid file specifier.

JP12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1.)

JP13 (Level 2): Either the specified device is not a disk drive, or the file (or entire diskette volume) has been write-protected.

JP19 (Level 2): Disk hardware error. (I/O error, drive not ready, or hardware write-protect error.)

JP21 (Level 2): Invalid Protection mode (must be 0 or 1).

JQ <Report-Device-Status>=Ec(J)(Q)<string>

JQ00 (Level 2): Unrecognized command. (Disk drive option is not installed.)

JQ10 (Level 2): Device does not exist or is not installed.

JQ11 (Level 2): Invalid device specifier.

JQ12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1.)

JR RENAME=<Rename-File>=Ec(J)(R)<string><string><string>

JR00 (Level 2): Unrecognized command. (Disk drive option is not installed.)

JR10 (Level 2): The specified device or file does not exist.

JR11 (Level 2): Invalid "old" file specifier.

JR12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1.)

JR13 (Level 2): Either the device specified in parameter 1 is not a disk drive, or the file (or entire diskette) has been write-protected.

JR19 (Level 2): Disk hardware error. (I/O error, drive not ready, or hardware write-protect error.)

JR21 (Level 2): Invalid separator string (must be empty string or "TO").

JR22 (Level 3): Parameter 2 memory error. (Out of memory while parsing parameter 2.)

JR30 (Level 2): Either the device specified in parameter 3 does not exist, or the "new" filename already is in use.

JR31 (Level 2): Invalid device specifier in parameter 3.

JR32 (Level 3): Parameter 3 memory error. (Out of memory while parsing parameter 3.)

JS SPOOL=<Spool>=Ec(J)(S)<string><string><string>

JS03 (Level 2): Command context error. (A spooling operation is already in progress.)

JS10 (Level 2): Specified source device does not exist.

JS11 (Level 2): Invalid source specifier. (Must be "HO:", "P0:", "P1:", "P2:", or a file specifier such as "F0: FILENAME". Specifying only the disk drive — such as "F0:" or "F1:" — is not allowed.)

JS12 (Level 3): Parameter 1 memory error. (Out of memory while parsing parameter 1, or while executing the command.)

JS13 (Level 2): Parameter 3 context error. (Not a valid input device.)

JS19 (Level 2): Disk hardware error or drive not ready on the source disk drive.

JS21 (Level 2): Invalid separator string (must be empty string or "TO").

JS22 (Level 3): Parameter 2 memory error. (Out of memory while parsing parameter 2.)

JS30 (Level 2): Specified destination device does not exist.

JS31 (Level 2): Invalid destination device specifier.

JS32 (Level 3): Parameter 3 memory error. (Out of memory while parsing parameter 3, or while executing the command.)

JS33 (Level 2): Parameter 3 context error. (Not a valid destination device.)

JS39 (Level 2): Disk hardware error or drive not ready on the destination disk drive.

JV SAVE=<Save>=Ec(J)(V)<string><int><string><string>

JV11 (Level 2): Invalid type-of-save string. (Must be MAC or SEG.)

JV12 (Level 3): Parameter 1 memory error. (Out of memory while parsing paramter 1, or while executing the command.)

JV20 (Level 2): The specified macro definition or segment does not exist.

JV21 (Level 2): Invalid item number or count.

JV31 (Level 2): Invalid separator string (must be empty string or "TO").

JV32 (Level 3): Parameter 3 memory error. (Out of memory while parsing the separator string.)

JV40 (Level 2): The specified destination device does not exist.

JV41 (Level 2): Invalid destination device specifier.

JV42 (Level 3): Parameter 4 memory error. (Out of memory while parsing the destination string, or while executing the command.)

JV43 (Level 2): Parameter 4 context error. (Not a valid destination device.)

JV49 (Level 2): Disk hardware error on destination disk drive (I/O error, drive not ready, or hardware write-protect error).

K0 : Keyboard System Errors

K0 (Level 3): Out of memory while initializing the keyboard system.

KA DAENABLE=<Enable-Dialog-Area>=Ec(K)(A)<int>

KA11 (Level 2): Parameter out of range (must be 0 or 1).

KC <Cancel>=Ec(K)(C)

No errors are detected for this command.

KD DEFINE=<Define-Macro>=^{E}c(K)(D)<int><int-array>

KD11 (Level 2): Invalid macro number (must be in range −1 to 32767).

KD21 (Level 2): Invalid character-code in <int-array>. (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.

KE ECHO=<Echo>=^{E}c(K)(E)<int>

KE11 (Level 2): Invalid Echo mode (must be 0 or 1).

KF LFCR=<LFCR>=^{E}c(K)(F)<int>

KF11 (Level 2): Invalid LFCR mode (must be 0 or 1).

KH <Hardcopy>=^{E}c(K)(H)<int>

KH11 (Level 2): Invalid hardcopy code (must be 0, 1, or 2).

KI IGNOREDEL=<Ignore-Deletes>=^{E}c(K)(I)<int>

KI11 (Level 2): Invalid Ignore-Deletes mode (must be 0 or 1).

KL LOCKKEYBOARD=<Lock-Keyboard>=^{E}c(K)(L)<int>

KL11 (Level 2): Invalid Keyboard-Lock mode (must be 0 or 1).

KN RENEW=<Renew-View>=^{E}c(K)(N)<int>

KN11 (Level 2): Parameter out of range (must be in the range from −32768 to + 32767). (This parameter **should** be in the range from −1 to 64; however, the terminal will substitute −1 in place of a value which is less than −1, and + 64 in place of a value in the range from 65 to 32767.)

KP PAGEFULL=<Set-Page-Full-Action>=^{E}c(K)(P)<int>

KP11 (Level 2): Invalid page-full-action code (must be in the range from 0 to 7).

KQ <Report-Errors>=^{E}c(K)(Q)

No errors are detected for this command.

KR CRLF=<CRLF>=^{E}c(K)(R)<int>

KR11 (Level 2): Invalid "CR-implies-LF" mode (must be 0 or 1).

KS SNOOPY=<Snoopy>=^{E}c(K)(S)<int>

KS11 (Level 2): Invalid parameter (must be 0 or 1).

KT ERRORLEVEL=<Set-Error-Threshold>=^{E}c(K)(T)<int>

KT11 (Level 2): Invalid error threshold (must be in range from 0 to 4).

KX **<Expand-Macro>=^{E}c(K)(X)<int>**

KX01 (Level 2): The maximum nesting depth (for <expand-macro> and <load> commands) has been exceeded. (The nesting depth should not exceed five. Greater nesting depths may result — but do not necessarily result — in type 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 0 to 32767).

KY **KEYEXCHAR=<Set-Execute-Toggle-Char>=^{E}c(K)(Y)<int>**

KY11 (Level 2): Invalid <key-execute-toggle> code (must be in range 0 to 127).

KZ **EDITCHARS=<Set-Edit-Chars>=^{E}c(K)(Z)<int><int><int**

KZ11 (Level 2): Invalid char-delete character (must be in the range from 0 to 127).

KZ21 (Level 2): Invalid line-delete character (must be in the range from 0 to 127).

KZ31 (Level 2): Invalid take-literally character (must be in the range from 0 to 127).

LB **DABUFFER=<Set-Dialog-Area-Buffer-Size>=^{E}c(L)(B)<<int+>**

LB11 (Level 2): Invalid number-of-lines parameter (must be in the range from 2 to 65535).

LC **DACHARS=<Set-Dialog-Area-Chars>=^{E}c(L)(C)<int+>**

LC11 (Level 2): Invalid number of characters per line (must be in the range from 5 to 819).

LE **<End-Panel>=^{E}c(L)(E)**

LE00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

LE03 (Level 1): No panel is currently being defined.

LE02 (Level 3): Out of memory while performing <end-panel> command.

LF **<Move>=^{E}c(L)(F)<xy>**

No errors are detected for this command.

LG **<Draw>=^{E}c(L)(G)<xy>**

No errors are detected for this command.

LH **<Draw-Marker>=^{E}c(L)(H)<xy>**

No errors are detected for this command.

LI **DAINDEX=<Set-Dialog-Area-Index>=^{E}c(L)(I)<int+><int+><int+>**

LI00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

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).

LK **<Include-Copy-of-Segment>=^{E}c(L)(K)<int>**

LK02 (Level 3): Out of memory while performing <include-copy-of-segment>.

LK10 (Level 2): Segment does not exist.

LK11 (Level 2): Invalid segment number (must be −3, −1, or in the range from 1 to 32767.)

LL **DALINES=<Set-Dialog-Area-Lines>=^{E}c(L)(L)<int+>**

LL11 (Level 2): Invalid parameter (must be in the range from 2 to 520).

LM **DAMODE=<Set-Dialog-Area-Writing-Mode>=^{E}c(L)(M)<int>**

LM11 (Level 2): Invalid Writing mode (must be 0 or 1).

LP **<Begin-Panel-Boundary>=^{E}c(L)(P)<xy><int>**

LP00 (Level 0): Unrecognized command. (Terminal is not a 4112.)

LP21 (Level 2): Invalid "draw border" mode (must be 0 or 1).

LS **<Set-Dialog-Area-Surface>=^{E}c(L)(S)<int>**

LS00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

LS10 (Level 2): Surface does not exist.

LS11 (Level 2): Invalid surface number (must be 1, 2, or 3).

LT **<Graphic-Text>=^{E}c(L)(T)<string>**

LT11 (Level 2): Invalid graphtext string. Invalid array count (must be in the range from 0 to 32767), or invalid <char> character in the array (must be in the range from ADE value 26 [space] to 126 [tilda]).

LT12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <string> parameter.)

LV **DAVIS=<Set-Dialog-Area-Visibility>=^{E}c(L)(V)<int>**

LV03 (Level 0): One or more of the dialog area parameters was altered when the dialog area was made visible.

LV11 (Level 2): Invalid dialog area visibility mode. (Must be 0 or 1; in Setup mode, must be YES or NO.)

LX **DAPOSITION=<Set-Dialog-Area-Position>=^{E}c(L)(X)<xy>**

No errors are detected for this command.

LZ **<Clear-Dialog-Scroll>=^{E}c(L)(Z)**

No errors are detected for this command.

MC <**Set-Graphtext-Size**>=^{E}c(M)(C)<int><int><int>

MC11 (Level 2): Invalid value in parameter 1 (must be in the range from 1 to 4095).

MC21 (Level 2): Invalid value in parameter 2 (must be in the range from 1 to 4095).

MC31 (Level 2): Invalid value in parameter 3 (must be in the range from 0 to 4095).

MD <**Begin-Fill-Pattern**>=^{E}c(M)(D)<int><int><int><int>

MD00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

MD02 (Level 3): Not enough memory available for fill pattern.

MD11 (Level 2): Invalid fill pattern number (must be in the range from 1 to 32767).

MD21 (Level 2): Invalid pattern width (must be in the range 1 to 32).

MD31 (Level 2): Invalid pattern height (must be in the range 0 to 480).

MD41 (Level 2): Invalid bits-per-pixel (must be 1, 2, 3, or 6).

ME <**End-Fill-Pattern**>=^{E}c(M)(E)

No errors are detected for this command.

MF <**Set-Graphtext-Font**>=^{E}c(M)(F)<int>

MF10 (Level 2): Font does not exist.

MF11 (Level 2): Invalid font number (must be in the range 0 to 32767).

MG **GAMODE=**<**Set-Graphics-Area-Writing-Mode**>=^{E}c(M)(G)<int>

MG00 (Level 2): Unrecognized command. (Terminal is not a 4112.)

MG11 (Level 2): Invalid parameter (must be 0 or 1; in SETUP mode, must be OVERSTRIKE or REPLACE).

MI <**Set-Pick-ID**>=^{E}c(M)(I)<int>

MI03 (Level 2): Command is invalid at this time (no segment is currently being defined).

MI11 (Level 2): Invalid pick identification number (must be in the range from 0 to 32767).

ML <**Set-Line-Index**>=^{E}c(M)(L)<int+>

ML11 (Level 2): Invalid line index (must be in the range from 0 to 32767).

MM <**Set-Marker-Type**>=^{E}c(M)(M)<int>

MM11 (Level 2): Invalid marker type (must be in the range from 0 to 10).

MP **<Select-Fill-Pattern>=^{E}c(M)(P)<int>**

MP00 (Level 2): Unrecognized command (terminal is not a 4112).

MP10 (Level 2): Specified fill pattern does not exist (has not been defined).

MP11 (Level 2): Invalid fill pattern number (must be in the range from −32768 to 32767).

MQ **<Set-Graphtext-Precision>=^{E}c(M)(Q)<int>**

MQ11 (Level 2): Invalid Precision mode (must be 1 or 2).

MR **<Set-Graphtext-Rotation>=^{E}c(M)(R)<real>**

MR11 (Level 2): Invalid rotation angle (must be in the range from −32767.0 to + 32767.0).

MS **<Set-Panel-Filling-Mode>=^{E}c(M)(S)<int><int><int>**

MS00 (Level 2): Unrecognized command. (Terminal is not a 4112.)

MS11 (Level 2): Invalid overstrike/replace mode parameter (must be 0 or 1).

MS21 (Level 2): Invalid boundary-filling mode parameter (must be 0 or 1).

MS31 (Level 2): Invalid pattern keying value (must be 0, 1, 2, or 3).

MT **<Set-Text-Index>=^{E}c(M)(T)<int+>**

MT11 (Level 2): Invalid text index (must be in the range from 0 to 65535).

MV **<Set-Line-Style>=^{E}c(M)(V)<int>**

MV11 (Level 2): Invalid line style (must be in the range from 0 to 7).

NB **STOPBITS=<Set-Stop-Bits>=^{E}c(N)(B)<int>**

NB11 (Level 2): Invalid number of stop bits (must be 1 or 2).

NC **EOMCHARS=<Set-EOM-Chars>=^{E}c(N)(C)<int><int>**

NC11 (Level 2): Invalid value in parameter 1 (must be in the range from 0 to 127).

NC21 (Level 2): Invalid value in parameter 2 (must be in the range from 0 to 127).

ND **XMTDELAY=<Set-Transmit-Delay>=^{E}c(N)(D)<int>**

ND11 (Level 2): Invalid transmit delay time (must be in the range from 0 to 65535 ms).

NE **EOFSTRING=<Set-EOF-String>=^{E}c(N)(E)<int-array>**

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 <int-array> parameter).

NF FLAGGING=<Set-Flagging-Mode>=^{E}c(N)(F)<int>

NF11 (Level 2): Invalid Flagging mode (must be in the range from 0 to 4).

NK BREAKTIME=<Set-Break-Time>=^{E}c(N)(K)<int+>

NK11 (Level 2): Invalid parameter. (Must be in the range from 0 to 65535.)

NL XMTLIMIT=<Set-Transmit-Rate-Limit>=^{E}c(N)(L)<int+>

NL11 (Level 2): Invalid transmit rate limit. (Must be in the range from 110 to 65535.)

NM PROMPTMODE=<Prompt-Mode>=^{E}c(N)(M)<int>

NM11 (Level 2): Invalid Prompt mode parameter (must be 0, 1,or 2).

NP PARITY=<Set-Parity>=^{E}c(N)(P)<int>

NP11 (Level 2): Invalid parity code (must be in the range from 0 to 4).

NQ QUEUESIZE <Set-Queue-Size>=^{E}c(N)(Q)<int+>

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).

NR BAUDRATE=<Set-Baud-Rates>=^{E}c(N)(R)<int+><int+>

NR11 (Level 2): Invalid transmit (terminal-to-host) data rate (must be 1, 50, 75, 110, 134, 150, 300, 600, 1200, 1200, 1800, 2000, 2400, 4800, 9600, 19200, or 38400).

NR21 (Level 2): Invalid receive (host-to-terminal) data rate (must be 0, 1, 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, or 38400).

NS PROMPTSTRING=<Set-Prompt-String>=^{E}c(N)(S)<int-array>

NS11 (Level 2): Invalid <int-array> parameter. (Must be an array holding from 0 to 10 <int> parameters. Each of the items in the array must be an <int> in the range from 0 to 127.)

NS12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <int-array>.

NT EOLSTRING=<Set-Term-String>=^{E}c(N)(T)<int-array>

NT11 (Level 2): Invalid array count in <int-array>. (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 <int-array> parameter.)

NU BYPASSCANCEL=<Set-Bypass-Cancel-Char>=^{E}c(N)(U)<int>

NU11 (Level 2): Invalid numeric equivalent of bypass-cancel character (must be in the range from 0 to 127).

OB BLOCKMODE=<Arm-For-Block-Mode>=^{E}c(O)(B)<int>

OB00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OB03 (Level 2): The communications queue size is smaller than the specified input block size.

OB11 (Level 2): Invalid "arm-for-block-mode" parameter (must be 0 or 1).

OC BCONTINUECHARS=<Set-Block-Continue-Chars>=^{E}c(O)(C)<int><int>

OC00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OC03 (Level 2): Command is invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OC11 (Level 2): Invalid first parameter (must be in the range from 0 to 127).

OC21 (Level 2): Invalid second parameter (must be in the range from 0 to 127).

OD DUPLEX=<Set-Duplex-Mode>=^{E}c(O)(D)<int>

OD00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OD01 (Level 2): Invalid duplex mode code (must be in range 0 to 3).

OE BENDCHARS=<Set-Block-End-Chars>=^{E}c(O)(E)<int><int>

OE00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OE03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OE11 (Level 2): Invalid first parameter (must be in the range from 0 to 127).

OE21 (Level 2): Invalid second parameter (must be in the range from 0 to 127).

OH BHEADERS=<Set-Block-Headers>=^{E}c(O)(H)<int-array><int-array>

OH00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OH02 (Level 3): Out of memory while performing <set-block-headers> command.

OH03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OH11 (Level 2): Invalid character code (0 to 127) or array count (must be in range 0 to 10) in "transmit" (terminal-to-host) block header.

OH12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <int-array>.)

OH21 (Level 2): Invalid character code (O to 127) or array count (must be in range 0 to 10) in "receive" (host-to-terminal) block header.

OH22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the <int-array>.)

OL BLINELENGTH=<Set-Block-Line-Length>=^{E}c(O)(L)<int>

OL00 (Level 2): Unrecognized command. (Option 1 not installed.)

OL03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OL11 (Level 2): Invalid maximum number of characters in line (must be in the range from 5 to 65535).

OM BMASTERCHARS=<Set-Block-Master-Chars>=^{E}c(O)(M)<int><int>

OM00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OM03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OM11 (Level 2): Invalid "transmit" (terminal-to-host) master char code (0 to 127).

OM21 (Level 2): Invalid "receive" (host-to-terminal) master char code (0 to 127).

ON BNONXMTCHARS=<Set-Block-Non-Xmt-Chars>=^{E}c(O)(N)<int-array><int-array>

ON00 (Level 2): Unrecognized command. (Option 1 is not installed.)

ON03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

ON11 (Level 2): Invalid character code or array count in list of terminal-to-host non-transmittable characters. (The array count must be in the range from 0 to 20, and the character codes must be in the range from 0 to 127.)

ON12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <int-array>.)

ON21 (Level 2): Invalid character code or array count in list of host-to-terminal non-transmittable characters. (The array count must be in the range from 0 to 20, and the character code must be in the range from 0 to 127.)

ON22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the <int-array>.)

OP BPACKING=<Set-Block-Packing>=^{E}c(O)(P)<int><int><int><int>

OP00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OP03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OP11 (Level 2): Invalid terminal-to-host unpacked-bits-per-byte (must be 7 or 8).

OP21 (Level 2): Invalid terminal-to-host packed-bits-per-pseudo-byte (must be 6, 7, or 8).

OP31 (Level 2): Invalid host-to-terminal unpacked-bits-per-byte (must be 7 or 8).

OP41 (Level 2): Invalid host-to-terminal packed-bits-per-pseudo-byte (must be 6, 7, or 8).

OS BLENGTH=<Set-Block-Length>=^{E}c(O)(S)<int+><int+>

OS00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OS03 (Level 2): Command invalid at this time. (Terminal must not be in Block mode or armed for Block mode.)

OS11 (Level 2): Invalid terminal-to-host block length (must be in the range from 5 to 65535.)

OS21 (Level 2): Invalid host-to-terminal block length (must be in the range from 5 to 65535.)

OT BTIMEOUT=<Set-Block-Timeout>=^{E}c(O)(T)<int+>

OT00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OT11 (Level 2): Invalid timeout (must be in the range from 0 to 65535 seconds).

PA PASSIGN=<Port-Assign>=^{E}c(P)(A)<string><string><int>

PA00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PA11 (Level 2): Invalid port identifier (must be "P0:","P1:", or "P2:").

PA12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PA13 (Level 2): Port in use.

PA21 (Level 2): Invalid protocol identifier (must be "4642," "4643," "4662," "4662/T," "4663," "4663/T," or "PPORT").

PA22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the <string>.)

PA31 (Level 2): Invalid device code (must be in the range from −32768 to + 32767).

PB PBITS=<Set-Port-Stop-Bits>=^{E}c(P)(B)<string><int><int>

PB00 (Level 0): Unrecognized command. (Option 10 is not installed).

PB11 (Level 2): Port identifier is invalid (must be "P0:", "P1:", or "P2:").

PB12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PB13 (Level 2): Port in use.

PB21 (Level 2): Invalid number of stop bits (must be 1 or 2).

PB31 (Level 2): Invalid number of data bits (must be 5, 6, 7, or 8).

PC PCOPY=<Port-Copy>=^{E}c(P)(C)<string>

PC00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PC02 (Level 3): 3PPI ISR transmit/receive error detected.

PC11 (Level 2): Invalid "source" port (must be "HO:", "P0:", "P1:", or "P2:").

PC12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the source specifier string.)

PC13 (Level 2): Port in use.

PC21 (Level 2): Invalid separator string (must be the empty string or "TO").

PC22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the separator string.)

PC23 (Level 2): Destination is busy.

PC31 (Level 2): Invalid "destination" port (must be "HO:", "P0:", "P1:" or "P2:", and must be different from the "source" port).

PC32 (Level 3): Parameter 3 memory error. (Out of memory while parsing the destination specifier.)

PE PEOF=<Set-Port-EOF-String>=^{E}c(P)(E)<string><int-array>

PE00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PE11 (Level 2): Port identifier is invalid (must be "P0:", "P1:", or "P2:").

PE12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PE13 (Level 2): Port in use.

PE21 (Level 2): Invalid EOF string. (The <int-array> must have from 0 to 10 elements, and each <int> in the array must be in the range from 0 to 127.)

PE22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the end-of-file string <int-array>.

PF PFLAG=<Set-Port-Flagging-Mode>=^{E}c(P)(F)<string><int><int><int>

PF00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PF11 (Level 2): Port identifier is invalid (must be "P0:", "P1:", or "P2:").

PF12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PF13 (Level 2): Port in use.

PF21 (Level 2): Invalid Flagging mode (must be 0, 1 or 2).

PF31 (Level 2): Invalid "GO" character (must be in range 0 to 127).

PF41 (Level 2): Invalid "STOP" character (must be in range 0 to 127; if non-zero, must be different from the "GO" character).

PI PMAP=<Map-Index-To-Pen>=EscC(P)(I)<string><int><int>

PI00 (Level 0): Unrecognized command (Option 10 is not installed).

PI02 (Level 3): No memory is available for the index map. (To guarantee an available index map for a particular peripheral port, the peripheral port should have a plotter device assigned to it when the terminal is turned on.)

PI11 (Level 3): Port identifier is invalid (must be "P0:", "P1:", or "P2:").

PI12 (Level 2): Parameter 1 memory error. (Out of memory while parsing the port name string.)

PI13 (Level 2): Port in use.

PI21 (Level 2): Invalid index (must be in the range from −2 to + 32767).

PI31 (Level 2): Invalid pen number (must be in the range from −1 to 255).

PI102 (Level 3): Not enough memory for index map.

PL PLOT=<Plot>=EscC(P)(L)<string><string>

PL00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PL11 (Level 2): Invalid first parameter (must be the empty string or "TO").

PL13 (Level 2): Device already in use.

PL12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <string> parameter.)

PL20 (Level 2): Non-existent destination device. (If destination specifier is of the form "F0:filename" or "F1:filename," then the corresponding disk drive must be installed, and the file named must exist on a disk loaded in that disk drive).

PL21 (Level 2): Invalid output specifier (must be "H0:", "P0:", "P1:", "P2:", "F0:filename" or "F1:filename").

PL22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the destination <string>.)

PM PEOL=<Set-Port-EOL-String>=EscC(P)(M)<string><int-array>

PM00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PM11 (Level 2): Invalid port identifier (must be "P0:", "P1:", or "P2:").

PM12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PM13 (Level 2): Port in use.

PM21 (Level 2): Invalid EOL string (must have 0, 1, or 2 characters).

PM22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the EOL-string <int-array>.)

PP PPARITY=<Set-Port-Parity>=^{E}c(P)(P)<string><int>

PP00 (Level 0): Unrecognized command. (Option 10 is not installed).

PP11 (Level 2): Port specifier is invalid (must be "P0:", "P1:", or "P2:").

PP12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name <string>.)

PP13 (Level 2): Port in use.

PP21 (Level 2): Invalid parity code (must be in range 0 to 4).

PQ PORTSTATUS=<Report-Port-Status>=^{E}c(P)(Q)<string>

PQ00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PQ02 (Level 3): Memory error. (Out of memory while trying to retrieve status information.)

PQ11 (Level 2): Invalid port identifier (must be "P0:", "P1:", or "P2:").

PQ12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port specifier string.)

PR PBAUD=<Set-Port-Baud-Rate>=^{E}c(P)(R)<string><int>

PR00 (Level 0): Unrecognized command. (Option 10 is not installed.)

PR11 (Level 2): Port identifier is invalid (must be "P0:", "P1:", or "P2:").

PR12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the port name string.)

PR13 (Level 2): Port in use.

PR21 (Level 2): Invalid baud rate (must be 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, or 9600).

RA <Set-View-Attributes>=$^{E_{e}]C}$(R)(A)<int><int+><int+>

RA00 (Level 0): Unrecognized command. (The terminal is not a 4112).

RA10 (Level 2): Surface does not exist (has not been defined with <set-surface-definitions> command).

RA11 (Level 2): Invalid surface number (must be in the range from −1 to + 3).

RA21 (Level 2): Invalid wipe index (must be in the range from 0 to 65535).

RA31 (Level 2): Invalid border index (must be in the range from 0 to 65535).

RB <Set-Background-Gray-Level>=^{E}c(R)(B)<int>

RB00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RB11 (Level 2): Invalid gray level (must be in the range from 0 to 100).

RC <Select-View> =^{E}c(R)(C)<int>

RC00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RC11 (Level 2): Invalid view number (must be in the range from −1 to 64).

RD <Set-Surface-Definitions> =^{E}c(R)(D)<int-array>

RD00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RD10 (Level 2): Occupied undefined surface. (This command would have resulted in a dialog area viewport, pixel viewport, or numbered graphic viewport residing on an undefined surface.)

RD11 (Level 2): Invalid <int-array> parameter. (The array count must be in the range from 1 to 3. Each <int> in the array must be in the range to 3. The sum of the <int>s in the array must be in the range from 1 to 3.)

RD12 (Level 3): Parameter 3 memory error. (Out of memory while trying to parse the <int-array> parameter.)

RE <Set-Border-Visibility> =^{E}c(R)(E)<int>

RE00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RE11 (Level 2): Invalid "border mode" parameter (must be 0, 1, or 2).

RF FIXUP=<Set-Fixup-Level> =^{E}c(R)(F)<int>

RF00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RF11 (Level 2): Invalid fixup level (must be in the range 0 to 6).

RG <Set-Surface-Gray-Levels> =^{E}c(R)(G)<int><int±array>

RG00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RG10 (Level 2): Surface does not exist (has not been defined with a <set-surface-definitions> command).

RG11 (Level 2): Invalid surface number (must be in the range 1 to 3).

RG21 (Level 2): Invalid surface-gray-levels array. (The array count must be even; the first <int+> in each pair must be a gray-index in the range from 1 to 65535; the second <int> in each pair must be a valid gray-level: a number in the range from 0 to 100.)

RG22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the <int-array> parameter.)

RH <Set-Pixel-Beam-Position> =^{E}c(R)(H)<xy>

RH00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RI **<Set-Surface-Visibility>=^{E}c(R)(I)<int-array>**

RI00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RI10 (Level 2): A surface number in the <int-array> is for a surface which does not exist (that is, a surface which has not been defined with a <set-surface-definitions> command).

RI11 (Level 2): Invalid <int-array>. (The <int-array> consists of (surface number, visibility) pairs; the "surface number" <int>s must be 1, 2, or 3; and the "visibility" <int>s must be 0, 1, or 2.)

RI12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <int-array> parameter.)

RJ **<Lock-Viewing-Keys>=^{E}c(R)(J)<int>**

RJ00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RJ11 (Level 2): Invalid locking mode parameter (must be 0 or 1).

RK **<Delete-View>=^{E}c(R)(K)<int>**

RK00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RK10 (Level 2): The designated view does not exist (has not been defined with a <select-view> command).

RK11 (Level 2): Invalid view number (must be in the range from −1 to 64).

RL **<Runlength-Write>=^{E}c(R)(L)<int±array>**

RL00 (Level 2): Unrecognized command. (The terminal is not a 4112.)

RL11 (Level 2): Invalid runlength-code array. (The array count must be in the range from 0 to 65535, and each <int+> in the array must also be in the range from 0 to 65535.)

RL12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the runlength-code array, or while attempting to execute the command.)

RN **<Set-Surface-Priorities>=^{E}c(R)(N)<int-array>**

RN00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RN10 (Level 2): Surface does not exist (has not been defined with a <set-surface-definitions> command).

RN11 (Level 2): Invalid surface priorities array. (Each <int> in the <int-array> — whether a surface number or a priority number — must be in the range from 1 to 3.)

RN12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the <int-array> parameter.)

RP **<Raster-Write> =^{E}c(R)W)<int><char-array>**

RP00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RP11 (Level 2): Invalid number of pixels (must be in the range from 0 to 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 <char-array> parameter.)

RR **<Rectangle-Fill> =^{E}c(R)(R)<xy><xy><int+>**

RR00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RR11 (Level 2): Invalid "lower left" coordinates. (X must be in the range from 0 to 639, and Y in the range from 0 to 479.)

RR21 (Level 2): Invalid "upper right" coordinates. (X must be in the range from 0 to 639, and Y in the range from 0 to 479.)

RR31 (Level 2): Invalid gray-index (must be in the range from 0 to 65535.)

RS **<Set-Pixel-Viewport> =^{E}c(R)(S)<xy><xy>**

RS00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RS11 (Level 2): Invalid "lower left" coordinate. (X must be in the range from 0 to 639, and Y in the range from 0 to 479.)

RS21 (Level 2): Invalid "upper right" coordinate. (X must be in the range from 0 to 639, and Y in the range from 0 to 479.)

RU **<Begin-Pixel-Operations> =^{E}c(R)(U)<int><int><int>**

RU00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RU10 (Level 2): Surface does not exist (has not been defined with a <set-surface-definitions> command).

RU11 (Level 2): Invalid surface number (must be in the range from −1 to 3).

RU21 (Level 2): Invalid ALU mode (must be in the range from 0 to 16).

RU31 (Level 2): Invalid bits-per-pixel (must be 0, 1, 2, 3, or 6).

RV **<Set-Viewport> =^{E}c(R)(V)<xy><xy>**

RV00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RV11 (Level 2): Invalid "lower left" corner. (X must be in the range from 0 to 4095, and Y in the range from 0 to 3071.)

RV21 (Level 2): Invalid "upper right" corner. (X must be in the range from 0 to 4095, and Y in the range from 0 to 3071.)

RW **<Set-Window>=^{E}c(R)(W)<xy><xy>**

RW00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RX **<Pixel-Copy>=^{E}c(R)(X)<int><xy><xy><xy>**

RX00 (Level 0): Unrecognized command. (The terminal is not a 4112.)

RX10 (Level 2): The specified surface does not exist.

RX21 (Level 2): Invalid destination-lower-left-corner. (X must be in the range from 0 to 639, and Y must be in the range from 0 to 479.)

RX31 (Level 2): Invalid first-source-corner. (Same range as for the destination-lower-left-corner.)

RX41 (Level 2): Invalid second-source-corner. (Same range as for the destination-lower-left-corner and the first-source-corner.)

SA **<Set-Segment-Class>=^{E}c(S)(A)<int><int-array><int-array>**

SA03 (Level 2): Command invalid at this time: the specified segment is currently being defined.

SA10 (Level 2): Segment does not exist.

SA11 (Level 2): Invalid segment number (must be in the range from −3 to −1, or from 1 to 32767).

SA21 (Level 2): Invalid "removal" class number array. (Each class number must be −1 or in the range from 1 to 64.)

SA22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the "removal" <int-array> parameter.)

SA31 (Level 2): Invalid "include" class number array. (Same rules as for the "removal" class number array.)

SA32 (Level 3): Parameter 3 memory error. (Out of memory while parsing the "addition" <int-array> parameter.)

SC **<End-Segment>=^{E}c(S)(C)**

SC03 (Level 2): Invalid at this time: no segment is currently being defined.

SC02 (Level 3): Out of memory while performing <end-segment> command.

SD **<Set-Segment-Detectability>=^{E}c(S)(D)<int><int>**

SD03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SD10 (Level 2): Segment does not exist.

SD11 (Level 2): Invalid segment number (must be in the range −3 to −1, or 1 to 32767).

SD21 (Level 2): Invalid "detectability mode" (must be 0 or 1).

SG **<Set-Graphtext-Font-Grid>=^{E}c(S)(G)<int><int><int>**

SG02 (Level 3): Out of memory while defining font grid.

SG10 (Level 2): Font already exists.

SG11 (Level 2): Invalid font number (must be in the range from 0 to 32767).

SG21 (Level 2): Invalid grid width (must be in the range from 1 to 4095).

SG31 (Level 2): Invalid grid height (must be in the range from 1 to 4095).

SH **<Set-Segment-Highlighting>=^{E}c(S)(H)<int><int>**

SH03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SH10 (Level 2): Segment does not exist.

SH11 (Level 2): Invalid segment number (must be in the range from −3 to −1, or from 1 to 32767).

SH21 (Level 2): Invalid Highlighting mode (must be 0 or 1).

SI **<Set-Segment-Image-Transform>=^{E}c(S)(I)<int><real><real><real><xy>**

SI03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SI02 (Level 3): Out of memory while transforming segment.

SI10 (Level 2): Segment does not exist.

SI11 (Level 2): Invalid segment number (must be in the range −3 to −1, or 1 to 32767).

SI21 (Level 2): Invalid X-scaling factor. (Must be in the range from −32767.0 to + 32767.0.)

SI31 (Level 2): Invalid Y-scaling factor. (Must be in the range from −32767.0 to + 32767.0.)

SI41 (Level 2): Invalid rotation angle. (Must be in the range from −32767.0 to + 32767.0.)

SK **<Delete-Segment>=^{E}c(S)(K)<int>**

SK10 (Level 1): Segment does not exist.

SK11 (Level 2): Invalid segment number (must be −3, −1, or in the range from 1 to 32767).

SL **<Set-Current-Matching-Class>=^{E}c(S)(L)<int-array><int-array>**

SL11 (Level 2): Invalid "include" segment-class array. (Class numbers must be in the range from 1 to 64.)

SL12 (Level 3): Parameter 1 memory error. (Out of memory while parsing the "include" segment-class array.)

SL21 (Level 2): Invalid "exclude" segment-class array. (Class numbers must be in the range from 1 to 64.)

SL22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the "exclude" segment-class array.)

SM <**Set-Segment-Writing-Mode**> =^{E}c**(S)(M)**<**int**><**int**>

SM03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SM10 (Level 2): Segment does not exist.

SM11 (Level 2): Invalid segment number (must be in the range from −3 to −1, or from 1 to 32767).

SM21 (Level 2): Invalid Writing mode (must be 0 or 1).

SO <**Begin-Segment**> =^{E}c**(S)(O)**<**int**>

SO02 (Level 3): Out of memory while defining segment.

SO03 (Level 2): Command is invalid at this time. (Another segment, a graphtext character, or a panel is currently being defined.)

SO10 (Level 2): Segment already exists.

SO11 (Level 2): Invalid segment number (must be in the range from 1 to 32767).

SP <**Set-Pivot-Point**> =^{E}c**(S)(P)**<**xy**>

No errors are detected for this command.

SQ <**Report-Segment-Status**> =^{E}c**(S)(Q)**<**int**><**char-array**>

SQ10 (Level 2): Segment does not exist.

SQ11 (Level 2): Invalid segment number (must be in the range −3 to + 32767).

SQ21 (Level 2): Invalid array of codes. (Must include only the uppercase letters A, D, H, I, M, P, S, V, and X. Also, the array count must be in the range from 0 to 32767.)

SQ22 (Level 3): Parameter 2 memory error. (Out of memory while parsing the aray-of-codes <char-array>.)

SR <**Rename-Segment**> =^{E}c**(S)(R)**<**int**><**int**>

SR03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SR10 (Level 2): Segment does not exist.

SR11 (Level 2): Invalid segment number (must be in the range from 1 to 32767).

SR20 (Level 2): A segment with that segment number already exists.

SR21 (Level 2): Invalid segment number (must be in the range from 1 to 32767).

SS **<Set-Segment-Display-Priority> =^{E}c(S)(S)<int><int>**

SS03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SS10 (Level 2): Segment does not exist.

SS11 (Level 2): Invalid segment number (must be in the range from −3 to −1, or from 1 to 32767).

SS21 (Level 2): Invalid priority number (must be in range −32768 to + 32767).

ST **<Begin-Graphtext-Character> =^{E}c(S)(T)<int><int>**

ST02 (Level 3): Out of memory while defining graphtext character.

ST03 (Level 2): Command is invalid at this time. (A segment, panel, or graphtext character is currently being defined.)

ST11 (Level 2): Invalid font number (must be in the range from 0 to 32767).

ST20 (Level 2): That character already exists in this font.

ST21 (Level 2): Invalid character number (must be in the range from 32 to 126).

SU **<End-Graphtext-Character> =^{E}c(S)(U)**

SU03 (Level 1): This command is invalid at this time. (No graphtext character is being defined.)

SV **<Set-Segment-Visibility> =^{E}c(S)(V)<int><int>**

SV03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SV10 (Level 2): Segment does not exist.

SV11 (Level 2): Invalid segment number (must be in the range from −3 to + 32767).

SV21 (Level 2): Invalid visibility mode (must be 0 or 1).

SX **<Set-Segment-Position> =^{E}c(S)(X)<int><xy>**

SX03 (Level 2): Command is invalid at this time: the specified segment is currently being defined.

SX10 (Level 2): Segment does not exist.

SX11 (Level 2): Invalid segment number (must be in the range from −3 to + 32767.)

SZ **<Delete-Graphtext-Character> =^{E}c(S)(Z)<int><int>**

SZ03 (Level 2): Command is invalid at this time. (A graphtext character is currently being defined.)

SZ10 (Level 1): The specified font does not exist (no characters have been defined for that font).

SZ11 (Level 2): Invalid font number (must be in the range from −1 to 32767).

SZ20 (Level 1): The character specified does not exist in this font.

SZ21 (Level 2): Invalid character number (must be −1, or in the range from 32 to 126).

Appendix B

POWER UP SEQUENCE AND SELF TEST

This appendix explains the messages which can appear on the screen during the power up sequence if a problem is detected. It also explains how to use the terminal's extensive self test procedure.

THE POWER UP SEQUENCE

Each time you turn the terminal on, it automatically tests parts of its memory and circuitry. This is called the "power up sequence." It takes from 15-60 seconds depending on what options your terminal has, and requires no operator interaction.

During the power up sequence the lights in the keyboard go through the following sequence:

- All lights turn on.
- The CAPS LOCK light turns off.
- All other lights turn off.
- The XMT and RCV lights flash once simultaneously.
- All lights turn off.

If the terminal bell does not ring during the power up sequence (within 60 seconds of turning the terminal on), the cursor flashes in the upper left corner of the screen. If the dialog area is enabled, the light in the DIALOG key is on and the cursor appears on the first available line of the dialog area. As soon as the cursor appears on the screen, the terminal is ready for use.

If the power up sequence detects an error the terminal bell rings one, three, or four times, depending on the severity of the problem. There may also be a message on the screen.

POWER UP ERROR

If the power up sequence encounters an error which will not prevent you from using the terminal (a non-fatal error), the bell rings one time and a message is displayed on the screen.

If the terminal bell rings and you cannot read the message that is displayed, press the RESET button (Figure B-1). This causes the screen to be erased and the power up sequence to be run again so you can read the message.

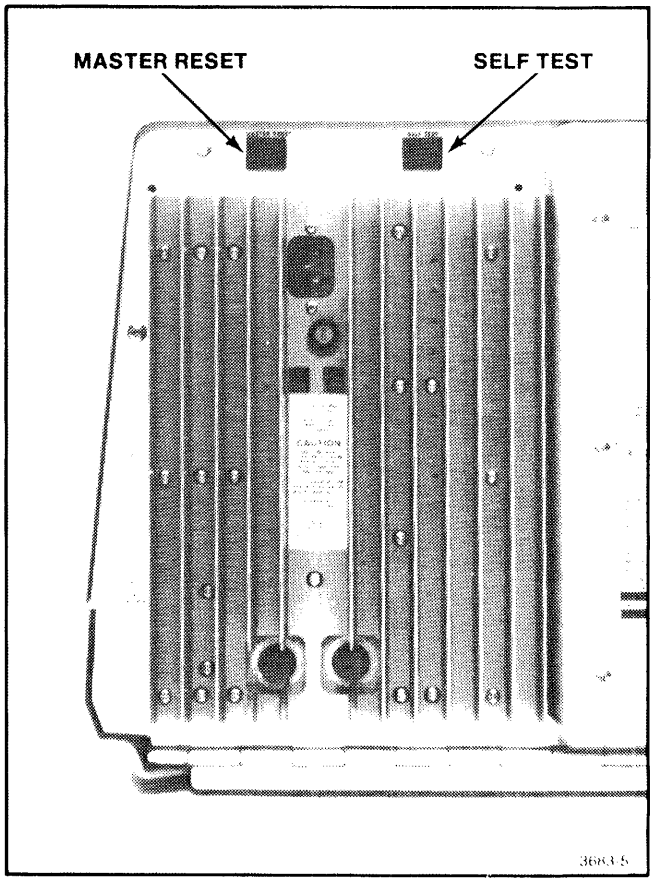


Figure B-1. The MASTER RESET and SELF TEST Buttons.

An example of a such an error is when a problem exists which prevents you from using one of the terminal's options. If this happens, the bell rings once and a message is displayed. For example, if the disk option failed the power up sequence, the following message would appear:

ROM Set: 42-Fail

If there is a problem with a tablet option, the following message appears:

ROM Set: 13-Fail

When an option fails, that option cannot be accessed but the rest of the terminal's features can still be used. You can report the failure to a service person, or run the self test procedure (described later in this appendix) to isolate the failure more specifically.

If an error occurs when the terminal is checking its keyboard, the following message can appear:

Keyboard I-D – XX

where XX is a keyboard ID value. If this occurs, record the message and contact a service person.

The terminal has a "setup memory," which remembers many of the terminal's parameters or settings after it is turned off. These attributes are reinstated the next time the terminal is turned on. If that memory fails, the terminal bell rings once and the following message appears on the screen:

Setup Defaults Reset

This occurs if the terminal's configuration has been altered by putting in or taking out a circuit board. It also occurs if the battery which powers the setup memory fails. It is important to note that if you change one or more boards, only the parameters affected by the change are reset. If the setup memory battery fails, all parameters are reset to default.

FATAL ERRORS

The terminal is not operational (fatal error) if its standard system fails. If that happens, the terminal bell rings four times, once when the following message appears on the screen and three times when the fatal error is reported:

ROM Set: 00-Fail

If the power up sequence encounters a fatal error before the screen is checked, the terminal bell rings three times and the indicator and keyboard lights blink in a particular sequence. The sequence in which they flash indicates the test which has failed.

If this occurs, you should record the light pattern and report it to a qualified service technician. You may also want to run the self test procedure, explained later in this section, which provides more detailed information that you can relay to a qualified service technician.

When there is a fatal error, the terminal cannot be operated except to run the self test procedure.

SELF TEST

GENERAL SELF TEST

The power up sequence is an abbreviated test of the terminal's memory and circuitry.

A more extensive test of the terminal's operating conditions is initiated by the SELF TEST button. The SELF TEST thoroughly tests the terminal and allows a service technician to make any necessary adjustments.

If the terminal detects an error in the power up sequence, you can run the self test procedure to isolate the problem. You can then report the specific problem to a technician.

The self test is initiated by pressing the SELF TEST and RESET buttons (Figure B-1) in the following sequence:

- Press SELF TEST.
- Continue to hold SELF TEST while you press and release RESET.
- Continue to hold SELF TEST until the indicator lights begin to ripple.
- Release SELF TEST.

For about the first three minutes of the self test, the light patterns continue to change as tests are run. If a module fails, the terminal bell rings twice and a light pattern remains on for 20 seconds, indicating the test that has failed. You can record that pattern to report it to a technician.

Sub Messages

In addition to this initial light-coded message, many messages have up to three "sub messages" which can provide a technician with even more information about the problem.

If you want to view the sub messages, start the following sequence within 20 seconds after the initial message appears. (If you take no action, the self test continues in 20 seconds.)

1. Press the RETURN key.
2. The terminal bell rings twice and another light pattern is displayed.
3. Record the light pattern and press RETURN again.
4. If there is another sub message, the bell rings twice and another light pattern is displayed.
5. Record the light pattern and press RETURN.
6. If there is a third sub message the bell rings twice and another light pattern is displayed.
7. Record the light pattern and press RETURN.
8. When you have seen all of the sub messages, there is no bell when you press RETURN, the light in the CAPS LOCK key turns off, and the indicator light pattern changes to indicate that the next test in the self test procedure is being run.

Error messages are displayed in this manner as the terminal checks its display. After the display has been checked, subsequent error messages are displayed on the screen instead by light patterns. If the test halts during this part of the test, it is because the screen is filled with error messages. Make a hard copy or write down the error messages. Press the PAGE key and the test will continue.

The self test procedure is explained in more detail in the 4112 Service Manual.

RESETTING SETUP MEMORY

You may find it necessary or desirable to restore the terminal's environment to the factory defaults. This is accomplished within the self test procedure. To do so, take the following steps:

1. Initiate the self test as explained earlier in this appendix.
2. Within 20 seconds of initiating the self test, press and hold the CTRL key while you press and release the C key, then release the CTRL key. This causes the light in the CAPS LOCK key to stop flashing and the following menu is displayed on the screen:

```
411X Menu
--
f1 4112 Display
f2 Processor Board
f3 Disk
f4 Tablet
--
Selection
*
```

3. Press function key 2 (F2). The following menu appears on the screen:

```
Processor Board Menu
--
f1 CMOS-Reset
f2 Keyboard
f3 Host Port
--
Selection
*
```

4. Press function key 1 (F1). The following message appears below the menu:

```
Processor Board Menu
--
f1 CMOS-Reset
f2 Keyboard
f3 Host Port
--
Selection
*f1
CMOS-Reset
Selection
*
```

   The term "CMOS RESET" means that the Setup memory has been restored to its factory default values.

5. Press and hold the CTRL key while you press and release the E key, and then release the CTRL key. The terminal exits from the self test procedure, runs the power up sequence and when the cursor appears, it is ready for use.

Appendix C

ESCAPE SEQUENCE COMMANDS

Commands which are most commonly used by a 4112 operator have an English-like format and are called setup commands, as explained in Section 6.

However the terminal can also execute many more commands, which are issued in a format called "escape sequences." They are called escape sequences because the first character in the command is always an escape. All terminal commands (including setup commands and command keys) have an escape sequence format.

Although escape sequences are primarily used by the host to send commands to the terminal, you can enter an escape sequence from the terminal as well, in either Setup or Local modes. The escape sequences are entered in a slightly different format for each of the two modes.

This appendix briefly discusses escape sequence commands and lists the format for each. See the 4112 Host Programmer's Manual and the 4110 Series Command Reference Manual for details on the use of escape sequences.

TWO FORMATS

There are two different formats for escape sequence commands.

One format consists of the escape character followed by a single character op code. Escape sequences in this format can only be entered in Local mode on a 4112. In a general form, this could be represented as follows (^{E}C represents the escape character, X represents a single ASCII character):

^{E}CX

The second (4110-family) format consists of the escape character followed by a two-character op code and sometimes one or more parameters. Escape sequences in this format can be entered in either Local or Setup modes, though they are entered slightly differently in each mode. This format could be represented as follows. (The ^{E}C represents the escape character. Each "(X)" represents a single ASCII character; the parentheses are included to distinguish between the characters. Each <parameter> may be an ASCII character, a string of ASCII characters, or a real number.)

^{E}C(X)(X)<parameter><parameter> . . .

ESCAPE SEQUENCES IN SETUP MODE

All escape sequence commands which consist of the escape character followed by a two-character op code can be executed in Setup mode. The following list details the format of escape sequences in Setup mode:

- The first character in the command is the "escape."
- All alphabetic characters must be upper case.
- There is no space between the "escape" character and the two-character op code of the command.
- If the command has parameters, enter one or more spaces after the op code and then enter the parameters with one or more spaces between each parameter.
- Enter keyword parameters in their English-like format; enter numeric values in a normal way (special encoding is not necessary as it is in Local mode).
- Terminate the command with a carriage return, which causes the command to be executed.

For example, if the dialog area is enabled but not visible, you press the DIALOG key to make the dialog area visible. There is also an escape sequence command which makes the dialog area visible.

You can execute that command in Setup mode as follows:

1. Put the terminal into Setup mode. If the light in the SET UP key is on, the terminal is already in Setup mode. If it is not on, press the SET UP key and the light turns on as the terminal enters Setup mode.
2. The terminal should not be in Local mode. The light in the LOCAL key should be off; if it is on, press the key and the light will go off as the terminal exits from Local mode.
3. The light in the CAPS LOCK key should be on, indicating that the key is active and all alphabetic characters will be upper case.
4. Enter the following command:

   $^{E}_{C}$LV YES $^{C}_{R}$

That is, press the "ESC," "L," and "V" keys with no spaces between them. Then enter a "space" and the keyword "YES" followed by the carriage return. The characters are echoed on the screen as you enter them and the command is executed when you enter the carriage return. If the dialog area was not visible before you entered the command, it becomes visible. If the dialog area was already visible, nothing happens.

The following command initializes the terminal for graphics input (GIN) mode, setting it to use the thumbwheels as the input device for ten points:

$^{E}_{C}$IE 0 10 $^{C}_{R}$

Press the SET UP key to exit from Setup mode. You can use the thumbwheels to locate the crosshair cursor and press ten alphabetic keys to exit from GIN mode. Or, you can put the terminal back into Setup mode and enter the following escape sequence command:

$^{E}_{C}$ID 0 $^{C}_{R}$

ESCAPE SEQUENCES IN LOCAL MODE

When the terminal is in Local mode, you can execute both 4010 and 4110-family escape codes. In both cases, escape sequences have the following characteristics. Notice that the first two characteristics are the same as using Setup mode, but the rest are different:

- The first character in the command is the "escape."
- All alphabetic characters must be upper case.
- No carriage return is necessary; the command is executed as soon as you enter the last character.
- Characters are not echoed on the screen as you type them.
- Parameters of a command are not delimited by a space (although a space is a valid character in some commands).
- Keywords and numeric values greater than nine are entered in an encoded form (see the 4110 Series Command Reference Manual).

The escape sequence to make the dialog area visible can be entered in Local mode as follows:

1. Put the terminal into Local mode. If the light in the LOCAL key is on, the terminal is in Local mode. If it is not on, press the LOCAL key and the light turns on as the terminal enters Local mode.
2. The terminal should not be in Setup mode. The light in the SET UP key should be off; if it is on, press the key and the light will go off as the terminal exits from Setup mode.
3. The light in the CAPS LOCK key should be on, indicating that the key is active and all alphabetic characters will be upper case.
4. Press the following keys; the characters will not be echoed on the screen.

   ^{E}cLV1

The command is executed when you enter the last character in the command. If the dialog area was not visible before you entered the command, it becomes visible. If the dialog area was already visible, nothing happens.

The following command initializes the terminal for graphics input (GIN) mode, setting it to use the thumbwheels as the input device for ten points (the colon at the end of the command is the coded form of the integer 10 in Local mode escape sequence commands):

^{E}cIE0:

Press the LOCAL key to exit from Local mode. You can use the thumbwheels to locate the crosshair cursor and press ten alphabetic keys to exit from GIN mode. Or, you can put the terminal back into Local mode and enter the following escape sequence command:

^{E}cID0

LIST OF ESCAPE SEQUENCES

The rest of this appendix lists the terminal's 4110-style escape sequences. They are listed alphabetically according to their two-character op code. The setup command is shown if the escape sequence has one.

Parameters for escape sequence commands are explained in the 4110 Series Command Reference Manual and the 4112 Programmer's Manual.

Escape Sequence	Command	Setup Command
$^{E}C(I)(!)<$ASCII-CHAR$>$	Disable-4953-Tablet-GIN	None
$^{E}C(I)(A)<$int$>$	Set-Pick-Aperture	PICKAPERTURE
$^{E}C(I)(C)<$int$><$int$>$	Set-GIN-Cursor	None
$^{E}C(I)(E)<$int$><$int+$>$	Enable-GIN	None
$^{E}C(I)(F)<$int$><$int$><$int$>$	Set-GIN-Stroke-Filtering	TBFILTER
$^{E}C(I)(G)<$int$><$int$><$int$>$	Set-GIN-Gridding	GRIDDING
$^{E}C(I)(H)<$int$>$	Set-Tablet-Header-Characters	TBHEADERCHARS
$^{E}C(I)(I)<$int$><$int$>$	Set-GIN-Inking	None
$^{E}C(I)(L)<$int$>$	Set-Report-Max-Line-Length	None
$^{E}C(I)(M)<$int$>$	Set-Report-EOM-Frequency	REOM
$^{E}C(I)(N)<$int-array$>$	Set-Tablet-Area	TBSIZE
$^{E}C(I)(P)<$int$>$	Report-GIN-Point	None
$^{E}C(I)(Q)<$char$><$char$>$	Report-Terminal-Settings	None
$^{E}C(I)(R)<$int$><$int$>$	Set-GIN-Rubberbanding	None
$^{E}C(I)(S)<$int$><$char$><$char$>$	Set-Report-Sig-Chars	None
$^{E}C(I)(T)<$int$>$	Set-Tablet-Status-Strap	TBSTATUS

Escape Sequence	Command	Setup Command
^{E}c(J)(C)<string> <string><string>	Copy	COPY
^{E}c(J)(D)<string> <string><string>	Directory	DIRECTORY
^{E}c(J)(E)	Stop-Spooling	STOP
^{E}c(J)(F)<string><int>	Format-Volume	FORMAT
^{E}c(J)(K)<string>	Delete-File	DELETE
^{E}c(J)(L)<string>	Load	LOAD
^{E}c(J)(P)<string><int>	Protect-File	PROTECT
^{E}c(J)(Q)<string>	Report-Device-Status	None
^{E}c(J)(R)<string> <string><string>	Rename-File	RENAME
^{E}c(J)(S)<string> <string><string>	Spool	SPOOL
^{E}c(J)(V)<string> <int><string><string>	Save	SAVE
^{E}c(K)(A)<int>	Enable-Dialog-Area	DAENABLE
^{E}c(K)(C)	Cancel	None
^{E}c(K)(D)<int><int-array>	Define-Macro	DEFINE
^{E}c(K)(E)<int>	Echo	ECHO
^{E}c(K)(F)<int>	LFCR	LFCR
^{E}c(K)(H)<int>	Hardcopy	None
^{E}c(K)(I)<int>	Ignore-Deletes	IGNOREDEL
^{E}c(K)(L)<int>	Lock-Keyboard	LOCKKEYBOARD
^{E}c(K)(M)<int>	Set-Margins	MARGINS

Escape Sequence	Command	Setup Command
$^{E}_{C}$(K)(N)<int>	Renew-View	RENEW
$^{E}_{C}$(K)(P)<int>	Set-Page-Full-Action	PAGEFULL
$^{E}_{C}$(K)(Q)	Report-Errors	None
$^{E}_{C}$(K)(R)<int>	CRLF	CRLF
$^{E}_{C}$(K)(S)<int>	Snoopy	SNOOPY
$^{E}_{C}$(K)(T)<int>	Set-Error-Threshold	ERRORLEVEL
$^{E}_{C}$(K)(X)<int>	Expand-Macro	None
$^{E}_{C}$(K)(Y)<int>	Set-Execute-Toggle-Char	KEYEXCHAR
$^{E}_{C}$(K)(Z)<int><int><int>	Set-Edit-Chars	EDITCHARS
$^{E}_{C}$(L)(B)<int+>	Set-Dialog-Area-Buffer-Size	DABUFFER
$^{E}_{C}$(L)(C)<int>	Set-Dialog-Area-Chars	DACHARS
$^{E}_{C}$(L)(E)	End-Panel	None
$^{E}_{C}$(L)(F)<xy>	Move	None
$^{E}_{C}$(L)(G)<xy>	Draw	None
$^{E}_{C}$(L)(H)<xy>	Draw-Marker	None
$^{E}_{C}$(L)(I)<int+><int+><int+>	Set-Dialog-Area-Index	DAINDEX
$^{E}_{C}$(L)(K)<int>	Include-Copy-of-Segment	None
$^{E}_{C}$(L)(L)<int>	Set-Dialog-Area-Lines	DALINES
$^{E}_{C}$(L)(M)<int>	Set-Dialog-Area-Writing-Mode	DAMODE
$^{E}_{C}$(L)(P)<xy><int>	Begin-Panel-Boundary	None
$^{E}_{C}$(L)(S)<int>	Set-Dialog-Area-Surface	DASURFACE
$^{E}_{C}$(L)(T)<string>	Graphic-Text	None
$^{E}_{C}$(L)(V)<int>	Set-Dialog-Area-Visibility	DAVIS
$^{E}_{C}$(L)(X)<xy>	Set-Dialog-Area-Position	DAPOSITION
$^{E}_{C}$(L)(Z)	Clear-Dialog-Scroll	None

Escape Sequence	Command	Setup Command
$^{E}_{C}$(M)(C)<int><int><int>	Set-Graphtext-Size	None
$^{E}_{C}$(M)(D)<int><int><int><int>	Begin-Fill-Pattern	None
$^{E}_{C}$(M)(E)	End-Fill-Pattern	None
$^{E}_{C}$(M)(F)<int>	Set-Graphtext-Font	None
$^{E}_{C}$(M)(G)<int>	Set-Graphics-Area-Writing-Mode	GAMODE
$^{E}_{C}$(M)(I)<int>	Set-Pick-ID	None
$^{E}_{C}$(M)(L)<int+>	Set-Line-Index	None
$^{E}_{C}$(M)(M)<int>	Set-Marker-Type	None
$^{E}_{C}$(M)(P)<int>	Select-Fill-Pattern	None
$^{E}_{C}$(M)(Q)<int>	Set-Graphtext-Precision	None
$^{E}_{C}$(M)(R)<real>	Set-Graphtext-Rotation	None
$^{E}_{C}$(M)(S)<int><int><int>	Set-Panel-Filling-Mode	None
$^{E}_{C}$(M)(T)<int+>	Set-Text-Index	None
$^{E}_{C}$(M)(V)<int>	Set-Line-Style	None
$^{E}_{C}$(M)(W)<int>	Set-Line-Width	None
$^{E}_{C}$(M)(Z)<int><int><int>	Set-Alphatext-Size	ALPHASIZE
$^{E}_{C}$(N)(B)<int>	Set-Stop-Bits	STOPBITS
$^{E}_{C}$(N)(C)<int><int>	Set-EOM-Chars	EOMCHARS
$^{E}_{C}$(N)(D)<int>	Set-Transmit-Delay	XMTDELAY
$^{E}_{C}$(N)(E)<int-array>	Set-EOF-String	EOFSTRING
$^{E}_{C}$(N)(F)<int>	Set-Flagging-Mode	FLAGGING
$^{E}_{C}$(N)(K)<int+>	Set-Break-Time	BREAKTIME
$^{E}_{C}$(N)(L)<int+>	Set-Transmit-Rate-Limit	XMTLIMIT
$^{E}_{C}$(N)(M)<int>	Prompt-Mode	PROMPTMODE
$^{E}_{C}$(N)(P)<int>	Set-Parity	PARITY

Escape Sequence	Command	Setup Command
E_C(N)(Q)<int+ >	Set-Queue-Size	QUEUESIZE
E_C(N)(R)<int+ ><int+ >	Set-Baud-Rates	BAUDRATE
E_C(N)(S)<int-array>	Set-Prompt-String	PROMPTSTRING
E_C(N)(T)<int-array>	Set-EOL-String	EOLSTRING
E_C(N)(U)<int>	Set-Bypass-Cancel-Char	BYPASSCANCEL
E_C(O)(B)<int>	Arm-For-Block-Mode	BLOCKMODE
E_C(O)(C)<int><int>	Set-Block-Continue-Chars	BCONTINUECHARS
E_C(O)(D)<int>	Set-Duplex-Mode	DUPLEX
E_C(O)(E)<int><int>	Set-Block-End-Chars	BENDCHARS
E_C(O)(H)<int-array><int-array>	Set-Block-Headers	BHEADERS
E_C(O)(L)<int>	Set-Block-Line-Length	BLINELENGTH
E_C(O)(M)<int><int>	Set-Block-Master-Chars	BMASTERCHARS
E_C(O)(N)<int-array><int-array>	Set-Block-Non-Xmt-Chars	BNONXMTCHARS
E_C(O)(P)<int><int><int><int>	Set-Block-Packing	BPACKING
E_C(O)(S)<int+ ><int+ >	Set-Block-Length	BLENGTH
E_C(O)(T)<int+ >	Set-Block-Timeout	BTIMEOUT
E_C(P)(A)<string><string><int>	Port-Assign	PASSIGN
E_C(P)(B)<string><int><int>	Set-Port-Stop-Bits	PBITS
E_C(P)(C)<string>	Port-Copy	PCOPY
E_C(P)(E)<string><int-array>	Set-Port-EOF-String	PEOF
E_C(P)(F)<string><int><int><int>	Set-Port-Flagging-Mode	PFLAG
F_C(P)(I)<string><int><int>	Map-Index-To-Pen	MAPINDEX
E_C(P)(L)<string><string>	Plot	PLOT
E_C(P)(M)<string><int-array>	Set-Port-EOL-String	PEOL
E_C(P)(P)<string><int>	Set-Port-Parity	PPARITY

Escape Sequence	Command	Setup Command
$^{E}_{C}$(P)(Q)<string>	Report-Port-Status	PORTSTATUS
$^{E}_{C}$(P)(R)<string><int>	Set-Port-Baud-Rate	PBAUD
$^{E}_{C}$(R)(A)<int><int+><int+>	Set-View-Attributes	None
$^{E}_{C}$(R)(B)<int>	Set-Background-Gray-Level	None
$^{E}_{C}$(R)(C)<int>	Select-View	None
$^{E}_{C}$(R)(D)<int-array>	Set-Surface-Definitions	None
$^{E}_{C}$(R)(E)<int>	Set-Border-Visibility	None
$^{E}_{C}$(R)(F)<int>	Set-Fixup-Level	FIXUP
$^{E}_{C}$(R)(G)<int><int±array>	Set-Surface-Gray-Levels	None
$^{E}_{C}$(R)(H)<xy>	Set-Pixel-Beam-Position	None
$^{E}_{C}$(R)(I)<int-array>	Set-Surface-Visibility	None
$^{E}_{C}$(R)(J)<int>	Lock-Viewing-Keys	None
$^{E}_{C}$(R)(K)<int>	Delete-View	None
$^{E}_{C}$(R)(L)<int±array>	Runlength-Write	None
$^{E}_{C}$(R)(N)<int-array>	Set-Surface-Priorities	None
$^{E}_{C}$(R)W)<int><char-array>	Raster-Write	None
$^{E}_{C}$(R)(R)<xy><xy><int+>	Rectangle-Fill	None
$^{E}_{C}$(R)(S)<xy><xy>	Set-Pixel-Viewport	None
$^{E}_{C}$(R)(U)<int><int><int>	Begin-Pixel-Operations	None
$^{E}_{C}$(R)(V)<xy><xy>	Set-Viewport	None
$^{E}_{C}$(R)(W)<xy><xy>	Set-Window	None
$^{E}_{C}$(R)(X)<int><xy><xy><xy>	Pixel-Copy	None
$^{E}_{C}$(S)(A)<int><int-array><int-array>	Set-Segment	None
$^{E}_{C}$(S)(C)	End-Segment	None
$^{E}_{C}$(S)(D)<int><int>	Seg-Segment-Detectability	None

Escape Sequence	Command	Setup Command
$^{E}_{C}$(S)(G)<int><int><int>	Set-Graphtext-Font-Grid	None
$^{E}_{C}$(S)(H)<int><int>	Set-Segment-Highlighting	None
$^{E}_{C}$(S)(I)<int><real> <real><real><real><xy>	Set-Segment-Image-Transform	None
$^{E}_{C}$(S)(K)<int>	Delete-Segment	None
$^{E}_{C}$(S)(L)<int-array><int-array><int-array>	Set-Current-Matching-Class	None
$^{E}_{C}$(S)(M)<int><int>	Set-Segment-Writing Mode	None
$^{E}_{C}$(S)(O)<int>	Begin-Segment	None
$^{E}_{C}$(S)(P)<xy>	Set-Pivot-Point	None
$^{E}_{C}$(S)(Q)<int><char-array>	Report-Segment-Status	None
$^{E}_{C}$(S)(R)<int><int>	Rename-Segment	None
$^{E}_{C}$(S)(S)<int><int>	Set-Segment-Display-Priority	None
$^{E}_{C}$(S)(T)<int><int>	Begin-Graphtext-Character	None
$^{E}_{C}$(S)(U)	End-Graphtext-Character	None
$^{E}_{C}$(S)(V)<int><int>	Set-Segment-Visibility	None
$^{E}_{C}$(S)(X)<int><xy>	Set-Segment-Position	None
$^{E}_{C}$(S)(Z)<int><int>	Delete-Graphtext-Character	None

Appendix D

ASCII CHARTS

This appendix includes a standard ASCII code chart and additional ASCII code charts which define the specific characters used as parameters (indicated by unshaded areas).

The code charts are:

Table	Description
D-1	ASCII Code Chart
D-2	Characters Used in <Char> Parameters
D-3	Characters Used in <Int> and <Int+> Parameters
D-4	Characters Used in <Int-Report> Parameters
D-5	Characters Used in <Xy> Parameters
D-6	Characters Used in <Xy-Report> Parameters

Table D-1

ASCII (ISO-7-US) CODE CHART

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	

Table D-2

CHARACTERS USED IN <CHAR> PARAMETERS

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

* | on some keyboards or systems

Table D-3

CHARACTERS USED IN <INT> AND <INT+> PARAMETERS

<HII> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

* | or some keyboards or systems

<LoI> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

Table D-4

CHARACTERS USED IN <INT-REPORT> PARAMETERS

<Hil-Report> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

<Lol-Report> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

Table D-5

CHARACTERS USED IN <XY> PARAMETERS

<HiY>,<HiX> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

<LoY>,<Extra> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

<LoX> Characters

B5 / BITS B4 B3 B2 B1	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	FIGURES			UPPERCASE		LOWERCASE	
0 0 0 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	CAN 24	(40	8 56	H 72	X 88	h 104	x 120
1 0 0 1	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 0 1 0	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	ESC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 0 0	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 0 1	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1 1 1 0	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1 1 1 1	US 31	/ 47	? 63	O 79	_ 95	o 111	RUBOUT (DEL) 127

Table D-6

CHARACTERS USED IN <XY-REPORT> PARAMETERS

<HiY-Report>, <Extra-Report>, <LoY-Report>,
<HiX-Report>, and <LoX-Report> Characters

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	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
0 0 0 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
0 0 1 0	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
0 0 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
0 1 0 0	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
0 1 0 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
0 1 1 0	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
0 1 1 1	BEL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 0 0 0	BS 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 0 1 1	VT 11	ESC 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	RUBOUT (DEL) 127

* | on some keyboards or systems

Appendix E

OPTIONAL KEYBOARDS

The following optional keyboards are available for the terminal. For information on how to install these optional keyboards, see the 4112 or 4114 Service Manual.

OPTION 4A: UNITED KINGDOM CHARACTER SET

This option permits the terminal to change to a United Kingdom keyboard layout so that the United Kingdom characters are displayed. The only change is that the "#" sign is replaced by the English "£" sign. When this key is pressed (or the appropriate code received by the terminal), the "£" sign is displayed on the screen. The revised keyboard configuration is shown in Figure E-1 and the revised ASCII code chart is shown in Table E-1.

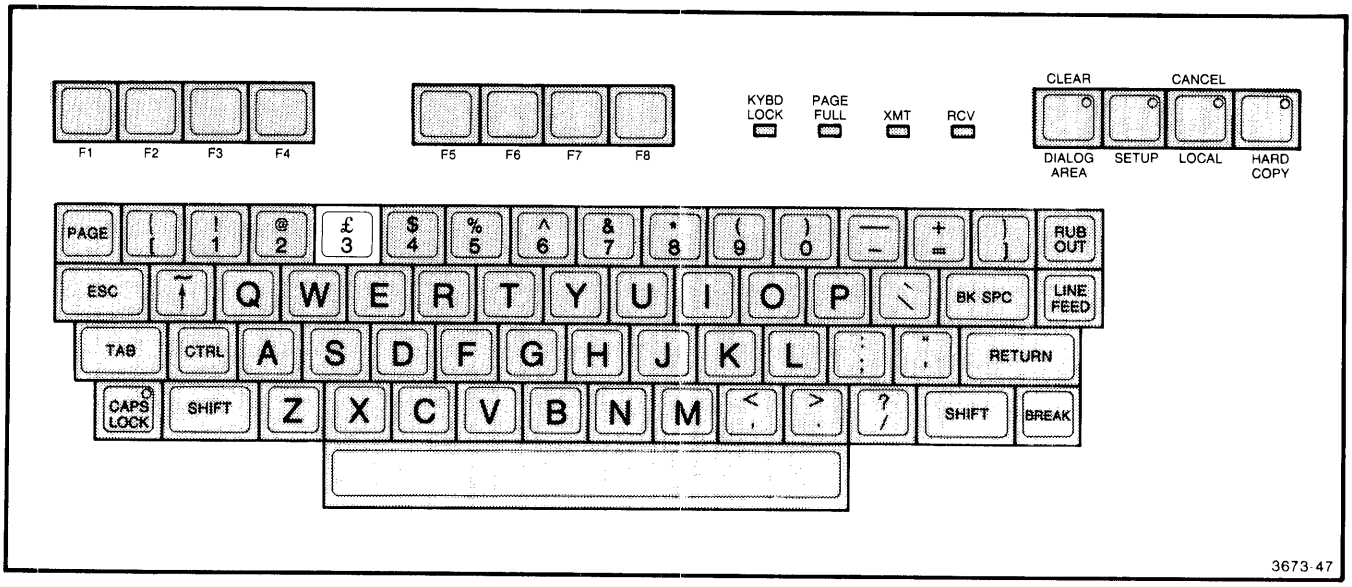


Figure E-1. United Kingdom Keyboard.

Table E-1

UNITED KINGDOM 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		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
Ø Ø Ø Ø	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	ETX 3	DC3 19	£ 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BEL BELL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS BACK-SPACE 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	ESC 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 RETURN 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	RUBOUT (DEL) 127

3673-51

OPTION 4C: SWEDISH CHARACTER SET

This option changes the standard keyboard configuration to a Swedish layout and allows Swedish characters to be displayed. There are 17 changes to the keyboard, with three of these changes being new alphabet characters. When these seventeen keys are pressed (or the appropriate codes are received by the terminal), the corresponding characters are displayed on the screen. The revised keyboard configuration is shown in Figure E-2 and the revised ASCII code chart is shown in Table E-2.

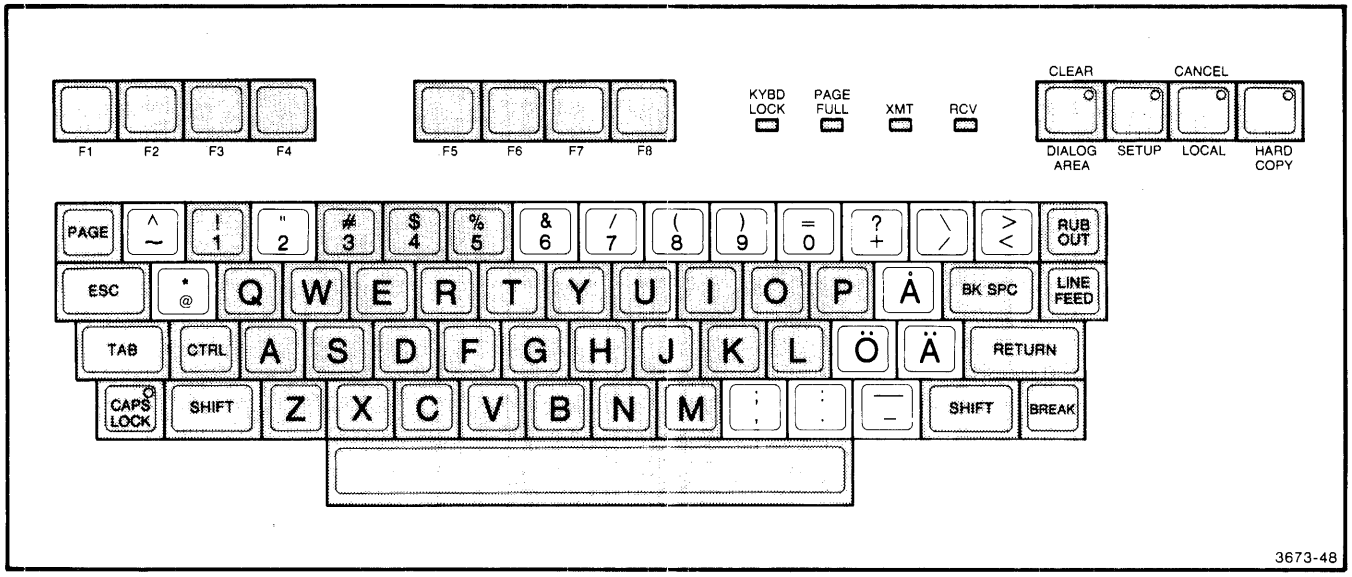


Figure E-2. Swedish Keyboard.

Table E-2
SWEDISH CHARACTER SET

BITS B4	B3	B2	B1	CONTROL		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
				B7 B6 B5 = Ø Ø Ø	B7 B6 B5 = Ø Ø 1	B7 B6 B5 = Ø 1 Ø	B7 B6 B5 = Ø 1 1	B7 B6 B5 = 1 Ø Ø	B7 B6 B5 = 1 Ø 1	B7 B6 B5 = 1 1 Ø	B7 B6 B5 = 1 1 1
Ø	Ø	Ø	Ø	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø	Ø	Ø	1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BEL BELL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS BACK-SPACE 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	ESC 27	+ 43	; 59	K 75	Ä 91	k 107	ä 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	Ö 92	l 108	ö 124
1	1	Ø	1	CR RETURN 13	GS 29	- 45	= 61	M 77	Å 93	m 109	å 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	RUBOUT (DEL) 127

3673-52

OPTION 4E: ASCII/APL CHARACTER SET

This option enables selection of either ASCII or APL characters to be entered and displayed. There are 47 different keys in this character set, with some of the symbols printed on the front of the keys as well as the tops of the keys (see Figure E-3). Tables E-3 and E-4 show the ASCII and APL coding.

When the terminal is turned on (and during normal operation), the keyboard will operate in ASCII mode. That is, the same as a standard keyboard. To obtain the APL (A Programming Language) character set, the ESC and CTRL N keys must be pressed. Once this is done, the terminal operates in APL mode. The APL codes are generated when a key is pressed and APL characters are displayed on the screen as a result of pressing keys or receiving data from the host computer.

To exit APL mode (back to ASCII), the ESC and CTRL O keys must be pressed.

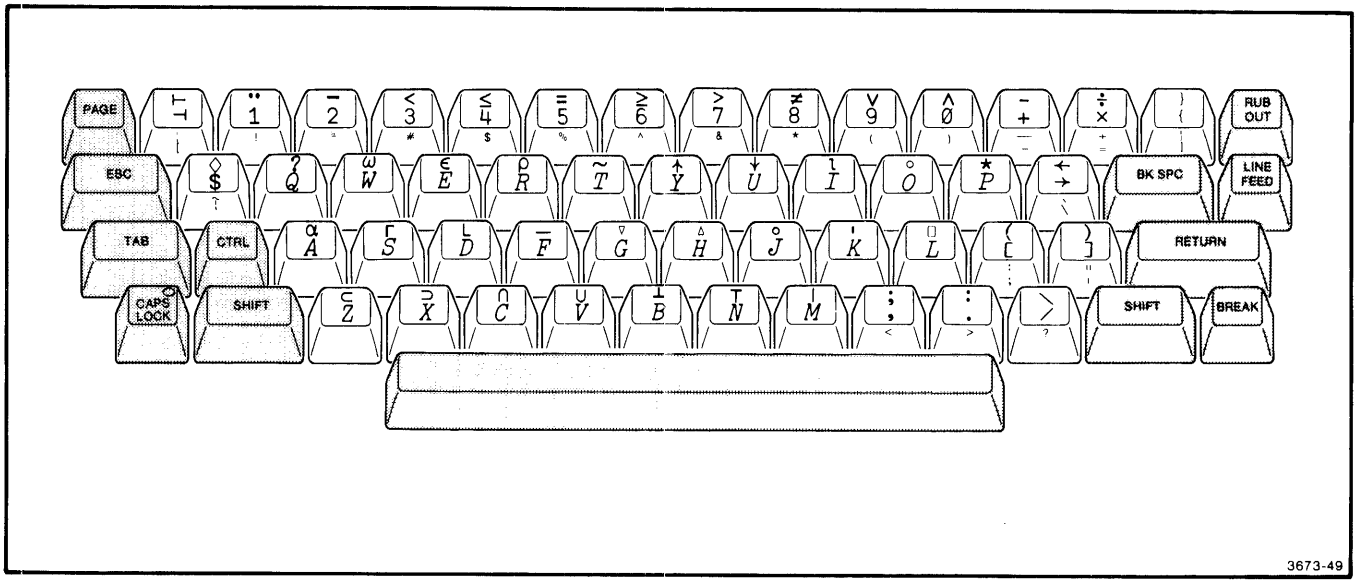


Figure E-3. APL Keyboard Option.

Table E-3

ASCII CODE CHART

BITS B7 B6 B5 / 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		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
Ø Ø Ø Ø	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BEL BELL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS BACK-SPACE 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	ESC 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 RETURN 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	RUBOUT (DEL) 127

3673-53

Table E-4
APL 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		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
Ø Ø Ø Ø	NUL 0	DLE 16	SP 32	0 48	− 64	* 80	◇ 96	P 112
Ø Ø Ø 1	SOH 1	DC1 17	•• 33	1 49	α 65	? 81	A 97	Q 113
Ø Ø 1 Ø	STX 2	DC2 18	) 34	2 50	⊥ 66	ρ 82	B 98	R 114
Ø Ø 1 1	ETX 3	DC3 19	< 35	3 51	∩ 67	⌈ 83	C 99	S 115
Ø 1 Ø Ø	EOT 4	DC4 20	≤ 36	4 52	⌊ 68	~ 84	D 100	T 116
Ø 1 Ø 1	ENQ 5	NAK 21	= 37	5 53	∈ 69	↓ 85	E 101	U 117
Ø 1 1 Ø	ACK 6	SYN 22	> 38	6 54	— 70	∪ 86	F 102	V 118
Ø 1 1 1	BEL BELL 7	ETB 23	] 39	7 55	∇ 71	ω 87	G 103	W 119
1 Ø Ø Ø	BS BACK-SPACE 8	CAN 24	∨ 40	8 56	Δ 72	⊃ 88	H 104	X 120
1 Ø Ø 1	HT 9	EM 25	∧ 41	9 57	ι 73	↑ 89	I 105	Y 121
1 Ø 1 Ø	LF 10	SUB 26	≠ 42	(58	∘ 74	⊂ 90	J 106	Z 122
1 Ø 1 1	VT 11	ESC 27	÷ 43	[59	' 75	← 91	K 107	{ 123
1 1 Ø Ø	FF 12	FS 28	, 44	; 60	□ 76	⊢ 92	L 108	⊣ 124
1 1 Ø 1	CR RETURN 13	GS 29	+ 45	× 61	| 77	→ 93	M 109	} 125
1 1 1 Ø	SO 14	RS 30	. 46	: 62	⊤ 78	≥ 94	N 110	$ 126
1 1 1 1	SI 15	US 31	/ 47	\ 63	○ 79	- 95	O 111	RUBOUT (DEL) 127

3673-54

OPTION 4F: DANISH/NORWEGIAN KEYBOARD

This option changes to a Danish/Norwegian standard keyboard layout so that the Danish/Norwegian characters are displayed. There are 17 changes to the keyboard, with three of the changes being new alphabetic characters. When these 17 keys are pressed (or the appropiate codes are received by the terminal), the corresponding characters are displayed on the screen. The revised keyboard configuration is shown in Figure E-4 and the revised ASCII code chart is shown in Table E-5.

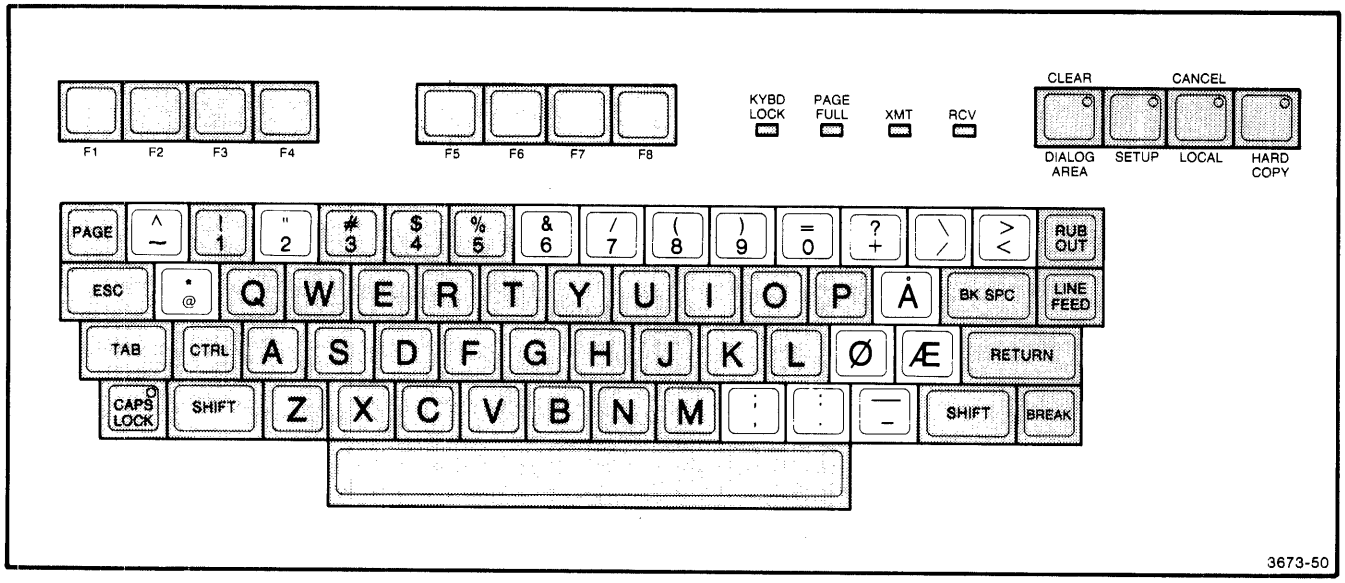


Figure E-4. Danish/Norwegian Keyboard.

Table E-5
DANISH/NORWEGIAN CHARACTER SET

BITS B4 B3 B2 B1	B7 B6 B5 = 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		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
Ø Ø Ø Ø	NUL 0	DLE 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SOH 1	DC1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	STX 2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	ETX 3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	EOT 4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	ENQ 5	NAK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	ACK 6	SYN 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BEL BELL 7	ETB 23	' 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS BACK-SPACE 8	CAN 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	SUB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	ESC 27	+ 43	; 59	K 75	Ä 91	k 107	ä 123
1 1 Ø Ø	FF 12	FS 28	, 44	< 60	L 76	Ø 92	l 108	ø 124
1 1 Ø 1	CR RETURN 13	GS 29	- 45	= 61	M 77	Å 93	m 109	å 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	RUBOUT (DEL) 127

3673-55