TEK COMMAND REFERENCE MANUAL

Part No. 070-5141-01
Product Group 16

4110/4120 SERIES
COMMAND REFERENCE WITH 3D

Please Check for
CHANGE INFORMATION
at the Rear of This Manual

First Printing FEB 1985
Revised SEP 1985

Tektronix®
COMMITTED TO EXCELLENCE

MANUAL REVISION STATUS

PRODUCT: 4111, 4115, M4115, 4125, 4128, 4129 Computer Display Terminals; 4115 P51

This manual supports the following versions of these products:
4111 — Firmware Version 1 and up.
4115, 4120 Series — Firmware Version V6 and up.
(Current versions: 4115, V7; 4120 Series, V21)

REV DATE	DESCRIPTION
FEB 1985	Original Issue
SEP 1985	Revised to include support for 4111 and new firmware; Section 9 added. Part number rolled to -01.

CONTENTS

Section 7 (cont) **Page**

Section 7 (cont) Page

Section 7 (cont) **Page**

Section 7 (cont) Page

Section 8 ANSI X3.64 COMMANDS

Appendix D MACRO NUMBER REFERENCE TABLES

Appendix E COLOR COORDINATES

Appendix F DISPLAY LIST OP CODES

Appendix G SHADED SURFACES

ILLUSTRATIONS

TABLES

Section 1

INTRODUCTION

The 4110/4120 Series Command Reference Manual With 3D is intended for the programmer who is writing host computer subroutines. This manual is primarily a reference manual and will be used by the programmer more than any other manual once the programmer becomes familiar with the terminal.

The *4110/4120 Series Host Programmers Manual* describes the terminal programming considerations and details which a programmer must be aware of when preparing the host computer subroutine package to communicate.

4110/4120 Series terminals operate in two major modes: TEK mode and ANSI mode. Since the syntax of the two modes are exclusive and overlapping, this manual separates the discussions of the commands sets of these two code sets. The 4111 has a third mode, VT52 mode, which also is described in a separate section.

MANUAL ORGANIZATION

This manual is divided into nine sections.

- Section 1 (this section) provides an introduction and overview, and lists the conventions used in the manual.
- Section 2 describes the syntax and general format of 4110/4120 Series commands of both code sets.
- Sections 3 through 6 are indexing sections, designed to make finding a specific command easier.
  - Section 3 contains a functional listing of all commands.
  - Section 4 lists the contents of Sections 7 and 8 that are not commands by type (i.e., parameter types, message types, etc.).
  - Section 5 indexes the commands by escape-sequence op codes.
  - Section 6 is a keyword-in-context index. That is, it contains an entry for each significant word in a command name or syntactic construct name.
- Section 7 is an alphabetical listing of all TEK mode commands. This section also includes (alphabetically) a description of such things as control characters, keys, message formats, parameters, etc. Note that in the alphabetic listings, numbers (e.g. 4010) follow the other listings.
- Section 8 is an alphabetical listing of the ANSI mode commands, and of the control characters that act differently in ANSI mode than in TEK mode.
- Section 9 is an alphabetical listing of the VT52 mode commands, which apply only to the 4111.

The remainder of the manual consists of various appendices:

- Appendix A — ASCII Code Charts — contains a standard ASCII Code chart and additional charts which define the ASCII characters used for various types of parameters.
- Appendix B — *Int* Parameters — contains tables of packed *int* parameters and an algorithm on how to pack an *int* using the tables.
- Appendix C — Error Codes — contains a complete listing of error codes. The error codes are listed alphabetically by command op codes.
- Appendix D — Macro Number Reference Tables — contains a listing of the macro numbers assigned the various keyboard keys.
- Appendix E — Color Coordinates and Color Map — describes the HLS, RBG, CMY, and Machine RGB systems of color coordinates. It also lists the RGB mode of the 256 colors in the default color map.
- Appendix F — Display List Op-Codes — lists the instructions executed by the picture processor.
- Appendix G — Shaded Surfaces — describes how shading is accomplished in the 4129.

MANUAL CONVENTIONS

In Sections 7 and 8, each command starts at the top of a page column. If the command is unique to a particular 4110/4120 Series Terminal, the terminal number appears beneath the command name. If the command is option-dependent, the option number also appears in this location. Other manual conventions are discussed in Section 2: the section on command syntax.

RELATED DOCUMENTATION

Other manuals which are available for the 4110/4120 Series terminals include:

- *4110/4120 Series Host Programmers Manual*
- Operators Manuals
- Service Manuals
- Manuals relating to specific options

Additional information on PLOT 10 Software which supports the 4110/4120 Series terminals is also available.

Contact your local Tektronix office for a complete list of manuals which are available for your particular terminal.

Section 2

COMMAND SYNTAX

SYNTAX NOTATION

All commands to the terminal are sent as a sequence of individual ASCII characters. To describe each operation in terms of individual characters would be tedious and confusing. Therefore, a number of notational conventions are used throughout this manual.

Syntax Rules

In this manual, command syntax is represented using the syntax and symbols summarized in Table 2-1. The following paragraphs describe each symbol in detail.

Table 2-1

COMMAND SYNTAX SYMBOLS

Meaning	**Symbol or Representation**
Literal characters (enter these characters literally)	Boldface type
Expressions (these words and phrases have special meaning)	Hyphenated
Specific examples	:
Has the following syntax	=
Alternate items (choose one)	{ }
Optional items (omit or choose)	[ ]
Values which can be repeated n times	. . .

Individual ASCII characters sent to or from the terminal are represented in bold face. Snoopy mode mnemonics are used to represent ASCII control characters.

For example:

A	The ASCII uppercase "A" character
a	The lowercase "a" character
$^{\mathrm{E}}{}_{\mathrm{C}}$	"Escape" (ESC) control character
$^{\mathrm{D}}{}_{\mathrm{T}}$	"Delete" (DEL) character (also called "rub out")
$^{\mathrm{U}}{}_{\mathrm{S}}$	"Unit separator" control character
$^{\mathrm{S}}{}_{\mathrm{O}}$	"Shift out" control character

An *expression* is a word or phrase that has special meaning in this manual. Expressions include command names, message types, special characters, parameter types, syntactic constructs, and parameter names.

Expressions in text are italicized, and if the expression is longer than one word, the words are joined by hyphens. This is true for all parameters and syntactic constructs except command names, which are fully capitalized and hyphenated whenever they appear in text, and except for minor variations within command description boxes to help you differentiate between parameter types and parameter names.

For example:

- Whenever the *int* parameter type is referred to in text, it is italicized, except when it specifies a parameter type in a command description.
- If a paragraph mentions the Panel-Definition syntactic construct, or any other multiword construct that is not a command name, the reference is italicized and hyphenated.
- If the BEGIN-FILL-PATTERN or COPY command is mentioned, the command name is capitalized and hyphenated if it is multiword.
- If the *fill-pattern-number* parameter of the BEGIN-FILL-PATTERN command is referred to, it is italicized and hyphenated.

The colon (:) expresses the meaning of parameters, especially in the more complicated syntax definitions. For instance, the SET-BAUD-RATES command syntax is:

${}^{E}C$**NR** int:*transmit-rate* int:*receive-rate*

The expressions *transmit-rate* and *receive-rate* tell the meanings of the *int* parameters.

When this manual represents the syntax of a syntactic construct, the equal sign (=) means "has the following syntax." For example:

char-array = *int* [*char* . . .]

In syntax definitions, a set of curly braces ({ }) around a stacked list of parameters or construct elements means "choose one." For example, the SET-ALPHATEXT-FONT command syntax is:

$${}^{E}C \begin{Bmatrix} {}^{S}I \\ {}^{S}O \end{Bmatrix}$$

To use this command choose either ${}^{S}I$ or ${}^{S}O$ as the second character.

In syntax definitions, parameters or construct elements that are enclosed in square brackets ([ ]) are optional. None, some, or all items enclosed may be included in the ASCII string sent to the terminal. For example:

int = [*HiI* [*HiI* [*HiI* [*HiI* [*HiI*]]]]] *LoI*

You could express this in words as, "An *int* consists of zero to five *HiIs*, followed by a *LoI*."

Syntactic constructs that may be repeated any number of times are followed by three dots. For example:

string = *int* [*char* . . .

This could be expressed in words as, "A *string* consists of an *int*, followed by zero or more *char*s."

TEK COMMAND SYNTAX

All commands to the terminal are sent from the host computer as a sequence of ASCII characters. A few of these commands consist of a single character; some consist of two characters; most consist of three or more characters.

One-Character Commands

The following commands consist of only one ASCII character:

ENTER-ALPHA-MODE = $^{U}_{S}$

ENTER-VECTOR-MODE = $^{G}_{S}$

ENTER-MARKER-MODE = $^{F}_{S}$

The ASCII control characters $^{C}_{R}$, $^{L}_{F}$, $^{B}_{S}$, $^{H}_{T}$, and $^{V}_{T}$ are also one-character commands. In Alpha mode, each printable character that the terminal receives is a command to print that character.

Two-Character Commands

Most commands consist of escape sequences — sequences of ASCII characters beginning with the $^{E}_{C}$ character. A few of these commands consist of only two characters:

ENABLE-4010-GIN = $^{E}_{C}{}^{S}_{B}$

ENTER-ALPHA-MODE = $^{E}_{C}{}^{U}_{S}$

ENTER-BYPASS-MODE = $^{E}_{C}{}^{C}_{N}$

ENTER-VECTOR-MODE = $^{E}_{C}{}^{G}_{S}$

ENTER-MARKER-MODE = $^{E}_{C}{}^{F}_{S}$

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

REPORT-4010-STATUS = $^{E}_{C}{}^{E}_{Q}$

SET-ALPHATEXT-FONT = $^{E}_{C}{}^{S}_{I}$ or $^{E}_{C}{}^{S}_{O}$

SET-4014-LINE-STYLE = $^{E}_{C}$ char:*line-style*

4010-HARD-COPY = $^{E}_{C}{}^{E}_{B}$

Commands of Three or More Characters

Most of the terminal commands are escape sequences of three or more characters. These commands take the following format:

1. The first character is $^{E}_{C}$. This serves as a flag to tell the terminal that the following characters make up a command.
2. The next two characters to identify the command.
3. Parameters, if any, follow. The parameter types are described in Section 7.
4. Finally, terminate the command. You can do this in two ways: send all the command's parameters, or send one of the command terminator characters. Command terminator characters are:
   - $^{E}_{C}$. The $^{E}_{C}$ character that begins a new command also terminates the previous command.
   - $^{U}_{S}$, $^{G}_{S}$, and $^{F}_{S}$. Any command in progress ends when the terminal receives a $^{U}_{S}$, $^{G}_{S}$, or $^{F}_{S}$ character. These characters are the single-character commands that put the terminal in Alpha mode, Vector mode, and Marker mode, respectively.

When the terminal receives the parameter for a command, it ignores any control characters except the command terminators ($^{E}_{C}$, $^{U}_{S}$, $^{G}_{S}$, and $^{F}_{S}$). Thus you can insert $^{C}_{R}$ characters or other interline characters within the command's parameters with no ill effect. (This is useful if the parameter is a very long *string* or *int-array*.)

Defaults for Missing Parameters

When you terminate a command early, the terminal assigns default values to the missing parameters. The standard defaults are:

- 0 for *int* parameters
- 0.0 for *real* parameters
- (0,0) for *xy* parameters
- NL for *char* parameters
- An array of 0 elements for array parameters

As you can see, the terminal assigns NL as the default for missing *char* parameters. This is an exception to the rule that *char* parameters must represent characters in the range from SP to ~ (ADEs 32 to 126). See the discussion of the *char* parameter in Section 7 for more details.

When an array parameter is terminated after being partially sent, the array count is adjusted to the number of elements already received.

NOTE

The terminal does not assign the standard defaults to some commands. To determine the default value for a particular command, consult the description of that command in Section 7.

PARAMETER TYPES

Parameters for escape sequence commands may be variables of several different parameter types. Each parameter type has its own syntax and coding scheme. The parameter types are *int,int-array*, *real*, *real-array*, *char*, *char-array* or *string*, *device*, *xy*, *xy-array*, *xy+*, and *xy+-array*.

REPORT MESSAGE TYPES

When the terminal sends report messages to the host computer, it packs the information in a format different from the parameter type. For each host-to-terminal parameter type there is a corresponding terminal-to-host message type.

For more information on *int-report*s, *intc-report*s, *int-array-report*s, *real-report*s, *real-array-report*s, *char-report*s, *char-array-report*s, *string-report*s, *xy-report*s, *xy-array-report*s, *xy+-report*s, and *xy+-array-report*s, see those descriptions in Section 7.

ANSI COMMAND SYNTAX

The ANSI X3.64 commands that are in Section 8 use a different syntax than the TEK commands in Section 7. ANSI commands (including alphatext) affect only the dialog area, whether the dialog area is enabled or not.

There are two syntax forms for the ANSI commands. Commands of one of these syntax forms consist of an $^{E}{}_{C}$ character and a unique final character. The final character is what identifies each command. The commands with this syntax form do not have parameters.

Commands of the other syntax form consist of a *control-sequence-introducer* (*CSI*), zero or more parameters of either *Pn* or *Ps* types, and a unique final character. The *CSI* and the terminator together identify individual commands.

The *CSI* syntax is:

$^{E}{}_{C}$[

Substitute these two characters whenever the *CSI* is called for.

The symbols that are used to represent syntax elements and conventions for ANSI commands are the same as those used for TEK commands (see Table 2-1).

ANSI PARAMETER TYPES

There are two parameter types for ANSI X3.64 commands, *Pn* and*Ps*.

Pn

Pn is a numeric parameter ranging from 0 to 32767. Send *Pn*'s as a sequence of digits. For example, send the number 75 as the two characters **7** and **5**. There is no special packing scheme for this parameter type. If the *Pn* is 0 or missing it is interpreted as 1 unless it is part of a *Ps* parameter.

Ps

Ps is a parameter selected from a given list. When the parameter type is *Ps*, the command description gives you a choice of parameters. There is no special packing scheme for this parameter type. They all have the format of a *Pn* or (char) *Pn* where (char) is **<**, **:**, **>**, or **?**.

The semicolon (;) separates parameters in a command string. Enter a semicolon between parameters when you are entering more than one parameter for a command. You can enter up to 46 characters of *Ps* parameters for a command that expects *Ps* parameters.

COMMAND TERMINATORS

ANSI mode uses the $^{C}{}_{N}$ and $^{S}{}_{B}$ characters as command terminators. When these characters are received in the middle of a command, the command is terminated and any characters belonging to that command that have already been received are discarded. When this occurs, a snoopy character $^{C}{}_{N}$ or $^{S}{}_{B}$ will appear on the terminal screen.

Section 3

INDEX OF COMMANDS BY FUNCTION

CONTROLLING THE DISPLAY

4010-HARDCOPY = $^{E}_{C}$$^{E}_{B}$ 7-337

HARDCOPY = $^{E}_{C}$**KH** int 7-90

SET-ERROR-THRESHOLD = $^{E}_{C}$**KT** int 7-208

SET-FIXUP-LEVEL = $^{E}_{C}$**RF** int. 7-209

SET-PAGE-FULL-ACTION = $^{E}_{C}$**KP** int 7-246

SET-SNOOPY-MODE = $^{E}_{C}$**KS** int .7-291

DIALOG AREA

CLEAR-DIALOG-SCROLL = $^{E}_{C}$**LZ** 7-25

DELETE-DIALOG-AREA = $^{E}_{C}$**KK** int 7-32

ENABLE-DIALOG-AREA = $^{E}_{C}$**KA** int 7-65

SET-DIALOG-AREA-ALTERNATE-INDEX
= $^{E}_{C}$**LJ** int 7-189

SET-DIALOG-AREA-BUFFER-SIZE = $^{E}_{C}$**LB** int 7-190

SET-DIALOG-AREA-CHARS = $^{E}_{C}$**LC** int 7-191

SET-DIALOG-AREA-INDEX = $^{E}_{C}$**LI** int int int 7-192

SET-DIALOG-AREA-LINES = $^{E}_{C}$**LL** int 7-194

SET-DIALOG-AREA-POSITION = $^{E}_{C}$**LX** xy. 7-195

SET-DIALOG-AREA-SURFACE = $^{E}_{C}$**LS** int. 7-196

SET-DIALOG-AREA-VISIBILITY = $^{E}_{C}$**LV** int 7-197

SET-DIALOG-AREA-WRITING-MODE = $^{E}_{C}$**LM** int 7-198

SET-DIALOG-HARDCOPY-ATTRIBUTES
= $^{E}_{C}$**QL** int int int. 7-199

GENERAL TERMINAL CONTROL

CANCEL = $^{E}_{C}$**KC** 7-21

RESET = $^{E}_{C}$**KV**. 7-143

SAVE-NONVOLATILE-PARAMETERS = $^{E}_{C}$**KU** 7-148A

SET-EDIT-CHARS = $^{E}_{C}$**KZ** int int int 7-204

GRAPHIC PRIMITIVES AND PRIMITIVE ATTRIBUTES

Alphatext

DISPLAY-ALPHA-TEXT = **char**. 7-54

ENTER-ALPHA-MODE = $^{U}_{S}$ 7-77

SET-4014-ALPHATEXT-SIZE
= $^{E}_{C}$**8**, $^{E}_{C}$**9**, $^{E}_{C}$**;**, OR $^{E}_{C}$**:** 7-315

SET-ALPHATEXT-FONT = $^{E}_{C}$$^{S}_{I}$ or $^{E}_{C}$$^{S}_{O}$ 7-164

SET-GRAPHICS-AREA-WRITING-MODE
= $^{E}_{C}$**MG** int. 7-225

SET-TEXT-INDEX = $^{E}_{C}$**MT** int 7-302

Defining Fill Patterns for Panels

BEGIN-FILL-PATTERN = $^{E}_{C}$**MD** int int int int. 7-5

END-FILL-PATTERN = $^{E}_{C}$**ME**. 7-75

Defining Graphtext Characters

BEGIN-GRAPHTEXT-CHARACTER = $^{E}_{C}$**ST** int int 7-6

DELETE-GRAPHTEXT-CHARACTER = $^{E}_{C}$**SZ** int int . . .7-35

END-GRAPHTEXT-CHARACTER = $^{E}_{C}$**SU** 7-76

SET-GRAPHTEXT-FONT-GRID = $^{E}_{C}$**SG** 7-228

Graphtext

GRAPHIC-TEXT = $^{E}_{C}$**LT** string 7-88

SET-GRAPHICS-AREA-WRITING-MODE
= $^{E}_{C}$**MG** int. 7-225

SET-GRAPHTEXT-FONT = $^{E}_{C}$**MF** int 7-226

SET-GRAPHTEXT-PRECISION = $^{E}_{C}$**MQ** int 7-230

SET-GRAPHTEXT-ROTATION = $^{E}_{C}$**MR** real 7-231

SET-GRAPHTEXT-SIZE = $^{E}_{C}$**MC** int int int 7-232

SET-GRAPHTEXT-SLANT = $^{E}_{C}$**MA** real 7-233

SET-TEXT-INDEX = $^{E}_{C}$**MT** int 7-302

Lines

Markers

Panels

Segment-Related Graphic Primitives

3D FACETS

HOST PORT COMMUNICATIONS

KEYBOARD CONTROL AND MACRO DEFINITIONS

MODE SWITCHING

MULTIPLE VIEWS

Defining and Selecting Views

OPTION 01 (HALF-DUPLEX AND BLOCK MODE)

Block Mode

Half-Duplex Mode

PERIPHERAL DEVICES

Option 3A or 3C (DMA Interface)

Option 19 (Color Hardcopy Interface)

Option 10 (Three Port Peripheral Interface)

Options 10, 13, and 14 (Graphics Tablet)

Options 42, 43, 45, 46, and 47 (Disk Drive Options)

ACTIVATE-LPOS = $^{E}_{C}$**JB**	7-2
DELETE-FILE = $^{E}_{C}$**JK** device	7-34
DIRECTORY = $^{E}_{C}$**JD** device string device	7-50
DISMOUNT = $^{E}_{C}$**JJ** device	7-53
FORMAT-VOLUME = $^{E}_{C}$**JF** device	7-83
PROTECT-FILE = $^{E}_{C}$**JP** device int	7-124
RENAME-FILE = $^{E}_{C}$**JR** device string device	7-130
SET-USER-NUMBER = $^{E}_{C}$**JU** int	7-305

Standard 4110/4120 Series Peripheral Support

COPY = $^{E}_{C}$**JC** device string device	7-28
LOAD = $^{E}_{C}$**JL** device	7-102
REPORT-DEVICE-STATUS = $^{E}_{C}$**JQ** device	7-136
SAVE = $^{E}_{C}$**JV** string int string device	7-147
SPOOL = $^{E}_{C}$**JS** device string device	7-318
STOP-SPOOLING = $^{E}_{C}$**JE**	7-320

PIXEL OPERATIONS

Data Transfer

PIXEL-COPY = $^{E}_{C}$**RX** int xy xy xy	7-115
RASTER-WRITE = $^{E}_{C}$**RP** int char-array	7-125
RECTANGLE-FILL = $^{E}_{C}$**RR** xy xy int	7-129
RESTORE-PIXELS-FROM-MEMORY = $^{E}_{C}$**UK** int xy xy	7-144
RUNLENGTH-WRITE = $^{E}_{C}$**RL** int-array	7-146
SAVE = $^{E}_{C}$**JV** string int string device	7-147
SAVE-PIXELS-TO-MEMORY = $^{E}_{C}$**UJ** int xy xy	7-149

Initialization

BEGIN-PIXEL-OPERATIONS = $^{E}_{C}$**RU** int int int	7-11
SET-PIXEL-BEAM-POSITION = $^{E}_{C}$**RH** xy	7-256
SET-PIXEL-VIEWPORT = $^{E}_{C}$**RS** xy xy	7-257
SET-PIXEL-WRITING-FACTORS = $^{E}_{C}$**RT** int int int	7-258

REPORTS FROM TERMINAL

Enabling and Disabling GIN

DISABLE-4953-TABLET-GIN = $^{E}_{C}$**!** char	7-53
DISABLE-GIN = $^{E}_{C}$**ID** int	7-52
ENABLE-4010-GIN = $^{E}_{C}{}^{S}_{B}$	7-71
ENABLE-4953-TABLET-GIN = $^{E}_{C}$**!** char	7-73
ENABLE-GIN = $^{E}_{C}$**IE** int int	7-67
REPORT-GIN-POINT = $^{E}_{C}$**IP** int	7-138

Setting GIN Parameters

SET-GIN-AREA = $^{E}_{C}$**IV** int int xy xy	7-211
SET-GIN-CURSOR = $^{E}_{C}$**IC** int int	7-213
SET-GIN-DISPLAY-START-POINT = $^{E}_{C}$**IX** int xy	7-214
SET-GIN-GRIDDING = $^{E}_{C}$**IG** int int int	7-215
SET-GIN-INKING = $^{E}_{C}$**II** int int	7-216
SET-GIN-LOGICAL-FUNCTION = $^{E}_{C}$**IO** int-array real-array xy + -array char-array	7-217
SET-PICK-APERTURE = $^{E}_{C}$**IA** int	7-253
SET-GIN-RATES = $^{E}_{C}$**IU** int	7-219
SET-GIN-REPORT-FORMAT = $^{E}_{C}$**IK** int	7-221
SET-GIN-RUBBERBANDING = $^{E}_{C}$**IR** int int	7-222
SET-GIN-STROKE-FILTERING = $^{E}_{C}$**IF** int int int	7-223
SET-GIN-WINDOW = $^{E}_{C}$**IW** xy xy	7-224

Requesting Non-GIN Reports

REPORT-COLORHARDCOPY-STATUS = $^{E}_{C}$**QQ**	7-136
REPORT-DEVICE-STATUS = $^{E}_{C}$**JQ** device	7-136
REPORT-ERRORS = $^{E}_{C}$**KQ**	7-137
REPORT-GIN-POINT = $^{E}_{C}$**IP**	7-138
REPORT-PORT-STATUS = $^{E}_{C}$**PQ** device	7-139
REPORT-SEGMENT-STATUS = $^{E}_{C}$**SQ** int char-array	7-140
REPORT-TERMINAL-SETTINGS = $^{E}_{C}$**IQ** char char	7-141
REPORT-4010-STATUS = $^{E}_{C}{}^{E}_{Q}$	7-142

Setting Parameters for Reports

SET-REPORT-EOM-FREQUENCY = $^{E}_{C}$**IM** int	7-270
SET-REPORT-MAX-LINE-LENGTH = $^{E}_{C}$**IL** int	7-271
SET-REPORT-SIG-CHARS = $^{E}_{C}$**IS** int int int	7-272

SEGMENTS

Defining Segments

Displaying Segments

Editing Segments

Reporting Segment Settings to the Host

Saving Segments

Setting Segment Attributes

SURFACES

ANSI COMMANDS

Cursor Commands

Tab Commands

Changing Modes and Graphic Renditions

Inserting and Deleting Lines

Inserting and Deleting Characters

Scrolling the Scroll Buffer

VT52 COMMANDS (4111 ONLY)

Section 4

INDEX OF NON-COMMANDS BY TYPE

CHARACTERS

Name

PARAMETER TYPES

Name

MESSAGE TYPES

Name

KEYS

Name

SYNTACTIC CONSTRUCTS

Name

Section 5

INDEX OF COMMANDS BY ESCAPE SEQUENCE OP CODES

TEK COMMANDS

Op Code	Command Name	Page
%!	SELECT-CODE	7-155
8	SET-4014-ALPHATEXT-SIZE	7-315
9	SET-4014-ALPHATEXT-SIZE	7-315
:	SET-4014-ALPHATEXT-SIZE	7-315
;	SET-4014-ALPHATEXT-SIZE	7-315
$^{C}_{N}$	ENTER-BYPASS-MODE	7-78
$^{E}_{B}$	4010-HARDCOPY	7-337
$^{E}_{Q}$	REPORT-4010-STATUS	7-142
$^{F}_{F}$	PAGE	7-111
$^{S}_{B}$	ENABLE-4010-GIN	7-71
$^{S}_{I}$	SET-ALPHATEXT-FONT	7-164
$^{S}_{O}$	SET-ALPHATEXT-FONT	7-164
!	DISABLE-4953-TABLET-GIN	7-53
!	ENABLE-4953-TABLET-GIN	7-73
IA	SET-PICK-APERTURE	7-253
IC	SET-GIN-CURSOR	7-213
ID	DISABLE-GIN	7-52
IE	ENABLE-GIN	7-67
IF	SET-GIN-STROKE-FILTERING	7-223
IG	SET-GIN-GRIDDING	7-215
IH	SET-TABLET-HEADER-CHARACTERS	7-299
II	SET-GIN-INKING	7-216
IK	SET-GIN-REPORT-FORMAT	7-221
IL	SET-REPORT-MAX-LINE-LENGTH	7-271
IM	SET-REPORT-EOM-FREQUENCY	7-270
IN	SET-TABLET-SIZE	7-300
IO	SET-GIN-LOGICAL-FUNCTION	7-217
IP	REPORT-GIN-POINT	7-138
IQ	REPORT-TERMINAL-SETTINGS	7-141
IR	SET-GIN-RUBBERBANDING	7-222
IS	SET-REPORT-SIG-CHARS	7-272
IT	SET-TABLET-STATUS-STRAP	7-301
IU	SET-GIN-RATES	7-219
IV	SET-GIN-AREA	7-211
IW	SET-GIN-WINDOW	7-224
IX	SET-GIN-DISPLAY-START-POINT	7-214
JB	ACTIVATE-LPOS	7-2
JC	COPY	7-28
JD	DIRECTORY	7-50
JE	STOP-SPOOLING	7-320

Op Code	Command Name	Page
JF	FORMAT-VOLUME	7-83
JH	SET-DMA-BLOCK-SIZE	7-201
JJ	DISMOUNT	7-53
JK	DELETE-FILE	7-34
JL	LOAD	7-102
JP	PROTECT-FILE	7-124
JQ	REPORT-DEVICE-STATUS	7-136
JR	RENAME-FILE	7-130
JS	SPOOL	7-318
JU	SET-USER-NUMBER	7-305
JV	SAVE	7-147
KA	ENABLE-DIALOG-AREA	7-65
KC	CANCEL	7-21
KD	DEFINE-MACRO	7-30
KE	SET-ECHO	7-203
KF	LFCR	7-101
KH	HARDCOPY	7-90
KI	IGNORE-DELETES	7-92
KK	DELETE-DIALOG-AREA	7-32
KL	LOCK-KEYBOARD	7-103
KN	RENEW-VIEW	7-133
KP	SET-PAGE-FULL-ACTION	7-246
KQ	REPORT-ERRORS	7-137
KR	CRLF	7-30
KS	SET-SNOOPY-MODE	7-291
KT	SET-ERROR-THRESHOLD	7-208
KV	RESET	7-143
KW	ENABLE-KEY-EXPANSION	7-71
KX	EXPAND-MACRO	7-81
KY	SET-KEY-EXECUTE-CHARACTER	7-235
KZ	SET-EDIT-CHARS	7-204
LB	SET-DIALOG-AREA-BUFFER-SIZE	7-190
LC	SET-DIALOG-AREA-CHARS	7-191
LE	END-PANEL	7-76
LF	MOVE	7-108
LG	DRAW	7-56
LH	DRAW-MARKER	7-62
LI	SET-DIALOG-AREA-INDEX	7-192
LJ	SET-DIALOG-AREA-ALTERNATE-INDEX	7-189
LK	INCLUDE-COPY-OF-SEGMENT	7-93
LL	SET-DIALOG-AREA-LINES	7-194
LM	SET-DIALOG-AREA-WRITING-MODE	7-198
LP	BEGIN-PANEL-BOUNDARY	7-10

ANSI COMMANDS

VT52 COMMANDS

Section 6

KEYWORD IN CONTEXT INDEX

This index is a "Keyword in Context" index. A Keyword in Context index contains an entry for each significant word in the command name. For example, the INCLUDE-COPY-OF-SEGMENT command can be found under C, I, and S as follows:

Include **Copy** of Segment
Include Copy of Segment
Include Copy of **Segment**

Since many 4110/4120 Series command names begin with "Set," the word "set" is not included as a keyword for indexing purposes.

TEK COMMANDS

ANSI COMMANDS

VT52 COMMANDS (4111 ONLY)

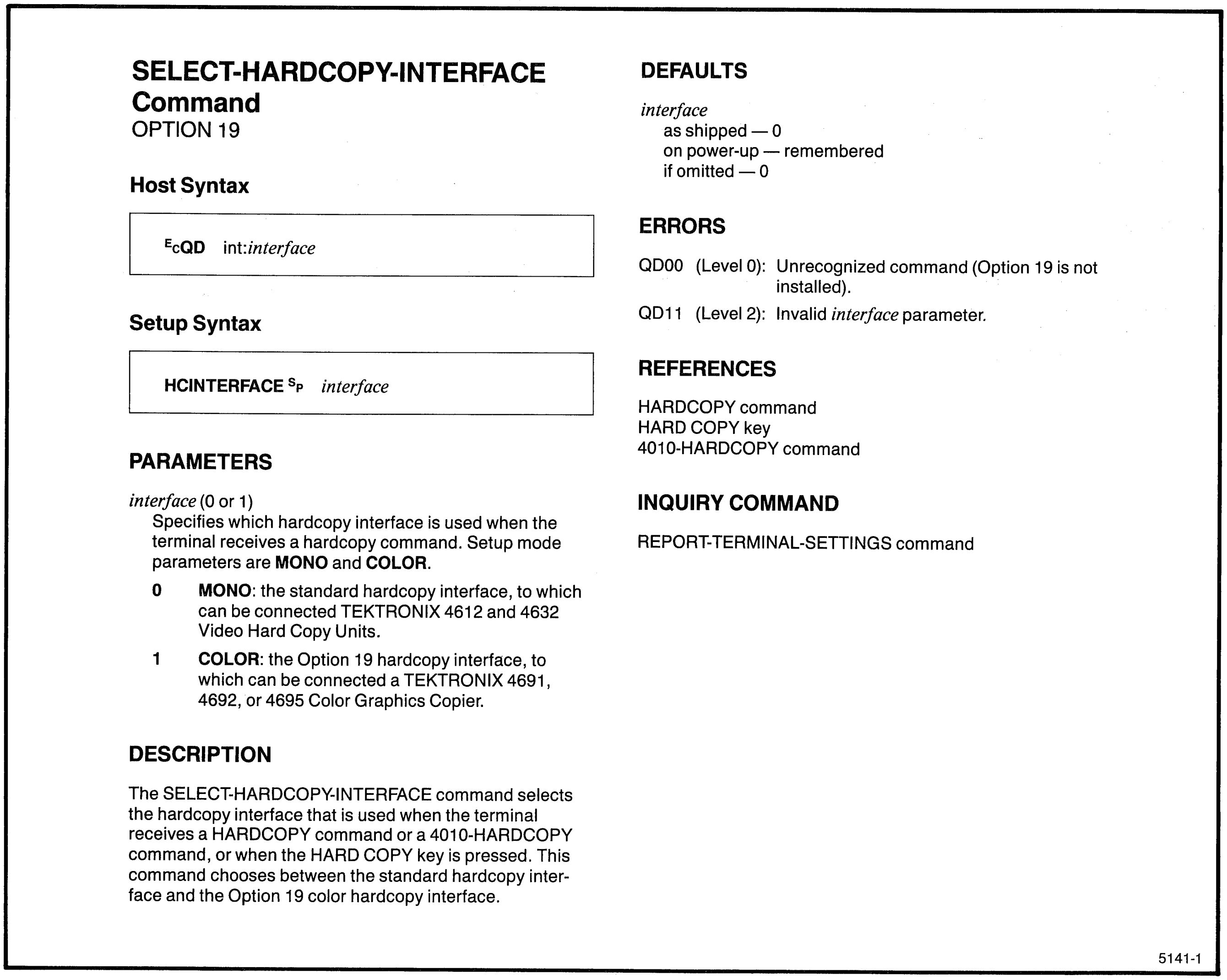
Section 7

TEK COMMANDS AND SYNTACTIC CONSTRUCTS

This section lists all the TEK commands, as well as message formats, parameter types and reports sent to the host.

Command descriptions include the syntax of the command, an explanation of the command's parameters (if any), the function of the command, defaults for the command's parameters (if any), errors that may occur, references to related commands and syntactic constructs, and whether the command is inquirable. If a command is only valid for a certain option or for certain terminals, those terminals or options are listed under the command description. Figure 7-1 shows a sample command description.

If you do not know the name of a command, refer to Sections 3, 4, 5, or 6, the indexing and cross-referencing sections. Most command names express the action that the command performs, such as SET-BAUD-RATES, SET-LINE-INDEX, and ENABLE-GIN.



SELECT-HARDCOPY-INTERFACE Command

OPTION 19

Host Syntax

$^{E}{}_{C}$**QD** int:*interface*

Setup Syntax

HCINTERFACE $^{S}{}_{P}$ *interface*

PARAMETERS

interface (0 or 1)
Specifies which hardcopy interface is used when the terminal receives a hardcopy command. Setup mode parameters are **MONO** and **COLOR**.

0 **MONO**: the standard hardcopy interface, to which can be connected TEKTRONIX 4612 and 4632 Video Hard Copy Units.

1 **COLOR**: the Option 19 hardcopy interface, to which can be connected a TEKTRONIX 4691, 4692, or 4695 Color Graphics Copier.

DESCRIPTION

The SELECT-HARDCOPY-INTERFACE command selects the hardcopy interface that is used when the terminal receives a HARDCOPY command or a 4010-HARDCOPY command, or when the HARD COPY key is pressed. This command chooses between the standard hardcopy interface and the Option 19 color hardcopy interface.

DEFAULTS

interface
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

QD00 (Level 0): Unrecognized command (Option 19 is not installed).

QD11 (Level 2): Invalid *interface* parameter.

REFERENCES

HARDCOPY command
HARD COPY key
4010-HARDCOPY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

5141-1

Figure 7-1. Sample Command Page.

ACTIVATE-LPOS Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

$^{E}_{C}$**JB** device:*boot-file*

Setup Syntax

LPOS $^{S}_{P}$ *boot-file*

PARAMETERS

boot-file

Device name and/or filename from which to boot local programmability. Valid devices are Option 42, 43, 45, 46, and 47 disk devices and files on those devices.

DESCRIPTION

This command activates the local programmability operating system (LPOS). If the operating system has not been loaded (is not resident in terminal memory), then this command loads it from a file on the disk mounted on the specified drive.

LPOS Not Loaded

The following are the four possible boot actions that you may specify (by entering the parameter in different ways) when LPOS is not resident in terminal memory:

- If the parameter is completely omitted, then LPOS boots from the file default operating system file in user area 15 of the default drive (F0: if Option 42 or 43 is installed, or the first device connected to the Option 45 interface).
- If the parameter is a filename without a device specifier, LPOS boots from the given file in user area 15 of the default drive (see above).
- If a device is specified without an explicit filename, LPOS boots from the default operating system file in user area 15 of the specified drive.
- If both device and filename are specified, then LPOS boots from the specified file in user area 15 of the specified device.

The default operating system filename is:

- If the terminal firmware version number is one digit, then the filename is **LP**v**.SYS** where "v" is the version number (e.g. **LP4.SYS**).
- If the terminal firmware version number is two digits, then the filename will be **L**vv**.SYS** where "vv" is the version number.

LPOS Loaded

If LPOS is resident in terminal memory, then the following may occur:

- If the *boot-file* parameter is specified, error JB03 is generated.
- If the parameter is omitted but LPOS has not been deactivated by the LPOS SLEEP command, error JB03 is generated.
- If the parameter is omitted and LPOS has been deactivated by the LPOS SLEEP command, then this command activates LPOS so that it resumes its task from where it left off.

See the *4110 Series CP/M-86 System Overview Manual* for information on the LPOS SLEEP command.

When the terminal receives this command, it activates the operating system immediately. When an operator issues this command from Setup mode, the operating system is activated but Setup is not disabled. Setup mode supersedes local programmability, so you must exit Setup mode before local programs can run. You can enter Setup mode as often as you want, but you cannot use local programs until you exit Setup mode.

When local programmability is activated, the LED in the LOCAL key is turned on and further communications from the host are placed in the input queue (as if the terminal were in Local mode). The operating system remains active until the terminal is reset or turned off, or until the local programmability SLEEP command is issued.

While local programmability is active, if the LED in the LOCAL key is on, an operator can interact directly with the operating system or a local program. When the LOCAL key light is off, the terminal reacts normally to host communication.

DEFAULTS

boot-file
as shipped — none
on power-up — none
if omitted — see previous description

ERRORS

JB03 (Level 2): A filename was specified when LPOS was resident in terminal memory, or a filename was not specified, but LPOS was active.

JB10 (Level 2): The *boot-file* was not found.

JB11 (Level 2): Illegal filename.

JB12 (Level 3): Out of memory while performing command.

JB13 (Level 2): Invalid device specifier, invalid medium format or the file is currently being written.

JB19 (Level 2): The drive is not ready or a hardware error occurred while reading from the drive.

REFERENCES

4110 Series CP/M-86 System Overview Manual
Device parameter type

INQUIRY COMMAND

none

ARM-FOR-BLOCK-MODE Command

4111 or OPTION 01

Host Syntax

```
ᴱcOB   int:block-mode-arming
```

Setup Syntax

```
BLOCKMODE ˢₚ  block-mode-arming
```

PARAMETERS

block-mode-arming (0 or 1)
Arms or disarms the terminal to accept block-mode transmission.

0 Disarms the terminal so that it will not enter block mode. Setup parameter is **NO**.

1 Arms the terminal to enter block mode with receipt of the next block header. Setup parameter is **YES**.

DESCRIPTION

This command determines whether the terminal will enter Block mode when it receives a block header. If the *int* parameter is 1, the terminal is armed for Block mode and will enter Block mode when it receives the *receive* block header specified by the most recent SET-BLOCK-HEADERS command.

If the *int* parameter is 0, subsequent block headers will not put the terminal in Block mode.

CAUTION

If you do not intend to use Block mode, do not arm the terminal for Block mode. The terminal cannot recognize EOF-string*s coming from the host computer when armed for Block mode. So, when the terminal is armed for Block mode, but is not yet actually in Block mode, the only way to terminate a copy or port-copy is with the CANCEL key.*

Before arming the terminal for Block mode, set all necessary Block mode parameters commands. The commands that set these parameters (except for SET-BLOCK-TIMEOUT) are invalid when the terminal is armed for Block mode. A list of these commands is found under "References" at the end of this command description.

Before arming the terminal for Block mode, be sure that the terminal's communications input queue is large enough to hold an entire block. Use the SET-QUEUE-SIZE command to make the communications queue larger, or use the SET-BLOCK-LENGTH command to make the block size smaller.

The Block mode protocol effectively suppresses any remote echoes that the host provides. So, before placing the terminal in Block mode, issue a SET-ECHO command so that the terminal will provide its own local echo of characters typed by an operator.

DEFAULTS

block-mode-arming
as shipped — 0 on
power-up — remembered
if omitted — 0

ERRORS

OB00	(Level 0):	Unrecognized command. (Not a 4111 or Option 01 is not installed)
OB03	(Level 2):	The communications queue size is smaller than the specified input block size.
OB11	(Level 2):	Invalid *block-mode-arming* parameter.

REFERENCES

Block syntactic construct
SET-BLOCK-CONTINUE-CHARS command
SET-BLOCK-END-CHARS command
SET-BLOCK-HEADERS command
SET-BLOCK-LENGTH command
SET-BLOCK-LINE-LENGTH command
SET-BLOCK-MASTER-CHARS command
SET-BLOCK-NON-XMT-CHARS command
SET-BLOCK-PACKING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

BEGIN-FILL-PATTERN Command

Host Syntax

```
EcMD  int:fill-pattern-number   int:pattern-width
      int:pattern-height   int:bits-per-pixel
```

Setup Syntax

```
EcMD Sp  fill-pattern-number
         pattern-width   pattern-height
         bits-per-pixel
```

PARAMETERS

fill-pattern-number (1 to 32767)
The number of the fill pattern being defined.

pattern-width (1 to 1280)
The width of the pattern in pixels.

pattern-height (0 to 1024)
The height of the pattern in pixels.

0 Deletes the fill pattern definition

>0 Assigns pattern height

bits-per-pixel (1 to 8)
Used to decode subsequent RASTER-WRITE and RUNLENGTH-WRITE commands.

DESCRIPTION

This command begins or deletes a *fill-pattern-definition.* A fill pattern is a rectangular array of color indices. These indices are all initialized to 0 by this command. See the discussion of *fill-pattern-definition* in this section for more details.

The *fill-pattern-number* specifies the pattern number assigned to the fill pattern you are creating. This can be any number from 1 to 32767. If a fill pattern with its same number already exists, the previous definition is deleted and the new definition takes its place.

Pattern-height can be from 0 to 1024. Specifying a 0 deletes the pattern.

The *bits-per-pixel* parameter is used to decode the RASTER-WRITE and RUNLENGTH-WRITE commands that follow. This affects which color indices may occur in the fill pattern.

DEFAULTS

fill-pattern-number
as shipped — none
on power-up — none
if omitted — error MD11

pattern-width
as shipped — none
on power-up — none
if omitted — error MD21

pattern-height
as shipped — none
on power-up — none
if omitted — error MD31

bits-per-pixel
as shipped — none
on power-up — none
if omitted — error MD41

ERRORS

MD02 (Level 3): Not enough memory available for fill pattern.

MD03 (Level 2): Another fill pattern is currently being defined.

MD11 (Level 2): Invalid *fill-pattern-number.*

MD21 (Level 2): Invalid *pattern-width.*

MD31 (Level 2): Invalid *pattern-height.*

MD41 (Level 2): Invalid *bits-per-pixel.*

REFERENCES

Fill-pattern-definition syntactic construct
RASTER-WRITE command
RUNLENGTH-WRITE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

BEGIN-GRAPHTEXT-CHARACTER Command

Host Syntax

EcST int:*font* int:*character*

Setup Syntax

EcST SP *font* *character*

PARAMETERS

font (0 to 32767)
The font number in which the character being defined resides.

character (32 to 126)
The ASCII decimal equivalent of the character being defined.

DESCRIPTION

This command opens the definition of a graphtext character in the specified font. See the *Graphtext-font-definition* syntactic construct for details on how graphtext characters are defined.

If the font does not have a grid set for it (by the SET-GRAPHTEXT-FONT-GRID command), if the character is already defined, or if the pivot-point (set by the SET-PIVOT-POINT command) is outside the 0 to 4095 range, the terminal detects an error and does not open the character definition. If a MOVE or DRAW command received during the definition is outside the 0 to 4095 range, the terminal detects an error and closes the definition.

DEFAULTS

font
as shipped — none
on power-up — none
if omitted — 0

character
as shipped — none
on power-up — none
if omitted — error ST21

ERRORS

ST01 (Level 2): MOVE or DRAW is out of range (X = 0 to 4095, Y = 0 to 4095).

ST02 (Level 3): Out of memory while defining graphtext character.

ST03 (Level 2): Command is invalid at this time. (A graphtext character is currently being defined, or the pivot-point coordinate is out of range X = 0 to 4095, Y = 0 to 4095, or coordinate mode is 2 (4128 and 4129 only)).

ST10 (Level 2): The specified font does not exist (no SET-GRAPHTEXT-FONT-GRID command has been issued for *font*).

ST11 (Level 2): Invalid *font* number.

ST20 (Level 2): The *character* specified has already been defined in this *font*.

ST21 (Level 2): Invalid *character* number.

REFERENCES

Graphtext-font-definition syntactic construct
SET-PIVOT-POINT command

INQUIRY COMMAND

none

BEGIN-HIGHER-SEGMENT Command

Host Syntax

$^{E}_{C}$SN

Setup Syntax

$^{E}_{C}$SN

DESCRIPTION

This command ends the definition of the segment which is currently being defined, and begins the definition of a segment with a number (segment ID) one greater than that of the segment just ended. If no segment is currently being defined, an error is detected and no action is taken.

The pivot point and position of the new segment are set to the current beam position. The current *pick-id* is set to 1.

Opening a 3D segment resets the 3D alpha position to the current 3D graphic beam position.

This command is equivalent to the following sequence of commands:

REPORT-TERMINAL-SETTINGS: **SO**
(returns open segment number)
END-SEGMENT
REPORT-GIN-POINT: –2 (returns current beam *xy*+)
REPORT-TERMINAL-SETTINGS: **SP**
(returns current pivot point *xy*+)
REPORT-SEGMENT-STATUS: –2, X
(returns future segment position *xy*+)
SET-PIVOT-POINT: (current beam *xy*+)
BEGIN-SEGMENT: (next higher segment ID)
SET-PIVOT-POINT: (previous pivot point *xy*+)
SET-SEGMENT-POSITION: –2,
(future segment position *xy*+)

See the *Segment-definition* syntactic construct for details.

When a segment created with this command is saved with the SAVE command, the segment definition is saved using the BEGIN-SEGMENT command instead of the BEGIN-HIGHER-SEGMENT command, since all segments are the same once they are created.

ERRORS

SN00 (Level 2): The indicated segment already exists.

SN01 (Level 2): Invalid value for next higher segment number (current segment ID is 32767).

SN02 (Level 3): Out of memory while ending or beginning segment definition.

SN03 (Level 2): Context error; command is invalid at this time. No segment is currently being defined, or a graphtext character is currently being defined.

REFERENCES

BEGIN-SEGMENT command
END-SEGMENT command
REPORT-GIN-POINT command
REPORT-SEGMENT-STATUS command
REPORT-TERMINAL-SETTINGS command
Segment-definition syntactic construct
SET-PIVOT-POINT command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command for
BEGIN-SEGMENT command ($^{E}_{C}$SO)

BEGIN-LOWER-SEGMENT Command

Host Syntax

^{E}c**SB**

Setup Syntax

^{E}c**SB**

DESCRIPTION

This command ends the definition of the segment which is currently being defined, and begins the definition of a segment with a number (segment ID) one less than that of the segment just ended. If no segment is currently being defined, an error is detected and no action is taken.

The pivot point and position of the new segment are set to the current beam position. The current *pick-id* is set to one.

Opening a 3D segment resets the 3D alpha position to the current 3D graphic beam position.

This command is equivalent to the following sequence of commands:

REPORT-TERMINAL-SETTINGS: **SO**
  (returns open segment number)
END-SEGMENT
REPORT-GIN-POINT: −2 (returns current beam *xy*+)
REPORT-TERMINAL-SETTINGS: **SP**
  (returns current pivot point *xy*+)
REPORT-SEGMENT-STATUS: −2, X
  (returns future segment position *xy*+)
SET-PIVOT-POINT: (current beam *xy*+)
BEGIN-SEGMENT: (next lower segment ID)
SET-PIVOT-POINT: (previous pivot point *xy*+)
SET-SEGMENT-POSITION: −2,
  (future segment position *xy*+)

See the *Segment-definition* syntactic construct for details.

When a segment created with this command is saved with the SAVE command, the segment definition is saved using the BEGIN-SEGMENT command instead of the BEGIN-LOWER-SEGMENT command, since all segments are the same once they are created.

ERRORS

SB00 (Level 2): The indicated segment already exists.

SB01 (Level 2): Invalid value for next lower segment number (current segment ID is 1).

SB02 (Level 3): Out of memory while ending or beginning segment definition.

SB03 (Level 2): Context error; command is invalid at this time. No segment is currently being defined, or a graphtext character is currently being defined.

REFERENCES

BEGIN-SEGMENT command
END-SEGMENT command
REPORT-GIN-POINT command
REPORT-SEGMENT-STATUS command
REPORT-TERMINAL-SETTINGS command
Segment-definition syntactic construct
SET-PIVOT-POINT command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command for
  BEGIN-SEGMENT command (^{E}cSO)

BEGIN-NEW-SEGMENT Command

Host Syntax

```
EcSE   int:segment-number
```

Setup Syntax

```
EcSE Sp   segment-number
```

PARAMETERS

segment-number (1 to 32767)
The number of the segment being defined.

DESCRIPTION

This command begins the definition of a new graphic segment with the specified segment identification number. If another segment is open, it is closed (as with an END-SEGMENT command).

Valid segment numbers range from 1 to 32767. If a segment with the specified segment number already exists, or if the segment number used is invalid, an error is detected and reported. If a graphtext character definition is currently open, or a panel is being defined with no segment also being defined, an error is detected and the command is ignored.

The pivot point and position of the new segment are set to the current beam position. The pick-ID is set to one.

Opening a 3D segment resets the 3D alpha position to the current 3D graphic beam position.

Issuing this command is equivalent to issuing the following sequence of commands:

END-SEGMENT (if segment is currently open)
REPORT-GIN-POINT: −2 (returns current beam *xy*+)
REPORT-TERMINAL-SETTINGS: **SP**
(returns current pivot point *xy*+)
REPORT-SEGMENT-STATUS: −2,
x (returns future segment position *xy*+)
SET-PIVOT-POINT: (current beam *xy*+)
BEGIN-SEGMENT: *segment-number*
SET-PIVOT-POINT: (previous pivot point *xy*+)
SET-SEGMENT-POSITION: −2,
(future segment position *xy*+)

If there is no segment currently open, issuing this command is equivalent to issuing the above series of commands without the initial END-SEGMENT command.

See the *Segment-definition* syntactic construct for details.

When a segment created with this command is saved with the SAVE command, the segment definition is saved using the BEGIN-SEGMENT command instead of the BEGIN-NEW-SEGMENT command, since all segments are the same once they are created.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SE11

ERRORS

SE02 (Level 3): Not enough memory to begin segment, or out of memory while defining segment.

SE03 (Level 2): Command is invalid at this time (a graphtext character or a panel is currently being defined).

SE10 (Level 2): Segment already exists.

SE11 (Level 2): Invalid *segment-number*.

REFERENCES

BEGIN-SEGMENT command
END-SEGMENT command
Segment-definition syntactic construct
SET-PIVOT-POINT command
SET-SEGMENT-IMAGE-TRANSFORM command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

BEGIN-PANEL-BOUNDARY Command

Host Syntax

$^{E}_{C}$**LP** xy:*first-point* int:*draw-boundary*

Setup Syntax

$^{E}_{C}$**LP** $^{S}_{P}$ *first-point* *draw-boundary*

PARAMETERS

first-point

The point where the panel boundary begins and ends. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = 2^{31} to $2^{31}-1$

draw-boundary (0 or 1)

Determines whether the panel boundary will be drawn or not.

0 not drawn (absent)

1 drawn (present)

DESCRIPTION

This command begins the definition of a panel boundary at the given point and sets the 2D graphic beam position to the given point. A panel boundary is a closed polygon consisting of zero or more edges. A panel is made up of one or more panel boundaries. If a panel-definition is not currently open, this command opens it. If a panel-definition is open, this command causes the terminal to close the current boundary and start another boundary.

If *draw-boundary* is 0, the boundary edges are not displayed. If *draw-boundary* is 1, all boundary edges are displayed as vectors in the current line index and current line style. (See the SET-LINE-STYLE and SET-LINE-INDEX commands.)

See the *Panel-definition* syntactic construct for details on panels.

DEFAULTS

first-point
as shipped — none
on power-up — none
if omitted — (0,0) 2D

draw-boundary
as shipped — none
on power-up — none
if omitted — 0

ERRORS

LP02	(Level 3):	Out of memory while defining panel.
LP03	(Level 2):	Alphatext and graphtext are not allowed within a *panel-definition* or a subpanel cannot be added to an existing panel being edited, or coordinate mode is not 0 or 1.
LP11	(Level 2):	*First-point* out of valid range.
LP21	(Level 2):	*Draw-boundary* out of valid range.

REFERENCES

Panel-definition syntactic construct
SET-LINE-INDEX command
SET-LINE-STYLE command

INQUIRY COMMAND

none

BEGIN-PIXEL-OPERATIONS Command

Host Syntax

$^{E}_{C}$**RU** int:*surface-number*
int:*ALU-mode*
int:*bits-per-pixel*

Setup Syntax

$^{E}_{C}$**RU** $^{S}_{P}$ *surface-number* *ALU-mode*
bits-per-pixel

PARAMETERS

surface-number (–1 to 8)
Specifies the surface(s) on which subsequent pixel operations will take place.

–1	super surface (all bit planes of all surfaces)
0	current surface
1 to **8**	a specific surface

ALU-mode (0, 7, 11, 12, 15, 17, 18)
Arithmetic logic unit writing mode. The function that relates the current pixel to the pixel information introduced by subsequent pixel operations. A value of 0 means no change.

bits-per-pixel (0 to 8)
The number of bits in each pixel. Used for decoding subsequent RASTER-WRITE and RUNLENGTH-WRITE commands.

0	current number of *bits-per-pixel*
1 to **8**	specific number of *bits-per-pixel*

DESCRIPTION

The BEGIN-PIXEL-OPERATIONS command sets three parameters for use by subsequent pixel operations commands. That is, commands that permit you to access directly the terminal's raster memory. RASTER-WRITE, RUNLENGTH-WRITE, RECTANGLE-FILL, PIXEL-COPY, and SAVE commands are pixel operations commands. Also, the parameters set by this command are used by the PX: pseudo device.

Surface-number. The *surface-number* parameter specifies the surface on which subsequent pixel operations commands will write (or read) their data. For detailed information on surfaces, see the *4110/4120 Series Host Programmers Manual.*

A value of 0 indicates that the existing surface is not to be changed. A value of –1 indicates that subsequent pixel operations occur on a "super surface" consisting of all bit planes in all currently-defined surfaces. This super surface is for advanced applications.

ALU-mode. The *ALU-mode* integer parameter can be 0, 7, 11, 12, 15, 17 or 18. A value of 0 indicates that the existing *ALU-mode* is not to be changed. Values from 7 to 18 specify how RASTER-WRITE, RUNLENGTH-WRITE, RECTANGLE-FILL, and PIXEL-COPY commands modify the existing contents of the terminal's raster memory buffer.

At each pixel position in raster memory space, the color index being written is regarded as a binary number and is combined, bit by bit, with the color index already stored at that pixel location. The result is a new color index, which is stored at that pixel location in the raster memory buffer.

Let A be one of the bits in the color index currently stored at a particular pixel location in raster memory space. Let B be the corresponding bit in the color index being supplied by the RASTER-WRITE, RUNLENGTH-WRITE, RECTANGLE-FILL, or PIXEL-COPY command. Let W be the corresponding bit of the color index which the terminal's ALUs (arithmetic logic units) actually write into that pixel location in raster memory space. Then W is some function of A and B:

$$W = f(A,B)$$

The *ALU-mode* parameter selects the function f from among six possible functions of A and B: f1(A,B) to f18(A,B). Table 7-1 lists the function fN selected by each value N of the ALU mode parameter. In the table, bit A is the bit already stored in the raster memory buffer; bit B is the bit supplied by the RASTER-WRITE, RUNLENGTH-WRITE, RECTANGLE-FILL, or PIXEL-COPY command, and bit W is the resulting bit which is actually written into the raster memory buffer.

Modes 17 and 18, rather than operating bit by bit, operate on all bits of the pixel surface. For instance, for Mode 17 and an eight-bit pixel surface, if A = 5 (binary 00000101) and B = 3 (binary 00000011), then A + B = 8 (binary 00001000). For an n-bit pixel surface, the addition or subtraction (Modes 17 or 18, respectively) is done in modulo 2^n arithmetic. Thus, on a 4-bit surface, 15 + 15 = 14.

Table 7-1

ALU VALUES TABLE

ALU-mode	W = f(A,B)
0	no change
7	A XOR B
11	B
12	A and B
15	A or B
17	A + B
18	A – B

Of the values in Table 7-1, the following are particularly useful:

Mode	W = f(A,B)	Use
7	A XOR B	This provides an opportunity to write an image which can later be completely removed without trace by repeating the same operation. That is because A = (A XOR B) XOR B.
11	B	This is the default *ALU-mode*. This causes a complete replacement of the existing image with the new pixels.
15	A or B	This is an "overstrike mode," in which the new image is written on top of the existing image. Zero pixel values in the command string do not affect the pixel buffer.

Bits-per-pixel. The *bits-per-pixel* parameter is used to decode subsequent RASTER-WRITE and RUNLENGTH-WRITE commands. It is also used by subsequent SAVE commands when saving a group of pixels using a stream of RASTER-WRITE and RUNLENGTH-WRITE commands. See the descriptions of the RASTER-WRITE, RUNLENGTH-WRITE, and SAVE commands for details.

A value of 0 indicates that the existing *bits-per-pixel* encoding mode is not to be changed.

If the *surface-number* parameter specifies a surface which has fewer bit planes than the number in the *bits-per-pixel* parameter, the least significant (low-order) bits are ignored in indices written on that surface.

DEFAULTS

surface-number
as shipped — 1
on power-up — 1
if omitted — unchanged

ALU-mode
as shipped — 11
on power-up — 11
if omitted — unchanged

bit-per-pixel
as shipped — 6
on power-up — 6
if omitted — unchanged

ERRORS

RU10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RU11 (Level 2): Invalid *surface-number.*

RU21 (Level 2): Invalid *ALU-mode.*

RU31 (Level 2): Invalid *bits-per-pixel.*

REFERENCES

PIXEL-COPY command
RASTER-WRITE command
RECTANGLE-FILL command
RUNLENGTH-WRITE command
SAVE command
SET-SEGMENT-WRITING-MODE command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

BEGIN-SEGMENT Command

Host Syntax

$^{E}{}_{C}$**SO** int:*segment-number*

Setup Syntax

$^{E}{}_{C}$**SO** $^{S}{}_{P}$ segment-number

PARAMETERS

segment-number (1 to 32767)
Number of segment to be defined.

DESCRIPTION

This command is one way to start a segment definition. *Segment-number* indicates the specified segment identification number. The command resets the current *pick-ID* to one. See the *Segment-definition* syntactic construct for details on how segments are defined. In the 4128 and 4129, the dimensionality of the segment is set to match the current coordinate mode of the terminal.

Opening a 3D segment resets the 3D alpha position to the current 3D graphic beam position.

Valid *segment-number*s are 1 to 32767. If a segment with the specified segment number already exists, or if the segment number used is invalid, an error is generated and the command is ignored. If another segment, graphtext character, or panel boundary definition is currently being defined, an error is generated and the command is ignored.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SO11

ERRORS

SO00 (Level 4): Picture Processor error. Segment transform error; segment scaled out of window, or window decreased beyond scaled segment. See Appendix C for more information.

SO02 (Level 3): Not enough memory to begin segment, or 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*.

REFERENCES

Segment-definition syntactic construct
SET-PICK-ID command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

$^{B}{}_{L}$ Character

DESCRIPTION

When a terminal receives a $^{B}{}_{L}$ character it sounds its bell.

A $^{B}{}_{L}$ in vector mode sets the move/draw flag to draw, so that $^{G}{}_{S}{}^{B}{}_{L}$ *xy* causes a vector to be drawn from the current position to the specified xy coordinate.

REFERENCES

ENTER-VECTOR-MODE command

Block Syntactic Construct

4111 or OPTION 01

Syntax

```
block  =  [block-"other-than-last"-line . . . ]
           block-last-line

block-"other-than-last"-line  =  block-header
                                 packed-data
                                 block-continue-char
                                 EOL-string

block-last-line  =  block-header
                    packed-data
                    block-end-char
                    EOL-string
```

PARTS OF A BLOCK

block-header
A sequence of up to ten characters, set by the SET-BLOCK-HEADERS command.

packed-data
Characters of packed data, the result of applying the packing algorithm selected by the SET-BLOCK-PACKING command.

block-continue-char
An ASCII character, selected by the SET-BLOCK-CONTINUE-CHARS command.

block-end-char
An ASCII character, selected by the SET-BLOCK-END-CHARS command.

EOL-string
A sequence of up to two characters, determined by the SET-EOL-STRING command.

DESCRIPTION

In block mode, each block sent to or from the terminal has the syntax described above. Each "line" of a block starts with a *block-header*, which is defined by the SET-BLOCK-HEADERS command. The end of each line is signalled by a *block-continue-char* (if it is not the last line in the block), or by a *block-end-char* (for the last line of the block).

Between the *block header* and the *block-continue-char* or *block-end-char* is a sequence of ASCII characters which hold the data being sent in a "packed" format; these characters comprise the *packed-data*. None of these characters may be the same as the characters reserved as the *block-continue-char* and *block-end-char*. The packing scheme is described together with the SET-BLOCK-PACKING command, elsewhere in this section.

The *EOM-character* (usually a CR) from the keyboard, which terminates a terminal-to-host block, is included in the block as the last data character.

The *block-header*, *block-continue-char*, *block-end-char*, and the block packing scheme may be different for blocks sent from the terminal to the host than for blocks which the terminal receives from the host. These parameters are set with the SET-BLOCK-HEADERS, SET-BLOCK-CONTINUE-CHARS, SET-BLOCK-END-CHARS, and SET-BLOCK-PACKING commands.

For blocks sent from the terminal to the host computer, the *EOL-string* is the current "end of line string," set by the SET-EOL-STRING command. (Typically, this is CR or CRLF.)

For blocks sent from the host computer to the terminal, the *EOL-string* consists of any characters which the host may send after the *block-continue-char* or *block-end-char* and before the *block-header* that starts the next block. All such characters are ignored by the terminal until (and unless) a printable character — ADE 32 to 126 — is encountered.

When the terminal detects the printable character it does a carriage return and prints the detected character and all following text (as unescorted text) until the next *block header* is detected.

REFERENCES

Block-control-bytes syntactic construct
SET-BLOCK-CONTINUE-CHARS command
SET-BLOCK-END-CHARS command
SET-BLOCK-HEADERS command
SET-BLOCK-LENGTH command
SET-BLOCK-MASTER-CHARS command
SET-BLOCK-NON-XMT-CHARS command
SET-BLOCK-PACKING command
SET-BLOCK-TIMEOUT command

Block-Control-Bytes Syntactic Construct

4111 or OPTION 01

Syntax

```
block-control-bytes   =   control-byte-1
                          control-byte-2
                          control-byte-3
                          control-byte-4
```

DESCRIPTION

In block mode, when the terminal or host composes a block to be sent over the data communications line, it appends four *block-control-bytes* to the characters or other data being packed into the block.

The four control bytes are packed into the block along with the other unpacked data; see the description of the SET-BLOCK-PACKING command for details. If the *unpacked-byte-size* (as set by the SET-BLOCK-PACKING command) is seven, then each control byte consists of seven binary bits. If the unpacked byte size is eight, then each control byte consists of eight binary bits.

Control-Byte-1

Let bit 1 be the least-significant bit of the byte; then bit 7 or bit 8 is the most-significant bit of the byte. (Bit 7 is the most-significant bit if the "unpacked byte size" is seven, since in that case there is no bit 8.) The individual bits are assigned as follows:

Bits 1 and 2:	Block count and end-protocol.
Bits 3, 4, 5:	Reserved (always zero)
Bit 6:	End of file
Bit 7:	End of message
Bit 8	Unused (not present in 7-bit bytes; always zero in 8-bit bytes)

In control-byte-1, bits 1 and 2 together serve two functions: they determine whether the terminal is to exit block mode, and — while the terminal is in block mode — they maintain an "odd/even" modulo two counter of blocks sent over the data communications line. Table 7-2 lists the four possible states of these bits, together with their meanings.

Table 7-2

MEANINGS OF LOW-ORDER BITS IN CONTROL-BYTE-1

Bit 2	Bit 1	Meaning
0	0	This is an "even" block; the terminal remains in block mode.
0	1	This is an "odd" block; the terminal remains in block mode.
1	0	In a block sent from the host to the terminal, this bit pattern acts as a command to the terminal to exit from block mode, but remain armed for block mode, and before exiting block mode, to acknowledge this command by sending an 'ACK' block to the host." In a block sent from the terminal to the host, this bit pattern means, "This is an 'ACK' block acknowledging receipt of a command to exit block mode. The terminal is now leaving block mode, but will remain armed for block mode."
1	1	In a block sent from the host to the terminal, these two bits comprise a command to the terminal: "Exit from block mode, but remain armed for block mode. Exit block mode immediately; do not send an 'ACK' block to the host." In a block sent from the terminal to the host, this combination of bits is not allowed.

Bit 6, the end-of-file bit, is set to one at the end of a file transfer; setting this bit serves the same purpose as the EOF-string used when the terminal is not in block mode.

In blocks which the terminal sends to the host, bit 7, the end-of-message bit, when set to one, indicates that the terminal has terminated the block because it encountered an *EOM-char* or *EOM-indicator* in the data being sent. When set to zero, this bit indicates that the block was terminated only because the maximum block length was reached, and that another block follows which contains more of the same message.

In blocks sent from the host to the terminal, bit 7 has a different meaning. If the bit is zero, the terminal is requested to acknowledge the block immediately (by sending an ACK block in reply). The terminal sends the ACK block immediately, whether or not it has a message to pack into that block. (The ACK block contains only the four block control bytes.)

If, however, the host sets bit seven to one, then the terminal does not acknowledge the block immediately. Instead, it waits until it has a block full of data to send, or until it encounters an *EOM-char* or *EOM-indicator* in the data it has to send to the host.

When sending a block to the terminal, the host should set bit 7 to zero, except when it expects a response message from the terminal. If a response message is expected, the host should set bit 7 to one.

Control-Byte-2

All the bits of control byte 2 are reserved; they are always zero.

Control-Byte-3 and Control-Byte-4

The last two control bytes carry a "check code" by which the receiving device (the terminal or the host computer) can verify that it has received the block with no errors. The check code is derived from all the unpacked data bytes which precede it : plus the first two control bytes. The process is as follows:

1. Two "checksum bytes" called H and L for this explanation — are both set equal to MaxByte. Here, MaxByte is 127 (for 7-bit bytes) or 255 (for 8-bit bytes).

2. Each byte in the preceding unpacked data is regarded as a binary numeral and added to L. The sum is computed as with "modulo 7 (or modulo 8) end-around-carry." That is, for a 7-bit "unpacked byte size," whenever the sum exceeds the maximum 7-bit numeral (127), the "carry" bit is omitted and one is added to the least-significant bit of the sum. Likewise, if the unpacked byte size is 8-bits, then whenever the sum exceeds 255, the carry bit is omitted and one is added to the least-significant bit of the sum.

   This process is equivalent to the following algorithm, in which MaxByte = 127 (for 7-bit bytes) or 255 (for 8-bit bytes):

   ```
   BEGIN
      L := L + Byte;
      IF (L > MaxByte) THEN L := L - MaxByte
   END
   ```

3. As each byte is added to L, the new value of L is added to H. The same "end-around-carry" method is used:

   ```
   BEGIN
      H := H + L;
      IF (H > MaxByte) THEN H := H - MaxByte
   END
   ```

4. When steps 2 and 3 have been performed for each of the unpacked bytes preceding the check code bytes, then the two check code bytes are computed as follows:

   ```
   BEGIN
      C1 := MaxByte - H - L;
      IF (C1 I 1) THEN C1 := C1 + MaxByte;
      ControlByte3 := C1;
      ControlByte4 := H
   END
   ```

Packing the Control Bytes into the Block

When all four control bytes have been computed, they are packed into the block along with any other unpacked data bytes; see the description of the set-block-packing command for details.

Checking a Received Block

When a block is received and unpacked, the H and L checksum bytes are computed as described above. As each byte is unpacked, the "unpacked byte" is added to L (with end-around carry), and L is added to H (with end-around carry). This is done on all bytes as they are unpacked, including all four control bytes. When the *block-end-char* is detected, H and L should both equal MaxByte. That is, if the unpacked bytes are 7-bit bytes, then H = L = 127; if they are 8-bit bytes, H = L = 255. If this is not the case, then a data transmission error has occurred. (In that case, the terminal or the host receiving the block would re-transmit the last block it had sent.)

REFERENCES

SET-BLOCK-PACKING command

BORDER Key

DESCRIPTION

Pressing this key (SHIFT-NEXTVIEW) changes the border visibility in the current view. If the border was not visible, it becomes visible, and if the border was visible, it becomes invisible.

This key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

SET-BORDER-VISIBILITY command
4128/4129 Operators Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

BREAK Key

DESCRIPTION

Pressing the BREAK key sends a "break" signal to the host. (In some time-sharing systems, the "break" signal is a way for the terminal operator to interrupt the host computer and request the use of the communications line. However, many time-sharing systems do not recognize "breaks".)

If the keyboard is locked, pressing BREAK also unlocks the keyboard. (This can also be accomplished by pressing CANCEL.)

For most host systems, the break signal should last for 200 milliseconds; that is the "default" setting to which this parameter is set when the terminal is shipped from the factory. However, this duration may be changed with the set-break-time command. (Indeed, by setting the break time to zero, the BREAK key can be disabled; this can be useful if the host computer does not tolerate "break" signals.)

If your terminal has Option 4K (the Katakana keyboard) installed, pressing this key causes the ASCII font to become active.

Full Duplex Mode

In full duplex mode, the break signal consists of a "space" condition sent on the TDATA line at the RS-232 connector.

Half Duplex Modes

In "half duplex normal," "half duplex with automatic request to send," and "half duplex with supervisor" modes, the the terminal responds to BREAK by turning off the SRTS (Secondary Request To Send) signal. This is a signal at the RS-232 interface between the terminal and its modem or other data communications equipment. If the modem uses the SRTS signal, then the modem turns off the secondary carrier. The host may or may not respond to the absence of secondary carrier by releasing the communications line so that the terminal may use it to transmit a message to the host.

REFERENCES

SET-BREAK-TIME command

^{B}s Character

DESCRIPTION

The effect of a ^{B}s character depends on whether a dialog area is enabled and on the current coordinate mode of the terminal.

Alpha Mode, Dialog Area Disabled

If no dialog area is enabled and the terminal is in alpha mode and either 2D (coordinate mode 0 or 1) or 3D (coordinate mode 2) mode, then a ^{B}s character moves the alpha cursor one character position to the left. If the terminal is in 2D mode, the 2D beam position is also moved left one space. If the new 2D beam position is to the left of the left edge of the current view's overview window, then:

- if a segment is being defined, no action occurs.
- if no segment is being defined, the alpha cursor and 2D beam position are moved to the right edge of the overview window.

Alpha Mode, Dialog Area Enabled

If a dialog area is enabled and the terminal is in alpha mode, a ^{B}s character moves the cursor one character position to the left. If the cursor is already at the left edge of the dialog area, then ^{B}s has no effect.

Vector and Marker Modes

If the terminal is in vector mode or marker mode, then the ^{B}s character has no effect.

REFERENCES

ENTER-ALPHA-MODE command
ENTER-MARKER-MODE command
ENTER-VECTOR-MODE command

INQUIRY COMMAND

none

CALL-SEGMENT Command

4111, 4115P51, or 4120 SERIES

Host Syntax

^{E}c**SF** int:*segment-number*
xy + :*position*
int:*attributes-flag*

Setup Syntax

^{E}c**SF** ^{S}P *segment-number*
position
attributes-flag

PARAMETERS

segment-number (–3, –1, 1 to 32767)
Specifies the segment to be called.

–3 All segments that match the current matching class and whose dimensionality matches the terminal's coordinate mode

–1 All segments whose dimensionality matches the terminal's coordinate mode

1 to **32767** A specific segment

position
Position of the called segment.

attributes-flag (0 to 3)
The two low bits of this integer parameter are two separate flag bits that control primitive attribute settings. Bit 0 controls saving and restoring primitive attributes. Bit 1 controls primitive attribute default setting.

0 Save primitive attributes before calling segment and restore the saved attributes when segment subroutine returns. Do not reset terminal attributes to default values.

1 Terminal attributes changed by the called segment remain in effect when the segment subroutine returns. Do not reset terminal attributes to default values.

2 Save terminal attributes before calling segment and restore the saved attributes when segment subroutine returns. Reset terminal attributes to default values when the segment is called.

3 Terminal attributes changed by the called segment remain in effect when the segment subroutine returns. Reset terminal attributes to default values when the subroutine segment is called.

DESCRIPTION

This command causes the specified segment's image to be scaled and rotated by the concatenation of transforms shown below and then displayed at the specified position. Primitive attribute settings are changed when the attribute's flag is a value of 1 or 3. If the called segment does not exist at the time of the call, no error is detected. If a panel definition is open when this command is received, the panel is automatically closed.

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist. Segment −1 refers to all currently defined segments (with numbers from 1 to 32767). Segment −3 refers to all segments that match the current segment matching class.

A called 2D segment is scaled and rotated by a concatenation of:

1. the called segment's image transform
2. the scale-rotate of segment −5
3. if a segment is open, the future segment's image transform (segment −2)

A called 3D segment is scaled and rotated by a concatenation of:

1. the called segment's matrix transform
2. the 3 x 3 matrix transform of segment −5
3. if a segment is open, the open segment's matrix transform (for segment −2)

The 2D scale and rotation of segment −5 are set by the SET-SEGMENT-SCALE-ROTATION command. The 3D scale and rotation of segment −5 are set by the SET-SEGMENT-MATRIX-TRANSFORM command. (Segment −5 is the called segment and is used only to set up these transforms.) Note that "nesting" calls to segments (1 calls 2, 2 calls 3, etc.) will result in multiple transforms being applied. The maximum nesting level is 24.

If the dimensionality of the called segment does not match the coordinate mode of the terminal at the time this command is received, an error is detected, nothing is displayed, and the call is not remembered in any segment-definition. If the call is remembered in a segment and the called segment does not exist (or exists and is subsequently deleted) and is later defined, and its dimensionality does not match that of the calling segment, the called segment will not be displayed and no error will occur.

It is possible for a called segment to "inherit" a current primitive attribute at the time of the call under these circumstances:

1. The *attributes-flag* is 0 or 1.
2. At the time the called segment was opened, the primitive attribute had the default setting, and the primitive attribute was not explicitly set prior to the affected primitives.

NOTE

> *If* attribute-flag *is 1 or 3, the status reported by the terminal for the primitive attributes will not necessarily reflect the current attribute settings as changed by the called segment.*

A CALL-SEGMENT command with *attributes-flag* value of **2** results in the same appearance with regard to primitive attributes as an INCLUDE-COPY-OF-SEGMENT command.

Because each call to a segment uses internal stack memory, it is possible for nested calls to segments to overflow the internal stack. This internal stack can support about 40 nested calls. A call to a segment is nested when a called segment calls another segment. When stack overflow occurs an error will be detected. When the error is detected the segment that caused the stack overflow is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when the segment is deleted.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SF11

position
as shipped — none
on power-up — none
if omitted — 0,0

attributes-flag
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

SF02 (Level 3): Not enough memory to make segment reference.

SF03 (Level 2): The specified segment is open or another segment is open for insertion inside a panel.

SF09 (Level 3): Internal stack overflow.

SF11 (Level 2): Invalid *segment-number*.

SF13 (Level 2): The specified segment's dimensionality does not match the current coordinate mode of the terminal.

SF31 (Level 2): Invalid *attribute-flag*.

SO00 (Level 4): Picture processor stack has overflowed (nesting depth up to 24 is valid). Picture processor error 0005 will also be part of the error message.

REFERENCES

INCLUDE-COPY-OF-SEGMENT command
SET-SEGMENT-MATRIX-TRANSFORM command
SET-SEGMENT-SCALE-ROTATION command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

none

CANCEL Command

Host Syntax

$^{E}{}_{C}$KC

Setup Syntax

$^{E}{}_{C}$KC

DESCRIPTION

This command resets to default values a number of terminal parameters and modes:

- Puts the terminal in Alpha mode. (Removes it from Vector or Marker mode.)
- Selects the standard alphatext font. (See SET-ALPHA-TEXT-FONT command.)
- Removes the terminal from GIN mode as if a command to disable all GIN was received. (See ENABLE-GIN and DISABLE-GIN commands.)
- Unlocks the keyboard. (See LOCK-KEYBOARD command.)
- Cancels bypass mode. (See ENTER-BYPASS-MODE command.)
- Flushes input/output queues. (See SET-QUEUE-SIZE command.) Characters not yet sent to the host will be discarded and ignored.
- Removes the terminal from prompt mode. (See PROMPT-MODE command.)
- Halts segment operations being done to segments specified by *segment-numbers* –1 and –3.
- Cancels Snoopy mode when issued from Setup mode.

REFERENCES

CANCEL Key
DISABLE-GIN command
ENABLE-GIN command
LOCK-KEYBOARD command
PROMPT-MODE command
SET-ALPHATEXT-FONT command
SET-QUEUE-SIZE command
SET-SNOOPY-MODE command

INQUIRY COMMAND

none

CANCEL Key

DESCRIPTION

The CANCEL key (the SHIFTed version of the LOCAL key) performs all the functions of the CANCEL command, and causes an exit from Snoopy mode (see the SET-SNOOPY-MODE command description).

Pressing this key also performs the following functions that are not accomplished by the CANCEL command:

- Aborts any file transfer (except spooling) in progress.
- Aborts any color hardcopy (except spooling) in progress. The number of copies numerical indicator remains at the current copy number when the CANCEL key is pressed.

The CANCEL key does not auto-repeat.

REFERENCES

CANCEL command
SET-SNOOPY-MODE command

Char-Array Parameter Type

Syntax

char-array = *int*[*char* . . .

DESCRIPTION

The *char-array* parameter type provides the capability to send a variable number of *char* parameters to various terminal commands. A *char-array* starts with an *int* that specifies the number of *chars* that follow. If the array is terminated with a command terminator (^{E}c, ^{F}s, ^{G}s, or ^{U}s) before the specified number of *chars* have been received by the terminal, the number is reduced to the number actually received. Note that *char-array* has the same syntax as *string* parameter type.

REFERENCES

Char parameter type
Int parameter type
String parameter type

Char-Array-Report Message Type

Syntax

char-array-report: *int-report*[*char-report* . . .

DESCRIPTION

The *char-array-report* is sent from the terminal to the host as part of various status reports.

REFERENCES

Char-report message type
Int-report message type

Char Parameter Type

Syntax

char = a "printing" ASCII character, that is, an ASCII character with ASCII Decimal Equivalent (ADE) ranging from 32 to 126.

DESCRIPTION

The *char* parameter type is used to send individual printing characters to the terminal. Groups of *chars* are sent with *char-array* and *string* parameters.

REFERENCES

Char-array parameter type
String parameter type

Char-Report Message Type

Syntax

char-report = [*EOM-indicator*]
ASCII-char

ASCII-char = (any ASCII character)

DESCRIPTION

The terminal sends *char-report* messages to the host as part of larger report sequences in response to report commands requesting the status of *char* parameters.

The *EOM-indicator* is sent as part of a *char-report* only if both of the following conditions are met:

- The terminal has already sent at least one character since the last *EOM-indicator*;
- If the *EOM-indicator* were not to be sent, then sending the character would cause the maximum line length to be exceeded.

REFERENCES

EOM-indicator syntactic construct

CLEAR Key

DESCRIPTION

The CLEAR key (the SHIFTed version of the DIALOG key) has the same effect as the CLEAR-DIALOG-SCROLL command: it erases the dialog area scroll.

The CLEAR key does not auto-repeat.

REFERENCES

CLEAR-DIALOG-SCROLL command

CLEAR-DIALOG-SCROLL Command

Host Syntax

$^{\mathrm{E}}{}_{\mathrm{C}}$**LZ**

Setup Syntax

$^{\mathrm{E}}{}_{\mathrm{C}}$**LZ**

DESCRIPTION

Erases (clears) the dialog area scroll buffer. The cursor is positioned at the top left of the dialog scroll buffer.

If *dialog-area* –2 (the special 3D Local Viewing System dialog area) is currently enabled on the 4128 or 4129, an error is detected and no action occurs.

ERRORS

LZ03	(Level 2)	Context error. The dialog area scroll buffer cannot be cleared when *dialog-area* –2 is enabled.

REFERENCES

CLEAR key

INQUIRY COMMAND

none

Colorhardcopy-Status-Report Message Type

4111 or OPTION 19

Syntax

```
colorhardcopy-status-report  =  {sig-char}
                                char-report: Q
                                char-report: Q
                                int-report: device-characteristics
                                int-report: device-ID
                                int-report: copier-version-and-option-number
                                int-report: long-axis-device-addressability
                                int-report: short-axis-device-addressability
                                int-report: copies-in-queue
                                int-report: error-code
                                EOM-indicator
```

DESCRIPTION

This report is sent by the terminal in response to a REPORT-COLORHARDCOPY-STATUS command. If a copier is not connected, not powered up, or is busy, all int reports except-*device-ID* will be reported as zero.

Device Characteristics

A 16-bit integer. The individual bits have the following meanings:

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
X	X	X	X	X	X	X	X	S	S	S	MT	CP	PA	PA	IM

X = Reserved for future use.
S = Media size; these three bits specify the size of the media loaded into the copier:

B7	B6	B5	Media Size
0	0	0	A — 8.5″ × 11″
0	0	1	B — 11″ × 17″ ENGLISH
0	1	0	C — 17″ × 22″
0	1	1	D — 22″ × 34″
1	0	0	A4 — 297 × 210 mm
1	0	1	A3 — 420 × 297 mm METRIC
1	1	0	A2 — 594 × 420 mm
1	1	1	A1 — 840 × 594 mm

MT = media type
- 0 Paper
- 1 Transparency

CP = copy process
- 0 Noninterruptible
- 1 Interruptible

PA = number of passes (a copier specification). Add 1 to the value value of the two-bit field (B2 and B1).

IM = imaging/media relationship
- 0 Image parallel to short axis of media
- 1 Image parallel to long axis of media

The 4695 always reports B size media in media type paper regardless of what is actually in the copier.

Device-ID

An integer that identifies the model of color copier. Possible values include 4690, 4691, 4692, and 4695. A value of 4690 is reported if a copier is not connected, not powered up, or is busy.

Copier-version-and-option-number

A four digit integer that, when interpreted hexadecimally, identifies the copier version and option number. The first two digits represent the copier version, and the last two digits represent the option number. For example, if the integer returned is "1000H", then the version number is 1.0, and the option number is 0.0 (i.e., no options installed).

Long-axis-device-addressability

An integer that specifies the maximum pixels the image contains in the direction of the long axis.

4691	A size paper	1559
	B size paper	2459
4692	all paper	1536
4695	all paper	32767

Short-axis-device-addressability

An integer that specifies the maximum pixels the image contains in the direction of the short axis.

4691	A size paper	1199
	B size paper	1559
4692	all paper	1152
4695	width 0	960
	width 1	1026

Copies in queue

The number of copies requested from the copier but not yet copied. The range of this report is 0 through 65535.

Error code

A 16-bit integer. The individual bits have the following meanings when set to one:

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
X	X	X	X	X	LC	BP	BC	TO	CS	IP	EC	DC	FE	MJ	OC

X = Reserved for future use.
LC = Line count error: error in number of lines transmitted per copy.
BP = Bit prompt error: error in number of bit prompts sent in status message.
BC = Byte count error: error in number of bytes sent in Raster Data Transmission.
TO = Time out error: idle time exceeded.
CS = Checksum error: command preamble data string in error.
IP = Inconsistent preamble data.
EC = Unexpected command.
DC = Undefined command.
FE = Fatal device error.
MJ = Media jam.
OC = Out of consumables: i.e., media, toner, ink, etc.

For the 4695 and the 4691 without Option 02 (the multiplexer), the error codes will always be zero.

REFERENCES

REPORT-COLORHARDCOPY-STATUS command

COPY Command

Host Syntax

ᴱcJC device:*source*
string:*separator*
device:*destination*

Setup Syntax

COPY ˢᴘ *source* **TO** *destination*

PARAMETERS

source
Specifies the source of the file being copied. See *Device* parameter type for details.

separator
Must be the empty string or **TO**. In Setup mode, must be **TO**.

destination
Names the destination for the file being copied. See*Device* parameter type for details.

DESCRIPTION

The COPY command transfers a file from the specified *source* to the specified *destination*.

Files are transferred as a sequence of 8-bit bytes; no format conversion is done, nor are embedded commands executed. During the transfer no other processing occurs in the terminal; everything is queued until the COPY process is complete.

The copy operation continues until an end-of-file (EOF) is reached or the CANCEL key is pressed. If you use the CANCEL key, the file is closed and any data already written is saved.

For an explanation of the valid devices and filenames (including syntax), see the discussion of the *Device* parameter type.

For a complete discussion of the transfers performed by this and other file transfer commands, see the *4110/4120 Series Host Programmers Manual.*

DEFAULTS

source
as shipped — none
on power-up — none
if omitted — error JC11

separator
as shipped — none
on power-up — none
if omitted — empty string

destination
as shipped — none
on power-up — none
if omitted — error JC31

ERRORS

JC00 (Level 2): Existence problem in op-code parameter (devices DS: and SG: only).

JC01 (Level 2): Data format error (devices DS:, SG:, and HC: only).

JC01 (Level 2): HC: cannot colormatch with data in old SC: format.

JC02 (Level 3): Out of memory while attempting DMA transfer (device DM: only.)

JC03 (Level 2): Attempt to copy an entire disk volume onto itself. Or HC: Image will not fit.

JC10 (Level 2): Specified *source* does not exist, or cannot be found.

JC11 (Level 2): Invalid *source* specifier.

JC12 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JC12 (Level 3): No low RAM for SC: task.

JC13 (Level 2): Context error (not an input device, or device is busy).

JC19 (Level 2): Disk hardware error or drive not ready on the *source* device, or error in DMA block transfer.

JC21 (Level 2): Invalid *separator* (must be empty string or **TO**, in Setup mode, must be **TO**).

JC22 (Level 3): Out of memory while parsing the parameter.

JC30 (Level 2): Specified *destination* does not exist.

JC31 (Level 2): Invalid *destination* specifier.

JC32 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JC32 (Level 3): No low RAM for HC: task

JC32 (Level 3): No low RAM for Line Out Task.

JC33 (Level 2): Context error. (Invalid destination device, device is busy, or existing disk file is protected.)

JC39 (Level 2): Disk hardware error on the *destination* device, or error in DMA block transfer.

REFERENCES

CANCEL key
Device parameter type
SET-USER-NUMBER command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

none

CR Character

DESCRIPTION

The effect of a CR character depends on whether a dialog area is enabled and on the current coordinate mode of the terminal.

Dialog Area Disabled. When the terminal receives a CR and the dialog area is disabled:

- The terminal enters alpha mode (if not already there)
- 4010-style GIN is disabled (if it is enabled)
- The current line style is reset to line style 0 (solid)

A "carriage return" action is then performed. If the terminal is in 2D mode (coordinate mode 0 or 1), this means moving the 2D beam position and alpha cursor to the left edge of the current view's overview window at the current y location. If the terminal is in 3D mode (coordinate mode 2), this means moving the alpha cursor to the current "carriage return position". The "carriage return position" is set to the 3D graphic beam position whenever the 3D graphic beam position is set, and is changed by LF and VT characters.

If "CR-implies-LF" mode is in effect (see CRLF command), then a "line feed" action is also performed, as if the terminal had received a LF character.

If the Dialog Area is Enabled. If the terminal is not in alpha mode, then the CR character has no effect.

If the terminal is in alpha mode, then the alpha cursor is moved to the left end of the current line. If "CRLF" mode is in effect, then a "line feed" action is also performed.

REFERENCES

CRLF command
ENABLE-DIALOG-AREA command
ENABLE-4010-GIN command
ENTER-ALPHA-MODE command
LF character
SET-LINE-STYLE command

INQUIRY COMMAND

none

CRLF Command

Host Syntax

EcKR int:*CRLF-mode*

Setup Syntax

CRLF SP *CRLF-mode*

PARAMETERS

CRLF-mode (0 or 1)
Indicates whether a carriage return will imply a line feed or not. (Setup mode parameters are **YES** and **NO**.)

1 YES; CR implies LF.

0 NO; CR does not imply LF.

DESCRIPTION

If YES (mode 1) is specified, then when a CR is received by the terminal, the terminal displays it as if CR and LF had been received; that is, the terminal executes both carriage return and line feed.

If NO (mode 0) is specified, when the terminal receives a CR it executes only a CR.

DEFAULTS

CRLF-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

KR11 (Level 2): Invalid *CRLF-mode.*

REFERENCES

CR character
LF character

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

DEFINE-MACRO Command

Host Syntax

EcKD int:*macro-number*
int-array:*macro-contents*

Setup Syntax

DEFINE SP *macro-number* *macro-contents*

PARAMETERS

macro-number (–32768 through –32742, –32740 through –32737, –32608 through –32513, and –178 through 32767)
The number of the macro being defined. Macro numbers –32768 through –32742, –32740 through –32737, and –32608 through –32513 specify "byte-macros", macro numbers –178 through –2, and 0 through 143 specify "key-macros", macro numbers 144 through 32767 specify "host-macros", and a macro number of –1 means "delete all macros".

macro-contents (0 to 127)
An int-array with 0 to 65535 elements, with element values 0 through 127. This parameter defines the characters (using their ASCII decimal equivalents) that result when the macro is expanded. An array of zero length deletes the macro definition for the specified macro number.

DESCRIPTION

The DEFINE-MACRO command defines or deletes a macro (or all macros). There are three types of macros: byte-macros, key-macros, and host-macros. There is no difference in the way that these three macro types are defined or stored, only in the ways they are expanded.

If the macro-number is –1, all macros are deleted, regardless of the macro-contents parameter. To delete a single macro, specify the macro-number and define the macro-contents as a zero length array. For macros –178 to –2 and 0 to 143, deleting the macro definition causes the corresponding key on the keyboard to revert to its "unprogrammed" meaning.

It is possible to save macro definitions specified by this command with the SAVE command. See the description of the SAVE command for details.

The macro-contents may include any combination of ASCII characters, including any 4110/4120 command. Nesting of macros is permitted, so that one macro may expand another.

Byte-macros. Byte-macros are expanded when the terminal's command processor receives a particular character. They have macro-numbers in the ranges –32768 through –32742, –32740 through –32737, and –32608 through –32513. These values correspond to the 8-bit ASCII character which is reached by adding 32768 to the macro-number. The ASCII characters that correspond to these macro numbers have the ASCII decimal equivalents 0 through 26, 28 through 31, and 160 through 255. For example, macro-number –32768 corresponds to $^{\mathrm{N}}\mathrm{L}$, and –32737 corresponds to $^{\mathrm{U}}\mathrm{S}$. The gaps in the range provided for byte-macros are to prevent sensitive characters (such as $^{\mathrm{E}}\mathrm{C}$ and the printable ASCII characters) from being redefined.

To access the byte-macros corresponding to the 8-bit ASCII values 160 through 255 (macro-numbers –32608 through –32513), you must use either 8-bit packing in block mode, or data parity with control over the eighth bit so that you can send the corresponding characters to the terminal command processor.

Key-macros. Key-macros have macro-numbers in the range –178 to –2 and 0 through 143. Macro-numbers 0 through 127 correspond to the ASCII characters (0 through 127) as they are typed on the keyboard. Macro-numbers 128 through 143 correspond to the terminal's function keys: 128 through 135 map onto F1 through F8 (the un-shifted function keys), and 136 through 143 map onto S1 through S8 (the shifted function keys). Macro-numbers –178 to –2 generally correspond to control and control-shift command and control keys (rub out, line feed, PAGE, DIALOG, etc.) and unshifted, shifted, control, and control-shift of the keypad, joystick, Local Viewing System, and mouse keys.

Appendix D contains macro reference tables that list the correspondence between macro numbers and keyboard keys for the serial interface keyboard (the one with the keypad).

Host-macros. The range of macro-numbers for host-macros includes the entire range of macro-numbers. Thus, macros –178 to –2 and 0 to 143 are both host-macros and key-macros, and macros with numbers less than –32512 are both host-macros and byte-macros. Host macros are defined as macros that can be expanded by the EXPAND-MACRO command only or by the EXPAND-MACRO command as well as by other methods.

Expanding Macros. Byte-macros and host-macros are expanded with commands from the host, keyboard, or file being loaded. Key-macros are expanded with commands from these sources and with keystrokes. See the EXPAND-MACRO command description for details on expanding macros.

DEFAULTS

macro-number
as shipped — none
on power-up — none
if omitted — 0

macro-contents
as shipped — none
on power-up — none
if omitted — empty array

ERRORS

KD11	(Level 2):	Invalid *macro-number.*
KD21	(Level 2):	Invalid *macro-contents.*
KD22	(Level 3):	Insufficient memory to define macro.

REFERENCES

EXPAND-MACRO command
LOAD command
SAVE command

INQUIRY COMMAND

none

$^{D}{}_{T}$ Character

DESCRIPTION

If the terminal is in snoopy mode, $^{D}{}_{T}$ is displayed as a filled in character space. If the terminal is in Alpha mode, $^{D}{}_{T}$ has no effect on the terminal.

If the terminal is in Vector or Marker mode, $^{D}{}_{T}$ is interpreted as a LoY or Extra byte for an *xy* parameter. If the terminal is parsing an *int* parameter, $^{D}{}_{T}$ is interpreted as a HiI byte.

This can present a problem if a host computer also intersperses $^{D}{}_{T}$ as a filler character. These filler characters could be misinterpreted by the terminal as meaningful data.

Two terminal features are provided to overcome this difficulty. Firstly, the IGNORE-DEL command lets you specify that the terminal is to ignore any $^{D}{}_{T}$s which the host may send as filler characters. Secondly, the terminal interprets the delete-equivalent string, $^{E}{}_{C}$**?**, as if it were the $^{D}{}_{T}$ character.

REFERENCES

Delete-equivalent syntactic construct
IGNORE-DEL command
Int parameter type
XY parameter type

DELETE-DIALOG-AREA Command

4111, 4115P51, or 4120 SERIES

Host Syntax

$^{E}{}_{C}$**KK** int:*dialog-number*

Setup Syntax

DADELETE $^{S}{}_{P}$ *dialog-number*

PARAMETERS

dialog-number (–1, 0, and 2 to 64)
Number of dialog area to be deleted.

–1	all dialog areas except dialog area 1 and –2.
0	the current dialog area unless the current dialog area is 1, 0, or –2.
2 to **64**	the number of the dialog area to be deleted.

DESCRIPTION

This command deletes the designated dialog area identified by *dialog-number*. Deletion of dialog area 1 or –2 is not permitted.

Dialog-number –1 deletes all dialog areas 2 to 64 and enables dialog area 1 if a dialog area was enabled when the command was issued.

Dialog-number 0 deletes the current dialog area if it is 2 to 64 and issues an error if the current dialog area is 0, 1, or –2.

The highest priority remaining dialog area (other than –2) becomes the current dialog area when the current dialog area is deleted. If there are no other (non –2) dialog areas, dialog area 1 becomes the current dialog area.

An error is issued if the specified dialog area does not exist.

DEFAULTS

dialog-number
as shipped — none
on power-up — none
if omitted — 0

ERRORS

KK00	(Level 0):	Command not recognized by terminal.
KK10	(Level 1):	Specified *dialog-number* does not exist.
KK11	(Level 2):	Invalid *dialog-number.*

REFERENCES

ENABLE-DIALOG-AREA command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

Delete-Equivalent Syntactic Construct

Syntax

^{E}C**?**

DESCRIPTION

The delete-equivalent string, ^{E}C**?**, is interpreted as replacing a ^{D}T. This feature, together with the ignore-DEL command, lets you deal with host computers which intersperse ^{D}T filler characters among the characters they send to the terminal. A host can substitute the Delete-equivalent whenever it would ordinarily send a ^{D}T.

REFERENCES

^{D}T character
Int parameter type
XY parameter type

DELETE-FILE Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

```
EcJK   device:file-name
```

Setup Syntax

```
DELETE Sp  file-name
```

PARAMETERS

file-name

Specifies the file to be deleted. See *Device* parameter-type for valid disk file specifiers.

DESCRIPTION

This command deletes the specified file from the specified disk drive.

If the file name is a file that has been write-protected with the PROTECT command or that is on a write-protected disk, the terminal detects error JK13 or error JK19.

DEFAULTS

file-name

as shipped — none
on power-up — none
if omitted — error JK11

ERRORS

JK00	(Level 0):	Unrecognized command. (Disk drive option is not installed.)
JK10	(Level 2):	The specified file or disk drive does not exist.
JK11	(Level 2):	Invalid *file-name*.
JK12	(Level 3):	Out of memory while parsing parameter.
JK13	(Level 2):	The specified device is not a disk drive, is write-protected, is busy, or detects a bit map error.
JK19	(Level 2):	Disk hardware error.

REFERENCES

Device parameter type
PROTECT command

INQUIRY COMMAND

none

DELETE-GRAPHTEXT-CHARACTER Command

Host Syntax

$^{E}{}_{C}$**SZ** int:*font-number* int:*char-number*

Setup Syntax

$^{E}{}_{C}$**SZ** $^{S}{}_{P}$ *font-number* *char-number*

PARAMETERS

font-number (–1 to 32767)
Specifies the font from which the character is deleted.

–1	all fonts
0 to **32767**	specifies a font

char-number (–1, or 32 to 126)
The numeric equivalent of a character within the specified font.

–1	all characters
32 to **126**	specifies a character

DESCRIPTION

The named character of the specified user-defined graphtext font is deleted. Using a number of "–1" deletes all items of that nature, as follows:

font-number = –1 : deletes all user-defined characters in all fonts.

char-number = –1 : deletes all user-defined characters in the specified font.

If you specify a *font-number* of –1, the *char-number* parameter is ignored.

If the specified character number does not exist in the specified font, a type SZ20 error is detected.

When a user-defined graphtext character is deleted, it is superseded by the corresponding predefined character for that font. For most fonts, that is the same as the corresponding character in the standard ASCII font (font 0). Fonts 1, 3, 7, 9, 10, or 11, correspond to characters used by the U.K., Swedish, APL, Danish/Norwegian, or Katakana fonts. See the SET-GRAPHTEXT-FONT command.

Specifying a *font-number* of –1 not only deletes all characters in all user-defined fonts, but also deletes all user-defined graphtext font grid definitions and resets the current graphtext font (as set by the SET-GRAPHTEXT-FONT command) to 0. Fonts numbered 0, 1, 3, 7, 9, 10, and 11 — the predefined fonts — cannot be deleted. If you try to delete these fonts, error SZ11 is generated.

Specifying a *char-number* of –1 not only deletes all the characters in the specified font, but also deletes the graphtext font grid for that font.

If the *font-number* specified is the current graphtext font, and the *char-number* is –1, then the current graphtext font is reset to 0.

DEFAULTS

font-number
as shipped — none
on power-up — none
if omitted — 0

char-number
as shipped — none
on power-up — none
if omitted — 0

ERRORS

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*.

SZ20 (Level 1): The character specified does not exist in this font.

SZ21 (Level 2): Invalid *character-number*.

REFERENCES

SET-GRAPHTEXT-FONT command
SET-GRAPHTEXT-FONT-GRID command

INQUIRY COMMAND

none

DELETE-PART-OF-SEGMENT Command

4111, 4115P51 or 4120 SERIES

Host Syntax

$^{E}_{C}$UD int:*segment-number* int:*pickIDfirst*
int:*pickIDlast*

Setup Syntax

$^{E}_{C}$UD $^{S}_{P}$ *segment-number* *pickIDfirst*
pickIDlast

PARAMETERS

segment-number (1 to 32767)
Segment number from which primitives and primitive attributes are to be deleted.

1 to **32767**	the segment number

pickIDfirst (–1 and 1 to 32767)
The first pickID group to be deleted

–1	the segment end
1	the segment beginning
2 to **32767**	a particular pickID group

pickIDlast (–1 and 1 to 32767)
The last pickID group to be deleted

–1	the segment end
1	the segment beginning
2 to **32767**	a particular pickID group

DESCRIPTION

This command deletes pickID groups from an existing segment whose dimensionality matches the current coordinate mode of the terminal.

A pickID group is defined as a sequence of primitives and primitive attributes that follow a pick-ID-number, up to, but not including the next pick-ID-number (other than 0 and a repetition of the same pick-ID-number).

If the segment part being deleted contains calls to segments (by the CALL-SEGMENT command), only the call is deleted, not the contents of the called segment.

The command selects the first occurrence of a pick-ID-number if it is used more than once in a segment definition.

To delete a single pickID group, enter the same pickID group number in both the *pickIDfirst* and *pickIDlast* parameters.

To delete the segment contents but not the segment, enter *pickIDfirst* **1** and *pickIDlast* **-1**.

Errors are reported if:

- *pickIDfirst* does not exist in the segment
- *pickIDlast* does not exist or does not follow *pickIDfirst* in the segment
- *pickIDfirst* is found before an included segment and-*pickIDlast* is found within the included segment (See the INCLUDE-COPY-OF-SEGMENT command for details.)
- *pickIDfirst* is found within an included segment and-*pickIDlast* is found after the included segment
- *pickIDfirst* is found before a panel definition and *pickIDlast* is found within a panel definition
- *pickIDfirst* is found within a panel definition and *pickIDlast* is found after a panel definition
- *pickIDfirst* is found within the contents of an included segment which had a non-default image transformation when it was included

The position and/or primitive attributes of the trailing portion of the segment (that part of the segment following the edit point) can be altered after deleting the specified pickID groups. Refer to the SET-SEGMENT-EDIT-MODE command for specific details.

The image of the edited segment will not change until the segment is redrawn.

If the current coordinate mode of the terminal does not match the dimensionality of the segment, an error is detected and nothing is deleted.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error UD11

pickIDfirst
as shipped — none
on power-up — none
if omitted — error UD21

pickIDlast
as shipped — none
on power-up — none
if omitted — error UD31

ERRORS

UD02	(Level 3):	Out of memory.
UD03	(Level 2):	Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.
UD10	(Level 2):	Specified segment does not exist.
UD11	(Level 2):	Invalid *segment-number.*
UD13	(Level 2):	The specified segment has a dimensionality attribute which does not match the coordinate mode of the terminal.
UD20	(Level 2):	Cannot find first *pickIDfirst.*
UD21	(Level 2):	Invalid *pickIDfirst.*
UD23	(Level 2):	Invalid because of transformations in the included segment.
UD30	(Level 2):	Cannot find *pickIDlast.*
UD31	(Level 2):	Invalid *pickIDlast.*
UD33	(Level 2):	Invalid because of included segment or panel begin/end boundaries.

REFERENCES

CALL-SEGMENT command
INCLUDE-COPY-OF-SEGMENT command
SET-SEGMENT-EDIT-MODE command

INQUIRY COMMAND

none

DELETE-SEGMENT Command

Host Syntax

$^{E}{}_{C}$**SK** int:*segment-number*

Setup Syntax

$^{E}{}_{C}$**SK** $^{S}{}_{P}$ *segment-number*

PARAMETERS

segment-number (–3, –1, or 1 to 32767)
The number of the segment to be deleted.

–3	all segments that match the current matching class
–1	all segments
1 to **32767**	a specific segment

DESCRIPTION

This command removes the specified segment from memory. If the segment is currently being defined, the segment definition is terminated, and then the segment is deleted.

The display is updated to the extent specified by the most recent SET-FIXUP-LEVEL command.

If the parameter is –1, all segments are deleted. (This does not include segment 0, the cross-hair cursor, which cannot be deleted.)

If the parameter is –3, then all segments that match the current matching class are deleted.

Segments cannot be protected against deletion. Before issuing the DELETE-SEGMENT command, take care that the command is really needed and that the segment-number being deleted is correct.

If the segment you want to delete is assigned as the GIN-cursor, and GIN is enabled, the terminal reports error SK13 and the segment is not deleted. If you specified –1 or –3 as the *segment-number* parameter, all segments except the GIN-cursor are deleted, and error SK13 is reported.

If all segments are to be deleted, it is faster to set the *fixup-level* to 0, delete the segments, renew the view, and then restore the original *fixup-level.*

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SK11

ERRORS

SK02	(Level 3):	Out of memory while attempting to delete a segment.
SK10	(Level 1):	Segment does not exist.
SK11	(Level 2):	Invalid segment number.
SK13	(Level 2):	Segment specified is an active GIN-cursor.

REFERENCES

SET-FIXUP-LEVEL command
SET-GIN-CURSOR command

INQUIRY COMMAND

none

DELETE-VIEW Command

Host Syntax

$^{E}_{C}$**RK** int:*view-number*

Setup Syntax

$^{E}_{C}$**RK** $^{S}_{P}$ *view-number*

PARAMETERS

view-number (–1 to 64)
Specifies the view to be deleted.

–1	all views
0	the current view
1 to **64**	specifies a view

DESCRIPTION

This command deletes the specified view. If *view-number* is 0, the current view is deleted. If *view-number* is –1, all views are deleted.

When all views are deleted, the visibility of all existing segments is turned off.

If the current view is deleted, the next view higher (or the lowest remaining view if none higher exist) becomes the current view. If there are no other views (e.g., after *view-number* = –1), then view 1 is selected with all the viewing parameters set to their power-up values. See the SELECT-VIEW command for a list of the viewing parameters.

DEFAULTS

view-number
as shipped — none
on power-up — none
if omitted — 0

ERRORS

RK10 (Level 2): The designated view does not exist (has not been defined with the SELECT-VIEW command).

RK11 (Level 2): Invalid *view-number*.

REFERENCES

SELECT-VIEW command

INQUIRY COMMAND

none

Device Parameter Type

Host Syntax

$$\text{device} = \text{string:} \left\{ \begin{array}{l} \textit{device-name} \\ \textit{device-name filename} \\ \textit{device-name parameter} \\ \textit{filename} \end{array} \right\}$$

Setup Syntax

$$\text{device} = \left\{ \begin{array}{l} \textit{device-name} \\ \textit{device-name filename} \\ \textit{device-name parameter} \\ \textit{filename} \end{array} \right\}$$

DESCRIPTION

Device parameters are used in file transfer commands (COPY, DIRECTORY, LOAD, PLOT, PORT-COPY, SAVE, and SPOOL) and in the REPORT-DEVICE-STATUS command. Device parameters are also used in commands that set attributes for the ports on Option 10.

Device parameters are sent as string parameters from the host. In Setup mode, no leading or trailing delimiters are needed (as are needed for normal string parameters). During all file transfer operations (except a spooling operation not involving the host port), the keyboard is disconnected. If a key is pressed during the transfer operation, its data goes into a 170-byte queue until the transfer is complete. When the queue becomes full, the terminal bell rings each time a key is pressed, and the terminal discards the character generated by that key.

Valid *device* parameters are:

- *device-name*
- *device-name* with a *filename*
- *device-name* with a *parameter* character(s)
- *filename*

The recognized *device-names* depend on the installed options, as shown in Table 7-2A:

Table 7-2A

RECOGNIZED DEVICE-NAMES

Device	*device-name*	Used with
Alpha display device	null-string	Opt 42,43,45
Color copier interface	HC:	4111; Opt 19
Color copier screen device	SC:	4111; Opt 19
Disk drive 0	F0:	Opt 42 or 43
Disk drive 1	F1:	Opt 43
DMA interface	DM:	4111; Opt 3A or 3C
External disk drive	S0: — Z7:	Opt 45
Host computer port	HO:	All
Internal devices	CM:, DS:, PX:, SG:	4111; 4115 with Opt 3A or 3C; 4125; 4128; 4129
Peripheral ports	P0:, P1:,	4111
Peripheral ports	P0:, P1:, P2:	Opt 10

Device-Name. A *device-name* consists of three characters. The first two characters must be any ASCII characters except ^{S}P, *, :, ;, <, = , >,?, ,, { , } , and _. The third character must be a colon (:). Lowercase characters ("a" to "z") are translated to uppercase and can be used interchangeably.

Filename. A *filename* consists of two parts, the name and the extension. The name can be from one to eight characters long, and the extension can be from one to three characters long. If you include an extension, it must be separated from the name with a period (.). All ASCII characters are valid except ^{S}P, *, ., :, ;, <, = ,>, ?, ,, { , } , and _. Control characters are ignored when they are received from the host and are not valid when entered in Setup mode. If extra valid characters are added to the name or the extension, an error is detected. Lowercase characters ("a" to "z") are translated to uppercase ("A" to "Z") and can be used interchangeably.

Parameter. Some devices have parameters. These parameters specify special action for the device. They can be any string of characters that are valid for *filename*. See the explanations of the different devices later in this description for the parameters and their meanings.

Alpha Display

The alpha display device is specified as a null-string, and is valid only as the *destination* in the DIRECTORY command. It treats the contents of the directory listing as alphatext with CRLF on, regardless of the current terminal mode or CRLF-setting.

Color Copiers

The 4111 or Option 19 support the 4690 Series Color Graphics Copiers on a 4111 or, if Option 19 is installed, two additional devices, HC: and SC:, are available.

HC: is valid only as a destination device for the COPY and SPOOL commands. It specifies the color hardcopy interface that is installed as part of the 4111 or of Option 19. There are three valid parameter characters for this device. HC:0 and HC: specify that black is copied as white on the color hardcopy device. HC:1 specifies that black is copied as black on the color hardcopy device. HC:2 specifies that the data will be interpreted as normal text and special copier commands as described in the following manuals:

Device Driver Development Guide for 4691 Color Graphics Copiers

4695 Device Driver and Interface Development Guide

In a file transfer to HC: (HC:0, HC:1), the pixel viewport area is centered and expanded up to four times to fit as much of the copier's maximum image area as possible. If the pixel viewport is larger than the copier's maximum image area, then an error is detected. Note that the default pixel viewport is larger than some of the maximum image areas, and must be made smaller to avoid this error.

SC:, a pseudo device, is valid as a source device for the COPY command only. It specifies the contents of the terminal's pixel viewport.

In Option 19, firmware version 6, there are two valid parameter characters for the SC: device. The parameter character determines the action of the SET-COLOR-COPIER-DATA-RESOLUTION command on the format of the copied image when dithering is enabled. SC:0 and SC: format the pixel viewport image with either one or two byte color resolution, depending on the SET-COLOR-COPIER-DATA-RESOLUTION setting. SC:1 ignores the SET-COLOR-COPIER-DATA-RESOLUTION setting; it formats and sends the image at full color resolution. An error will be generated if SC:0 or SC: is used when color matching is enabled.

Data from SC: is in a very strict format, and HC:, HC:0, and HC:1 accept data only in the format generated by SC:. An error is detected for data in any other format.

Disk Drives (Internal and External)

If Option 42 is installed in your terminal, device F0: addresses the single disk unit. If Option 43 is installed, F0: addresses the right disk unit and F1: addresses the left one. External Option 45 disk drive devices are addressed with the first character of S, T, U, V, W, X, Y, or Z and the second character 0, 1, 2, 3, 4, 5, 6 or 7. The specific device name used depends on how the option hardware was set at installation.

Disk drive devices can include a *filename* after the three character device specifier. Do not include a space between the last character of the *device-name* (:) and the first character of the *filename*.

If only a *filename* is used without the three character device specifier, the device F0: is assumed, when Option 42 or 43 is installed. If neither Option 42 or 43 is installed, but an external disk drive is installed, the assumed drive is the first device attached to the Option 45 interface.

All disk drive devices are valid as *source*s and *destination*s of all file transfer commands, except PORT-COPY. Only files with the current user-number can be accessed as *source*s. *Destination* files are given the current user-number. If a *destination* file already exists, it is deleted at the start of the file transfer unless it is protected.

All data is stored on disk files as eight-bit bytes. If the file came from the host port, the high bit in each byte is the parity bit as received from the host.

DMA Interface

The DMA interface device (DM:) is standard on the 4111 and is available when Option 3A or 3C is installed in other terminals. It can be specified as a *source* or as a *destination* in all file transfer commands except PORT-COPY. If DM: is selected as the *source* device, the direction of DMA transfer is from the host to the workstation. If DM: is selected as the *destination* device, the direction of DMA transfer is from the workstation to the host.

The DMA transfer process and the host computer CPU operate on alternate cycles of the host computer clock. The transfer does not lock up the host system bus. In addition, using DM: does not cancel RS-232 communications with the host. RS-232 command are held in a queue and are executed at the end of the transfer. Additional information about DMA transfer is found in the *4110/4120 Series Host Programmers Manual.*

Host Port

The host communication port (HO:), is a standard device, available on all terminals. The host port is valid as *source* and *destination* for all file transfer commands.

When HO: is used as a *source* device in a file transfer operation, the terminal sends all characters that it receives at the host port after the operation is started to the destination device. The terminal continues to send until it detects an End-Of-File (EOF). The EOF is either an *EOF-string* as set by a SET-EOF-STRING command if the terminal is not in Block mode, or an end-of-file-control-bit if the terminal is in Block mode. After the EOF, all further characters are routed to the terminal's command processor as usual.

When HO: is the *destination* device in a data transfer, the contents of the file are routed through the terminal's report sending system. When the terminal detects an EOF from the source device during the transfer, if block mode is not in use, it sends an *EOL-string, EOF-string, EOL-string* sequence to the HO: port to terminate the transfer, if block mode is in use, the end-of-file-control-bit is set and the block is sent.

Internal Devices

The internal devices are CM:, DS:, PX:, and SG:. On the 4111, 4125, 4128, and 4129 these devices are standard. See Appendix F for a description of the op-codes.

CM:. CM: specifies the color map. It is valid as both *source* and *destination* with the COPY command only. If CM: is the source device, the terminal sends the entire color map. If CM: is the destination device, the terminal accepts either the whole color map or part of the color map.

CM: requires that data be in a special form. The entire color map of 256 colors may be specified in 768 8-bit bytes. Each color consists of three bytes, with each byte representing the intensity of the three primary colors: red, green, and blue. Figure 7-2 illustrates the data stream for the color map.

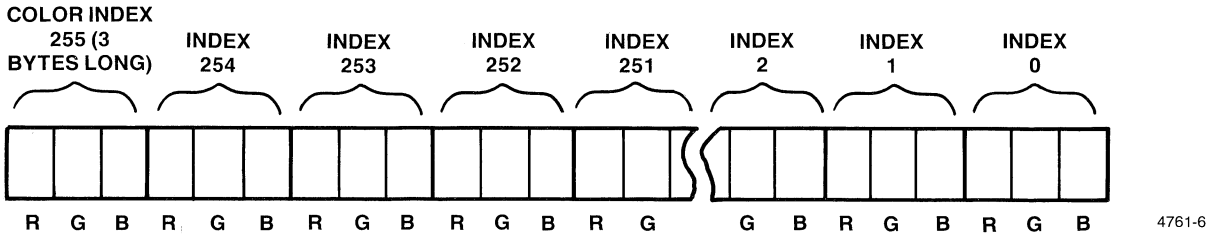


Figure 7-2. CM: Data Stream.

When CM: is the destination device, the color map is updated only for the data received, starting with the highest index, 255. So if only five bytes were received, color index 255 and the red and green components of index 254 would be updated.

There are no parameters for CM:.

DS:. DS: specifies a list of picture processor instructions (called a display list) which are processed by terminal hardware to create a viewable display. It is valid as a *destination* device with the COPY command only. Using this device does not retain the display list in terminal memory.

DS: makes the current viewport available to the terminal file system. When DS: is specified, the current beam position, matrix transform, and primitive attributes are automatically saved with a PUSHBC instruction (see Appendix F for an explanation of this instruction). DS: then receives a list of picture processor op codes, which it sends to the picture processor. The picture processor performs the operations indicated by the instructions, resulting in a picture in the current viewport.

When the transfer of data to DS: is complete, the beam position, transform, and primitive attributes are restored to their previous values, and other screen activity is allowed to continue. Because of the saving and restoring of attributes, those set during the transfer will not necessarily be in effect when the transfer is completed.

When using DS: you may first want to set up a viewport and window.

Data received by DS: must be in the format shown in Figure 7-3. Not all picture processor op codes are allowed by DS:. The allowed op codes are limited to graphic primitives (Moves, Draws, Alpha-text, Markers, and Panels) and primitive attributes (line style, etc.).

DS: reads each op code, and then, because each op code has a specific number of bytes that follow, DS: skips that number of bytes before looking for another op code to indicate the start of the next instruction.

DS: has no parameters.

PX:. PX: specifies the current pixel viewport as a *source* and *destination* device for COPY command only. This device allows information on the screen to be directly transferred to another device or information from another device to be directly displayed on the terminal screen.

Before you specify PX: as a device, you may want to set up a non-default pixel viewport.

If PX: is the *source* device, all pixels from the current pixel viewport are read by the method indicated by the accompanying *parameter*, and then sent to the destination device. All the pixels are read, starting in the home position and reading in the normal writing direction. The beam position is left just beyond the read position at the end of this command.

If PX: is the *destination* device, the incoming pixels are decoded as they are received and written, by the method indicated by the device parameter, to the current pixel viewport. Writing starts at the current pixel beam position with pixels written in the normal writing direction. This continues as pixels are received, wrapping to the home position as necessary, until the last block of data is received. After receiving the last block, the beam position is left at the pixel following the last pixel write.

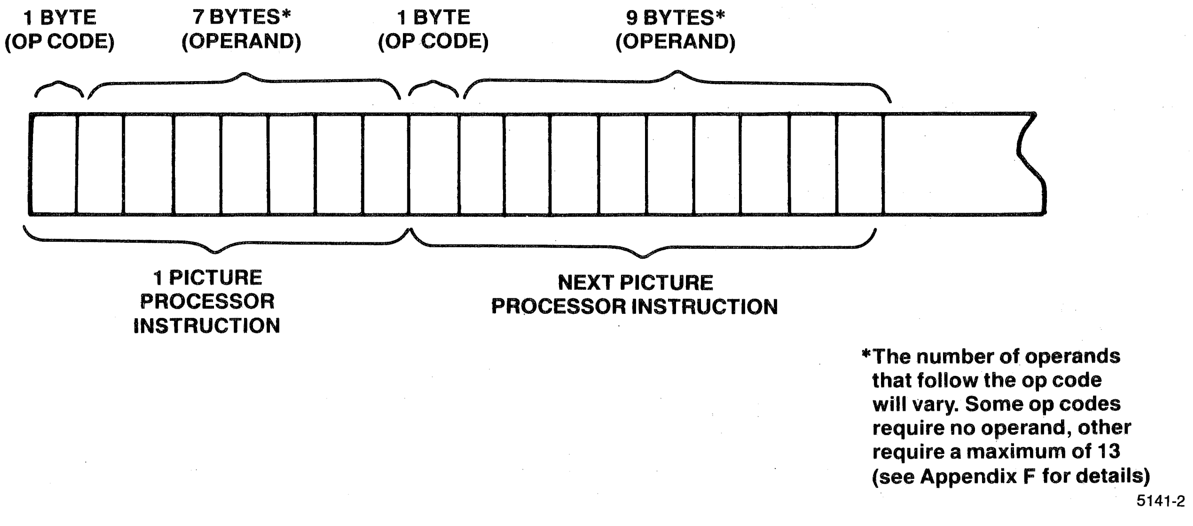


Figure 7-3. DS: Data Stream.

You can specify whether you want the pixel data (either-*source* or *destination*) to be *raw* or *normalized*, and whether you want it *runlength encoded* or *non-runlength encoded* by adding a parameter after the *device-name*.

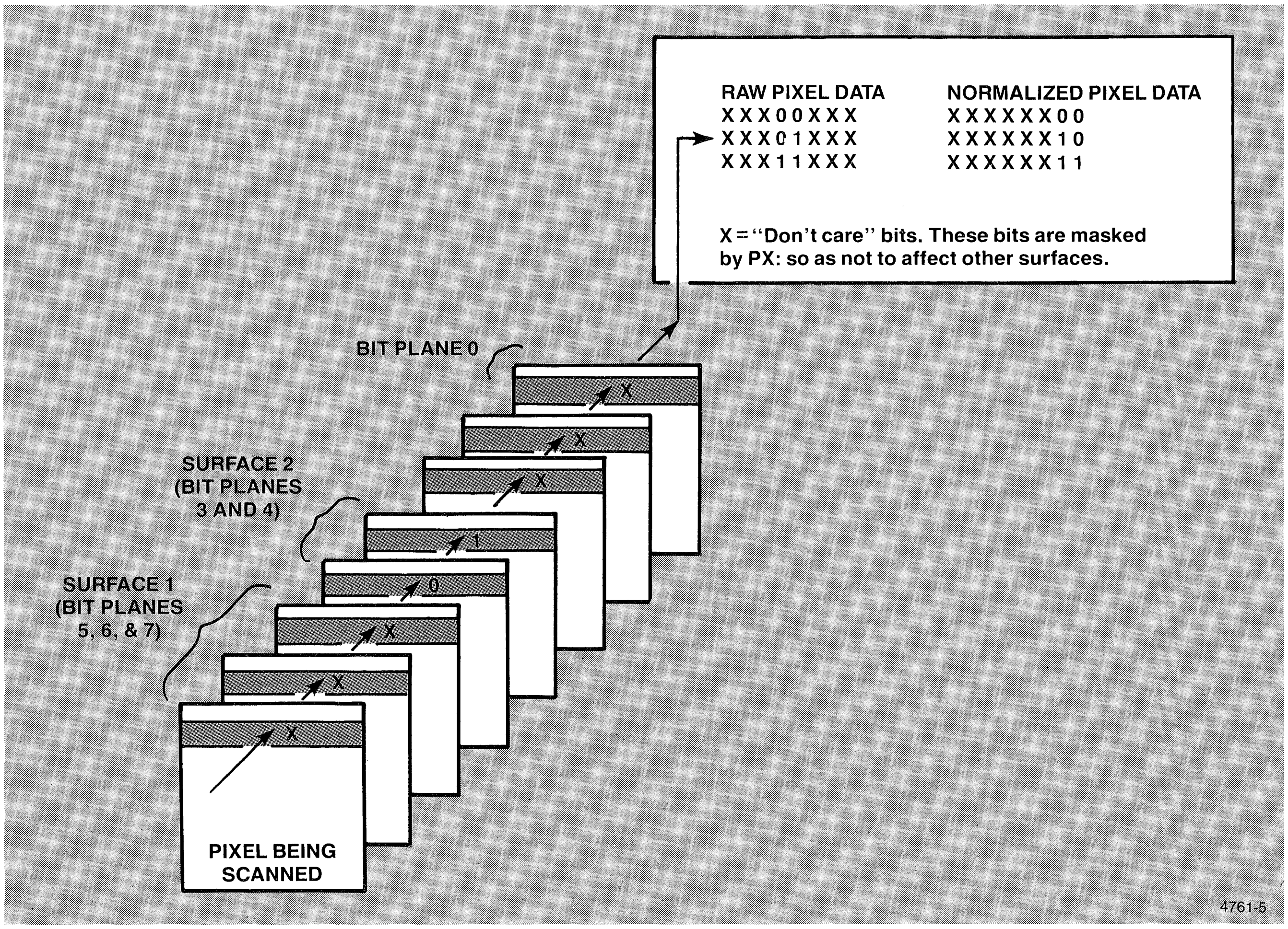
The parameter defines four ways to code pixel data for sending or decoding pixel data for display. The four types of pixel data formats are:

- Raw, Non-Run-Length-Encoded Pixel Format
- Normalized, Non-Run-Length-Encoded Pixel Format
- Raw, Run-Length-Encoded Pixel Format
- Normalized, Run-Length-Encoded Pixel Format

These terms are defined in the following paragraphs.

Raw. The valid bits of pixel data byte for current viewport surface correspond to the bit planes that are being used for that surface. For example, if Bit Planes 3 and 4 are used in Surface 2, only Bits 3 and 4 are valid bits in the pixel data byte. Figure 7-4 illustrates this.

Normalized. The valid bits for the current pixel viewport surface are always the lowest order bits in the pixel data byte. Unused bits are always in the highest order bits of the data byte. Refer to Figure 7-4 for an example of normalized pixel data.



RAW PIXEL DATA	NORMALIZED PIXEL DATA
XXX00XXX	XXXXXX00
XXX01XXX	XXXXXX10
XXX11XXX	XXXXXX11

X = "Don't care" bits. These bits are masked by PX: so as not to affect other surfaces.

BIT PLANE 0
SURFACE 2 (BIT PLANES 3 AND 4)
SURFACE 1 (BIT PLANES 5, 6, & 7)
X
X
X
1
0
X
X
X
PIXEL BEING SCANNED
4761-5

Figure 7-4. Raw and Normalized Pixel Data.

Non-Run-Length-Encoded. Each pixel data byte sent or received, whether it is raw or normalized, represents one pixel on the screen (see Figure 7-5).

Run-Length-Encoded. Each three bytes sent or received tells how many pixels to fill with the specified color. The first two bytes are the number of pixels to fill. The third bytes is the pixel data byte. The data stream for run-length-encoded pixel information is shown in Figure 7-6.

The parameters that specify the different forms of data are:

0	raw, non-runlength encoded data
1	normalized, non-runlength encoded data
2	raw, runlength encoded data
3	normalized, runlength encoded data

The power-up default parameter specifies raw, non-runlength encoded data (0). If you do not include a parameter, the last parameter specified is used; or, if none has been previously specified, the power-up default is used.

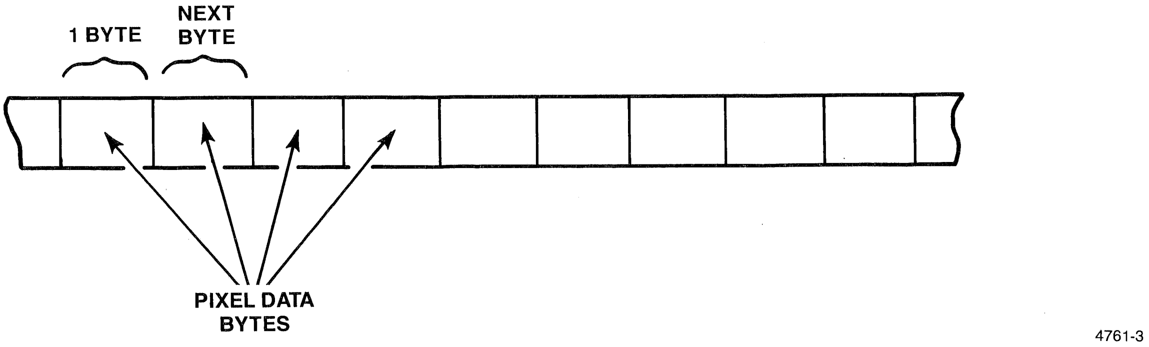


Figure 7-5. Data Stream for Non-Run-Length-Encoded Pixel Data.

SG:. SG: specifies the retained segment list as a *source* or *destination* device for the COPY command only. When SG: is the source device, individual segments, a matching class of segments, or all segments are read from terminal memory and transferred to the *destination* device. When SG: is the *destination* device, the contents of the data stream are put into the segment list in terminal memory. Any number of segments may be received in one transfer. SG: is similar to DS: except that the data received by the terminal is retained for later use rather than being immediately processed and displayed. All picture processor instructions that are legal in a DS: data stream are also legal in a SG: data stream.

Individual segments have certain attributes associated with them, including segment visibility, segment class, and segment priority. When SG: is the source device, these attributes are initially set to the default segment (Segment −2) values for all segments created. (These are not necessarily the values of previously displayed segments.) It is not possible to change segment attributes during the transfer; however, they can be changed using 4120 Series commands after the transfer is complete.

If a segment, panel, or graph-text character is open when SG: is specified as a *destination* device, SG: automatically closes the item before allowing new segments to be created.

The data stream for 2D coordinate mode SG: consists of three parts for each segment (see Figure 7-7, Part A). Part B of Figure 7-7 shows the data stream for 3D coordinate mode.

- A header which has the number of the segment and up to five other special instructions
- The list of picture processor instructions required to define the segment (the display list)

NOTE

If a segment will be transformed (that is, translated, scaled, rotated, or 3D skewed), all moves and draws after the required AMOVE in the header must be relative rather than absolute.

- A return code (hexadecimal 13), which also serves as a delimiter between segments

If another segment is sent, it must immediately follow the return code for the preceding segment.

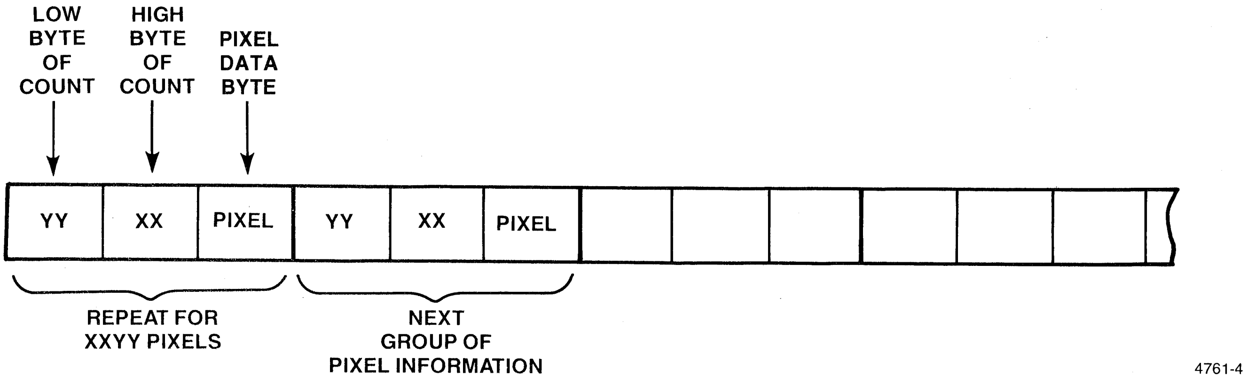
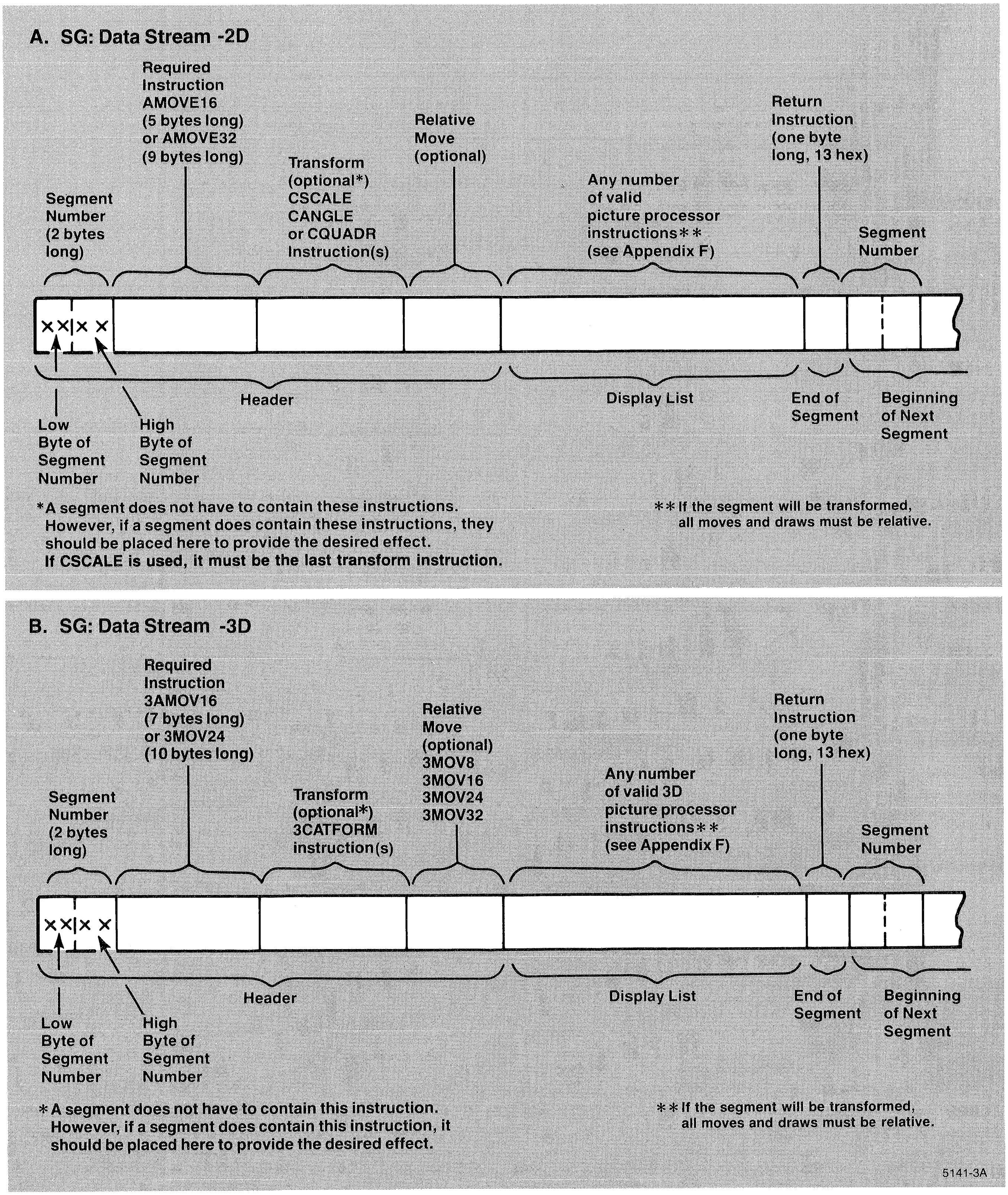


Figure 7-6. Data Stream for Run-Length-Encoded Pixel Data.



A. SG: Data Stream -2D

*A segment does not have to contain these instructions. However, if a segment does contain these instructions, they should be placed here to provide the desired effect. **If CSCALE is used, it must be the last transform instruction.**

** If the segment will be transformed, all moves and draws must be relative.

B. SG: Data Stream -3D

* A segment does not have to contain this instruction. However, if a segment does contain this instruction, it should be placed here to provide the desired effect.

** If the segment will be transformed, all moves and draws must be relative.

Figure 7-7. SG: Data Stream.

The information in the header must appear in the order shown in Figure 7-7. Notice, however, that some of the information in the header is optional, and does not have to be included if the segment does not require it.

Valid parameters are:

–1	Specifies all segments (equivalent to *segment-number* –1 in segment commands).
–3	Specifies all segments that match the current matching class (equivalent to *segment-number* –3 in segment commands).
1 to **32767**	Specifies the individual segments of the given number.

If you specify –1 or –3 as the parameter, segment visibility, priority, and class are set the same as those of the default segment (segment number = –2). This is because it is not possible to set these attributes for each of multiple segments sent in one transfer.

Peripheral Ports

On the 4111, two ports, P0: and P1:, are available for connecting RS-232C compatible peripherals. On the 4115/4120 Series, if Option 10 is installed, three ports, P0:, P1:, and P2: are available. The terminal contains several device drivers for different peripherals which may be connected. The peripherals for which the terminal has device drivers are:

- TEKTRONIX 4662 Interactive Digital Plotter
- TEKTRONIX 4663 Interactive Digital Plotter
- TEKTRONIX 4643 Line Printer
- TEKTRONIX 4957/4958 Graphics Tablet

The ports may also be used for any compatible device, such as the TEKTRONIX 4923 Cartridge Tape Unit.

The peripheral port devices are valid as *sources* and *destinations* for all file transfer commands.

The data transferred to and from a peripheral port is formatted according to the current settings for that port. If a plotter or printer device driver is assigned to a port, that port cannot be used as a source for any file transfer. If you specify a port assigned either the plotter or printer device driver as a source, an error is detected.

When an EOF is detected in a transfer to a peripheral port, the *port-EOF-string* is sent to the attached peripheral and the transfer is terminated. The *port-EOF-string* is also used to mark the end-of-file for a transfer from a peripheral port. Any characters received at the port after the *port-EOF-string* are lost.

REFERENCES

Appendix F, "Display List Op-Codes"
COPY command
DIRECTORY command
LOAD command
PLOT command
PORT-COPY command
SAVE command
SET-COLOR-COPIER-DATA-RESOLUTION command
SET-COLORHARDCOPY-DITHERING command
SET-EOF-STRING command
SET-PARITY command
SPOOL command

Device-Status-Report Message Type

Syntax

device-status-report = [char-report:*sig-char*]
char-report:*first-device-char*
char-report:*second-device-char*
int-report:*status-word*
EOM-indicator

DESCRIPTION

The *Device-status-report* message is sent by the terminal to the host computer in response to a REPORT-DEVICE-STATUS command.

When the terminal sends a report to the host bypass mode is entered. (See ENTER-BYPASS-MODE.)

The *sig-char* in the report is the signature character for non-GIN reports, as set by the most recent SET-REPORT-SIG-CHARS command. If this character is NU, then it is omitted from the report.

First-device-char and *second-device-char* are the the first two characters from the REPORT-DEVICE-STATUS command's *device-specifier* parameter if the parameter was valid. If the parameter was invalid, *first-device-char* and *second-device-char* are both SP.

Status-word is in the range from 0 to 65535. Its bits hold status information about the device.

If the REPORT-DEVICE-STATUS command had an invalid *device-specifier*, or if the device is not present, then *status-word* is 0.

The *status-word* for the host communication port (**HO**) is always 1 if the port is not busy and 3 if the port is busy.

Tables 7-3 through 7-8 show the meanings of the bits in *status-word* for the other devices.

Table 7-3

DISK DRIVES

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0	W	R	0	D	0	B	P

The meanings of the bits (when set to 1 rather than 0) are as follows:

W The disk drive write protect switch was on when the disk drive door was last closed. (Non-MSIB board only.)

R The disk drive is ready.

D A double-sided diskette is installed.

B The disk drive is busy.

P This disk drive is installed in the terminal.

Table 7-4

RS-232 PERIPHERAL PORTS

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	B	P

The meanings of these bits (when set to 1 rather than 0) are as follows:

B Peripheral port is busy.

P This peripheral port is installed in the terminal. (This does not necessarily mean that there is a peripheral device attached to the the peripheral port.)

Table 7-5

COLOR HARDCOPY INTERFACE (SC:)

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	P

The meanings of these bits (when set to 1 rather than 0) are as follows:

X Reserved for future use.

P The screen pseudo device is present.

Table 7-6

COLOR HARDCOPY INTERFACE (HC:)

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	I	A	0	0	C	F	B	P

The meanings of these bits (when set to 1 rather than 0) are as follows:

I	The image is produced parallel with the long axis of the media (as opposed to with the short axis).
A	The copier has acknowledged a data transfer.
C	The copier is connected and powered-up.
F	A copier fault condition exists.
B	The copier is busy.
P	The color copier interface is present.

When the 4691 is connected but turned off, F and P (only) are 1.

When the 4695 is connected but turned off, I, A, C, B, and P (only) are 1.

Table 7-7

DMA INTERFACE (DM:)

B15	B14	B13	B12	B11	B10	B9	B9	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0	0	0	N	0	M	B	P

The meanings of these bits (when set to 1 rather than 0) are as follows:

M	DMA/host interaction is in Single-cycle mode (as opposed to Burst mode
B	The device is busy.
P	The DMA interface is present.
N	Identifies DMA option installed (0 = Option 3A, 1 = Option 3C).

Table 7-8

DIRECT ACCESS PSEUDO DEVICES (SG:, DS:, PX:, CM:)

B15	B14	B13	B12	B11	B10	B9	B9	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	P

The meanings of these bits (when set to 1 rather than 0) are as follows:

P	The pseudo device is present.

EOM-Indicator. The *EOM-indicator* is always sent to end the report. See the *EOM-indicator* syntactic construct description for details.

REFERENCES

Char-report message type
EOM-indicator syntactic construct
Int-report message type
REPORT-DEVICE-STATUS command

DIALOG Key

DESCRIPTION

The DIALOG key has an effect similar to the SET-DIALOG-AREA-VISIBILITY command. Pressing the key when its light is off causes the terminal to execute a SET-DIALOG-AREA-VISIBILITY : 1 command, making the dialog area visible. Pressing the key when its light is on causes the terminal to execute a SET-DIALOG-AREA-VISIBILITY : 0 command, making the dialog area invisible.

When a 3D Local Viewing function is entered, the light on the DIALOG key reflects the visibility status of dialog area –2. When all 3D Local Viewing functions are exited, the DIALOG key light reflects the visibility status of the currently enabled dialog area.

This key does not auto-repeat.

REFERENCES

SET-DIALOG-AREA-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

DIRECTORY Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

```
EcJD  device:source
      [string:separator [device:destination]]
```

Setup Syntax

```
DIRECTORY Sp  source  TO  [destination]
```

PARAMETERS

source
The device or filename for which a directory is desired. Valid specifiers are all disk drive devices. See *Device* parameter type.

separator
Empty string or **TO**.

destination
Specifies the device to which the directory information is to be sent. See *Device* parameter type.

DESCRIPTION

The DIRECTORY command sends a directory listing of the specified device or file to the specified destination.

If *source* is a disk device without a filename, the entire directory of the current user number is sent to the destination device. An empty string in this field is equivalent to the default device. If a filename is specified, only the directory entry for that file is sent.

See the SET-USER-NUMBER command for information about *user-numbers*.

Figure 7-8 shows a typical directory report, both for a disk (Figure 7-8A) and for a single file (Figure 7-8B). Note that file size is reported only with a single file directory report.

DEFAULTS

source
as shipped — none
on power-up — none
if omitted — empty string

separator
as shipped — none
on power-up — none
if omitted — empty string

destination
as shipped — none
on power-up — none
if omitted — empty string

ERRORS

JD00 (Level 0): Unrecognized command. (Disk drive option is not installed.)

JD10 (Level 2): The specified *source* device does not exist or cannot be found.

JD11 (Level 2): Invalid *source* specifier.

JD12 (Level 3): Out of memory while parsing the parameter, 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* (must be empty string or **TO**).

JD22 (Level 3): Out of memory while parsing the parameter.

JD30 (Level 2): The specified *destination* does not exist.

JD31 (Level 2): Invalid *destination* specifier.

JD32 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JD33 (Level 2): Context error. (The device specified is not a valid destination device, or is write-protected.)

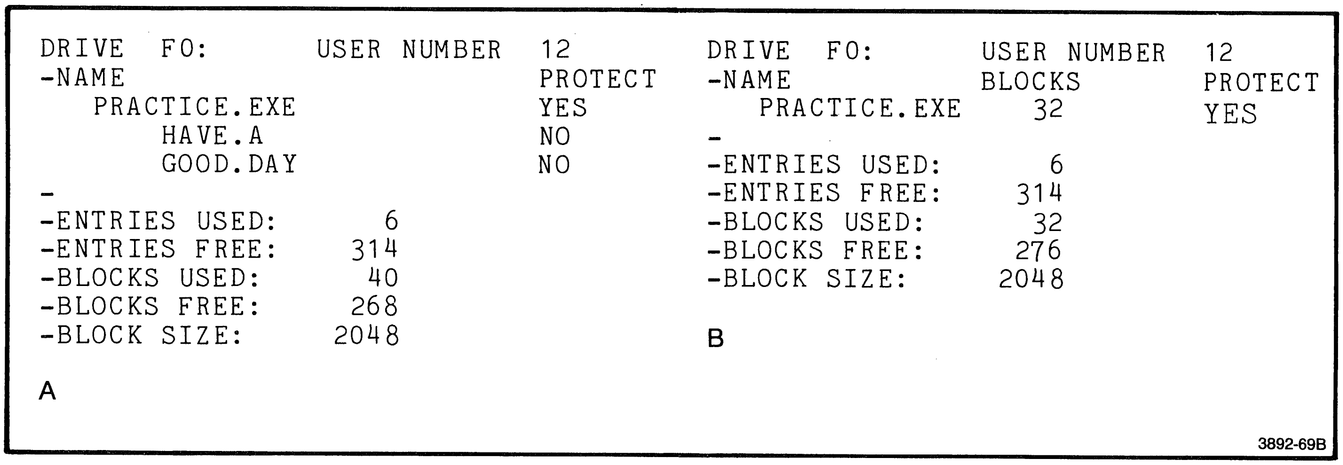
JD39 (Level 2): Hardware error for the destination device.

REFERENCES

Device parameter type
SET-USER-NUMBER command

INQUIRY COMMAND

none



```
DRIVE  FO:      USER NUMBER  12
-NAME                        PROTECT
   PRACTICE.EXE              YES
       HAVE.A                NO
       GOOD.DAY              NO
-
-ENTRIES USED:      6
-ENTRIES FREE:    314
-BLOCKS USED:      40
-BLOCKS FREE:     268
-BLOCK SIZE:     2048
```

A

```
DRIVE  FO:       USER NUMBER  12
-NAME            BLOCKS       PROTECT
   PRACTICE.EXE    32         YES
-
-ENTRIES USED:      6
-ENTRIES FREE:    314
-BLOCKS USED:      32
-BLOCKS FREE:     276
-BLOCK SIZE:     2048
```

B

Figure 7-8. Directory Command Report Format.

DISABLE-GIN Command

Host Syntax

^{E}cID int:*device-function*

Setup Syntax

GINDISABLE ^{S}P *device-function*

PARAMETERS

device-function
Specifies the GIN *device-function* which is to be disabled. Valid values are -1 and those valid for the ENABLE-GIN command.

DESCRIPTION

This command disables the specified GIN *device-function* combination. The *device-function* codes are the same as for the ENABLE-GIN command; see the description of that command for details. However, a *device-function* code : –1 disables all devices for which graphic input is enabled.

If the specified function is already disabled, the command is ignored. Likewise, if the specified device is for an option which is not present, the command is ignored; an error is not generated.

When a *device-function* combination is disabled, the terminal sends the *GIN-termination-report.* See *GIN-report* message type for details.

This command disables the GIN function on the named device. It leaves the device and its port (if any) active and addressable for further action.

ERRORS

I011	(Level 2):	Invalid *device-function.* (See the ENABLE-GIN command for a table of *device-function* codes.)

REFERENCES

ENABLE-GIN command
GIN-report message type

INQUIRY COMMAND

none

DISABLE-4953-TABLET-GIN Command

4111 or OPTION 10

Host Syntax

```
Ec!   char:disable-code
```

PARAMETERS

disable-code
An ASCII character whose second-least-significant bit is zero, and whose fourth-least-significant bit is one.

DESCRIPTION

This disables the terminal's emulation of a TEKTRONIX 4010 Series terminal equipped with a 4953 or 4954 Graphics Tablet.

After a DISABLE-4953-TABLET-GIN command is received, any points stored in the buffer will still be sent.

ERRORS

I100 (Level 0): Unrecognized command. (Option 10 is not installed.)

REFERENCES

ENABLE-4953-TABLET-GIN command

INQUIRY COMMAND

none

DISMOUNT Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

```
EcJJ   device:device
```

Setup Syntax

```
DISMOUNT Sp   device
```

PARAMETERS

device
Specifies the disk device to dismount. See *Device* parameter type.

DESCRIPTION

This command tells the terminal that the disk has been changed in the specified device. The terminal then scans the new disk to determine data allocation and block information. This command is necessary only for performing the DIRECTORY command on 5 1/4 inch flexible disks. All other operations and disk drives automatically scan before read or write operations.

DEFAULTS

device
as shipped — none
on power-up — none
if omitted — the default disk drive

ERRORS

JJ00 (Level 0): Unrecognized command (there are no disk options installed).

JJ10 (Level 2): *Device* is not installed.

JJ11 (Level 2): Invalid *device* parameter.

JJ13 (Level 2): Context error (device is busy).

REFERENCES

DIRECTORY command

INQUIRY COMMAND

none

DISPLAY-ALPHA-TEXT Command

Host Syntax

char:*text-char*

PARAMETERS

text-char (SP to ~ (ASCII characters with ASCII Decimal Equivalent (ADE) ranging from 32 to 126))
Specifies the printable character to be displayed.

DESCRIPTION

The DISPLAY-ALPHA-TEXT command is implicit. When the terminal is in alpha mode and is not parsing an explicit command, any printable character received is an implicit command to display alphatext. (There is no explicit command to display alphatext, although the GRAPHIC-TEXT command can cause string precision graphtext in the graphics area which looks like alphatext.)

Dialog Area Enabled

When a dialog area is enabled, the alphatext character is placed into the scroll buffer of that dialog area. Before this occurs the alphatext cursor can be in one of three conditions:

1. not at the right edge of the dialog area; in this case the character is printed at the cursor location and the cursor moves one space to the right.
2. just arrived at the right edge (with no character at the right edge); in this case a character is displayed at the cursor location and the cursor stays at the right edge (no wrapping occurs).
3. at the right edge after a character has been displayed there (cursor is under character); in this case, if auto-wrap is enabled the character is displayed at the beginning of the next line and the cursor moves one position to the right of the character; if auto-wrap is disabled, the character is displayed at the right edge (replacing the character currently there) and the cursor doesn't move.

The character is actually displayed the next time that area of the scroll buffer is visible.

The way alphatext is displayed in the dialog area is controlled by the SELECT-GRAPHIC-RENDITION, SET-ALPHATEXT-FONT, SET-DIALOG-AREA-ALTERNATE-INDEX, SET-DIALOG-AREA-INDEX, SET-DIALOG-AREA-WRITING MODE, and SET-4014-ALPHATEXT-SIZE commands.

Dialog Area Not Enabled

If no dialog area is enabled, the alphatext character is displayed at the current alpha cursor position, and the alpha cursor is moved one space to the right. One character space is eight or sixteen pixels on the screen, eight for the small character size and sixteen for the large size, as set by the SET-4014-ALPHATEXT-SIZE command. The alpha cursor position is located in pixel space, and is changed to match the graphic beam position whenever the graphic beam position is changed by commands such as MOVE and DRAW. The alpha cursor is also moved by the alpha positioning control characters ^{B}S, ^{C}R, ^{H}T, ^{L}F, and ^{V}T.

If the terminal is in 2D mode (coordinate mode 0 or 1), the 2D graphic beam position is updated through a viewport-to-window transform to match the alpha position. If the initial alpha position was within one character space of the right edge of the overview window, and if no segment is open, the alpha position is first wrapped either to the beginning of the next line at the left edge of the overview window or to the view home position (see the SET-OVERVIEW-WINDOW command) if the next line is below the bottom of the overview window. If a segment is open and the initial alpha position is within one space of the right edge of the overview window, no character is displayed and the alpha position is not moved.

In 3D mode (coordinate mode 2), the 3D graphic beam position is not moved and there is no right edge where autowrap occurs. On the 4129, when alphatext is displayed as part of a segment during a RENEW-VIEW or PAGE operation, and the view's object-surface-display attribute is "shaded-surface" or "hidden-line", the characters are obscured by anything in front of them.

The way alphatext is displayed in the graphics area is controlled by the SET-ALPHATEXT-FONT, SET-BACKGROUND-INDICES, SET-GRAPHICS-AREA-WRITING-MODE, SET-TEXT-INDEX, and SET-4014-ALPHATEXT-SIZE commands.

REFERENCES

PAGE command
RENEW-VIEW command
SELECT-GRAPHIC-RENDITION (ANSI) command
SET-ALPHATEXT-FONT command
SET-BACKGROUND-INDICES command
SET-DIALOG-AREA-ALTERNATE-INDEX command
SET-DIALOG-AREA-INDEX command
SET-GRAPHICS-AREA-WRITING-MODE command
SET-TEXT-INDEX command
SET-4014-ALPHATEXT-SIZE command

INQUIRY COMMAND

none

DRAW Command

Host Syntax

Explicit:
^{E}cLG xy + :*position*
or
^{G}s^{B}L xy + :*position*

Implicit:
xy + :*position*

Setup Syntax

^{E}cLG ^{S}P *position*

PARAMETERS

position
Specifies the end point of a line (a vector) in terminal space drawn from the current beam position. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y $\geqslant$ 0 to 4095;
coordinate mode 1: X,Y $\geqslant -2^{31}$ to $2^{31}-1$;
coordinate mode 2: X,Y,Z $\geqslant -2^{23}$ to $2^{23}-1$

DESCRIPTION

The DRAW command causes a line (a vector) to be drawn, in the current line style and line index, from the current beam position to the point specified by *position.*

If the terminal is in 2D mode (coordinate mode 0 or 1), the line is drawn from the 2D beam position; if in 3D mode (coordinate mode 2), then the line is drawn from the 3D beam position. The point specified with the command becomes the new beam position. On a 4128 and 4129, when a view where object-surface-display attribute (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is "hidden-line" or "shaded surface" is renewed, all vectors are displayed as a solid line (line style 0) regardless of the vector's line style.

The DRAW command has three formats: two explicit and one implicit.

The first explicit DRAW command syntax is ^{E}cLG *xy* + . It does not depend upon or change the terminal's vector/marker/alpha mode; it is executed as any other escape-sequence command. The second explicit DRAW command syntax is ^{G}s^{B}L *xy* + . The ^{G}s character puts the terminal into vector mode and sets the internal move/draw flag to move; the ^{B}L character sets the internal move/draw flag to draw. The *xy* + causes a line to be drawn from the current beam position to the *xy* + position.

The implicit DRAW command syntax is *xy* + when the terminal is in vector mode and the internal move/draw flag is set to draw. The second explicit DRAW command, above, and the implicit MOVE command, set the internal move/draw flag to draw.

NOTE

> *The terminal displays the first pixel in a vector if that pixel was just* moved to*; the terminal does not display the first pixel in a vector if that pixel had just been* drawn to.

DEFAULTS

position
as shipped — none
on power-up — none
if omitted — 0,0 (2D)
0,0,0 (3D)

ERRORS

LG11 (Level 2): *Position* out of valid range.

REFERENCES

ENTER-VECTOR-MODE command
SET-LINE-INDEX command
SET-LINE-STYLE command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-GIN-POINT command

DRAW-CURVE Command

4111, 4115P51 or 4120 SERIES

Host Syntax

```
EcUC   int:curve-type
       xy-array:list-of-points
```

Setup Syntax

```
EcUC Sp   curve-type   list-of-points
```

PARAMETERS

curve-type (0 to 32767)

Specifies the type of curve to be drawn.

1 A circular arc from the current beam position through pairs of points.

2 A circular arc from the current beam position through pairs of points with chord drawn back to the current beam position.

3 A circular arc from the current beam position through pairs of points with a vector drawn from the end point of each arc to arc center, and a vector drawn from arc center to the current beam position.

list-of-points (length 0 to 8192)

Array of pairs of xy points that define the curve to be drawn. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$

DESCRIPTION

This command approximates a circular curve with one or more vectors drawn in the current line style and line index. These are: circular arcs, circular arcs plus a chord, and circular arcs plus radii. The smoothness of the curves (number of vectors used in approximations) is determined by the SET-CURVE-SMOOTHNESS command.

Three points specify all arcs: the first point is the current beam position and the second and third points are specified by *list-of-points*. This implies that an even number of points must be specified in the array.

Special cases are handled as follows:

1. The three points are the same — draws a dot.
2. 2nd point = 1st or 3rd — draws a vector from 1st to 3rd point.
3. The three points are not equal but are colinear — draws a vector from the 1st point to 2nd point and draws another vector from the 2nd point to 3rd point.
4. 3rd point = 1st point — draws a circle, with the 1st and 2nd points as diameter endpoints.
5. Coordinates generated for the arc which are greater than $2^{31}-1$ are set to $2^{31}-1$, less than -2^{31} are set to -2^{31}.

For *curve-type*s 2 and 3, the additional vector(s) are not drawn if the specified arc is a circle or straight line.

Curve-type 1 leaves the beam position at the third point of the arc so connected and continuous arcs can be drawn with one command.

*Curve-type*s 2 and 3 leave the beam position at the starting point (the beam position when this command is issued).

If a graphtext character is being defined, or if the 4128 or 4129 terminal is in 3D mode (coordinate mode 2), an error is detected and no action is taken.

DEFAULTS

curve-type
as shipped — none
on power-up — none
if omitted — error UC11

list-of-points
as shipped — none
on power-up — none
if omitted — error UC21

ERRORS

UC03 (Level 2): A graphtext character is currently being defined or the 4128 or 4129 terminal is in coordinate mode 2.

UC11 (Level 2): Invalid *curve-type*.

UC21 (Level 2): Invalid *list-of-points* array.

REFERENCES

SET-CURVE-SMOOTHNESS Command

INQUIRY COMMAND

none

DRAW-FACETS Command

4128, 4129

Host Syntax

```
EcUF  int-array:action
      xy+-array:points
      int-array:indices
      xy+-array:normals
```

Setup Syntax

```
EcUF Sp action points indices normals
```

PARAMETERS

action (length 0 to 3)

first element: *facet-type* (0 to 4)
Specifies the type of facet.

0	triangles
1	quadrilaterals
2	spokes
3	strips
4	meshes

second element: *outlining* (0 to 63)
Specifies which edges to outline (depends on facet type).

third element: *columns* (2 to 255)
Specifies the number of columns (mesh type only).

points (length 0 to 5461)
Specifies the vertices of the facets.

All elements = -2^{23} to $2^{23}-1$. The array must provide an even multiple of the number of points needed for each facet type.

indices (length 0, 1, number-of-facets, or the same as length of *points*)
Specifies the Extended Range Color Indices (ERCI's) of each vertex, all vertices in each triangle or quadrilateral facet, or all vertices of all facets, or the power-of-two for normals, or none.

All elements = 0 to 4095 (ERCI's)
0 to 22 (power-of-two)

normals (length 0, 1, number-of-facets or the same as length of *points*)

Specifies one normal for each vertex, one normal for each triangle or quadrilateral facet, one normal for each command, or none.

All elements: XYZ = -2^{23} to $2^{23}-1$

DESCRIPTION

This command draws five types of facets: triangles, quadrilaterals, spokes, strips and meshes. Spokes and strips are more efficient ways of defining adjacent triangles, while meshes are a more efficient way to define grids of quadrilaterals. Quadrilaterals need not be planar, since they are "pixelated" as two triangles, although the common edge of the triangles crossing the quadrilateral cannot be outlined. (See Appendix G for details on "pixelating".)

Each type of facet is described individually after the following paragraphs applying to all types.

The way facets are displayed depends on when they are displayed and the current view's object-surface-display attribute as set by the SET-VIEW-DISPLAY-ATTRIBUTES command. They are displayed as wireframe except when they are part of a segment and the view is renewed with the view's object-surface-display attribute set to "shaded-surface" or "hidden-line".

"Wireframe" display of facets consists of drawing the outlined edges once each as vectors in the current line style and line index. The second element of the *actions* parameter determines which edges are outlined.

"Shaded-surface" display of facets consists of filling the facets much as 2D panels are filled (see Appendix G for details), and drawing the outlined edges as solid vectors in the current line index if the "outlining" bit of the segment's highlighting setting is "on".

"Hidden-line" display of facets is the same as shaded surface display except that the facets are filled with the view's wipe index as set by the SET-VIEW-ATTRIBUTES command, and facets are always outlined according to the *outlining* parameter in this command.

The 3D graphic beam position is set to the last point in the *points* array.

To specify a normal, send a single value in the *indices* array, set to 0 to 22 (8 is recommended for the highest precision with the least transmission), and send an *xy+* in the *normals* array for each normal. The terminal computes the intended normal from the *xy+* and index by dividing the coordinates of the *xy+* by two raised to the power of the index. For example, to send the unit normal <.707,.707,0.>, send 8 as the index and 181,181,0 as the *xy+*.

NOTE

The length of each normal should be 1.0 or slightly less than 1.0. Due to internal optimization for speed, normals with a length greater than 1.0 may have one or more of the three coordinates' signs changed, resulting in an improper direction.

The Extended Range Color Indices (ERCI's) at the vertices of the facet are derived from the *indices* array, the *normal* array and/or the current object-surface-color as shown in Table 7-9. In this table, OSC means "the object-surface-color when it is non-positive or the range is positive but has been deleted for it after the segment has been defined (then the ERCI will be set to 0)". See Appendix G for details on how the Local Lighting Model computes ERCI's from normals.

Table 7-9

DERIVING ERCIs

Length *points*	**Length** *indices*	**Length** *normals*	**Source of ERCI**
N	0	0	OSC or one ERCI computed per facet (triangles and quadrilaterals only)
N	1	0	one ERCI given for all vertices
N	N/3-N/4	0	one ERCI given per facet (triangles and quadrilaterals only)
N	N	0	one ERCI given per vertex
N	1	1	OSC or one ERCI computed for all vertices
N	1	N/3-N/4	OSC or one ERCI computed per facet (triangles and quadrilaterals only)
N	1	N	OSC or one ERCI computed per vertex

The low order bits of the *outlining* element of the *actions* parameter control which edges of a facet are outlined. The descriptions below give the resulting decimal numbers. For example, to outline the second and third edges of triangles, use 6.

The order the vectors are traversed is the natural order for triangles and quadrilaterals and as indicated below for strokes, strips, and meshes. See the SELECT-TRAVERSAL-DIRECTION command description for details of the effect of this order of traversal.

Triangles

Facet type: 0

Outlining:	0	none
	1	first edge of each triangle
	2	second edge of each triangle
	3	combination of 1 and 2
	4	third edge of each triangle
	5-7	combinations of 1,2,4
	8	no action
	9-15	combinations of 1,2,4,8
	16	no action
	17-31	combinations of 1,2,4,8,16
	32	no action
	33-63	combinations of 1,2,4,8,16,32

Points must have an even multiple of three elements. *Indices* may have 0 or 1 elements, or one per facet, or one per vertex. *Normals* may have 0 or 1 elements, or one per facet, or one per vertex.

Quadrilaterals

Facet type: 1

Outlining:	0	none
	1	first edge of each quadrilateral
	2	second edge of each quadrilateral
	3	combination of 1 and 2
	4	third edge of each quadrilateral
	5-7	combinations of 1,2,4
	8	fourth edge of each quadrilateral
	9-15	combinations of 1,2,4,8
	16	no action
	17-31	combinations of 1,2,4,8,16
	32	no action
	33-63	combinations of 1,2,4,8,16,32

Points must have an even multiple of four elements. *Indices* may have 0 or 1 elements, or one per facet, or one per vertex. *Normals* may have 0 or 1 elements, or one per facet, or one per vertex.

Spokes

Facet type: 2

Outlining:	0	none
	1	first edge
	2	all edges except first and last
	3	combination of 1 and 2
	4	last edge
	5-7	combinations of 1,2,4
	8	all internal spokes
	9-15	combinations of 1,2,4,8
	16	no action
	17-31	combinations of 1,2,4,8,16
	32	no action
	33-63	combinations of 1,2,4,8,16,32

Points must have three or more elements. *Indices* may have 0 or 1 elements or one per vertex. *Normals* may have 0 or 1 elements or one per vertex. Each spoke with N points defines N-2 triangles with point numbers [1,2,3], [1,3,4], [1,4,5], and so on.

Strips

Facet type: 3

Outlining:	0	none
	1	edges between odd numbered vertices
	2	edge between last two vertices
	3	combination of 1 and 2
	4	edges between even numbered vertices
	5-7	combinations of 1,2,4
	8	edge between first two vertices
	9-15	combinations of 1,2,4,8
	16	edges between even and odd vertices (except first and last)
	17-31	combinations of 1,2,4,8,16
	32	no action
	33-63	combinations of 1,2,4,8,16,32

Points must have three or more elements. *Indices* may have 0 or 1 elements or one per vertex. *Normals* may have 0 or 1 elements or one per vertex. Each strip with N points defines N-2 triangles with point numbers [1,2,3], [2,4,3], [3,4,5], [4,6,5], and so on. Note that the triangles starting with an even numbered point are ordered so they will traversed in the same direction as the triangles starting with an odd numbered point.

Meshes

Facet type: 4

Outlining:

0	none
1	first row edges
2	last column edges
3	combination of 1 and 2
4	last row edges
5-7	combinations of 1,2,4
8	first column edges
9-15	combinations of 1,2,4,8
16	other row edges
17-31	combinations of 1,2,4,8,16
32	other column edges
33-63	combinations of 1,2,4,8,16,32

Number of columns: 2 or more

Points must have four or more elements which is an even multiple of the number of columns and representing two or more rows. *Indices* may have 0 or 1 elements or one per vertex. *Normals* may have 0 or 1 elements or one per vertex. Each mesh with M rows and N columns defines (M-1)∗(N-1) quadrilaterals with point numbers [1,2,N + 2,N + 1], [2,3,N + 3,N + 2], and so on. The first N points in points specifies the first row, the second N points the second row, etc.

ERRORS

UF00	(Level 0):	Unrecognized command (terminal is not a 4128 or 4129).
UF03	(Level 2):	Command is invalid at this time (coordinate mode 2 must be in effect).
UF11	(Level 2):	Invalid *type* code in *action* array.
UF12	(Level 3):	Out of memory while parsing *action*.
UF21	(Level 2):	Invalid *points* array.
UF22	(Level 3):	Not enough memory for *points* array.
UF31	(Level 2):	Invalid *indices* array.
UF32	(Level 3):	Not enough memory for *indices* array.
UF41	(Level 2):	Invalid *normals* array.
UF42	(Level 3):	Not enough memory for *normals* array.

REFERENCES

Appendix G, Shaded Surfaces
SELECT-TRAVERSAL-DIRECTION command
SET-LINE-INDEX command
SET-LINE-STYLE
SET-VIEW-ATTRIBUTES command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-GIN-POINT command

DRAW-MARKER Command

Host Syntax

Explicit:
^{E}cLH xy + :*marker-position*
or
^{F}s xy + :*marker-position*

Implicit:
xy + :*marker-position*

Setup Syntax

^{E}cLH ^{S}p *marker-position*

PARAMETERS

marker-position
Specifies the position in terminal space where a marker is drawn. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$
coordinate mode 2: X,Y,Z = -2^{23} to $2^{23}-1$

DESCRIPTION

The DRAW-MARKER command draws a marker of the current marker type at the point specified by *marker-position* and the graphic beam is updated to the point specified. The marker is drawn in the current line index, as set by the most recent SET-LINE-INDEX command. If the terminal is in 2D mode (coordinate mode 0 or 1), the 2D beam position is updated; if in 3D mode (coordinate mode 2), the 3D beam position is updated.

Markers are drawn as dot-matrix patterns in pixel space centered around the specified position regardless of coordinate mode.

The DRAW-MARKER command has three formats: two explicit and one implicit. The first explicit DRAW-MARKER command syntax is ^{E}cLH *xy* + . It does not depend upon or change the terminal's vector/marker/alpha mode; it is executed as any other escape-sequence command. The second explicit DRAW-MARKER command syntax is ^{F}s *xy* + . The ^{F}s character puts the terminal into marker mode. The *xy* + causes a marker to be drawn at the given position.

The implicit DRAW-MARKER command syntax is *xy* + when the terminal is in marker mode.

A DRAW-MARKER command received during a panel definition does not draw a marker. Instead, it defines a vertex of the panel. See the *Panel-definition* syntactic construct for details on the use of markers as a panel vertex.

DEFAULTS

marker-position
as shipped — none
on power-up — none
if omitted — 0,0 (2D); 0,0,0 (3D)

ERRORS

LH11 (Level 2): *Marker-position* out of valid range.

REFERENCES

ENTER-MARKER-MODE command
Panel-definition syntactic construct
SET-LINE-INDEX command
SET-MARKER-TYPE command

INQUIRY COMMAND

REPORT-GIN-POINT command: –2

DRAW-RECTANGLE Command

Host Syntax

$^{E}_{C}$**UR** xy-array:*opposite-corners*

Setup Syntax

$^{E}_{C}$**UR** $^{S}_{P}$ *opposite-corners*

PARAMETERS

opposite-corners (array length from 0 to 8192)
An *xy-array* containing pairs of xy coordinates. Each pair of xys represents opposite corners of a rectangle to be drawn The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$

DESCRIPTION

This command draws rectangular panels in 2D terminal space. If the terminal is in 3D mode (coordinate mode 2), an error is detected with no other action. The actual action depends on whether a *panel-boundary-definition* is currently open.

For each pair of xy coordinates in the *opposite-corners* parameter, actions equivalent to the following commands are performed:

BEGIN-PANEL-BOUNDARY: upper-right xy, *draw-boundary-mode*
DRAW: lower-right xy
DRAW: lower-left xy
DRAW: upper-left xy
DRAW: upper-right xy
END-PANEL (see note)

Draw-Boundary-Mode. The *draw-boundary-mode* is as set by the last SET-DRAW-BOUNDARY-MODE command.

Beam Position. The final beam position after this command is the upper-right corner of the last rectangle drawn.

NOTE

If the DRAW-RECTANGLE command is issued within a panel-boundary-definition*, the END-PANEL action is not performed for the last pair of* xys. *This allows the panel to be extended beyond the series of rectangles with MOVE, DRAW, and DRAW-MARKER commands. Also, if rectangles overlap, they behave as multiple boundaries to one panel.*

The rectangular panels are filled according to the most recent SELECT-FILL-PATTERN and SET-PANEL-FILLING-MODE commands. They are filled fastest if the fill pattern is negative, and, if in a segment, the rotation of the segment is the power-up default.

DEFAULTS

opposite-corners
as shipped — none
on power-up — none
if omitted — error UR11

ERRORS

UR03 (Level 2): Context error. (A new panel boundary cannot be added to an existing panel during segment editing. Rectangles are not allowed when the terminal is in coordinate mode 2).

UR11 (Level 2): Invalid *opposite-corners* array.

UR12 (Level 3): Out of memory while parsing parameter.

REFERENCES

BEGIN-PANEL-BOUNDARY command
DRAW command
END-PANEL command
Panel-definition syntactic construct
SELECT-FILL-PATTERN command
SET-COORDINATE-MODE command
SET-DRAW-BOUNDARY-MODE command
SET-PANEL-FILLING-MODE command

INQUIRY COMMAND

REPORT-GIN-POINT command

ENABLE-DIALOG-AREA Command

Host Syntax

${}^{E}{}_{C}$**KA** int:*dialog-area*

Setup Syntax

DAENABLE ${}^{S}{}_{P}$ *dialog-area*

PARAMETERS

dialog-area (0 to 64, –2)
Specifies which dialog area is to be enabled. Setup mode parameters are **YES**, **NO**, **2** to **64**, and **–2**.

0	NO; none.
1	YES; dialog area 1.
2 to **64**	dialog area 2 to 64. (4115 requires 4115P51.)
–2	dialog area –2 (the special 3D Local Viewing System dialog area on 4125, 4128, and 4129.)

DESCRIPTION

This command enables the specified dialog area.

If *dialog-area* is 0, all dialog areas are disabled. Alphatext is directed to the graphics area, at the current graphic beam position. With the dialog areas disabled, the terminal more closely emulates TEKTRONIX 4010-series terminals, which do not have a dialog area. All changes to dialog area attributes affect dialog area 1.

If *dialog-area* is 1 through 64, the specified dialog area is enabled. All alphatext is directed to the specified dialog area's scroll buffer, at the position of the dialog area cursor. This occurs whether or not the dialog area is visible. (If the dialog area is invisible, the operator cannot see the alphatext being sent there.) All changes to dialog area attributes affect the currently enabled dialog area. Insufficient memory to support a new dialog area causes an error to be detected.

On the 4128 and 4129, if *dialog-area* is –2, the special 3D Local Viewing System dialog area is enabled so that its position and visibility can be set. Alphatext continues to be displayed in the previously enabled dialog area (0 to 64). The dialog area attributes of characters, lines, and buffer size may not be changed when dialog area –2 is enabled. (Characters, lines, and buffer size are fixed at 60, 4, and 4 respectively.) Inquiries of dialog area parameter settings return reports for the last enabled *dialog-area* (0 to 64). Dialog area –2 cannot be cleared.

On the 4125, dialog area –2 may be selected; however, it will not be seen since coordinate mode 2 and 3D local viewing are not available on the 4125.

NOTE

> *When ANSI code is selected (with the SELECT-CODE command), all alphatext goes into the dialog area which is currently enabled (1 to 64). If no dialog area is enabled, all alphatext will go into dialog area 1.*

Dialog Area Attributes

If *dialog-area* 2 to 64 has not been previously enabled or has been deleted, it inherits attributes from the most recently enabled dialog area. If no other dialog area has been enabled since a terminal reset, the dialog area will inherit dialog area 1 attributes. The exceptions are dialog area buffer size and visibility.

A newly enabled dialog area inherits a buffer size equal to DALINES and is invisible. Once enabled, the dialog area attributes can be set in the normal manner and are remembered when the appropriate dialog area command is issued.

Enabled Dialog Areas After Power-up/Reset

If the enabled dialog area is 1 when power is turned off or before resetting the terminal, dialog area 1 will still be enabled and visible at power-up or after the reset.

If the currently enabled dialog area is 2 to 64 when power is turned off or the terminal is reset, dialog area 1 will be enabled and visible at power-up or after the reset.

Table 7-10 lists the differences in terminal operation with the dialog area disabled and with it enabled.

Table 7-10

FEATURES AFFECTED BY THE ENABLE-DIALOG-AREA Command

Feature	Effect With Dialog Area 0 Enabled	Effect with Dialog Area 1 to 64 Enabled
Destination for Alphatext	Alphatext is displayed at the current graphic beam position in terminal space.	Alphatext is displayed at the current dialog area cursor position.
PAGE Key, PAGE Command	Erases the viewport for the current view.	Erases the viewport for the current view.
	Redraws all visible segments.	Redraws all visible segments.
	Removes the terminal from 4010 GIN mode.	
	Resets line style to 0.	
	Moves the graphic beam to the Home position.	
	Puts terminal in alpha mode.	
CR Character	Puts the terminal in alpha mode.	If in alpha mode, performs "carriage return" in the dialog area.
	Performs "carriage return" action in the graphics area.	If in vector mode or marker mode, does nothing.
	Resets line style to 0.	
	Removes the terminal from 4010 GIN mode.	

DEFAULTS

dialog-area
as shipped — 0
on power-up — remembered (0 or 1 only)
if omitted — 1

ERRORS

KA02 (Level 3): Insufficient memory to enable a new dialog area.

KA11 (Level 2): Invalid *dialog-area.*

REFERENCES

CR character
PAGE command
PAGE key
PAN key
RENEW-VIEW command
SELECT-CODE command
SET-DIALOG-AREA-BUFFER-SIZE command
SET-DIALOG-AREA-CHARS command
SET-DIALOG-AREA-LINES command
SET-DIALOG-AREA-POSITION command
SET-DIALOG-AREA-VISIBILITY command
ZOOM key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

ENABLE-GIN Command

Host Syntax

$^{E}_{C}$IE int:*GIN-code* int:*GIN-count*

Setup Syntax

GINENABLE $^{S}_{P}$ *GIN-code* *GIN-count*

PARAMETERS

GIN-code

An integer of the form 2048M + 8D + F, where F is a 3-bit function code, D is an 8-bit device code, and M is a mode code.

Valid values for F are:

0	Locate
1	Pick
2	Stroke

Valid values for D are:

0	Thumbwheels
1	Tablet (absolute mode)
3	Peripheral Port 0 (P0:)
4	Peripheral Port 1 (P1:)
5	Peripheral Port 2 (P2: — not available on 4111)
6	Tablet (relative mode)
8	Mouse
9	Joystick

Valid values for M are:

0	GIN events on key depression
1	GIN events on key depression and also on key release

GIN-count (0 to 65535)

Specifies the number of GIN events possible before the device-function combination is automatically disabled.

DESCRIPTION

This command enables GIN (graphic input) for the device, function, and mode specified in the *GIN-code* parameter.

GIN-code. The *GIN-code* parameter combines a mode code (M) with the device-function code used by other GIN commands.

NOTE

This is the only GIN command that uses a GIN-code *containing mode information as well as device and function code information. Other GIN commands use the* device-function *code, which consists of the least-significant 11 bits of the* GIN-code. *Device-function codes take the form 8D + F, whereas GIN-codes take the form 2048M + 8D + F.*

The mode code (M) specifies when GIN events occur for the locator and pick functions. When M is 0, GIN events occur only on key depressions. When M is 1, GIN events occur both on key depressions and also on key releases. Key-release GIN events are not allowed to occur in conjunction with inking or rubberbanding. If you attempt to enable GIN with the mode in the GIN code set for a device-function combination for which rubberbanding or inking has already been set, the terminal will detect an error.

NOTE

GIN events are caused by changes in the status of the GIN device keys; they do not depend on the state of the GIN device keys before GIN is enabled. The first GIN event after ENABLE-GIN will be the first valid transition (down, if M is 0; up or down, if M is 1) detected after GIN is enabled.

Table 7-11 summarizes the supported GIN devices and the features that trigger GIN events.

Table 7-11

GIN DEVICE CODES (D)

Device Code	Motion Device	Event Device
0	Thumbwheels	Keyboard keys
1, 6	Stylus or puck	Stylus or puck button
3	Peripheral Port 0; joystick on 4662 and 4663 plotter; stylus or puck on 4957 and 4958 tablets	CALL button on 4662 plotter; the POINT button on 4663 plotter; stylus or puck button on tablet.
4	Peripheral Port 1	Same as Device Code 3
5	Peripheral Port 2	Same as Device Code 3
8	Mouse	Mouse buttons
9	Joystick	Joystick trigger and position switches (if macros are defined for them)

Table 7-12 lists the valid *device-function* codes for currently supported devices. *Use these codes with other GIN commands.*

Table 7-12

DEVICE-FUNCTION CODES
(8D + F)

	Function Code (F)		
Device Code (D)	0 (Locate)	1 (Pick)	2 (Stroke)
0 — Thumbwheels	0	1	
1 — Tablet (absolute)	8	9	10
3 — Peripheral Port 0	24	25	
4 — Peripheral Port 1	32	33	
5 — Peripheral Port 2	40	41	
6 — Tablet (relative)	48	49	
8 — Mouse	64	65	66
9 — Joystick	72	73	

Table 7-13 lists the valid *GIN-codes*, which include those shown in Table 7-12 for the case in which M is 0. *Use these codes (≥ 2048) only with the ENABLE-GIN command.*

Table 7-13

GIN-CODES
(2048M + 8D + F)

	Mode code (M) = 0 Function Code			Mode code (M) = 1 Function Code		
Device Code	0	1	2	0	1	2
0	0	1				
1	8	9	10	2056	2057	
3	24	25		2072	2073	
4	32	33		2080	2081	
5	40	41		2088	2089	
6	48	49		2096	2097	
8	64	65	66	2112	2113	
9	72	73				

GIN-count. The *GIN-count* specifies how many events of the function are to occur before the function is automatically disabled. A function can be essentially enabled permanently by specifying a very large count (for instance, 65535).

At the time GIN is enabled some of the dynamic attributes of the cursor segment are saved and set. See SET-GIN-CURSOR command for details about what happens to the cursor segment when GIN is enabled. For each device-function combination, the terminal remembers a separate GIN space location. When the device and function is enabled, this GIN-space location changes as the device is manipulated. The GIN-space location is transformed into terminal space coordinates through a GIN-area to GIN-window transform to update the GIN cursor position and for the GIN reports. (See the SET-GIN-AREA and SET-GIN-WINDOW commands for details.)

When a 3D segment is used as a GIN cursor, these GIN window space coordinates are used to update the 3D position of the cursor according to the current GIN logical function for the device and function. (See the SET-GIN-LOGICAL-FUNCTION command.)

The GIN cursor segment "blinks" once for each GIN event.

CAUTION

If you want to simultaneously enable more than one device, use the ENABLE-GIN command. Undesired results may occur if the terminal is simultaneously enabled for graphic input with two different style enabling commands: ENABLE-GIN, ENABLE-4010-GIN, ENABLE-4953-TABLET-GIN.

GIN Devices

There are five GIN device types: thumbwheels, tablet, peripheral ports, mouse, and joystick. (A tablet can be used in two modes, absolute and relative.)

Thumbwheels. The operator positions the GIN cursor by moving the thumbwheels and signals a GIN event by pressing a key on the keyboard. Every character which would be sent to the host as a result of pressing the key causes a GIN event. This includes the contents of key-macros. Each of these characters is sent as the key *char-report* of a *GIN-report*. If pressing a key causes no characters to be sent to the host, it does not cause a GIN event.

If the terminal is in local mode, pressing a keyboard key does not signal a GIN event, even though GIN may be enabled with the thumbwheels as the GIN device. In local mode, characters typed on the keyboard are treated as if they came from the host rather than the keyboard.

Only locator and pick functions are valid functions for the thumbwheels.

Tablet. (Requires 4111 or Option 10) The operator positions the GIN cursor segment by moving the stylus or puck on the graphic tablet and signals a locator, pick, or first-stroke-point GIN event by pressing the stylus against the tablet, or by pressing a button on the puck. Subsequent-stroke-point GIN events are signalled automatically as the stylus/puck is moved across the tablet surface. A last-stroke-point GIN event is signalled when the stylus or puck button is released or when the puck is lifted off the tablet. Due to the magnetic fields involved in digitizing points on a tablet, if the last-point is signalled by lifting the puck or stylus off the tablet surface, the *xy + -report* may not be accurate.

When a tablet device is enabled for any GIN function, one (and only one) of the following ports is accessed under the described conditions:

- Port P0: if a tablet protocol is assigned to P0: before P1: or P2:.
- Port P1: if a tablet protocol is assigned to P1: before P0: or P2:.
- Port P2: if a tablet protocol is assigned to P2: before P0: or P1:.

For more information on assigning protocols to peripheral ports, see the PORT-ASSIGN command.

The key code returned for tablet locator/pick functions is given by Table 7-13A. The same table applies for a mouse when the mouse buttons are not defined. Note that the code for a button release is just the code for the button press plus 32 (that is, shifted two places to the right in an ASCII code chart).

Table 7-13A

TABLET/MOUSE KEY REPORTS

Switch Pressed or Released	Key Code if Pressed	Key Code if Released
None (Response to Report-GIN-POINT command)	SP (ADE 32)	SP (ADE 32)
Stylus/Mouse	0 (ADE 48)	P (ADE 80)
Tablet button	Z (ADE 90)	z (ADE 122)
Tablet/mouse	1 (ADE 49)	Q (ADE 81)
Tablet/mouse	2 (ADE 50)	R (ADE 82)
Tablet button	▪	▪
▪	▪	▪
▪	▪	▪
▪	9 (ADE 57)	Y (ADE 89)
▪	A (ADE 65)	a (ADE 97)
▪	▪	▪
▪	▪	▪
▪	▪	▪
▪	F (ADE 70)	f (ADE 102)

The table covers the button press and release reports for Option 13/14 tablets, 4957/4958 tablets, and the mouse (stylus (Z), four-button puck (Z, 1, 2, 3), and 16-button puck (Z, 1 — 9, A — F)).

The GIN-key-report for subsequent points of a stroke is J or SB (ADE 74 or 26, respectively), except that it is O or US (ADE 79 or 31) for the last point of the stroke. The SET-TABLET-HEADER-CHARACTERS command controls whether these characters are the letters or the control characters.

Relative Mode Tablet Device. The tablet that is assigned to Device Code 1 also may be used as a "relative mode" device referred to by Device Code 6.

Most GIN device-parameters — signature character, cursor, inking, rubberbanding, gin display start point, and gin gridding — are shared by devices 1 and 6. Thus, for example, setting gridding for Device-Function 9 also sets gridding for Device-Function 49. However, GinArea specifications are stored separately for devices 1 and 6. Devices 1 and 6 cannot be enabled simultaneously, and an attempt to do so will generate an error.

When enabled by device-function codes 48 or 49, the GIN cursor position is not mapped directly from the position of the tablet puck (stylus), but is moved in the same direction as the puck (stylus) is moved.

This mode of operation is very similar to that of thumbwheel or mouse GIN. Large motions of the cursor can be accomplished with repeated small movements of the puck. The 4957 or 4958's proximity zone is approximately 1.5 inches (4 cm), so to keep the cursor from moving whenever the puck is moved, the puck must be moved out of proximity.

When GIN is first enabled, the GIN cursor is positioned at the last cursor position or at the position specified by the SET-GIN-START-POINT command.

Peripheral ports. (Requires a 4111 or Option 10) If a TEKTRONIX 4662/4663 plotter is attached, the operator positions the plotter pen by moving the joystick and signals a GIN event by pressing the CALL button on a 4662 or a POINT button on a 4663. If a 4957/4958 Graphics Tablet is connected, it operates as described above under Tablet.

NOTE

> *When graphic input is done from a plotter, the position on the screen of the graphic cursor is not updated until the operator signals a GIN event by pressing a button on the plotter.*

The key *char-report* included in *GIN-reports* for the plotter is **0**, **1**, or **2**: on a TEKTRONIX 4662 Plotter, **0** if the stylus is up, **1** if the stylus is down, and **2** if the CALL button is held down until the bell rings once; on a TEKTRONIX 4663 Plotter, **0** if the MOVE button is pressed, **1** if the DRAW button is pressed, and **2** if the LAST POINT button is pressed. The key *char-report* for tablet *GIN-reports* is as shown for the tablet device, above.

Only locator and pick functions are valid on peripheral ports, even with tablet protocol assigned. To access stroking on a 4957 or 4958 tablet, use the tablet device above.

NOTE

> *When GIN device 1 corresponds to a peripheral interface port (2PPI or 3PPI) tablet, the port belonging to this tablet may not be enabled for GIN as device-code 3, 4, or 5 when device 1 is enabled, and vice-versa.*

Mouse. (Requires Option 4M). The operator positions the GIN cursor segment by moving the mouse across any flat surface, and signals a GIN event by pressing one of the mouse buttons. The mouse buttons do not auto-repeat.

Every character that would be sent to the host as the result of pressing a mouse button causes a locator, pick, or first-stroke-point GIN event. This includes the contents of key-macros. Every character which would be sent to the host as the result of releasing a mouse button causes a locator or pick event. Subsequent-stroke-point GIN events are signalled automatically while the mouse button is held down. If no release-mouse-button macro is defined, or if one is defined with one or more characters which would be sent to the host computer, releasing the mouse button causes a last-stroke-point GIN event. The characters in the release-mouse-button macro are not sent to the host for the stroke function, they are ignored.

NOTE

> *If the mouse and thumbwheels are moved simultaneously, the mouse movement may be ignored.*

The default character assigned the mouse is 1 for the left button, 2 for the center button, and 3 for the right button.

Table 7-13A shows the key code returned for mouse locator/pick functions when the mouse buttons have not been defined.

The key *char-report* included in locator, pick and first-stroke point *GIN-reports* for the mouse are:

> **1**, **2**, **3**, or a host bound character in a press-mouse-button macro.

For subsequent stroke points the key *char-report* is:

> **J**

For last stroke point, the key *char-report* is :

> **O**

When mouse keys have been defined with the DEFINE-MACRO command, the character codes reported in GIN-key-reports are those in the macro definition (except, of course, characters inside Key-Execute-Character brackets; see the SET-KEY-EXECUTE-CHARACTER command for details). If a mouse key *press* is defined but the corresponding key *release* is not, then the character codes sent whenever a key release causes a GIN event are those defined for the key *press* plus 32 modulo 128. For example, if a mouse key press is defined to be "M" (ADE 77) and if the key release is not defined, then the key release code will be "m" (ADE 109). Likewise, if a mouse key press is defined to be "m" (ADE 109) and if the key release is not defined, then the key release code will be ᶜR (ADE 13) because MOD 128 (109 + 32) is 13.

Joystick. The operator moves the joystick to position the GIN cursor segment and presses the joystick trigger to signal a GIN event. The joystick trigger will auto-repeat if it is held down for more than half a second. (When GIN is not enabled for it, the joystick trigger behaves as a regular keyboard key.) Each character which would be sent to the host computer causes a GIN event, including characters in macros defined for the trigger and position switches. If a macro is defined for a position switch, the switch will not move the GIN cursor segment.

The default key character of the joystick trigger is **0**. The position switches have no default characters.

Only locator and pick functions are valid for the joystick.

NOTE

The internal GIN table for the joystick overlaps the table for Peripheral Port 2. This overlap means that the joystick and Peripheral Port 2 share all GIN parameters such as cursor segment and signature characters. Both devices cannot be enabled at the same time. Trying to enable one while the other is enabled causes an error to be detected.

GIN Functions

Locator Function. The GIN locator function returns a single report for each GIN event. The report contains a key *char-report*, an *xy + -report*, and maybe *view-number*. See *GIN-report* message type. All five GIN devices may be used for the locator function.

Pick Function. The GIN pick function returns one or more reports for each operator action. Each report contains a key *char-report*, an *xy + -report*, maybe *view-number*, and reports on which segment was picked. See *GIN-report* message type.

When a pick GIN event occurs, the terminal compares all detectable segments visible in the current view, in display priority order, to an area the size of the pick aperture centered on the current GIN location. If more than one segment is within the pick aperture, the 2D segment with the highest display priority is picked; if no 2D segment is within the pick aperture, the 3D segment with the highest display priority is picked. For segments which have the same display priority, the order of picking is not defined. When a primitive whose pick-ID is not 0 appears in the pick aperture area, the segment number and pickID of the primitive are reported in a *GIN-report*. Either only one *GIN-report* is sent for the first picked segment, or one *GIN-report* is sent for each picked segment depending on GIN-report-format, as set by SET-GIN-Report-FORMAT.

If, during a pick operation, the pick aperture covers primitives in a segment with different *pick-ID* numbers, then the terminal returns the *pick-ID* which was defined last. If no visible, detectable segment falls within the pick aperture, the terminal sends a *GIN-pick-report* in which the *pick-ID* number is zero.

If the image of more than one 3D segment falls within the pick aperture, then which 3D segment picked is determined as follows:

> If the view's object-surface-display attribute is "wire-frame", then the segment picked will be the one with the highest display priority.
>
> If the view's object-surface-display attribute is "shaded-surface" or "hidden-line", then the segment which is picked will be the one which is "closest" to the front plane in the W-axis direction. If there is more than one segment with the same minimum W-coordinate in the pick aperture, then the segment picked will be the one with the highest display priority; if they have the same display priority, the order of picking is undefined.

Stroke Function. The GIN stroke function returns two or more *GIN-reports* for each stroke action: one or more first-point, zero or more subsequent points and one last-point. When the operator presses the tablet's stylus or puck button or a mouse button, one (or more) first-point is sent to the host computer. As the stylus/puck/mouse is moved across the surface, subsequent points are sent at a rate determined by SET-GIN-STROKE-FILTERING. When the stylus/puck/mouse button is released, a last-point is sent to the host computer. Each point sent is considered to be one GIN event. If the GIN count expires in the middle of a stroke, the remaining points of the stroke are not digitized or sent.

It is possible to digitize points faster than the terminal can send the corresponding *GIN-reports* to the host computer. When this happens, the terminal uses blocks of main memory to store the digitized points. If (when) no more main memory is available, the terminal rings the bell and stops digitizing. When more memory becomes available, you can resume digitizing points. To avoid this condition, set stroke filtering to slow down digitizing, or just be sure there is sufficient main memory available.

Disabling GIN

The device and function combination is disabled when any of the following occurs:

- The count specified in the ENABLE-GIN command expires.
- The terminal receives a DISABLE-GIN command for that device and function.
- The terminal receives a CANCEL command.
- The operator presses the CANCEL key.

DEFAULTS

GIN-code
as shipped — none
on power-up — none
if omitted — 0

GIN-count
as shipped — none
on power-up — none
if omitted — 65536

ERRORS

I011 (Level 2): Invalid device-function code.

IE00 (Level 2): The cursor segment for the specified device and function does not exist. (It has been deleted since the SET-GIN-CURSOR command which assigned it to that device and function.)

IE03 (Level 2): Command is invalid at this time. (The segment being used as the cursor for the specified device and function is a segment which is currently being defined; or, in the 4128 and 4129, a 3D cursor is specified for a pick or stroke function.)

IE10 (Level 2): The specified GIN device is not installed in the terminal.

IE13 (Level 2): The specified device is already enabled, the protocol assigned to the peripheral port does not support GIN, the specified GIN device-function mode is invalid because inking or rubberbanding is set for that device-function and the mode is set for key-releases, or, on the 4128 and 4129, a locator function is enabled for GIN with a 3D cursor and either inking or rubberbanding is specified.

IE21 (Level 2): Invalid *GIN-count*.

REFERENCES

CANCEL command
CANCEL key
ENABLE-4010-GIN command
ENABLE-4953-TABLET-GIN command
GIN-report message type
PORT-ASSIGN command
SET-GIN-AREA command
SET-GIN-CURSOR command
SET-GIN-LOGICAL-FUNCTION command
SET-GIN-Report-FORMAT command
SET-GIN-STROKE-FILTERING command
SET-GIN-WINDOW command
SET-TABLET-HEADER-CHARACTERS command

INQUIRY COMMAND

none

ENABLE-KEY-EXPANSION Command

4111

Host Syntax

${}^{E}{}_{C}$**KW int:***mode*

Setup Syntax

KEYEXPAND ${}^{S}{}_{P}$ *mode*

PARAMETERS

mode (0 or 1)
Specifies whether key macros are enabled or disabled. Setup mode parameters are **YES** and **NO**.

0	**NO**; disables key expansion.
1	**YES**; enables key expansion.

DESCRIPTION

This command enables or disables all key macros.

When key expansion is enabled, the user can expand a key's macro by simply pressing the key. When key expansion is disabled, all keys temporarily revert to their default values.

This command does not delete macros. All key macros remain in memory and can be reenabled later.

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

NOTE

While key expansion is disabled, all programmed keys revert to their factory default values.

DEFAULTS

mode
as shipped — 1
on power-up — 1
if omitted — 1

REFERENCES

EXPAND-MACRO command

ENABLE-4010-GIN Command

Host Syntax

${}^{E}{}_{C}{}^{S}{}_{B}$

DESCRIPTION

This command is provided for compatibility with software written for earlier TEKTRONIX terminals. It provides an abbreviated way of enabling the thumbwheels for one GIN (graphic input) "locator" event. The report which the terminal sends in response to this GIN event is in the *4010-GIN-report* format, rather than the *GIN-report-sequence* format used with the ENABLE-GIN command.

CAUTION

If you want to simultaneously enable more than one device, use the ENABLE-GIN command. Undesired results may occur if the terminal is simultaneously enabled for graphic input with two different style enabling commands: ENABLE-GIN, ENABLE-4010-GIN, ENABLE-4953-TABLET-GIN.

When the terminal receives an ENABLE-4010-GIN command — the sequence of characters ${}^{E}{}_{C}{}^{S}{}_{B}$ — it displays the graphic cursor assigned to *device-function* zero. (The default cursor is the crosshair cursor; however, this may be changed with the SET-GIN-CURSOR command.)

The operator may then position the cursor by moving the thumbwheels. (The possible cursor positions are determined by the most recent SET-GIN-GRIDDING command for *device-function* zero — thumbwheels device, locator function.)

When the cursor is at the desired location, the operator presses any of the ASCII keys on the keyboard; this signals a "GIN event." In response to this GIN event, the terminal sends a *4010-GIN-report*. That report tells the host (a) which key the operator pressed, and (b) the location of the crosshair cursor in the terminal space.

When 4010 GIN is enabled, pressing a key which has a key-macro defined for it will cause an event for the first character of the macro which would normally be sent to the host. The remainder of the macro is expanded normally.

When the terminal sends a report to the host, it enters Bypass mode. (See ENTER-BYPASS-MODE.)

After sending the *4010-GIN-report*, the 4110/4120 sets its graphic beam position to the position of the graphics cursor, and enters Alpha mode. This is done for compatibility with TEKTRONIX 4010 Series terminals.

If the dialog area is not enabled, then receiving a CR character, or a PAGE command, cancels the effect of the ENABLE-4010-GIN command and places the terminal in Alpha mode. Pressing the PAGE key has the same effect as the PAGE command. For details, see the description of the ENABLE-DIALOG-AREA command.

If the dialog area is enabled, then the PAGE key, the PAGE command, and the CR character have no effect on graphic input; they do not cancel the effect of the ENABLE-4010-GIN command.

Terminal Settings For Emulating 4010-Series Graphic Input. To properly emulate a TEKTRONIX 4010-series terminal during graphic input, several of the terminal's settings must be set a certain way. The exact settings may vary from one computer installation to another. Table 7-14 lists settings which should work for most host computers.

Table 7-14

TERMINAL SETTINGS TO EMULATE 4010-SERIES GRAPHIC INPUT

Escape-Sequence Command	Equivalent Setup Mode Command
SET-REPORT-EOM-FREQUENCY: 1 = Ec**IM1**	REOM 1
SET-EOM-CHARS: 0, 0 = Ec**NC00**	EOMCHARS

SET-EOL-STRING:
CR = Ec**NT**
int-array:**13** = Ec**NT1** =
	EOLSTRING CR

SET-BYPASS-CANCEL-CHAR = Ec**NU** int:*bypass-cancel-char*	BYPASSCANCEL CR BYPASSCANCEL NU BYPASSCANCEL LF

Set the *bypass-cancel-char* to whatever is the last character which the host echoes in response to a report message from the terminal. If the host is not echoing characters sent from the terminal, set the *bypass-cancel-char* to NU. If the host echoes CR as CRLF, set the *bypass-cancel-char* to LF. If the host echoes CR as exactly CR, set the *bypass-cancel-char* to CR.

REFERENCES

4010-GIN-report syntactic construct
ENABLE-DIALOG-AREA command
ENABLE-GIN command
ENTER-BYPASS-MODE command
SET-EOL-STRING command
SET-EOM-CHARS command
SET-GIN-CURSOR command
SET-GIN-GRIDDING command

INQUIRY COMMAND

none

ENABLE-4953-TABLET-GIN Command

4111 or OPTION 10

Host Syntax

^{E}c! char:*enable-code*

PARAMETERS

enable-code
An ASCII character whose five least-significant bits hold tablet control information. If the fourth least-significant bit in the character is 1, the second least-significant bit must be 1 as well.

DESCRIPTION

This command is provided for (partial) compatibility with software written for earlier TEKTRONIX terminals. When graphic input is enabled with this command (rather than the ENABLE-GIN command), the 4110/4120 emulates a TEKTRONIX 4010 Series terminal equipped with a TEKTRONIX 4953 or 4954 Graphics Tablet. Also see the SET-TABLET-SIZE command.

To use this command with Option 10, a 4957 or 4958 protocol must be assigned to the port where the 4957/4958 tablet is attached.

CAUTION

Undesired results may occur if the terminal is simultaneously enabled for graphic input with two different enabling commands. Do not use more than one of these commands at the same time: ENABLE-GIN, ENABLE-4010-GIN, ENABLE-4953-TABLET-GIN.

Enable-Code. The *enable-code* parameter is a seven-bit ASCII character whose least-significant bits hold control information for the tablet, as follows:

b7	b6	b5	b4	b3	b2	b1
X	X	d	d	d	d	X

where

X = "Don't care" (may be zero or one)

d = Data (control information for the tablet)

b7, b6: "Don't care."

b5: Bit b5 controls whether the 4110/4120 is to return 10-bit or 12-bit coordinate data to the host. If this bit is zero, then x- and y-coordinates which the terminal sends to the host will each include only 10 binary data bits. If this bit is one, then the x- and y-coordinates reported to the host will each include 12 data bits. (For details, see the description of the *4953-TABLET-GIN-REPORT* message type.)

b4: Bit b4 determines how the tablet is disabled. If this bit is 1, and b2 is 1, then lifting the pen or puck out of presence disables the tablet. If b4 is 1, and b2 is 0, the tablet is immediately disabled. If bit b4 is 0, then another command must be sent to disable the tablet.

b3, b2: For 4010 Series terminals equipped with the 4953 or 4954 graphics tablets, bit b3 is the "presence" bit, and bit b2 is the "multiple-point" bit. In the 4110/4120 terminal, these bits may perhaps best be explained by saying that they together determine which 4110/4120 ENABLE-GIN command most closely resembles this ENABLE-4953-TABLET-GIN command:

b3	b2	Function
0	0	Analogous to an enable-GIN command for the tablet device, locator function, and one GIN event.
0	1	Analogous to an enable-GIN command for the tablet device, locator function, and 65535 GIN events.
1	0	Analogous to an enable-GIN command for the tablet device, locator function, and one GIN event.
1	1	Analogous to an enable-GIN command for the tablet device, stroke function, and 65535 GIN events.

b1: In a 4010 Series terminal equipped with a 4953 graphic tablet, bit b1 determines whether the terminal is to display locally the position data which is being sent to the host computer. The 4110/4120 always assumes that this bit is 1, causing the terminal to display the graphic cursor. If the "stroke" function or "multi-point locator function" is enable (bit b2 set to one), then the 4110/4120 performs "inking," as if a SET-GIN-INKING command had been issued.

After the Report is Sent. If the terminal is enabled for single-point GIN (that is, if the *enable-code*'s bit b2 is zero), then it sends a single *4953-tablet-GIN-report* to the host. After sending this report, the 4110/4120 enters alpha mode. On entering alpha mode, the alpha cursor's lower left corner is at the point whose position was just reported to the host in the *4953-tablet-GIN-report*.

Likewise, if the terminal is enabled for multiple-point GIN (that is, if the *enable-code*'s bit b2 is one), then it enters a graphic input mode and can send many *4953-tablet-GIN-reports* to the host. It exits this graphic input mode on receiving a DISABLE-4953-TABLET-GIN command or a CANCEL command, when 65535 points have been sent to the host (i.e., the report count has been satisfied), or when the the CANCEL key is pressed. When this occurs, the terminal enters alpha mode, and the alpha cursor is at the last point whose position was reported to the host.

Emulating 4953/4954 Strap Settings

As mentioned before, the purpose of the ENABLE-4953-TABLET-GIN command is to let the 4110/4120 terminal emulate an earlier TEKTRONIX 4010-series terminal with an accessory 4953 or 4954 graphics tablet. However, to do this correctly, the 4110/4120 must emulate certain strap settings on the 4953/4954 Tablet Control Board.

Table 7-15 lists the 4953/4954 Tablet Control Board strap settings, together with the commands for the 4110/4120 which emulate these strap settings. Before using a 4110/4120 with an existing program designed for the 4953/4954 graphics tablet, you should first issue the appropriate commands from this table.

Also, If a tablet overlay is used with the existing applications program, then you should issue a SET-TABLET-SIZE:1 command. This reduces the tablet's active area, so that overlays developed for Tektronix 4953 and 4954 tablets can be used with the 4957 and 4958 tablets.

Emulating Other 4010-Series Terminal Settings

Besides the settings which emulate straps on the 4953/4954 Tablet Control Board, there are certain other terminal settings which must be set to emulate 4010-series graphics terminals. Please refer to Table 7-14, in the description of the ENABLE-4010-GIN command.

Table 7-15

EMULATING 4953/4954 STRAP SETTINGS

Strap	Strap Setting	Equivalent 4115/4120 Commands
CR	IN	SET-REPORT-EOM-FREQUENCY: 1 SET-EOL-STRING: (13)
	OUT	SET-REPORT-EOM-FREQUENCY: 0 SET-REPORT-MAX-LINE-LENGTH:0
HEADER	CONTROL	SET-TABLET-HEADER-CHARACTERS: 1
	LETTER	SET-TABLET-HEADER-CHARACTERS: 0
LARGE/SMALL		The terminal automatically emulates this strap. That is, the terminal automatically determines whether the small tablet or the large tablet is attached.
DELAY	IN	SET-TRANSMIT-DELAY: 25 or SET-TRANSMIT-DELAY: 50
	OUT	SET-TRANSMIT-DELAY: 0
ESUP		In the 4110/4120 terminal, echo suppression is handled by the "bypass mode" associated with GIN. There is no need to emulate the ESUP (echo suppression) strap.
COMSUP		The 4110/4120 terminal always emulates the IN position of the COMSUP strap.
STATUS	IN	SET-TABLET-STATUS-STRAP: 1
	OUT	SET-TABLET-STATUS-STRAP: 0

ERRORS

I100 (Level 0): Unrecognized command. (Not a 4111 or Option 10, Version 6 or later is not installed.)

I003 (Level 2): A 3D segment is being used as the cursor tablet GIN device.

REFERENCES

CANCEL command
CANCEL key
DISABLE-4953-TABLET-GIN command
ENABLE-GIN command
ENABLE-4010-GIN command
SET-GIN-INKING command
SET-TABLET-SIZE command
4010-status-report message type
4953-tablet-GIN-report message type

INQUIRY COMMAND

none

END-FILL-PATTERN Command

Host Syntax

```
EcME
```

Setup Syntax

```
EcME
```

DESCRIPTION

This command terminates a *fill-pattern-definition*. The remainder (if any) of the fill pattern rectangle is filled with index 0.

ERRORS

This command detects no errors.

REFERENCES

Fill-pattern-definition syntactic construct

INQUIRY COMMAND

none

END-GRAPHTEXT-CHARACTER Command

Host Syntax

${}^{E}{}_{C}$SU

Setup Syntax

${}^{E}{}_{C}$SU

DESCRIPTION

This command terminates the graphtext character currently being defined.

ERRORS

SU03 (Level 1): This command is invalid at this time. (No graphtext character is being defined.)

REFERENCES

BEGIN-GRAPHTEXT-CHARACTER command
Graphtext-definition syntactic construct

INQUIRY COMMAND

none

END-PANEL Command

Host Syntax

${}^{E}{}_{C}$LE

Setup Syntax

${}^{E}{}_{C}$LE

DESCRIPTION

This command terminates a panel-definition, closing the current panel boundary and setting the 2D graphic beam position to the *first-point* of the panel boundary (that is, the coordinate specified in the last BEGIN-PANEL-BOUNDARY command). (See the *Panel-definition* syntactic construct for details on panels.)

If the internal stack is nearly full when a panel is being closed the internal stack may overflow causing an error to be detected. When the error is detected the segment that caused the stack overflow is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when the segment is deleted.

ERRORS

LE02 (Level 3): Out of memory while performing END-PANEL command.

LE03 (Level 1): No panel is currently being defined.

LE09 (Level 3): Internal stack overflow.

REFERENCES

BEGIN-PANEL-BOUNDARY command
Panel-definition syntactic construct

INQUIRY COMMAND

none

END-SEGMENT Command

Host Syntax

$^{E}_{C}$SC

Setup Syntax

$^{E}_{C}$SC

DESCRIPTION

If a segment is currently being defined, this command terminates that segment definition. If a panel is currently being defined within the segment, that panel-definition is also terminated, as if an END-PANEL command had been received. See the *Segment-definition* syntactic construct.

ERRORS

SC02	(Level 3):	Out of memory while performing END-SEGMENT command.
SC03	(Level 1):	Context error (no segment is currently being defined).

REFERENCES

Panel-definition syntactic construct
Segment-definition syntactic construct

INQUIRY COMMAND

none

ENTER-ALPHA-MODE Command

Host Syntax

$^{U}_{S}$

DESCRIPTION

This command puts the terminal in alpha mode rather than vector mode or marker mode. If no dialog area is enabled and the current graphic beam position is outside the alphatext window (as set by the SET-OVERVIEW-WINDOW command), the graphic beam position is moved to the nearest point on the edge of the alphatext window.

While the terminal is in alpha mode, subsequent printable characters are interpreted as DISPLAY-ALPHA-TEXT commands. When the terminal is in alpha mode, the alpha cursor blinks at the current alpha 2D position in the graphics area if no dialog area is enabled or at the current position in the currently enabled (and visible) dialog area.

The ENTER-ALPHA-MODE command (that is, the $^{U}_{S}$ character) also terminates any command which precedes it. This is described in Section 2, under the heading "Commands of More Than Three Characters."

The terminal processes this command whether it is in alpha, vector, or marker mode.

REFERENCES

DISPLAY-ALPHA-TEXT command
Section 2, "Commands of More Than Three Characters"
SET-OVERVIEW-WINDOW command

INQUIRY COMMAND

REPORT-4010-STATUS command

ENTER-BYPASS-MODE Command

Host Syntax

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

DESCRIPTION

When the terminal is in Bypass mode, characters coming from the host are ignored. Without Bypass mode, characters echoed by the host would be received by the terminal and executed as commands. If the current *bypass-cancel-char* is ${}^{N}{}_{U}$, bypass mode cannot be entered.

Bypass mode is automatically entered when the terminal sends report messages and files to the host.

The terminal also enters Bypass mode after it processes an ENTER-BYPASS-MODE command, which is at least 25 milliseconds after it has received the command. To be sure all intended characters are bypassed, either with one-half second after the command or pad with one-half second of no-op characters (such as <SYN>) before sending the characters to be bypassed.

The terminal exits bypass mode when it receives the current bypass cancel character, as set by the most recent SET-BYPASS-CANCEL-CHAR command, and when the CANCEL command is processed or the CANCEL key is pressed.

REFERENCES

CANCEL command
CANCEL key
SET-BYPASS-CANCEL-CHAR command

INQUIRY COMMAND

none

ENTER-MARKER-MODE Command

Host Syntax

${}^{F}{}_{S}$

DESCRIPTION

This command places the terminal in marker mode rather than alpha mode or vector mode. Once in marker mode, subsequent printable characters are interpreted as *xy*+ parameters, which are interpreted as implicit DRAW-MARKER commands. When the terminal is in marker mode, the alpha cursor is not displayed.

The ENTER-MARKER-MODE command (that is, the ${}^{F}{}_{S}$ character) also terminates any command which precedes it. This is described in Section 2, under the heading "Commands of More Than Three Characters."

The terminal processes this command whether it is in alpha, vector, or marker mode.

REFERENCES

DRAW-MARKER command
Section 2, "Commands of More Than Three Characters"
XY+ parameter type

INQUIRY COMMAND

REPORT-4010-STATUS command

ENTER-VECTOR-MODE Command

Host Syntax

GS

DESCRIPTION

This command places the terminal in vector mode rather than alpha or marker mode and sets the internal move/draw flag to move. Once in vector mode, subsequent printable characters are interpreted as *xy+* parameters, which in turn are interpreted as implicit MOVE or DRAW commands (depending on the setting of the move/draw flag). After all implicit MOVE or DRAW, the move/draw flag is set to draw. When the terminal is in vector mode the alpha cursor is not displayed.

The ENTER-VECTOR-MODE command (that is, the GS character) also terminates any command which may precede it. This is described in Section 2, under the heading "Commands of Three or More Characters."

If the terminal is in marker mode, it ignores the ENTER-VECTOR-MODE command. The terminal must be in alpha mode or vector mode before it will process this command.

REFERENCES

DRAW command
ENTER-ALPHA-MODE command
ENTER-MARKER-MODE command
MOVE command
Section 2, "Commands of More Than Three Characters"
XY+ parameter type

INQUIRY COMMAND

REPORT-4010-STATUS command

EOM-Indicator Syntactic Construct

Syntax

EOM-indicator = $\left\{\begin{array}{l} \textit{EOL-string} \\ \text{block-end-of-message} \end{array}\right\}$

DESCRIPTION

*EOM-indicator*s are used by the terminal to terminate reports and to break long reports into shorter segments. All reports from the terminal end with an *EOM-indicator.*

As the terminal sends reports, it counts the characters. When the count reaches the current *report-max-line-length,* the terminal inserts an *EOM-indicator.* The terminal always sends report units such as *int-report*s and *xy-report*s in their entirety between *EOM-indicator*s.

If the terminal is not in block mode, then it sends the current *EOL-string* as each *EOM-indicator.* Having sent the *EOL-string,* the terminal pauses for the current transmit delay before sending the first character of the next line.

In block mode, the terminal terminates the current block and sets the end-of-message bit to one in the *block-control-bytes* for each *EOM-indicator.*

See the *4110/4120 Series Host Programmers Manual* for details on the uses of *EOM-indicator*s.

REFERENCES

4110/4120 Series Host Programmers Manual
Block-control-bytes syntactic construct
SET-EOL-STRING command
SET-REPORT-EOM-FREQUENCY command
SET-TRANSMIT-DELAY command

EQ Character (Enquiry)

4111

DESCRIPTION

Receipt of this character by the terminal causes the answerback message to be sent to the host. If the terminal is in local mode, the EQ character is treated as a no-op and the answerback message is not transmitted.

The terminal does not enter bypass mode when transmitting the answerback message or append optional EOM indicators to the message. If you want to suppress the echo of the message from the host, you must do so at the host.

REFERENCES

SET-EOM-CHARS command
SET-ANSWERBACK-MESSAGE command
SET-REPORT-MAX-LINE-LENGTH command

Error-Report Message Type

Syntax

```
error-report  =  [report-for-one-error . . . ]
                 [term-sig-char]
                 EOM-indicator

report-for-one-error  =  [char-report: sig-char]
                         char-report: first-command-char
                         char-report: second-command-char
                         char-report: parameter-number
                         char-report: error-type
                         int-report: severity-level
                         int-report: error-count
                         [EOM-indicator]
```

DESCRIPTION

The terminal sends an *error-report* message in response to a REPORT-ERRORS command. In that *error-report*, there is one *report-for-one-error* message for each of the eight most recently detected error codes. If fewer than eight errors have been detected since power-up or since the last REPORT-ERRORS command then there are fewer than eight *reports-for-one-error* in the *error-report* message.

The *sig-char* is the one for non-GIN reports, set by the SET-REPORT-SIG-CHARS command.

Besides the *sig-char*, each *report-for-one-error* includes a four-character error code, a severity level number, and the number of times the terminal has detected that error since power-up or the last REPORT-ERRORS command. (If the terminal has detected the error more than 255 times, it reports that the error has occurred 255 times.) The error codes and severity levels are described in Appendix C.

After the last *report-for-one-error*, the terminal sends a *term-sig-char* and an *EOM-indicator*.

REFERENCES

Appendix C, "Error Codes"
REPORT-ERRORS command
SET-REPORT-SIG-CHARS command

^{E}C Character

DESCRIPTION

The ASCII "escape" character — ^{E}C — is used to signal the terminal that the next few characters comprise a command for it. When this character begins a command, it also terminates any command currently being sent to the terminal.

REFERENCES

Section 2, "Command Syntax"

EXPAND-MACRO Command

Host Syntax

$^{E}{}_{C}$KX int:*macro-number*

Setup Syntax

$^{E}{}_{C}$KX int:*macro-number*

PARAMETERS

macro-number (–32768 to –32742, –32740 to –32737, –32608 to –32513, –178 to –2, and 0 to 32767)
The number of the macro whose definition is being invoked.

DESCRIPTION

The EXPAND-MACRO command causes the terminal to insert into its current input data stream the contents of a macro definition. It is a companion command to the DEFINE-MACRO command, which defines a macro.

If the EXPAND-MACRO command comes from the host computer, it is treated by the terminal as if the host computer had sent the contents of the macro which is being expanded.

If the EXPAND-MACRO command comes to the terminal from a disk file which is being loaded, it is treated by the terminal as if the disk file had included the contents of that macro.

If the EXPAND-MACRO command is entered from the keyboard while the terminal is in Local or Setup mode, the macro contents are treated as input from the keyboard.

Any macro which is being expanded may contain byte-macros or other expand-macro commands. (Information on byte-macros can be found in the DEFINE-MACRO command description.) Commands may be nested this way to a nesting depth limited only by the number of defined macros. However, if one of these commands attempts to expand a macro which is currently being expanded at a higher nesting level, the command is not executed; that is, recursion does not occur in macro expansions.

The rest of this description discusses how the three types of macros can be expanded.

Byte-macros. Byte-macros are expanded when the 8-bit ASCII character that corresponds to the byte-macro's macro-number is received by the terminal's command processor or the *macro-number* is included as the *macro-number* parameter of the EXPAND-MACRO command. The character may come from the host, from the keyboard when the terminal is in ECHO YES or Local mode, or from the data stream that results from a LOAD command.

Byte-macros are expanded in Setup mode only if they are included as the *macro-number* parameter of the EXPAND-MACRO command. If the character to which the byte-macro corresponds is typed on the keyboard (without an EXPAND-MACRO command), the byte-macro is not expanded.

Any character within the byte-macro's macro-contents which corresponds to a byte-macro is also expanded (because that character is received by the command processor). Any EXPAND-MACRO command in the macro-contents expands the appropriate key- or host-macro, including any byte-macros that are part of the key or host-macro expansion.

While a byte-macro is being expanded, the byte to which it corresponds is no longer recognized as having a byte-macro definition (unmapped) for as long as it takes to complete the expansion. This prevents the occurrence of an infinite loop. Once the macro is fully expanded, the byte is again recognized as a byte-macro (remapped).

If macros are nested, each byte-macro is unmapped and remapped each time its expansion is begun and finished, so that there may be more than one expansion of a byte-macro within another macro so long as that individual byte-macro is not nested within itself.In other words, recursive macro expansion is not allowed.

To access the byte-macros corresponding to the 8-bit ASCII values 160 through 255 (macro-numbers -32608 through -32513), you must use either 8-bit packing in block mode, or data parity with control over the eighth bit so that you can send the corresponding characters to the terminal command processor.

Key-macros. Key-macros are expanded with the EXPAND-MACRO command or by pressing the key (or combination of keys) on the terminal keyboard to which the macro corresponds. The expansion of key-macros by pressing a key cannot be nested, since the terminal cannot press one of its own keys.

When a key-macro is expanded with an EXPAND-MACRO command received from the host, or from the keyboard when the terminal is in ECHO YES or Local mode, any EXPAND-MACRO commands or byte-macros in its contents are expanded as if they came from the host.

When a key-macro is expanded by its corresponding keystroke(s), and Setup mode is disabled and the terminal is not in Local mode, the macro-contents are sent to the host communication port as if the they had been entered at the keyboard. You may use the *key-execute-character* within a key-macro's macro-contents to toggle the flow of the macro-contents back and forth between the terminal's command processor and the host communication port. This method only works if the key-macro was expanded by a keystroke and the terminal is not in Setup mode. If Local mode or local echo are enabled, the contents, including the *key-execute-character*, are sent to the command processor, so that any EXPAND-MACRO commands or byte-macros in the contents are treated as if they had come from the host.

When a key-macro is expanded while Setup mode is enabled (either with keystroke(s) or with the expand-macro command), all characters within the macro-contents are treated as if they had been typed into the keyboard with Setup mode enabled, including EXPAND-MACRO and other terminal commands.

EXPAND-MACRO commands or keystrokes for macros that are already being expanded are ignored. Recursive macro expansions are not allowed.

With thumbwheel GIN enabled, pressing a key with a macro defined for it will cause a GIN event for each character in the macro which would normally be sent to the host (see the ENABLE-GIN command).

Host-macros. Host-macros are expanded with the EXPAND-MACRO command.

If the EXPAND-MACRO command is received from the host or while the terminal is in Local mode, any byte-macros within the macro-contents are expanded (unless they are already being expanded), as are any EXPAND-MACRO commands.

EXPAND-MACRO commands for macros that are already being expanded are ignored. Recursive macro expansions are not allowed.

If the EXPAND-MACRO command is entered when the terminal is in SETUP mode, the contents of the specified macro are treated as if they had been typed on the keyboard while the terminal was in Setup mode.

DEFAULTS

macro-number
as shipped — none
on power-up — none
if omitted — 0

ERRORS

KX11 (Level 2): Invalid *macro-number*.

REFERENCES

DEFINE-MACRO command
ENABLE-GIN command
LOAD command
SET-KEY-EXECUTE-CHARACTER command

INQUIRY COMMAND

none

Fill-Pattern-Definition Syntactic Construct

Syntax

fill-pattern-definition = BEGIN-FILL-PATTERN
[*pixel-def . . .*]
[END-FILL-PATTERN]

pixel-def = RUNLENGTH-WRITE
or RASTER-WRITE

DESCRIPTION

To define a fill pattern (for later use in filling panels):

1. Issue a BEGIN-FILL-PATTERN command.
2. Issue one or more RASTER-WRITE and/or RUNLENGTH-WRITE commands to specify the content of the fill-pattern rectangle as a matrix of color-indices.
3. Issue an END-FILL-PATTERN command to explicitly terminate a *fill-pattern-definition* before defining all indices in the pattern rectangle. The remaining indices default to index 0. (This command is ignored if the RASTER-WRITE or RUNLENGTH-WRITE operations filled the fill-pattern rectangle.)

When enough indices are received to fill the fill-pattern rectangle, the *fill-pattern-definition* is implicitly ended. If more color-indices then are needed are received within the final RASTER-WRITE and RUNLENGTH-WRITE command, the extras are discarded.

REFERENCES

BEGIN-FILL-PATTERN command
END-FILL-PATTERN command
RASTER-WRITE command
RUNLENGTH-WRITE command

FORMAT-VOLUME Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

$^{E}{}_{C}$**JF** device:*disk-drive* {**/C**} {/int:*track*}

Setup Syntax

FORMAT $^{S}{}_{P}$ *disk-drive* {**/C**} {*/track*}

PARAMETERS

disk-drive

Specifies the drive containing the disk you want to format. Recognized disk drives are:

F0:	the Option 42 and 43 disk drives.
F1:	
S0: — Z7:	the Option 45 disk drives.

/C

Include the optional parameter **/C** to format a Tektronix 4926 10-megabyte hard disk for compatibility with 4110/4120 Series Local Programmability.

/track

Include this optional parameter to specify known bad tracks on a hard disk. Valid track numbers are **/1** to **/65535**. Within that range, the maximum bad track value depends on the type of hard disk in use. (For example, the highest track number on a Tektronix 4926 hard disk is track 1223.)

DESCRIPTION

The FORMAT-VOLUME command formats a disk on the specified disk drive. Formatting is necessary before using a disk because it verifies which areas of the disk are good and builds directory information.

If a previously formatted disk is formatted again, the existing directory and any files on the disk are wiped out; the space is reused for the new directory and files. When a disk is formatted, all files, even those that have been write-protected with the PROTECT-FILE command, are deleted. (However, if you are trying to format a disk that is write-protected by the disk's write-protect notch or the terminal's write-protect switch, then the disk is not formatted and error JF13 or JF19 occurs.)

The FORMAT-VOLUME command creates a predetermined number of directory entries on the disk, depending on the type of disk:

Option 42, 43 (8" flexible disks)	320 entries
Tektronix 4925 Dual Flexible Disk Drive (5-W" flexible disks)	64 entries
Tektronix 4926 Hard Disk Drive (10-megabyte hard disk)	2048 entries

To format a 10-megabyte hard disk (such as the Tektronix 4926) for Local Programmability, enter the optional parameter **/C** after the device name. The **/C** argument formats the hard disk for compatibility with Local Programmability.

To format a hard disk and exclude known bad tracks from use, you may specify up to five bad track numbers after the device name. If Local Programmability is specified, then the limit is four. (A list of bad track numbers may be shipped with the drive from the disk drive manufacturer.) Express bad tracks in the form/*track*, where *track* is an integer from 1 to 65535. For example, to format a hard disk that has bad tracks 25, 145, 856, 299, and 1187, enter these parameters:

```
W2:/25/145/856/299/1187
```

If you are also formatting the disk for Local Programmability, include the **/C** parameter, like this:

```
W2:/C/25/145/856/299/1187
```

Listing the known bad tracks prevents them from being accessed, but allows the rest of the disk to be used normally.

During formatting, the terminal writes to the disk and immediately reads what it has written. This detects any "bad blocks" (any defective parts of the disk surface). If the terminal finds any bad blocks, it does not record them. Instead, it detects and maps out up to 200 bad tracks (in addition to the five bad tracks you may have specified). These additional bad tracks are not accessed during disk operations. If the terminal detects more than 200 bad tracks, then it halts the formatting operation; as a result, the disk is left unformatted and a type JF13 error occurs.

DEFAULTS

disk-drive
as shipped — none
on power-up — none
if omitted — error JF11

ERRORS

JF00 (Level 0): Unrecognized command. (Disk drive option not installed.)

JF10 (Level 2): *Device* is not installed.

JF11 (Level 2): Invalid *device* specifier.

JF12 (Level 3): Out of memory while parsing the parameter.

JF13 (Level 2): The device specified is not a disk drive, is write-protected, is busy, detects a verify error, detects a bit map error, or is not mounted.

JF19 (Level 2): Hardware error at the specified disk drive. (Format error, drive not ready, or write-protect switch or notch error.)

REFERENCES

DIRECTORY command
DISMOUNT command
PROTECT-FILE command

INQUIRY COMMAND

none

GIN-REPORT Message Type

Syntax

```
                        ⎧ [EOM-indicator][1]              ⎫
                        ⎪ GIN-locator-report              ⎪
                        ⎪ [EOM-indicator][3]              ⎪
GIN-report =            ⎪ [EOM-indicator][1]              ⎪
                        ⎪ GIN-pick-report                 ⎪
                        ⎨ [EOM-indicator][3]              ⎬
                        ⎪ [EOM-indicator][1]              ⎪
                        ⎪ GIN-stroke-report               ⎪
                        ⎪ [EOM-indicator][3]              ⎪
                        ⎪ GIN-termination-report          ⎪
                        ⎩                                 ⎭
GIN-locator-report = [char-report:sig][2]
                     Gin-key-report
                     xy + -report: GIN-location
                     [int-report:view-number][4]
GIN-pick-report = [char-report:sig][2]
                  GIN-key-report
                  xy + -report:GIN-location
                  [int-report:view-number][4]
                  segment-pickID-report
segment-pickID-report = int-report:segment-number
                        int-report:pickID-number
                           or
                        int-array-report:segment-number/pickID
GIN-stroke-report = [char-report:sig][2]
                    GIN-key-report
                    xy + -report:GIN-location
                    [int-report:view-number][4]
GIN-termination-report = [term-sig-char][2]
                         EOM-indicator
GIN-key-report = char-report:key
```

[1] This *EOM-indicator* is sent only if: (1) at least one character has been sent since the last *EOM-indicator* and (2) this GIN-report would cause the maximum report line length to be exceeded.

[2] A *sig-char* or *term-sig-char* is not sent if it is NL.

[3] An *EOM-indicator* occurs at the end of each *GIN-locator/pick/stroke-report* only if the most recent SET-Report-EOM-FREQUENCY command specifies *EOM-frequency* to be "more frequently".

[4] A *view-number* is sent only when the current GIN report format is 4, 5, 6, or 7.

DESCRIPTION

The terminal sends a *GIN-report* to the host computer when GIN is enabled (see ENABLE-GIN command) and a "GIN event" occurs. A single GIN pick event can cause more than one GIN report to be sent when the GIN report format is 2, 3, 6, or 7 (see SET-GIN-REPORT-FORMAT).

More than one GIN device can be enabled at a time. When more than one GIN device is enabled, the *GIN-reports* sent to the host computer may be intermixed. Signature characters can be used to distinguish the *GIN-report* for one GIN device from the *GIN-report* for another GIN device, and also to distinguish the *GIN-reports* of different functions. (See SET-REPORT-SIG-CHAR command).

Unless the bypass-cancel-character is ^{N}L, the terminal enters bypass mode when it sends a report to the host computer. In bypass mode the terminal ignores all characters received from the host. (See ENTER-BYPASS-MODE command.)

A GIN-report consists of a possible *EOM-indicator*, a signature character, an *xy + -report*, a possible view number *int-report*, pick information for pick reports and another possible EOM indicator. Figure 7-9 is a flow chart of the GIN-Report Message Type that shows the possible contents of *GIN-reports*.

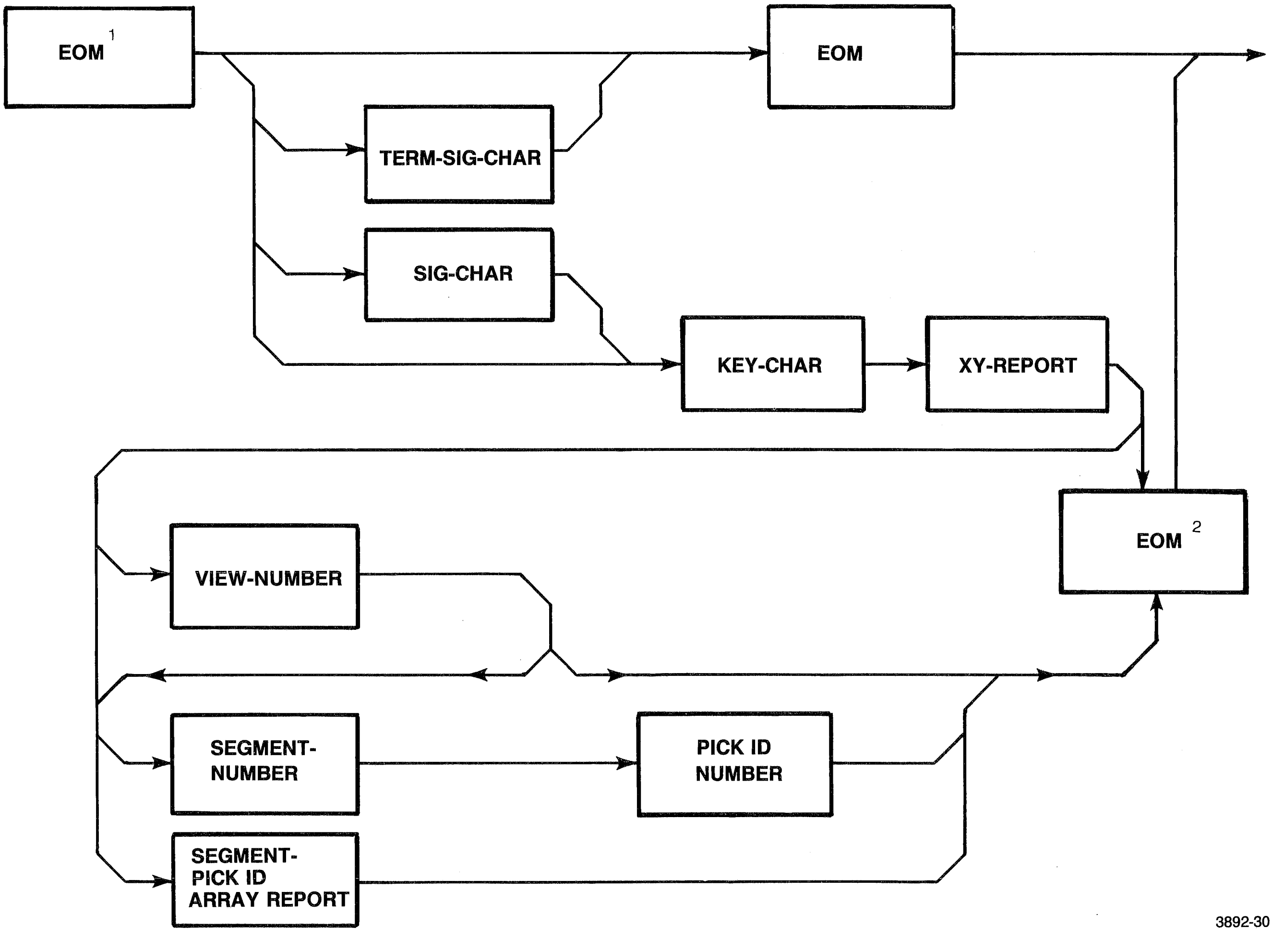


Figure 7-9. GIN-Report Flowchart

EOM-Indicators

An EOM-indicator is sent as an EOL-string when the terminal is not in block mode, and as the end-of-block when the terminal is in block mode. (See EOM-indicator syntactic construct for details).

An *EOM-indicator* will be sent at the start of a *GIN-report* (other than *GIN-termination-report*) to keep the report maximum line length from being exceeded. This condition happens when *EOM-frequency* (SET-Report-EOM-FREQUENCY) is **0** (less frequent), *max-line-length* (SET-Report-MAX-LINE-LENGTH) is set so a line can hold two or more *GIN-report* items, and enough *GIN-report* items have been sent on the current line so that one more *GIN-report* item would cause the maximum line length to be exceeded. The terminal automatically inserts this *EOM-indicator* when maximum line length is going to be exceeded.

An *EOM-indicator* is sent at the end of a *GIN-report* (other than *GIN-termination-report*) when *EOM-frequency* is set to 1 (more frequently).

An *EOM-indicator* is always sent in the *GIN-termination-report*.

Signature Character. Different signature characters can be set for different device-functions, so that the originating device and the function can be deduced from the signature character.

The most recent SET-REPORT-SIG-CHARS command for the particular GIN *device-function* code determines the two signature characters (*sig-char* and *term-sig-char*). For details, see the description of the SET-REPORT-SIG-CHARS command.

Key. The key *char-report* indicates which button or key was pressed or released to signal the "GIN event."

Every character which would be sent to the host as the result of pressing or releasing a key or button on an enabled device causes a GIN-event. This includes all the contents of a key-macro. For example, pressing a keyboard key with a macro defined for it with five characters in it (with no key-execute-toggle characters) causes five thumbwheel GIN events.

The release of a button or key triggers a GIN-event only if the mode M is 1 in the *GIN-code* of the ENABLE-GIN command that enabled the device. This value for M indicates that GIN events should occur on key depression and also on key release (see ENABLE-GIN command).

GIN-Location. The *GIN-location xy+-report* specifies the GIN device position or a point near a picked 3D segment (in terminal space coordinates) at the time of the event. The contents of the *GIN-location* report depend on the dimensionality of the GIN cursor segment and the coordinate mode of the terminal.

When the terminal is in 2D mode (coordinate mode 0 or 1) and a 2D segment is used as the GIN cursor, the *xy+-report* contains the x and y coordinates of the 2D GIN location. (Segment 0 (the crosshair cursor) is considered to be a 2D segment.)

When the terminal is in 2D mode and a 3D segment is used as the GIN cursor, the *xy+-report* contains only the x and y coordinates of the 3D GIN location.

When the terminal is in 3D mode (coordinate mode 2) and a 2D segment is used as the GIN cursor, the *xy+-report* for locate and stroke functions contains the x and y coordinates of the 2D GIN locations and a z of 0. The *xy+-report* for a pick function depends on the dimensionality of the picked segment. If a 2D segment (or none) is picked, then the *xy+-report* is the same as for locate and stroke. If a 3D segment is picked, the *xy+-report* contains xyz coordinates near the picked primitive in the 3D segment. This report for wireframe images may be different than the report for shaded images.

When the terminal is in 3D mode and a 3D segment is used as the GIN cursor, the *xy+-report* contains the x,y, and z coordinates of the 3D GIN location.

Stroke and pick functions are not allowed with 3D GIN cursors in any coordinate mode.

See the SET-GIN-CURSOR command and the *XY+-report* message type for further details.

View-Number. The *view-number int-report* indicates the view the event occurred in. It is sent only when the current GIN report format is 4 to 7 (bit 2 is 1). See SET-GIN-REPORT-FORMAT command.

Segment-PickID-Report. A *segment-pickID-report* contains either a segment number *int-report* and pickID number *int-report* or an *int-array-report* for segment number and pickID number. The *int-array-report* is sent instead of the two *int-report*s when the current GIN report format is 1, 3, 5, or 7. See the SET-GIN-Report-FORMAT command for details.

Segment-Number. The *segment-number int-report* gives the number of the segment being "picked." If no visible, detectable segment with a non-zero pickID falls within the current pick aperture, this *int-report* is 0.

PickID-Number. The *pickID-number int-report* gives the "pick identification number" of a part of the segment within the pick aperture. If no visible-detectable segment with a non-zero pickID falls within the pick aperture, then this *int-report* is 0. This pickID is the one associated with the picked primitive, at the lowest level called segment (if any).

Segment-Number/PickID. This *int-array-report* contains segment-number/pickID pairs which define the pick path through the segment calling tree. The first segment-number/pickID pair is for the top-level segment and the last segment-number/pickID pair is for the deepest called segment. The last pickID is the one associated with the picked primitive, while the other pickIDs are the ones in effect for each of the segment calls.

GIN-Termination-Report

The terminal sends a *GIN-termination-report* to the host computer when the graphic input function is disabled. This occurs when the ENABLE-GIN command's "count" is exhausted, when the terminal receives a DISABLE-GIN or CANCEL command, or when the operator presses the CANCEL key. The *GIN-termination-report* consists of a *term-sig-char*, followed by an *EOM-indicator*.

The *term-sig-char* is set by the SET-REPORT-SIG-CHARS command.

REFERENCES

CANCEL command
CANCEL key
DISABLE-GIN command
ENABLE-GIN command
EOM-indicator syntactic construct
SET-GIN-CURSOR command SET-GIN-REPORT-FORMAT command
SET-REPORT-EOM-FREQUENCY command
SET-REPORT-MAX-LINE-LENGTH command
SET-REPORT-SIG-CHARS command
XY+report message type

GRAPHIC-TEXT Command

Host Syntax

$^{E}{}_{C}$**LT** string:*text-string*

Setup Syntax

$^{E}{}_{C}$**LT** $^{S}{}_{P}$ *text-string*

PARAMETERS

text-string
A string that specifies the characters to be displayed.

DESCRIPTION

This command displays both string precision and stroke precision graphtext. String precision graphtext is text displayed in the same manner as alpha text in the graphics area. See the DISPLAY-ALPHA-TEXT command for details. Stroke precision graphtext is text displayed as vectors, but can only be used when the terminal is in 2D mode (coordinate mode 0 or 1).

To display stroke precision graphtext characters, the terminal first determines if the character will fit inside terminal space. If the final beam position is outside terminal space, the character is not displayed (not even partially). The terminal then finds the character vector definitions in the font tables.

Each vector is scaled, slanted, and rotated about the character's origin, translated to the graphic beam position, and displayed. The vector's x and y coordinates are scaled by multiplying by the graphtext-size width/height and then dividing by the font grid width/height. The scaled result is then slanted by adding y times the tangent of the slant angle to x. The slanted result is then rotated. The rotated result is then translated by adding it to the current beam position, and it is displayed (this is actually done by a matrix transform). As they are displayed, the vectors are clipped at the edge of the window. Finally, the terminal moves the graphic beam position to the ending point of the character.

The appearance of stroke precision graphtext is governed by the most recent SET-TEXT-INDEX, SET-GRAPHTEXT-SIZE, SET-GRAPHTEXT-FONT, SET-GRAPHTEXT-SLANT, and SET-GRAPHTEXT-ROTATION commands. The vectors are always solid, regardless of the current line style. The font grid width and height are 6 and 8 for the predefined fonts and as set by the SET-GRAPHTEXT-FONT-GRID command for user-defined fonts.

A GRAPHIC-TEXT command within a panel-definition, or when the terminal is in coordinate mode 2 when precision is set to stroke, causes the terminal to detect a type LT03 error.

If the internal stack is nearly full when you attempt to use this command (due to nested CALL-SEGMENT commands ot INCLUDE-COPY-OF-SEGMENT commands), the internal stack may overflow causing an error to be detected. When the error is detected the segment that causes the stack overflow is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when the segment is deleted.

DEFAULTS

text-string
as shipped — none
on power-up — none
if omitted — empty string

ERRORS

LT03 (Level 2): Command is invalid at this time. (Graphtext is not allowed within a panel-definition or stroke precision graphtext is not allowed in coordinate mode 2.)

LT09 (Level 3): Internal stack overflow.

LT11 (Level 2): Invalid array count (must be in range from 0 to 65535).

LT12 (Level 3): Out of memory while parsing the parameter.

REFERENCES

DISPLAY-ALPHA-TEXT command
SET-GRAPHTEXT-FONT command
SET-GRAPHTEXT-FONT-GRID command
SET-GRAPHTEXT-PRECISION command
SET-GRAPHTEXT-ROTATION command
SET-GRAPHTEXT-SIZE command
SET-GRAPHTEXT-SLANT command
SET-TEXT-INDEX command

INQUIRY COMMAND

none

Graphtext-Font-Definition Syntactic Construct

Syntax

```
font-definition  =  SET-GRAPHTEXT-FONT-GRID
                    [char-definition, . . . ]

char-definition  =  [SET-PIVOT-POINT]
                    BEGIN-GRAPHTEXT-CHARACTER
                    [poly-line, . . . ]
                    END-GRAPHTEXT-CHARACTER

poly-line  =  [MOVE]
              [DRAW, . . . ]
```

DESCRIPTION

A graphtext font is a list of MOVE and DRAW commands remembered locally in the terminal's memory that can be displayed by the GRAPHIC-TEXT command when the graphtext precision is set to 'stroke'. The main advantage of user-defined graphtext fonts is that any font which can be defined as vectors can be defined in the terminal for fast access and display.

Before any characters in a font can be defined, the font grid must be established by the SET-GRAPHTEXT-FONT-GRID command. This command sets up terminal memory to accept character definitions, and sets the grid size for the font. The grid is not used during character definition, but is used in the scaling of the characters when they are displayed. It is possible to redefine some or all of a predefined font; the font grid you set for it applies only to the characters you define and not to the predefined characters (whose grid is 6x8). When a defined character is deleted, if the font is predefined, the predefined character definition is restored.

Before each character you may either set the pivot point for that character's definition with the SET-PIVOT-POINT command or you may leave the pivot point at its current setting.

The BEGIN-GRAPHTEXT-CHARACTER command opens the character definition. Any MOVE and DRAW commands are remembered as part of the character after subtracting the value of the pivot point from the xy of the MOVE or DRAW. For example, if the pivot point is 20,30 and a MOVE to 25,38 is received, the terminal remembers a MOVE to 5,8 in the character definition. No other commands affect the contents of the character definition. The DRAW commands cause vectors to be drawn as they would be otherwise. The xy coordinates of the MOVEs and DRAWs must be in the range 0 to 4095; if any are outside this range, an error is detected and the definition is closed.

The END-GRAPHTEXT-CHARACTER command closes the character definition.

Although the font grid does not restrict the vectors of the character definitions, it does define a virtual upper-case character size for the font. That is, most upper-case characters will just fill the grid. Lower-case characters with descenders and some special characters will extend outside the grid.

HARDCOPY Command

Host Syntax

```
EcKH   int:hard-copy-code
```

Setup Syntax

```
EcKH Sp   hard-copy-code
```

PARAMETERS

hard-copy-code (0, 1, 2, or 3)
Selects the type of hardcopy operation that is generated with respect to the type generated by the HARD COPY key.

0	Same as the HARD COPY key.
1	Same as the HARD COPY key.
2	Same as the Shift-HARD COPY key.
3	Same as CTRL-HARD COPY key.

DESCRIPTION

This command causes the terminal to generate a hardcopy as if the HARD COPY key had been pressed. See HARD COPY key for details.

DEFAULTS

hard-copy-code
as shipped — none
on power-up — none
if omitted — 0

ERRORS

KH01	(Level 2):	Copier fault condition; operator assistance required (Option 19 only).
KH03	(Level 2):	Image too large for copier (Option 19 only).
KH11	(Level 2):	Invalid *hard-copy-code*.
KH13	(Level 2):	Wrong copier connected (Option 19, version 5 or later).
KH19	(Level 2):	Color hardcopy device not ready (Option 19 only).

REFERENCES

HARD COPY key

INQUIRY COMMAND

none

HARD COPY Key

DESCRIPTION

Pressing the HARD COPY key generates a full-screen hard copy. This requires that a TEKTRONIX 4632 or 4612 Video Hard Copy Unit, or a TEKTRONIX 4690 Series color copier, be attached to the appropriate hardcopy interface on the back of the terminal.

During a hard copy, terminal operation is suspended. Input from the host is queued.

When and how the copy is made depends on the copier interface currently selected and whether SHIFT or CTRL is held down when the HARD COPY key is pressed. The monochrome interface is standard, The color copier interface is standard on the 4111 and is available as Option 19 on other terminals.

The full-screen area is larger than some of the maximum color copier image areas as shown on the chart in the description of the SET-IMAGE-ORIENTATION command. When this condition exists, the HARDCOPY key will cause an error to be detected. To avoid this, set the form width, image orientation, and/or media size to a larger size.

Monochrome Copiers. Pressing HARD COPY causes an "inverted" image to be copied, white appearing darkest and black appearing lightest, and other colors at gray levels between. Pressing SHIFT-HARD COPY causes a "normal" image to be copies, white as white and black as black. While the copy is being made, the screen is blanked. If no hard copy unit is attached, the screen flashes once.

CAUTION

While the terminal is making a color hard copy, do not press the HARD COPY key on the monochrome copier. It will lock up the terminal.

Color Copiers. Pressing HARD COPY causes an "inverted" image to be copied, with white and black only copied as black and white; other colors are copied as themselves. Pressing SHIFT-HARD COPY causes a normal image to be copied, white as white and black as black. Pressing CTRL-HARD COPY causes the dialog area to be copied to a 4695; if another copier (or none) is attached, an error is detected.

The HARD COPY key does not auto-repeat.

REFERENCES

HARDCOPY command
SELECT-HARDCOPY-INTERFACE command
SET-COLORHARDCOPY-FORM-WIDTH command
SET-DIALOG-HARDCOPY-ATTRIBUTES command
SET-IMAGE-ORIENTATION command
SET-PIXEL-VIEWPORT command

HT Character

DESCRIPTION

The effect of an HT character depends on whether a dialog area is enabled and on the current coordinate mode of the terminal.

The HT character causes different actions depending on whether the terminal is in TEK mode or ANSI mode (as set by the SELECT-CODE command). See the ANSI Commands section in this manual for details on how to set tab stops and on what happens when the terminal is in ANSI mode.

Alpha Mode. When the terminal is in alpha mode and the dialog area is disabled, the HT "horizontal tab" character is interpreted as a SP character.

If the dialog area is enabled, an HT causes the alpha cursor to move right to the next defined tab stop or the right edge of the dialog area if no tab stops are encountered.

Marker and Vector Modes. When the terminal is in marker mode or vector mode, the HT character has no effect.

REFERENCES

ANSI Commands section
ENTER-ALPHA-MODE command
ENTER-MARKER-MODE command
ENTER-VECTOR-MODE command
HT CHARACTER ANSI command
SELECT-CODE command

INQUIRY COMMAND

none

IGNORE-DELETES Command

Host Syntax

ECKI int:*ignore-deletes-mode*

Setup Syntax

IGNOREDEL SP *ignore-deletes-mode*

PARAMETERS

ignore-deletes-mode (0 or 1)
Specifies whether DT characters should be ignored or not. Setup mode parameters are **YES** and **NO**.

0 **NO**; do not ignore DT characters.

1 **YES**; ignore DT characters

DESCRIPTION

This command determines if the terminal parser will ignore DT (delete) characters. If the parameter is 1, the terminal ignores any DT characters that it receives. If this is the case, the host should substitute **EC?** for any DT characters it would send when sending *int* or *xy* parameters to the terminal.

If the parameter is 0, the terminal parser does not ignore DT characters.

DEFAULTS

ignore-deletes-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

KI11 (Level 2): Invalid *ignore-deletes-mode.*

REFERENCES

Delete-equivalent syntactic construct
Int parameter type
XY parameter type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

INCLUDE-COPY-OF-SEGMENT Command

Host Syntax

```
EcLK    int:segment-number
```

Setup Syntax

```
EcLK Sp segment-number
```

PARAMETERS

segment-number (–3, –1, or 1 to 32767)
Specifies the segment to be included.

–3	all existing segments that match the current matching class and whose dimensionality matches the terminal's coordinate mode
–1	all existing segments whose dimensionality matches the terminal's coordinate mode
1 to **32767**	the specified segment

DESCRIPTION

This command causes the following to occur:

- If a panel is being defined, the definition is terminated.
- The current primitive attributes are saved. These include the graphic beam position, line style, line index, text index, marker type, pick ID, graphics area writing mode, background indices, panel style, fill pattern, and 4014 alphatext size.
- The specified segment (scaled, rotated, and positioned according to its current transform) is copied into the graphics area and any segment being defined. The initial default pick-ID is not copied, but all explicitly set pick-IDs are copied. None of the included segment's attributes, such as highlighting or detectability, are carried over into the new segment.
- The primitive attributes that were saved are restored.
- The included segment is displayed in the writing mode of segment –2, the future segment.

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist or if the dimensionality of the specified segment does not match the coordinate mode of the terminal.

Segment –1 refers to all currently defined segments (with numbers from 1 to 32767) whose dimensionality matches the current coordinate mode. Segment –3 refers to all segments that match the current segment matching class and whose dimensionality matches the current coordinate mode of the terminal.

NOTE

> *Including a copy of a segment whose writing-mode is XOR while segment -2's writing mode is XOR causes the included segment's image to be erased.*

On the 4128 and 4129, the dimensionality of the included segment must match the current coordinate mode of the terminal. Thus a copy of a 2D segment may be included only in coordinate modes 0 and 1, and a copy of a 3D segment may be included only in coordinate mode 2.

Because each INCLUDE-COPY-OF-SEGMENT command segment uses internal stack memory, it is possible to overflow the internal stack. This internal stack can support about 40 nested inclusions. An included segment is nested when an included segment is included in another segment. When stack overflow occurs an error will be detected. When the error is detected the segment that caused the stack overflow isnot displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when the segment is deleted.

If a segment contains an included segment which had a non-identity transform when it was included, when it is saved with the SAVE command and reloaded with the LOAD command, the image of the included segment may be slightly different. This is because the included transformed primitives are saved as integral values.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — LK11

ERRORS

LK02	(Level 3):	Out of memory while performing INCLUDE-COPY-OF-SEGMENT.
LK09	(Level 3):	Internal stack overflow.
LK10	(Level 2):	Segment does not exist.
LK11	(Level 2):	Invalid *segment-number.*
LK13	(Level 2):	The segment specified is currently being defined, or, on a 4128 and 4129 terminal, the dimensionality of the segment does not match the current coordinate mode of the terminal.
SO00	(Level 4):	INCLUDE-COPY-OF-SEGMENT commands are nested too deep.

REFERENCES

LOAD command
Panel-definition syntactic construct
SAVE command
SET-SEGMENT-WRITING-MODE command

INQUIRY COMMAND

none

INSERT-INTO-SEGMENT Command

4111, 4115P51, OR 4120 SERIES

Host Syntax

```
EcUI int:segment-number
     int:pickID-group
     int:beforeorafter
```

Setup Syntax

```
EcUI Sp  segment-number
         pickID-group
         beforeorafter
```

PARAMETERS

segment-number (1 to 32767)
Segment to which primitives and primitive attributes will be added.

pickID-group (–1 and 1 to 32767)
Specifies the position in the segment at which additional primitives and primitive attributes will be added.

–1	The segment end.
1	The segment beginning.
2 to **32767**	A particular pickID group.

beforeorafter (0 to 2)
Specifies where insertion occurs.

0	Before pickID group.
1	After pickID group.
2	Just after the pickID that starts this pickID group.

DESCRIPTION

This command reopens an existing segment to allow insertion of new primitives and primitive attributes. The command initiates the first part of a three-step segment editing sequence:

1. In response to this command, most primitive attributes are reset to their defaults, then the attributes and beam position are updated by scanning the segment up to the point of insertion. See "Effect of Reopening a Segment" (which follows) for details. The updated attributes and beam position are remembered for possible restoration later.
2. Any primitive and primitive attribute commands received are entered into the segment beginning at the insertion point.
3. When editing is terminated by an END-SEGMENT command, the attributes and beam position remembered in the first step may be restored, depending on the current segment-edit-mode. See "Effect on Trailing Portion" for details. The remainder of the segment is updated by the next PAGE or RENEW-VIEW command.

The insertion point is identified by *pickID-group* and *beforeorafter*. A pickID group is defined as the sequence of primitives and primitive attributes that follow the pickID-number, up to, but not including the next pickID-number (other than 0 and a repetition of the same pickID-number).

For *pickID-group*s 2 to 32767, a *beforeorafter* of 0 opens for insertion at the end of the preceding pickID group (i.e., before the pickID-number specified). A *beforeorafter* of 1 opens for insertion at the end of the specified pickID group and a *beforeorafter* of 2 opens for insertion immediately after the pickID-number specified.

For a *pickID-group* of –1, all values for *beforeorafter* open for insertion at the end of the segment. For a *pickID-group* of 1, a *beforeorafter* of 0 or 2 opens for insertion at the very beginning of the segment, while a *beforeorafter* of 1 opens for insertion at the end of the first pickID group.

Primitives and primitive attributes in the trailing portion of the segment (that is, after the insertion point) are "hidden away" until the segment is closed. These trailing primitives will not be visible if PAGE is done while still in insert mode.

Inserted primitives are transformed according to the segment's image or matrix transform. However, if the *pickID-group* is found within the contents of an included segment that had a non-default image transformation when it was included, an error is detected (see INCLUDE-COPY-OF-SEGMENT command).

If the dimensionality of the specified segment does not match the current coordinate mode of the terminal, an error is detected and the segment is not opened.

Effect of Reopening a Segment. At the time the segment is reopened, most primitive attributes are reset to their default values. Then the segment is scanned up to the insertion point, and any primitive attribute commands encountered during the scan change the value of those attributes. The graphic beam position is also updated during this scan to be where it was at the time the segment was created.

The following primitive attributes are changed by this command:

line-style
line-index
line-gap-index
text-index
text-background-index
4014-alphatext-size
marker-type
fill-pattern (2D)
panel-filling-mode (2D)
draw-boundary-mode (2D)
object-surface-color (3D)
translucency-pattern (3D)

If these primitive attributes are not set in the segment, they will be left at their power-up defaults when editing is terminated.

The following primitive attributes are *not* changed by this command:

graphtext-font
graphtext-precision
graphtext-rotation
graphtext-size
graphtext-slant

The current setting of the attributes may be inquired with the REPORT- TERMINAL-SETTINGS command.

Effect on Trailing Portion. The effect of inserted attribute commands on the trailing portion of the segment depends on two factors:

- If the segment-edit-mode is 0 or 1, the previously remembered primitive attributes are restored when the segment is closed by adding any required attributes ahead of the trailing portion. Consequently, the trailing portion will not change in appearance. See SET-SEGMENT-EDIT-MODE.
- The way the terminal stores attributes in a segment determines how they can be edited later. See the "Segment-Definition-syntactic-construct" for details. Unless the attributes were explicitly placed in a different pickID group, you will have to edit the primitive along with its attribute in order to change the attribute. See the *4110/4120 Series Host Programmers Manual* for suggestions.

If the segment-edit-mode is 0 or 2, the beam position is similarly remembered and restored, so that the trailing part is not moved.

Editing String-Precision Graphtext. Because of the way string-precision (that is, dot matrix) graphtext is stored in a segment, editing it, or the graphics immediately following it, requires additional care. See "Segment-Definition-syntactic-construct" for details on how graphtext is stored, and see the *4110/4120 Host Programmers Manual"* for hints on editing.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error UI11

pickID-group
as shipped — none
on power-up — none
if omitted — error UI21

beforeorafter
as shipped — none
on power-up — none
if omitted — 0

ERRORS

UI02	(Level 3):	Out of memory.
UI03	(Level 2):	Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.
UI10	(Level 2):	Specified segment does not exist.
UI11	(Level 2):	Invalid *segment-number.*
UI13	(Level 2):	The dimensionality of the specified segment does not match the coordinate mode of the terminal.
UI20	(Level 2):	Cannot find *pickID-group.*
UI21	(Level 2):	Invalid *pickID-group.*
UI23	(Level 2):	Context error because of image transformation of the included segment.
UI31	(Level 2):	Invalid *beforeorafter.*

REFERENCES

INCLUDE-COPY-OF-SEGMENT command
Segment-Definition syntactic construct
SET-SEGMENT-EDIT-MODE command
4110/4120 Host Programmers Manual

INQUIRY COMMAND

none

Int-Array Parameter Type

Syntax

```
int-array   =   int[int . . . ]
```

DESCRIPTION

The *int-array* parameter type provides the capability to send a variable number of *int* parameters to various terminal commands. An *int-array* starts with an *int* that specifies the number of *ints* that follow. If the array is terminated with a command terminator (^{E}c, ^{F}s, ^{G}s, or ^{U}s) before the specified number of *ints* have been received by the terminal, the number is reduced to the number actually received.

REFERENCES

Int parameter type

Int-Array-Report Message Type

Syntax

```
int-array-report:   int-report   [int-report . . . ]
```

DESCRIPTION

The *int-array-report* is sent from the terminal to the host as part of various status reports.

REFERENCES

Int-report message type

Int Parameter Type

Syntax

```
int   =   [HiI . . . ]  LoI
HiI   =   an ASCII character with bits 1dddddd
LoI   =   an ASCII character with bits 01sdddd
```

DESCRIPTION

When the terminal is expecting an *int* parameter as part of a command, it first sets its int-register to 0 and then interprets the next characters it receives as follows.

All control characters (ASCII Decimal Equivalent (ADE) 0 to 31) are ignored except ^{F}s, ^{G}s, and ^{U}s, which terminate the parameter interpretation (and commands, too) and set the int-register back to 0. The ^{E}c character also has this effect when the next character is not <**?**>; the sequence ^{E}c<**?**> is always interpreted as the ^{D}T character.

When a *HiI* character (ADE 64 to 127) is received, the int-register is shifted left six bits and the low six bits of the *HiI* are placed in the low six bits of the int-register.

When a *LoI* character (ADE 32 to 63) is received, the int-register is shifted left four bits and the low four bits of the *LoI* are placed in the low four bits of the int-register. The value of the int-register is negated if the fifth bit of the *LoI* is 0. The *int* parameter interpretation is then terminated.

The resulting value in the int-register is used by the command. If this value is not within the valid range for the command, an error is detected. When the valid range is –32767 to 65535 or less, and when more than two *HiI*s are received before a *LoI*, an error is detected for each extra *HiI*. When the valid range is -2^{31} to $2^{31}-1$, and when more than five *HiI*s are received before a *LoI*, an error is detected for each extra *HiI*.

See the *4110/4120/4120 Series Host Programmers Manual* for a general-purpose algorithm showing how to pack *int* parameters.

REFERENCES

^{E}c character
^{F}s character
^{G}s character
^{U}s character
4110/4120/4120 Series Host Programmers Manual

Int-Report Message Type

Syntax

```
int-report    =  {EOM-indicator}
                 1st-HiI-report
                 2nd-HiI-report
                 LoI-report

1st-HiI-report  =  an ASCII character whose
                   numeric equivalent is in the
                   range from 32 to 95.

2nd-HiI-report  =  an ASCII character whose
                   numeric equivalent is in the
                   range from 32 to 95.

LoI-report  =  an ASCII character whose
               numeric equivalent is in the
               range from 32 to 63.
```

DESCRIPTION

The terminal sends *int-report* messages to the host as part of larger report sequences in response to report commands.

EOM-Indicator. The *EOM-indicator* is sent only when there are less than 3 characters left before the report-max-line-length is reached.

Parsing an Int-Report. This is the algorithm for parsing an *int-report*:

```
(int) = ( (1st-HiI-report)–32) * 1024 +
        ( (2nd-HiI-report)–32) * 16 +
        (LoI-report) MOD 16
if (LoI-report) < 48
  then (int) = –(int)
```

REFERENCES

EOM-indicator syntactic construct
SET-REPORT-MAX-LINE-LENGTH command

Intc-Array-Report Message Type

Syntax

```
intc-array-report  =  int[intc . . . ]
```

DESCRIPTION

The *intc-array-report* is sent from the terminal to the host as part of various status reports. It reports arrays of integers whose ranges are too large for *int-reports*. This message type is currently only used in the *terminal-settings-report* for the SET-VIEWING-TRANSFORM command.

REFERENCES

Intc-report message type
SET-VIEWING-TRANSFORM command
Terminal-settings-report message type

Intc-Report Message Type

Syntax

```
intc-report  =  {EOM-indicator}
                {HiI-report   {HiI-report   {HiI-report
                {HiI-report   {HiI-report} } } } LoI-report

HiI-report   =  an ASCII character whose numeric equivalent is in the range from 32 to 95.

LoI-report   =  an ASCII character whose numeric equivalent is in the range from 32 to 63.
```

DESCRIPTION

The terminal sends *intc-reports* to the host as part of larger report sequences in response to report commands. *Intc-reports* are sent as part of *xy-report* and *xy + -reports* in coordinate modes 1 and 2. They are also sent in *terminal-settings-reports* for those *int* parameters which have maximum possible values greater than 65535.

When an *intc-report* is sent, the value being reported is encoded into the full six characters, and then only the number of characters specified by the SET-COORDINATE-MODE command is sent. The "low order" characters of the possible six are sent, resulting in a modulo function for those values which are too large to fit into the report.

EOM-Indicator. An *intc-report* includes an *EOM-indicator* only when there are less than *intc-report-size* characters left before the report-max-line-length is reached.

Parsing an Intc-Report. The size of an *intc-report* is determined by the *intc-report-size* parameter specified to the SET-COORDINATE-MODE command. In the following algorithm for parsing a single *intc-report*, "report-length" is the size of the *intc-report*:

```
(int) = 0
for (counter) = 1 to ( (report-length) -1)
    input-character: (HiI-report)
    (int) = (int) * 64 + ( (HiI-report) -32)
input-character: (LoI-report)
(int) = (int) * 16 + (LoI-report) MOD 16
if (LoI-report) < 48
    then (int) = -(int)
```

REFERENCES

EOM-indicator syntactic construct
SET-COORDINATE-MODE command
SET-REPORT-MAX-LINE-LENGTH command

LF Character

DESCRIPTION

The effect of an LF character depends on whether a dialog area is enabled and on the current coordinate mode of the terminal.

Vector and Marker Modes. The LF character has no effect when the terminal is in vector mode or marker mode.

Alpha Mode, Dialog Area Disabled. LF moves the alpha cursor down one line. If the terminal is in 2D mode (coordinate mode 0 or 1), the 2D beam position is also moved down one line. If the new 2D beam position is below the bottom of the current view's overview window, then:

- if a segment is currently being defined, no vertical movement occurs.
- if no segment is being defined, the alpha cursor and beam position move to the top of the alphatext window (see the SET-OVERVIEW-WINDOW command) and the page-full condition occurs. What happens next depends on the most recent SET-PAGE-FULL-ACTION command.
- if LFCR mode is enabled, a carriage return action also occurs.

If the 4128 or 4129 terminal is in 3D mode (coordinate mode 2), the 3D graphic beam position is not affected. The "carriage return position" is moved down one line in pixel (frame buffer) space.

Alpha Mode, Dialog Area Enabled. If the dialog area is enabled, then LF moves the cursor down one line in the dialog area without affecting the column position. If the original cursor position was on the bottom visible line, the dialog area scrolls. If the cursor was originally located on the bottom line of the scroll buffer (regardless of whether it was visible or not), receipt of a LF character deletes the top scroll buffer line, renumbers the original lines (subtracting 1 from each line number), and inserts an empty line at the bottom of the scroll buffer. If the LFCR mode is enabled, the cursor also moves to the beginning of the line.

REFERENCES

ENTER-ALPHA-MODE command
ENTER-MARKER-MODE command
ENTER-VECTOR-MODE command
LFCR command
SET-OVERVIEW-WINDOW command
SET-PAGE-FULL-ACTION command

INQUIRY COMMAND

none

LFCR Command

Host Syntax

```
ECKF   int:LFCR-mode
```

Setup Syntax

```
LFCR SP   LFCR-mode
```

PARAMETERS

LFCR-mode (0 or 1)
Specifies whether line feed actions imply carriage return actions. Setup mode parameters are **YES** and **NO**.

0 **NO**; LF characters are treated as LF characters only.

1 **YES**; LF characters are treated as LFCR sequences.

DESCRIPTION

If the parameter is 1, LFCR mode is enabled. When LFCR mode is enabled, each LF character received when the terminal is in Alpha mode is processed as a line feed followed by a carriage return.

If the LINE-FEED key on the terminal keyboard is pressed, no carriage return is sent to the host, regardless of LFCR mode.

If the parameter is 0, LFCR mode is disabled.

DEFAULTS

LFCR-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

KF11 (Level 2): Invalid *LFCR-mode.*

REFERENCES

CRLF command
LF character

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

LOAD Command

Host Syntax

${}^{E}{}_{C}$**JL** device:*source*

Setup Syntax

LOAD ${}^{S}{}_{P}$ *source*

PARAMETERS

source
The source of the file to be executed as commands. See *Device* parameter type.

DESCRIPTION

This command causes the terminal to interpret the contents of the file from *source* as commands.

The commands being loaded may contain other LOAD commands. LOAD commands can be nested five deep; deeper nesting causes error JL03.

Keyboard data entered during a load operation is queued until execution is completed. An load operation can be aborted by pressing the CANCEL key.

Report commands in the file cause reports to be sent to the host, regardless of the source of the load operation.

DEFAULTS

source
as shipped — none
on power-up — none
if omitted — error JL11

ERRORS

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 *source* specifier.

JL12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JL13 (Level 2): Context error. (Not a valid source device, device is busy, or command detects a disk format error.)

JL19 (Level 2): Device hardware error.

REFERENCES

CANCEL key
Device parameter type

INQUIRY COMMAND

none

LOCAL Key

DESCRIPTION

Pressing the LOCAL key when its light is off causes the terminal to enter Local mode and turns on the light in the key. Pressing the key when its light is on turns off the light and removes the terminal from Local mode.

When the terminal is in Local mode, the terminal stores characters from the host in the "communications queue" part of its memory. When the queue is full, the oldest characters are discarded to make room for the new ones. You can change the size of the communications queue with the SET-QUEUE-SIZE command.

Also, while the terminal is in Local mode, characters typed on the keyboard are not sent to the host computer. Instead, they are sent to the terminal's command processor, as if they had come from the host.

If you attempt to transfer files to the host port (HO:), the data goes to the terminal's alpha-exchange and are printed on the terminal screen.

If both the SETUP and LOCAL lights are on, the terminal is in Setup mode *and* Local mode. Messages to the host are echoed to the display, including reports and file transfers. *Key-execute-characters* are displayed when the terminal is in both modes simultaneously.

The LOCAL key does not auto-repeat.

REFERENCES

SET-KEY-EXECUTE-CHARACTER command
SET-QUEUE-SIZE command
SETUP key

INQUIRY COMMAND

none

LOCK-KEYBOARD Command

Host Syntax

EcKL int:*locking-mode*

Setup Syntax

LOCKKEYBOARD Sp

PARAMETERS

locking-mode (0 or 1)
Specifies whether the keyboard is locked or unlocked. Setup mode parameters are **YES** and **NO**.

0 **NO**; unlocks the keyboard.

1 **YES**; locks the keyboard.

DESCRIPTION

The LOCK-KEYBOARD command lets the host computer disable the keyboard keys. (This is useful at times when a host computer program cannot tolerate input from the operator.)

To lock the keyboard, send the following command:

lock-keyboard : 1 = **EcKL** int: 1 = **EcKL1**

When the keyboard is locked, the KYBD LOCK light is on, and all keys in the basic keyset, except the CANCEL and BREAK keys, are inoperative.

The keyboard can be unlocked by issuing a KEYBOARD-LOCK command in which the parameter is 0. It can also be unlocked by pressing the CANCEL or BREAK key, or issuing a CANCEL command.

NOTE

While in full duplex and echo Y modes, if the terminal is busy receiving a graphtext command and the user types on the keyboard, the text displayed on the screen contains spurious lines and graphtext characters can be changed.

DEFAULTS

locking-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

KL11 (Level 2): Invalid *locking-mode.*

REFERENCES

CANCEL key
CANCEL command
LOCK-VIEWING-KEYS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

LOCK-VIEWING-KEYS Command

Host Syntax

$^{E}_{C}$**RJ** int:*locking-mode*

Setup Syntax

$^{E}_{C}$**RJ** $^{S}_{P}$ *locking-mode*

PARAMETERS

locking-mode (0 or 1)
Specifies whether the viewing keys are locked or unlocked. Setup mode parameters are **YES** and **NO**.

0 **NO**; unlocks the viewing keys.

1 **YES**; locks the viewing keys.

DESCRIPTION

The LOCK-VIEWING-KEYS command lets the host program disable the terminal's four viewing keys. These are the four keys clustered together just to the left of the thumbwheels: ZOOM, PAN, VIEW, NEXTVIEW (and their SHIFTed versions, NORMAL, OVERVIEW, RESTORE, and BORDER).

If *locking-key* is 1:

- If the terminal is in a Local Viewing function (that is, if the light on the ZOOM key or the PAN key is turned on), then the terminal is removed from that function, just as if the operator had pressed the lighted key.
- The four viewing keys are disabled in the same way that the LOCK-KEYBOARD: 1 command disables the entire keyboard. While these keys are disabled, pressing them only sounds the bell and has no other effect.

The host computer can inquire whether the viewing keys are locked by issuing a REPORT-TERMINAL-SETTINGS command for the "RJ" op code. The *terminal-settings-report* sent in response to such a command not only tells the host whether the viewing keys are locked, but also reports whether the terminal is in the ZOOM or PAN Local Viewing function.

DEFAULTS

locking-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

RJ11 (Level 2): Invalid *locking-mode.*

REFERENCES

LOCK-KEYBOARD command
REPORT-TERMINAL-SETTINGS command
Terminal-settings-report message type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

MAP-INDEX-TO-PEN Command

4111 or OPTION 10

Host Syntax

```
ECPI  device:port
      int:index-to-be-mapped
      int:pen-ID-number
```

Setup Syntax

```
PMAP SP  port   index-to-be-mapped
         pen-ID-number
```

PARAMETERS

port
Specifies the RS-232 peripheral port for which color indices are to be mapped to plotter pen numbers. See *Device* parameter type.

index-to-be-mapped (–1 to 255)
Specifies the color index that is assigned to the specified pen.

-1	all color indices
0 to 255	a particular color index

pen-ID-number (0 to 255)
Specifies the pen that will draw the specified color index.

0	no pen
1 to 255	a particular pen

DESCRIPTION

This command assigns a specific color index to a particular plotter pen number at the specified peripheral port. When graphic data is drawn on the plotter at that port, all with that color index will be drawn using the specified pen.

Peripheral Port Identifier. The peripheral port identifier is a string parameter identifying the peripheral port to which the plotter is connected. This string may be "P0:", "P1:", or "P2:".

Index-to-be-Mapped. This parameter specifies the color index which is to be assigned to the particular plotter pen. If this parameter is –1, it means that all color indices are assigned to that pen.

Pen-ID-Number. This parameter specifies to which pen the color index is to be assigned. Values of 1 to N name a particular plotter pen. Here, N is the maximum number of pens for the plotter in question. For a standard 4662 plotter, N = 1. For a 4662 equipped with Option 31, N = 8. For a 4663 plotter, N = 2.

Pen number 0 means "no pens." For example, selecting index 2 and pen zero causes color index 2 to be assigned to no pen. (Lines drawn in color index 2 will not be plotted.).

Assigning a color index to one pen automatically deletes that color index from any other pen to which it may previously have been assigned.

Examples.

MAP-INDEX-TO-PEN	:	"P0:",–1, 0	Use no pens.
MAP-INDEX-TO-PEN	:	"P0:",–1, 3	Map all indices to pen number 3.
MAP-INDEX-TO-PEN	:	"P0:",5, 0	Do not draw color index 5. (Color index 5 is not assigned to any of the pens at peripheral port zero.)
MAP-INDEX-TO-PEN	:	"P0:",5, 2	Color index 5 is assigned to pen number 2 at peripheral port zero.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PI11

index-to-be-mapped
as shipped — -1
on power-up — -1
if omitted — 0

pen-ID-number
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

PI00 (Level 0): Unrecognized command. (Not a 4111 or 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 MAP-INDEX-TO-PEN command should be issued immediately after power-up, or immediately after a RESET command.)

PI11 (Level 2): Invalid *port* identifier.

PI12 (Level 3): Out of memory while parsing the parameter.

PI13 (Level 2): Port busy.

PI21 (Level 2): Invalid *index-to-be-mapped.*

PI31 (Level 2): Invalid *pen-ID-number.*

REFERENCES

PLOT command
SET-LINE-INDEX command
SET-TEXT-INDEX command

INQUIRY COMMAND

none

MOVE Command

Host Syntax

Explicit:
EcLF xy+ :*position*

Implicit:
Gs xy+ :*position*

Setup Syntax

EcLF Sp *position*

PARAMETERS

position
Specifies the new terminal space graphic beam position. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095;
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$;
coordinate mode 2: X,Y,Z = -2^{23} to $2^{23}-1$

DESCRIPTION

The MOVE command moves the 2D (coordinate mode 0 or 1) or 3D (coordinate mode 2) graphic beam position to a point in terminal space and resets the dashed-line generator to the beginning of the current line-style pattern to force the dash pattern to start with a solid section.

MOVE has two formats: explicit and implicit. The explicit format, EcLF *xy*+, sets the value of the current beam position. If the terminal is in 2D mode, the 2D beam position is set; if in 3D mode, then the 3D beam position is set. The explicit form does not depend on or change vector/marker/alpha mode. The implicit format is Gs *xy*+. Like the explicit format, it sets the value of the current beam position. The Gs puts the terminal into vector mode and sets the internal move/draw flag to move. The *xy*+ causes the beam position to move and sets the internal move/draw flag to draw.

DEFAULTS

position
as shipped — (0, 3179) 2D; (0,0,0) 3D
on power-up — (0,3179) 2D; (0,0,0) 3D
if omitted — (0,0) 2D; (0,0,0) 3D

ERRORS

LF11 (Level 2): *Position* out of valid range.

REFERENCES

ENTER-VECTOR-MODE command
SET-LINE-STYLE command

INQUIRY COMMAND

REPORT-GIN-POINT command

NEXTVIEW Key

DESCRIPTION

This key is useful only when multiple views have been defined (by the SELECT-VIEW command). Pressing the key saves the status of the current view and selects the next higher-numbered view. Pressing NEXTVIEW while holding the CTRL key down selects the next lower-numbered view. The selected view's border is blinked once.

If the terminal is in a Local Viewing function (that is, if the light in the PAN key or the ZOOM key is on) and has a pan-dimensionality attribute of 2D (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command), then pressing NEXTVIEW does not alter the location of the Local Viewing function box — the "proposed new window" in 2D terminal space. (That is, the function box frames the same window in the new view.)

This key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

PAN key
SELECT-VIEW command
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEWING-TRANSFORM command
ZOOM key
4128/4129 Operators Manual

NORMAL Key

DESCRIPTION

The NORMAL key is the SHIFTed version of the ZOOM key.

This key is in effect only when the terminal is in 2D (coordinate mode 0 or 1) Local Viewing function; that is, when the Local Viewing function box is displayed and the light on the ZOOM key or the PAN key is turned on.

When the terminal is in 2D Local Viewing function, pressing the NORMAL key restores the shape (i.e., the ratio of height to width) of the Local Viewing function box to that of the original window (see below) for that view. The size of the adjusted Local Viewing function box is determined by the average of the ratios of the X and Y extents of the original window and Local Viewing function box. The location of the Local Viewing function box, defined by the location of its center, is only changed if the new Local Viewing function box would extend outside terminal space, unless moved. This operation may result in defining the Local Viewing function box larger than terminal space, so that when the VIEW key is pressed error RW21 level 2 is detected.

"Original window" means the window defined by the most recent SET-WINDOW command for that view. (The operator's manipulation of the viewing keys does not count.) If no SET-WINDOW command has been issued, then the original window is the window in effect when the view was created.

On the 4128 and 4129, when the terminal is in a 3D Local Viewing function, pressing this key has no effect.

This key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

PAN key
SET-WINDOW command
VIEW key
ZOOM key
4128/4129 Operators manual

OVERVIEW Key

DESCRIPTION

Pressing this key (the SHIFTed version of the PAN key) causes the following to occur when the terminal is in 2D mode (coordinate mode 0 or 1):

- The current view's viewport is erased.
- The partial overview window (as set by the SET-OVERVIEW-WINDOW command) is selected.
- If the terminal is in Local Viewing function (that is, if the PAN or ZOOM key light is on), the terminal remains in Local Viewing function and the Local Viewing function box showing the "proposed new window" remains at the same position in terminal space.

Pressing CTRL-OVERVIEW (pressing the PAN key while holding down the SHIFT and CTRL keys) selects the full overview window (as set by the SET-OVERVIEW-WINDOW command).

It is possible to group several views together in a "view display cluster". (See the SET-VIEW-DISPLAY-CLUSTER command for details.) If this is done, then pressing OVERVIEW or CTRL-OVERVIEW affects not only the current view, but also all other views in the same display cluster as the current view.

On the 4128 and 4129, when the terminal is in a 3D Local Viewing function or in 3D coordinate mode, pressing either OVERVIEW or CTRL-OVERVIEW has no affect.

This key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

PAN key
RESTORE key
SET-OVERVIEW-WINDOW command
SET-VIEW-DISPLAY-CLUSTER command
ZOOM key
4128/4129 Operators manual

PAGE Command

Host Syntax

$E_C F_F$

DESCRIPTION

The effect of the PAGE command depends on whether the dialog area is enabled.

If the dialog area is enabled, the PAGE command is equivalent to a RENEW-VIEW 0 command.

If the dialog area is disabled, the PAGE command is equivalent to RENEW-VIEW 0 command followed by these actions:

1. The current line style is reset to 0 (solid).
2. The 4010-GIN is canceled.
3. The graphic beam position and the alpha cursor are moved to home position (as set by the SET-OVERVIEW-WINDOW command).
4. The terminal enters Alpha mode.

REFERENCES

4110/4120 Series Host Programmers Manual
ENABLE-DIALOG-AREA command
RENEW-VIEW command
SET-OVERVIEW-WINDOW command

INQUIRY COMMAND

none

PAGE Key

DESCRIPTION

Pressing the PAGE key has the same effect as issuing a PAGE command. This key does not auto-repeat.

REFERENCES

PAGE command

PAN Key

DESCRIPTION

Pressing this key when its light is off enables the 2D or 3D (4128 and 4129 only) PAN Local Viewing function and turns the key's light on. Pressing this key when the terminal is the PAN Local Viewing function disables the function and turns the key's light off. Pressing this key when its light is on, but the terminal is in a different 3D Local Viewing function, enables the 3D Local Viewing function and leaves the key's light on. The key does not auto-repeat.

When the PAN Local Viewing function is enabled, the PAN Local Viewing function box is displayed on the screen. The 2D PAN symbol includes a large plus symbol (+) in the center. On the 4128 and 4129, the 3D PAN symbol includes the plus symbol with perpendicular lines extending from each end. If the ZOOM or DEPTH Local Viewing function was enabled when the PAN key was pressed, then the PAN symbol is the same size as the previous symbol.

On the 4128 and 4129, if the view's pan-dimensionality (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is 2D, the 2D PAN Local Viewing function is enabled. If the view's pan-dimensionality is 3D, then the 3D PAN Local Viewing function is enabled.

The PAN symbol can be moved around the screen by moving the thumbwheels. The horizontal thumbwheel moves the symbol to the right or left; the vertical thumbwheel moves it up or down. Pressing the SHIFT key while moving either thumbwheel causes the symbol to move more slowly. This allows for finer adjustments in the position of the symbol.

The boundary of the 2D PAN symbol indicates the 2D window which will take effect if the VIEW key is pressed. The size and center of the 3D PAN symbol indicates the size of the 3D UV-window and the location of the 3D view-reference-point which will take effect if the VIEW key is pressed.

When the terminal is in a 3D PAN function, other 3D Local Viewing functions can be enabled by pressing the CTRL key and the numeric keypad keys 1 through 9.

When the terminal is in a 3D Local Viewing function, dialog area –2 will be visible if its visibility is turned on. When the terminal is not in a Local Viewing function, dialog area –2 will be invisible regardless of its visibility setting.

See the *4128/4129 Operators Manual* for another description of the PAN key and a full description of the Local Viewing System.

REFERENCES

SET-DIALOG-AREA-VISIBILITY command
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEWING-TRANSFORM command
SET-WINDOW command
VIEW key
ZOOM key
4128/4129 Operators Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

Panel-Definition Syntactic Construct

Syntax

```
panel-definition   =   set-up-panel-environment
                       panel-boundary-definition
                       {panel-boundary-definition . . . }
                       panel-termination

set-up-panel-environment  =   ⎧SET-PANEL-FILLING MODE    ⎫
                              ⎨SELECT-FILL-PATTERN       ⎬
                              ⎩SET-DRAW-BOUNDARY-MODE    ⎭

panel-boundary-definition  =  BEGIN-PANEL-BOUNDARY
                              {boundary-point . . . }

boundary-point     =  ⎧ MOVE            ⎫
                      ⎪ DRAW            ⎪
                      ⎨ DRAW-CURVE      ⎬
                      ⎪ DRAW-MARKER     ⎪
                      ⎩ DRAW-RECTANGLE  ⎭

panel-termination  =   ⎧END-PANEL                 ⎫
                       ⎪END-SEGMENT               ⎪
                       ⎪BEGIN-NEW-SEGMENT         ⎪
                       ⎪BEGIN-HIGHER-SEGMENT      ⎪
                       ⎪BEGIN-LOWER-SEGMENT       ⎪
                       ⎨INCLUDE-COPY-OF-SEGMENT   ⎬
                       ⎪CALL-SEGMENT              ⎪
                       ⎪SET-COORDINATE-MODE[1]    ⎪
                       ⎪DISPLAY-ALPHA-TEXT[2]     ⎪
                       ⎩GRAPHIC-TEXT[2]           ⎭
```

[1] only when changing dimensionality

[2] error condition

DESCRIPTION

A panel is a filled polygon whose boundary(ies) may or may not be displayed as vectors. The main advantage of using panels is that they provide a quick and efficient way to fill enclosed areas. Panels are 2D constructs only; to build 3D panels, use the DRAW-FACETS command.

Before the Panel-Definition

All panel boundary vectors are drawn in the current line index and current line style.

The SET-PANEL-FILLING-MODE command specifies how the panel is to be filled, the SELECT-FILL-PATTERN command specifies what the panel's interior is to be filled with, and the SET-DRAW-BOUNDARY-MODE command specifies whether panel boundaries defined by the DRAW-RECTANGLE command are drawn or not.

Defining A Panel Boundary

To define a panel boundary, first send a BEGIN-PANEL-BOUNDARY command. This command specifies the first vertex of the panel and whether to draw the panel's boundary. Subsequent vertices are defined by the MOVE, DRAW, DRAW-MARKER, DRAW-CURVE, and DRAW-RECTANGLE commands. Each BEGIN-PANEL-BOUNDARY command after the first one closes the previous panel boundary while continuing to define the same panel. Thus several disjoint boundaries can be defined within one panel.

The terminal displays the drawn boundary edges as the commands defining the edges are received, depending on the fixup level of the terminal. (See the SET-FIXUP-LEVEL command.)

Example. Figure 7-10 shows one panel with two boundaries. Figure 7-11 shows two intersecting panels with one boundary each. The panels were created with the following commands:

Figure 7-10

```
EcLP 0 1000 1
EcLG 1000 1000
EcLG 1000 2000
EcLG 0 2000
EcLP 500 500 1
EcLG 1500 500
EcLG 1500 1500
EcLG 500 1500
EcLE
```

Figure 7-11

```
EcLP 2000 1000 1
EcLG 3000 1000
EcLG 3000 2000
EcLG 2000 2000
EcLE
EcLP 2500 500 1
EcLG 3500 500
EcLG 3500 1500
EcLG 2500 1500
EcLE
```

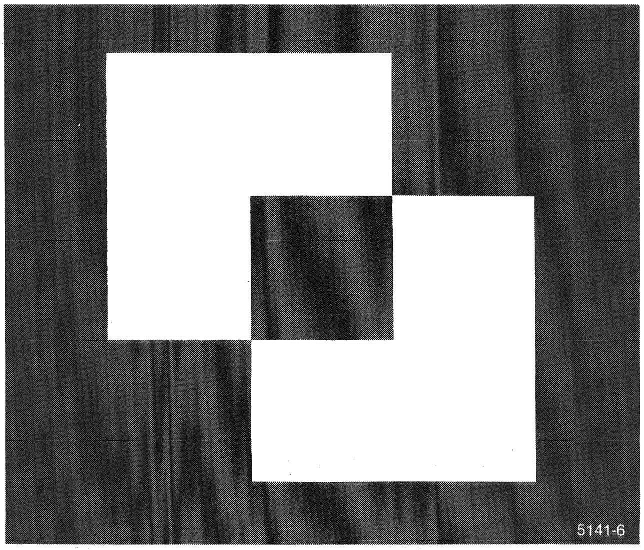


Figure 7-10. One Panel With Two Square Boundaries.

Terminating the Panel-Definition

The panel definition ends when an END-PANEL, END-SEGMENT, BEGIN-NEW-SEGMENT, BEGIN-HIGHER-SEGMENT, BEGIN-LOWER-SEGMENT, INCLUDE-COPY-OF-SEGMENT, CALL-SEGMENT, DISPLAY-ALPHA-TEXT, or GRAPHIC-TEXT command occurs. The panel boundary is closed, updating the 2D graphic beam position to the point specified by the *xy* parameter in the last BEGIN-PANEL-BOUNDARY command. (The SET-COORDINATE-MODE:2 command also terminates panel definitions.)

When a panel is terminated, it may be filled with the current fill pattern according to the current panel-filling mode, depending on the fixup level of the terminal.

The terminal fills the panel using the appropriate pattern. The boundary may or may not be filled depending on the settings included in the BEGIN-PANEL-BOUNDARY command.

Error Conditions

Neither alphatext nor graphtext is permitted within a *panel-definition.* Graphtext within a *panel-definition* causes a type LT03 error. If the dialog area is enabled, any alphatext received by the terminal during a *panel-definition* is sent to the dialog area and does not interfere with the definition. If, however, the dialog area is not enabled, the receipt of alphatext causes a type LP03 error and the *panel-definition* is closed.

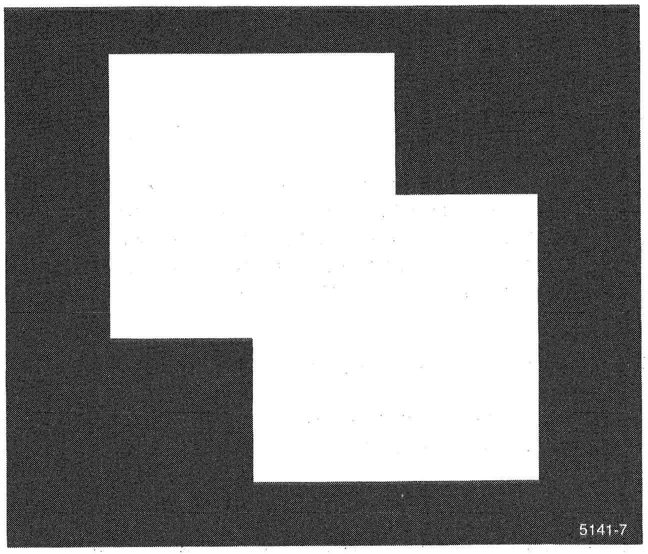


Figure 7-11. Two Square Panels, Each With One Boundary.

PIXEL-COPY Command

Host Syntax

```
EcRX  int:destination-surface
      xy:destination-lower-left-corner
      xy:first-source-corner
      xy:second-source-corner
```

Setup Syntax

```
EcRX Sp  destination-surface
         destination-lower-left-corner
         first-source-corner
         second-source-corner
```

PARAMETERS

destination-surface −1 to 8

Specifies the surface to which pixels are to be copied.

−1	super surface: all bit planes on all defined surfaces.
0	the current surface as defined by the last BEGIN-PIXEL-OPERATIONS command.
1 to 8	the surface with the specified number.

destination-lower-left-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)

Specifies the lower-left corner of the region on the destination surface to which pixels are to be copied.

first-source-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)

Specifies one corner of a rectangular region on the current pixel surface. The pixel at this corner is copied to the lower-left corner of the destination region.

second-source-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)

Specifies the corner opposite the first-source-corner in the "source" rectangular region.

DESCRIPTION

This command copies pixels from a rectangular region on the current pixel surface in raster memory space on to a rectangular region (of the same dimensions) on the same or another surface.

The current pixel surface is the surface specified in the most recent BEGIN-PIXEL-OPERATIONS command. Pixels are copied to the destination region using the ALU mode specified in the most recent BEGIN-PIXEL-OPERATIONS command.

In this command, all xy coordinates specify pixel positions in raster memory space.

Destination Surface. The first parameter specifies to which writing surface the pixels are to be copied. The special surface number −1 designates a "super surface" consisting of all bit planes of all defined surfaces. See the *4110/4120 Series Host Programmers Manual* if you intend to use the super surface. Surface number 0 represents the current pixel surface as set in the most recent BEGIN-PIXEL-OPERATIONS command.

Destination Lower Left Corner. The first *xy* parameter specifies the lower left corner of a rectangular region on the destination surface in raster memory space. This destination region is the same width and height as the source region specified by the source corners, as limited by the edge of pixel space.

Source Corners. The last two *xy* parameters are opposite corners of a rectangular region on the current pixel surface. The PIXEL-COPY command copies each pixel in this rectangular region onto a corresponding pixel in the destination region on the destination surface (or on all surfaces, if the destination surface number is −1.)

The two "source corners" need not be the lower left and upper right corners, respectively, of the source region. However, if they are not, then a "mirror" or "inversion" operation is performed when copying. That is, the pixels written to the destination region may form a mirror image, or an inverted image, of the picture formed by the pixels in the source region. The pixel at the first source corner is copied onto the pixel at the lower left corner of the destination region. The pixel at the second source corner is copied onto the pixel at the upper right corner of the destination region.

To facilitate "mirror" and "inversion" operations, the preliminary pixel-copy in place operation is not done if the following conditions are met:

- The *destination-surface* is the same as the source surface.
- The *destination-lower-left-corner* is the same as the lower-left source corner.
- The *first-source-corner* is not the lower-left corner of the source region, and the *second-source-corner* is not the upper-right corner of the source region.

DEFAULTS

destination-surface
as shipped — none
on power-up — none
if omitted — 0

destination-lower-left-corner
as shipped — none
on power-up — none
if omitted — (0,0)

first-source-corner
as shipped — none
on power-up — none
if omitted — (0,0)

second-source-corner
as shipped — none
on power-up — none
if omitted — (0,0)

ERRORS

RX10 (Level 2): The specified *destination-surface* does not exist.
RX11 (Level 2): Invalid *destination-surface.*
RX21 (Level 2): Invalid *destination-lower-left-corner.*
RX31 (Level 2): Invalid *first-source-corner.*
RX41 (Level 2): Invalid *second-source-corner.*

REFERENCES

BEGIN-PIXEL-OPERATIONS command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

none

PLOT Command

4111 or OPTION 10

Host Syntax

```
EcPL   string:separator
       device:destination
```

Setup Syntax

```
PLOT Sp  TO  destination
```

PARAMETERS

separator
Must be the empty string or **TO**

destination
Specifies the destination for the plot. See the *Device* parameter type.

DESCRIPTION

The PLOT command saves segments that are visible in the current view to the specified destination.

The SET-WINDOW and SET-VIEWPORT commands are sent prior to the visible segments. When the destination is a peripheral port with a plotter driver assigned to it, segments are clipped to the designated window and viewport.

DEFAULTS

separator
as shipped — none
on power-up — none
if omitted — error PL11

destination
as shipped — none
on power-up — none
if omitted — error PL21

ERRORS

PL00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PL02	(Level 3):	Out of memory while attempting DMA transfer (Option 3A or 3C or 4111 only).
PL11	(Level 2):	Invalid first parameter. (Must be the empty string or **TO**.)
PL12	(Level 3):	Out of memory while parsing parameter.
PL20	(Level 2):	Destination device not installed.
PL21	(Level 2):	Invalid destination.
PL22	(Level 3):	Out of memory while parsing parameter.
PL23	(Level 2):	Destination device is busy or is a write-protected file.
PL29	(Level 2):	Hardware error on destination device.

REFERENCES

Device parameter type

INQUIRY COMMAND

none

PORT-ASSIGN Command

OPTION 10

Host Syntax

$^{E}{}_{C}$**PA** device:*port* string:*protocol-identifier*

Setup Syntax

PASSIGN $^{S}{}_{P}$ *port* *protocol-identifier*

PARAMETERS

port

Specifies the RS-232 peripheral port to which a protocol is being assigned. Valid ports are:

P0:

P1:

P2: (Not valid for 4111)

protocol-identifier

A string specifying the protocol being assigned to the peripheral port. Valid values are:

PPORT

4643

4662

4662/MP

4662/NT

4663

4663/NT

4663/NB

4957

4957/S

4958

4958/S

KANA

DMA (4111 only; must be assigned to P0:)

DESCRIPTION

The PORT-ASSIGN command assigns a "device protocol" to a particular RS-232 peripheral port. (This command requires that Option 10, the Three Port Peripheral Interface, be installed in the terminal.)

The command has two parameters: the *port* identifier and the *protocol-identifier.*

Port Identifier. The port identifier is a three-character string parameter: "P0:" for peripheral port 0, "P1:" for peripheral port 1, or "P2:" for peripheral port 2. These identify the three RS-232 peripheral connectors on the rear panel of the terminal.

Protocol Identifier. The protocol identifier specifies the communications protocol which the terminal will use in communicating with a peripheral device connected at the specified peripheral port. Table 7-16 lists the valid protocol identifiers.

If a peripheral port is to be the *source* in a COPY, LOAD, or SPOOL command, or if it is to be either the *source* or the *destination* in a PORT-COPY command, then it is necessary that the port have been assigned the **PPORT** protocol identifier. If the protocol identifier is 4957, 4957/S, 4958, or 4958/S, a context error results.

All plotter protocols expect that the attached plotter's device address is set to Device A. See the manual for your plotter for details on setting this parameter. Some of the plotter protocols translate some 4110/4120-series commands into plotter commands. The 4110/4120 commands that are translated include the 2D graphic primitive commands (e.g., MOVE, DRAW, DISPLAY-ALPHA-TEXT, etc.). Other 4110/4120-series commands, including 3D graphic primitive commands (e.g., DRAW-FACETS, etc.) are ignored.

In the 4111, when a 4957, 4957/S, 4958 or 4958/S Graphic Tablet is PASSIGNED to port PO: or P1:, the following actions occur automatically:

The port baud rate is set to	9600
The port bits are set to	1 7
The port parity is set to	ODD
The port flagging is set to	NONE

When any other device is PASSIGNED after the previous action, the user has to insure that the port is properly set for the new environment.

DEFAULTS

port
as shipped — none
on power-up — remembered
if omitted — error PA11

protocol-identifier
as shipped — none
on power-up — remembered
if omitted — error PA21

ERRORS

PA00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PA11	(Level 2):	Invalid *port.*
PA12	(Level 3):	Out of memory while parsing the parameter.
PA13	(Level 2):	Port is in use.
PA21	(Level 2):	Invalid *protocol-identifier.* (Must be **PPORT**, **4643**, **4662**, **4662/MP**,**4662/NT**, **4663**, **4663/NB**,**4663/NT**, **4957**, **4957/S**, **4958**,**4958/S**, **KANA**, or **DMA** [4111 only]).
PA22	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

COPY command
DIRECTORY command
ENABLE-GIN command
ENABLE-4953-TABLET-GIN command
LOAD command
MAP-INDEX-TO-PEN command
PLOT command
PORT-COPY command
SAVE command
SET-PORT-EOF-STRING command
SPOOL command

INQUIRY COMMAND

REPORT-PORT-STATUS command

Table 7-16

PERIPHERAL PORT PROTOCOL IDENTIFIERS

Protocol Identifier	Meaning
PPORT	Assigns a general-purpose RS-232 communications protocol (a protocol that makes no assumptions about the nature of the attached device). When data is transferred to this port, the end of the data is marked by appending the port's end-of-file string, as set by the SET-PORT-EOF-STRING command.
4643	4643 Printer. Assigns a communications protocol which is suitable for a TEKTRONIX 4643 Printer. Any CR characters sent to this port will be replaced by the port's current end-of-line string, as set by the SET-PORT-EOL-STRING command. As with the PPORT protocol, the end of a data transfer is marked with the port's end-of-file string.
4662	4662 Plotter. Assigns a communications protocol which assumes that the device at his port is a TEKTRONIX 4662 Interactive Digital Plotter. The plotter's block mode communications protocol is used. Some 4110/4120-series commands are translated into plotter commands.
4662/MP	4662 Plotter with Multiple Pens. Similar to the "4662" protocol, except that the 4662 plotter is assumed to be equipped with MULTIPLE PENS (Option 31). When 4110/4120-series escape-sequence commands are translated into plotter language commands, lines (or text) drawn with different color indices are drawn on the plotter using different pens. (See the description of the MAP-INDEX-TO-PEN command for details.)
4662/NT	4662, No Translation. Similar to**4662** protocol; the device at this peripheral port is assumed to be a 4662 plotter, and the plotter's block mode protocol is used. However, here is no translation from 4110/4120-series escape-sequence commands to plotter commands.
4663	4663 Plotter. Assigns a communications protocol which assumes that the device at this port is a TEKTRONIX 4663 Interactive Digital Plotter. The plotter's block mode communications protocol is used. Some 4110/4120-series commands are translated into plotter commands.
4663/NB	4663, No Blockmode. Similar to**4663** protocol; the device at this port is assumed to be a 4663 plotter, and 4110/4120-series escape-sequence commands are translated into plotter commands. However, the plotter;s block mode is not used.
4663/NT	4663, No Translation. Similar to**4663** protocol; the device at this port is assumed to be a 4663 plotter, and the plotter's block mode is used. However, there is no translation from 4110/4120-series escape-sequence commands to plotter commands.
4957	4957 Tablet with 4-Button Puck. Assigns a communications protocol which assumes that the device at this port is a TEKTRONIX 4957 Graphics Tablet.
4957/S	4957 Tablet with Stylus.
4958	4958 Tablet with 4- or 16-Button Puck. Assigns a communications protocol which assumes that the device at this port is a TEKTRONIX 4958 Graphics Tablet.
4958/S	4958 Tablet with stylus.
KANA	Katakana, Option 4K. Allows the port to act as a general purpose RS-232 peripheral port, with the ability to transmit and receive the JIS 7 bit coded character set.
DMA	When assigned to P0: of a 4111, deactivates RS-232 signals, then activates RS-422 signals and the DMA serial interface device. Thereafter, DM: can be used as a port-identifier.

PORT-COPY Command

4111 or OPTION 10

Host Syntax

```
ECPC  device:source
      string:separator
      device:destination
```

Setup Syntax

```
PCOPY SP  source  TO  destination
```

PARAMETERS

source
Specifies the first of two devices between which data flows. See *Device* parameter type.

separator-string (Empty String or "TO ")
Separates the *source* and *destination* specifiers.

destination
Specifies the second device for the data transfer. See *Device* parameter type.

DESCRIPTION

The PORT-COPY command establishes a data path between two devices, so that a two-way data transfer can take place. The "destination" device can "talk back" to the "source" device during the data transfer. This permits the following:

- The two devices can use a device-dependent "handshaking" protocol.
- The source device (typically the host computer) can directly query the destination device, and parse the destination device's responses.

The data connection is established by the PORT-COPY command; the connection is broken when either device sends an end-of-file indicator (EOF). At the end of the PORT-COPY operation, the terminal always sends an EOF-string to the host port if HO: is either *source* or *destination*, and a *port-EOF-string* to the peripheral port.

Only HO: and the peripheral port devices with the PPORT protocol are valid for PORT-COPY.

All other terminal activity is suspended until the end-of-file is detected. This two-way file transfer can also be terminated by pressing the CANCEL key on the keyboard.

DEFAULTS

source
as shipped — none
on power-up — none
if omitted — error PC11

separator
as shipped — none
on power-up — none
if omitted — error PC21

destination
as shipped — none
on power-up — none
if omitted — error PC31

ERRORS

PC00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PC02 (Level 3): Out of memory while processing command.

PC10 (Level 2): *Source* device does not exist.

PC11 (Level 2): Invalid *source* (must be **HO:**, **P0:**,**P1:**, or **P2:** [P2: not valid for 4111]).

PC12 (Level 3): Out of memory while parsing the parameter.

PC13 (Level 2): *Source* is busy, or is a peripheral port that is not assigned the **PPORT** protocol.

PC21 (Level 2): Invalid *separator.*

PC22 (Level 3): Out of memory while parsing the parameter.

PC30 (Level 2): *Destination* device is not installed.

PC31 (Level 2): Invalid *destination* (must be **HO:**,**P0:**, **P1:** or **P2:** [P2: not valid for 4111] , and must be different from the *source* port).

PC32 (Level 3): Out of memory while parsing the parameter.

PC33 (Level 2): *Destination* is busy is a peripheral port that is not assigned the **PPORT** protocol.

REFERENCES

CANCEL key
COPY command
Device parameter type
PORT-ASSIGN command

INQUIRY COMMAND

REPORT-PORT-STATUS command

Port-Status-Report Message Type

OPTION 10

Syntax

```
port-status-report  =  [char-report:sig-char]
                       char-report:P or SP
                       char-report:0, 1, 2 or SP
                       int-report:baud-rate
                       int-report:parity
                       int-report:stop-bits
                       int-report:data-bits
                       int-report:flagging-mode
                       int-report:flagging-"go"-character
                       int-report:flagging-"stop"-character
                       string-report:protocol
                       int-array-report:EOF-string
                       int-array-report:EOL-string
                       [EOM-indicator]
```

DESCRIPTION

The terminal sends a *port-status-report* to the host to in response to a Report-PORT-STATUS command.

When the terminal sends a report to the host, bypass mode is entered. (See ENTER-BYPASS-MODE.)

The *sig-char* is the current *sig-char* for non-GIN reports, as set by the most recent SET-Report-SIG-CHARS command. If the current *sig-char* is NU, then it is not sent.

The next two char-reports, **P0**, **P1**, **P2**, or SP SP, names the RS-232 peripheral port to which the *port-status-report* pertains.

SP SP indicates that the Report-PORT-STATUS command had an invalid *port-specifier* string. If that is the case, then the rest of the *port-status-report* is not sent.

The rest of the *port-status-report* contains the current values of the peripheral port's parameters: baud rate, parity, number of stop bits, etc.

The *protocol* string report is always ten characters long, with the actual protocol characters in the first positions.

REFERENCES

EOM-indicator syntactic construct
PORT-ASSIGN command
REPORT-PORT-STATUS command
SET-PORT-BAUD-RATE command
SET-PORT-EOF-STRING command
SET-PORT-EOL-STRING command
SET-PORT-FLAGGING-MODE command
SET-PORT-PARITY command
SET-PORT-STOP-BITS command
SET-REPORT-SIG-CHARS command

PROMPT-MODE Command

Host Syntax

```
EcNM    int:prompt-mode
```

Setup Syntax

```
PROMPTMODE Sp   prompt-mode
```

PARAMETERS

prompt-mode (0, 1, or 2)
Turns Prompt mode on and off, and specifies when Prompt mode should be turned on. Setup mode parameters are **YES** and**NO**.

0	**NO**; turn Prompt mode off.
1	**YES**; turn Prompt mode on after the next EOM indicator or *EOM-indicator*.
2	turn Prompt mode on immediately.

DESCRIPTION

This command enables or disables Prompt mode.

If *prompt-mode* is 0, Prompt mode is turned off and any characters in the terminal's output queue are transmitted.

If *prompt-mode* is 1, the terminal does not enter Prompt mode until it has encountered an *EOM-indicator* or EOM character in the data it sends to the host. After sending the *EOM-indicator* or EOM character the terminal enters Prompt mode.

If *prompt-mode* is 2, the terminal enters Prompt mode immediately.

When the terminal is in Prompt mode, characters are not sent to the host, but are placed in the output queue until a prompt is received. When a prompt is received, all characters up to and including the next *EOM-indicator* or EOM character are sent.

When the terminal is in Prompt mode, the terminal's output queue can be filled while waiting for a prompt from the host computer. When the output queue is full, the keyboard is temporarily locked and pressing a key simply rings the bell. The keyboard unlocks when the terminal receives a prompt to empty its output queue, when the host computer sends a command to exit Prompt mode, when the CANCEL key is pressed, or when the terminal is reset.

DEFAULTS

prompt-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

NM11 (Level 2): Invalid *prompt-mode.*

REFERENCES

EOM-indicator syntactic construct
SET-EOM-CHARS command
SET-PROMPT-STRING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

PROTECT-FILE Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

$^{E}{}_{C}$**JP** device:*file-specifier*
int:*write-protect-mode*

Setup Syntax

PROTECT $^{S}{}_{P}$ *file-specifier*
write-protect-mode

PARAMETERS

file-specifier
The disk file whose write-protection you are specifying. See *Device* parameter type.

write-protect-mode (0 or 1)
The protection you want for the specified file. Setup mode parameters are **YES** and **NO**.

0 **NO**; unprotects the file.

1 **YES**; protects the file.

DESCRIPTION

This command sets or clears the write-protect flag on the specified file. This operation is distinct from the hardware write-protect that is controlled by a notch in the disk and/or the front panel switch.

To protect or unprotect a specific disk file, include both the disk drive number and the filename in the string parameter.

Protected files may not be deleted by the DELETE-FILE command or overwritten with a COPY, SPOOL, SAVE, PLOT, or DIRECTORY command. A FORMAT-VOLUME command can overwrite individual protected files.

DEFAULTS

file-specifier
as shipped — none
on power-up — none
if omitted — error JP11

write-protect-mode
as shipped — 1
on power-up — 1
if omitted — 1

ERRORS

JP00 (Level 0): 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): Out of memory while parsing parameter.

JP13 (Level 2): The specified device is not a disk drive.

JP19 (Level 2): Disk hardware error. (I/O error, drive not ready, or hardware write-protect error.)

JP21 (Level 2): Invalid *write-protect-mode*.

REFERENCES

FORMAT-VOLUME command

INQUIRY COMMAND

none

RASTER-WRITE Command

Host Syntax

$^{E}_{C}$**RP** int:*number-of-pixels*
char-array:*indices-array*

Setup Syntax

$^{E}_{C}$**RP** $^{S}_{P}$ *number-of-pixels* *indices-array*

PARAMETERS

number-of-pixels (0 to 65535)
Specifies the number of pixels represented in the following char-array.

indices-array (length 0 to 65535)
Specifies the color indices of individual pixels. Each ASCII character must be in the rage 32 to 96.

DESCRIPTION

This command set the color indices of individual pixels in the pixel viewport or in a fill-pattern definition.

If no fill-pattern definition is open, for each specified index all pixels within the pixel rectangle defined by the current pixel writing factors are set to that index. The pixel beam position is then moved according to the current pixel writing factors. See SET-PIXEL-WRITING-FACTORS command for details.

If a fill-pattern definition is open, for each specified index one pixel is to that index, and the next pixel position is moved on to the right. When it goes past the right edge of the pattern, the next pixel position is moved down one row and to the left edge. When it goes past the right edge of the last row, the fill-pattern definition is closed and any remaining indices in the command are discarded.

The indices are packed into *indices-array* using a bit-packing scheme similar to block mode bit packing. Where N is the number of bits per pixel (as set by BEGIN-PIXEL-OPERATIONS for pixel viewport indices or by BEGIN-FILL-PATTERN for pattern definitions), the terminal uses this algorithm to extract color indices from *indices-array*:

```
Set Ptr to 0
Set Bits to 0
Set R to 0
For J = 1 to number-of-pixels:
   Until Bits ⩾ N:
   Add 1 to Ptr
   If indices-array (Ptr) = 96:
      Move pixel position to start of next row
   Else:
      Shift R left 6 bits
      Add indices-array (Ptr) –32 to R
      Add 6 to Bits
   Until Bits < 6:
      Set index-of-pixel to R and $2^N -1$
      Move pixel position
      Shift R right N bits
      Subtract N from bits
```

Note that the character   , (ASCII decimal equivalent 96), moves the pattern or pixel beam position to the start of the following row of pixels. The   code is not included in the pixel count.

If *number-of-pixels* is different from the number of indices possible in *indices-array* (given the current bits-per-pixel), an error is detected after all the indices have been set.

DEFAULTS

number-of-pixels
as shipped — none
on power-up — none
if omitted — error RP11

indices-array
as shipped — none
on power-up — none
if omitted — 0

ERRORS

RP11 (Level 2): Invalid *number-of-pixels.*

RP21 (Level 2): There are too many or too few pixels in the*indices* array, or invalid code(s) are present (range is from ADE 32 to 96).

RP22 (Level 3): Out of memory while parsing the parameter.

REFERENCES

BEGIN-FILL-PATTERN command
BEGIN-PIXEL-OPERATIONS command
SET-PIXEL-VIEWPORT command
SET-PIXEL-WRITING-FACTORS command

INQUIRY Command

none

Real-Array Parameter Type

Syntax

real-array = *int*[*real* . . .

DESCRIPTION

The *real-array* parameter type provides the capability to send a variable number of *real* parameters to various terminal commands. A *real-array* starts with an *int* that specifies the number of *reals* (pairs of *int*s) that follow. If the array is terminated with a command terminator (^{E}C, ^{F}S, ^{G}S, or ^{U}S) before the specified number of *reals* have been received by the terminal, the number is reduced to the number actually received.

REFERENCES

Int parameter type
Real parameter type

Real-Array-Report Message Type

Syntax

real-array-report: *int-report*[*real-report* . . .

DESCRIPTION

The *real-array-report* is sent from the terminal to the host as part of various status reports.

REFERENCES

Int-report message type
Real-report message type

Real Parameter Type

Syntax

real = int: *mantissa* int:*power-of-two*

COMPONENTS

mantissa
An integer, represented as an *int* parameter.

power-of-two
An *int* telling the power of two by which the mantissa is multiplied.

DESCRIPTION

To send a real number to the terminal (as a parameter for one of its commands), represent it in the form "A times two to the Bth power," where A and B are integers. Send A first, and then B, packed in the *int* parameter format.

Examples. The number 1.5 may be expressed as 3 times 1/2, or 3 times 2 to the power −1. Here, A = 3 and B = −1; hence

real: 1.5 = int: 3 int: −1 = **3!**

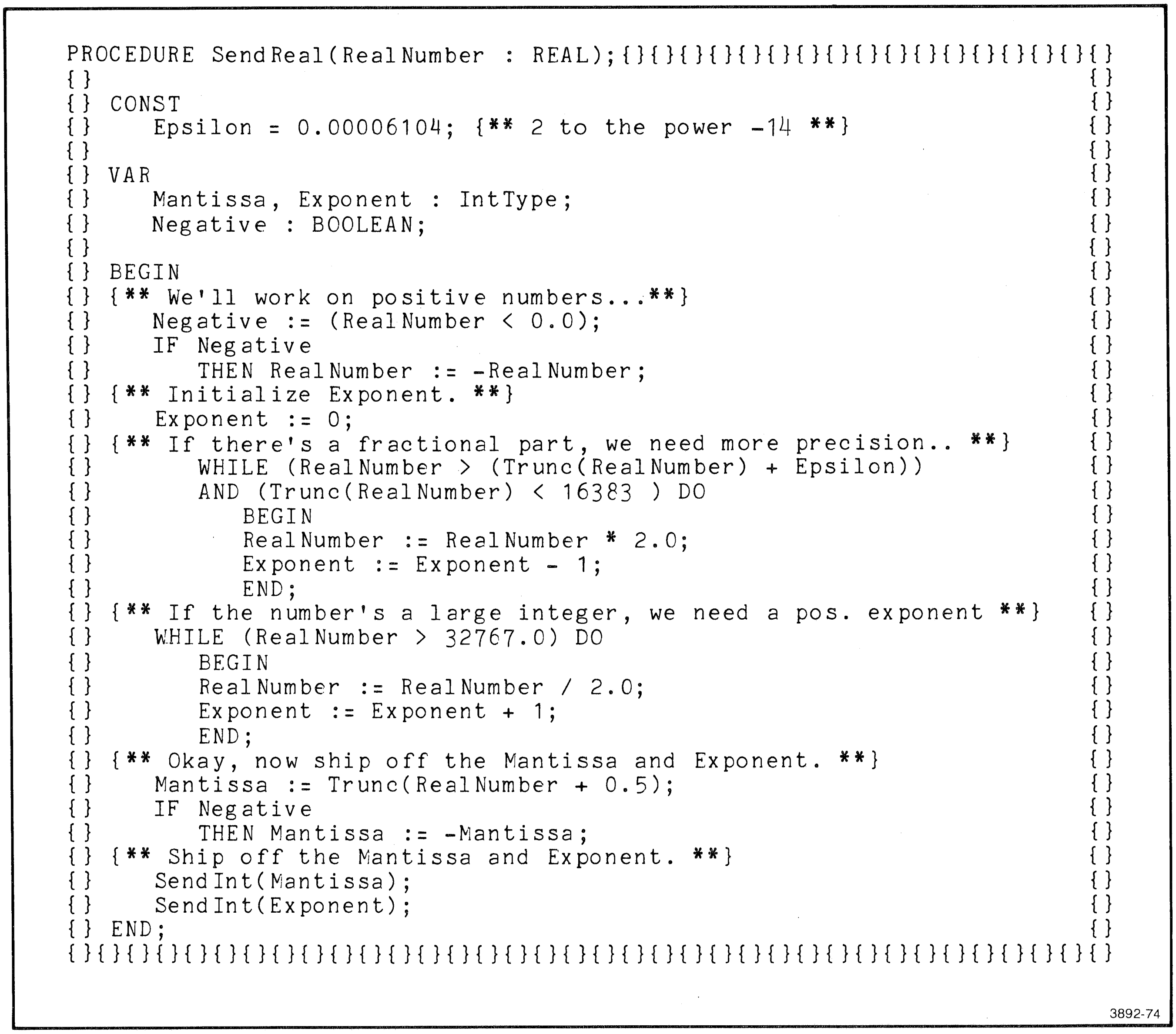
The number pi may be approximated as 25736/8192, or 25736 times 2 to the power −13. Here A = 25736 and B = −13; hence

real: pi = int: 25736 int: −13 = **YH8-**

A Sample Routine. Figure 7-12 shows a PASCAL procedure to issue real parameters to the terminal.

REFERENCES

Int parameter type



```
PROCEDURE SendReal(RealNumber : REAL);{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}
{}                                                                    {}
{} CONST                                                              {}
{}    Epsilon = 0.00006104; {** 2 to the power -14 **}                {}
{}                                                                    {}
{} VAR                                                                {}
{}    Mantissa, Exponent : IntType;                                   {}
{}    Negative : BOOLEAN;                                             {}
{}                                                                    {}
{} BEGIN                                                              {}
{} {** We'll work on positive numbers...**}                           {}
{}    Negative := (RealNumber < 0.0);                                 {}
{}    IF Negative                                                     {}
{}       THEN RealNumber := -RealNumber;                              {}
{} {** Initialize Exponent. **}                                       {}
{}    Exponent := 0;                                                  {}
{} {** If there's a fractional part, we need more precision.. **}     {}
{}       WHILE (RealNumber > (Trunc(RealNumber) + Epsilon))           {}
{}       AND (Trunc(RealNumber) < 16383 ) DO                          {}
{}          BEGIN                                                     {}
{}          RealNumber := RealNumber * 2.0;                           {}
{}          Exponent := Exponent - 1;                                 {}
{}          END;                                                      {}
{} {** If the number's a large integer, we need a pos. exponent **}   {}
{}    WHILE (RealNumber > 32767.0) DO                                 {}
{}       BEGIN                                                        {}
{}       RealNumber := RealNumber / 2.0;                              {}
{}       Exponent := Exponent + 1;                                    {}
{}       END;                                                         {}
{} {** Okay, now ship off the Mantissa and Exponent. **}              {}
{}    Mantissa := Trunc(RealNumber + 0.5);                            {}
{}    IF Negative                                                     {}
{}       THEN Mantissa := -Mantissa;                                  {}
{} {** Ship off the Mantissa and Exponent. **}                        {}
{}    SendInt(Mantissa);                                              {}
{}    SendInt(Exponent);                                              {}
{} END;                                                               {}
{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}{}
```

3892-74

Figure 7-12. Sending a Real Parameter in PASCAL.

Real-Report Message Type

Syntax

```
real-report =   int-report:mantissa
                int-report:power-of-two
```

DESCRIPTION

The terminal sends *real-report* messages to the host as part of larger report sequences in response to report commands. The *real-report* consists of two *int-reports*. The real number reported is derived from the two integers by multiplying the first integer by two raised to the power of the second integer.

REFERENCES

Int-report parameter type

RECTANGLE-FILL Command

Host Syntax

```
EcRR  xy:first-corner
      xy:second-corner
      int:fill-index
```

Setup Syntax

```
EcRR Sp  first-corner
         second-corner
         fill-index
```

PARAMETERS

first-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)
Specifies one corner of a rectangle in raster memory space.

second-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)
Specifies the opposite corner of that rectangle.

fill-index (0 to 32767)
The color index with which the rectangle is to be filled.

DESCRIPTION

This command sets all pixels of the specified rectangle, on the current surface, to the specified color index. The "current surface" is the one specified in the most recent BEGIN-PIXEL-OPERATIONS command. The color indices are set according to the ALU mode specified in that same BEGIN-PIXEL-OPERATIONS command.

If the first and second corners of the rectangle have the same x-value, then the rectangle filled will be one pixel wide. Likewise, if the first and second y-values are the same, then the rectangle filled will be one pixel high.

DEFAULTS

first-corner
as shipped — none
on power-up — none
if omitted — (0,0)

second-corner
as shipped — none
on power-up — none
if omitted — (0,0)

fill-index
as shipped — none
on power-up — none
if omitted — 0

ERRORS

RR11	(Level 2):	Invalid *first-corner* coordinates.
RR21	(Level 2):	Invalid *second-corner* coordinates.
RR31	(Level 2):	Invalid *fill-index*.

REFERENCES

BEGIN-PIXEL-OPERATIONS command

INQUIRY COMMAND

none

RENAME-FILE Command

OPTIONS 42, 43, 45, 46, 47

Host Syntax

```
EcJR  device:old-filename
      string:separator
      device:new-filename
```

Setup Syntax

```
RENAME Sp  old-filename  TO  new-filename
```

PARAMETERS

old-filename
The file name you want to change. See *Device parameter type.*

separator
The empty string or **TO**.

new-filename
The name with which you want to replace the old-filename. See *Device* parameter type.

DESCRIPTION

This command renames the file specified by the *old-filename* parameter to the name specified by the *new-filename* parameter.

The two file names may have the same device name fields.

The file is not renamed and an error occurs if the file is write-protected, if the *old-filename* does not exist, if a file already exists with the *new-filename*, or if the disk unit is not present.

DEFAULTS

old-filename
as shipped — none
on power-up — none
if omitted — error JR11

separator
as shipped — none
on power-up — none
if omitted — error JR21

new-filename
as shipped — none
on power-up — none
if omitted — error JR31

ERRORS

JR00 (Level 0): Unrecognized command. (Disk drive option is not installed.)

JR10 (Level 2): The specified file does not exist or the device is not installed.

JR11 (Level 2): Invalid *old-filename* specifier.

JR12 (Level 3): Out of memory while parsing parameter.

JR13 (Level 2): The specified file is write-protected or*old-filename* is not a disk device.

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): Out of memory while parsing parameter.

JR30 (Level 2): The device specified is not installed or is different from the device specified in parameter 1, or the *new-filename* is already present.

JR31 (Level 2): Invalid *new-filename* specifier.

JR32 (Level 3): Out of memory while parsing parameter.

JR33 (Level 2): *New-filename* is not a disk device.

JR39 (Level 2): Disk hardware error. (I/O or drive not ready.)

REFERENCES

Device parameter type
PROTECT-FILE command

INQUIRY COMMAND

none

RENAME-SEGMENT Command

Host Syntax

$^{E}_{C}$SR int:*old-segment-number*
int:*new-segment-number*

Setup Syntax

$^{E}_{C}$SR $^{S}_{P}$ *old-segment-number*
new-segment-number

PARAMETERS

old-segment-number (1 to 32767)
Number of the segment being renamed.

new-segment-number (1 to 32767)
New number for the segment.

DESCRIPTION

This command renames (renumbers) an existing segment. The segment number is changed to the *new-segment-number*, shown above as the second int parameter. If a segment with the new segment number already exists, an error occurs and the segment is not renamed.

DEFAULTS

old-segment-number
as shipped — none
on power-up — none
if omitted — error SR11

new-segment-number
as shipped — none
on power-up — none
if omitted — error SR21

ERRORS

SR02 (Level 3): Out of memory while renaming a segment.

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 *old-segment-number.*

SR20 (Level 2): A segment with that segment number already exists.

SR21 (Level 2): Invalid *new-segment-number.*

REFERENCES

BEGIN-NEW-SEGMENT command
BEGIN-SEGMENT command

INQUIRY COMMAND

none

RENEW-VIEW Command

Host Syntax

ECKN int:*view-number*

Setup Syntax

RENEW SP *view-number*

PARAMETERS

view-number (–32768 to 32767)
Specifies the view to be renewed.

–32768 to **–1**	all views
0	the current view
1 to **64**	a specific view
65 to **32767**	view 64

DESCRIPTION

This command causes the terminal to wipe the specified view with the view's wipe index (if the wipe index is not –1) and redraw all segments visible in that view, plus the border and the Local Viewing function box if they are visible.

Specifying *view-number* 0 causes the current view to be renewed. Specifying view number –1 causes the terminal to erase the screen and then renew all views in sequence, starting with the next higher view than the current view, followed by all the defined views up to 64, then by the defined views starting from 1 and ending with the current view. If the specified view does not exist, an error is detected and no action occurs.

When a view is renewed on a 4128 and 4129, 3D segments are displayed before 2D segments. If the view's object-surface-display attribute is "wireframe", the 3D segments are displayed in priority order, with the highest priority displayed last. If the view's object-surface-display attribute is "shaded-surface" or "hidden-line", the 3D segments are displayed first in priority order as "wireframe" and then all at the same time in "shaded-surface" or "hidden-line" images. While the terminal is "shading" the view, the light on the VIEW key is lit. When two 3D primitives have the same "depth", the one drawn first in the segment with the higher priority is the one displayed (on a pixel-by-pixel basis). If back-facing surfaces are suppressed (as set by the SELECT-TRAVERSAL-DIRECTION command), individual triangles in facets which are not front-facing are not shaded; in this case the shading operation may take noticeably less time.

After all 3D segments have been displayed, 2D segments are displayed in priority order, so that the 2D segment(s) are displayed last.

For segments which have the same display priority, the order of display is not defined.

If a segment is being defined, it is redisplayed last, regardless of display priority or dimensionality. If a 3D segment is being defined, it is redisplayed in "wireframe" mode regardless of the view's object-surface-display attribute.

When a RENEW-VIEW command is issued for a single view in a view display cluster, all views in the cluster are renewed. For details, see the description of the SET-VIEW-DISPLAY-CLUSTER command.

DEFAULTS

view-number
as shipped — none
on power-up — none
if omitted — 0

ERRORS

KN02 (Level 3): Out of memory while attempting to renew a view. (This error can also occur as a result of pressing the PAGE key.)

KN10 (Level 2): The view specified does not exist.

KN11 (Level 2): Invalid *view-number*.

REFERENCES

SELECT-VIEW command
SELECT-TRAVERSAL-DIRECTION command
SET-SEGMENT-DISPLAY-PRIORITY command
SET-SEGMENT-VISIBILITY command
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEW-DISPLAY-CLUSTER command

INQUIRY COMMAND

none

REPLACE-PART-OF-SEGMENT Command

4115, 4115P51, or 4120 SERIES

Host Syntax

$^{E}_{C}$**UE** int:*segment-number*
int:*pickIDfirst* int:*pickIDlast*

Setup Syntax

$^{E}_{C}$**UE** $^{S}_{P}$ *segment-number* *pickIDfirst*
pickIDlast

PARAMETERS

segment-number (1 to 32767)
Segment in which primitives and primitive attributes will be replaced.

1 to **32767** the segment number.

pickIDfirst (–1 and 1 to 32767)
The pickID group at which replacement will begin.

–1	The segment end.
1	The segment beginning.
2 to **32767**	A particular pickID group.

pickIDlast (–1 and 1 to 32767)
The pickID group after which replacement ends.

–1	The segment end.
1	The segment beginning.
2 to **32767**	A particular pickID group.

DESCRIPTION

This command deletes the specified pickID group(s) from an existing segment and leaves the segment open for insertion at that point. This is equivalent to a DELETE-PART-OF-SEGMENT command with the same parameters as this command, followed by an INSERT-INTO-SEGMENT command with the same *segment-number*, a *pickID-group* indicating the pickID group preceding the deleted group(s), and a *beforeorafter* of 1.

See the descriptions of DELETE-PART-OF-SEGMENT and INSERT-INTO-SEGMENT commands for details.

If the dimensionality of the specified segment does not match the current coordinate mode of the terminal an error is detected and no action occurs.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error UE11

pickIDfirst
as shipped — none
on power-up — none
if omitted — error UE21

pickIDlast
as shipped — none
on power-up — none
if omitted — error UE31

ERRORS

UE02 (Level 3): Out of memory.

UE03 (Level 2): Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.

UE10 (Level 2): Specified segment does not exist.

UE11 (Level 2): Invalid *segment-number*.

UE13 (Level 2): The dimensionality of the specified segment does not match the current coordinate mode of the terminal.

UE20 (Level 2): Cannot find *pickIDfirst*.

UE21 (Level 2): Invalid *pickIDfirst*.

UE23 (Level 2): Invalid because of transformations in an included segment.

UE30 (Level 2): Cannot find *pickIDlast*.

UE31 (Level 2): Invalid *pickIDlast*.

UE33 (Level 2): Invalid because of included segment or panel begin/end boundaries.

REFERENCES

DELETE-PART-OF-SEGMENT command
INSERT-INTO-SEGMENT command

INQUIRY COMMAND

none

REPORT-COLORHARDCOPY-STATUS Command

4111 or OPTION 19

Host Syntax

```
EcQQ
```

Setup Syntax

```
EcQQ
```

DESCRIPTION

The REPORT-COLOR-HARDCOPY-STATUS command causes the terminal to send a *color-hardcopy-status-report* to the host computer.

ERRORS

QQ00	(Level 0):	Unrecognized command. (Must be 4111 or Option 19 must be installed.)

REFERENCES

Color-hardcopy-status-report message type

INQUIRY COMMAND

none

REPORT-DEVICE-STATUS Command

Host Syntax

```
EcJQ    device:device-specifier
```

Setup Syntax

```
EcJQ Sp    device-specifier
```

PARAMETERS

device-specifier
Specifies the device whose status is to be reported to the host. See *Device* parameter type.

DESCRIPTION

This command causes the terminal to send a *device-status-report* for the specified device to the host computer.

If a valid but not installed device is specified, the terminal detects error JQ10 and returns 0 as the *status-integer* in the *device-status-report*.

If an invalid device code is specified as the string parameter, the terminal detects error JQ11. Nevertheless, the terminal sends a *device-status-report* to the host computer with SpSp as the returned two-character *device-mnemonic*, and 0 as the *status-integer*.

DEFAULTS

device-specifier
as shipped — none
on power-up — none
if omitted — error JQ11

ERRORS

JQ10	(Level 2):	Device is not installed.
JQ11	(Level 2):	Invalid device specifier.
JQ12	(Level 3):	Out of memory while parsing parameter.

REFERENCES

Device parameter type
Device-status-report message type

INQUIRY COMMAND

none

REPORT-ERRORS Command

Host Syntax

$^{E}{}_{C}$KQ

Setup Syntax

$^{E}{}_{C}$KQ

DESCRIPTION

The Report-ERRORS command causes the terminal to send an *errors-report* message to the host computer. See the description of the *error-report* message type for details.

REFERENCES

Appendix C, "Error Codes"
Error-report message type

INQUIRY COMMAND

none

REPORT-GIN-POINT Command

Host Syntax

$^{E}_{C}$**IP** int:*code*

Setup Syntax

$^{E}_{C}$**IP** $^{S}_{P}$ *code*

PARAMETERS

code
Valid values are –2 and the valid *device-functions* for the ENABLE-GIN command.

DESCRIPTION

This command forces the terminal to send a GIN report for the specified *code* to the host without any operator interaction.*Code* specifies either the current graphic beam position or a GIN device and function.

If *code* is –2, the terminal reports the current graphic beam position. The format for the report is the same as for the locator function with $^{S}_{P}$ as the *key char-report*. Separate signature characters may be assigned for this report. (See the *GIN-report* message type and the SET-REPORT-SIG-CHARS command.)

In the 4128 and 4129, the beam position reported corresponds to the current coordinate mode. If the terminal is in 2D mode (coordinate mode 0 or 1), then the 2D beam position will be reported; if the terminal is in 3D mode (coordinate mode 2), the 3D beam position will be reported.

If *code* is a GIN device-function, the terminal causes a GIN event for the device function. If the device function is not currently enabled, the terminal first enables it for one point, as if the ENABLE-GIN command had been received. The *key char-report* is $^{S}_{P}$ in the one or more *GIN-report*s caused by the GIN event. See the *GIN-report* message type for details. (If this GIN event exhausts the GIN count, a *GIN-termination-report* is also sent.)

The GIN cursor segment "blinks" once for each GIN event.

DEFAULTS

code
as shipped — none
on power-up — none
if omitted — error IP11

ERRORS

I011 (Level 2): Invalid *code*.

IE03 (Level 2): Context error. Command is invalid at this time. (The segment being used as the cursor for the specified *device-function* is a segment which is currently being defined; or, in the 4128 and 4129, a 3D segment is specified as a GIN cursor for a pick or stroke function.)

IE10 (Level 2): The specified GIN device is not installed in the terminal.

IE13 (Level 2): The protocol assigned to the peripheral port does not support GIN, or, on the 4128 and 4129, a locator function is enabled for GIN with a 3D cursor and either inking or rubberbanding is specified.

IP13 (Level 2): The *code* names a GIN device which has already been enabled for a different GIN function.

REFERENCES

ENABLE-GIN command
GIN-report message type
SET-REPORT-SIG-CHARS command

INQUIRY COMMAND

none

REPORT-PORT-STATUS Command

4111 or OPTION 10

Host Syntax

```
EcPQ   device:port
```

Setup Syntax

```
STATUS 3PPI
```

PARAMETERS

port
Specifies the peripheral port whose status is to be reported. Valid devices are:

P0:
P1:
P2: (4115/4120 Series only)

DESCRIPTION

When the REPORT-PORT-STATUS command is received from the host or in local mode, the terminal sends a *port-status-report* for the specified RS-232 peripheral port to the host computer. When the command is invoked by the operator in Setup mode, the terminal displays status information about all ports on its screen.

If *port* is not **P0:**, **P1:**, or **P2:**, the terminal detects a type PQ11 error. Nevertheless, it still sends a *port-status-report* to the host computer. That report, however, is abbreviated; its *port-ID-code* consists of two "space" characters, and the *port-information* is omitted. For details, see the description of the *Port-status-report* message type.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PQ11

ERRORS

PQ00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PQ11	(Level 2):	Invalid port identifier (must be **P0:**,**P1:**, or **P2:** [P2: not valid for 4111]).
PQ12	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

Port-status-report message type

INQUIRY Command

none

REPORT-SEGMENT-STATUS Command

Host Syntax

ECSQ int:*segment-number*
char-array:*status-codes*

Setup Syntax

ECSQ SP *segment-number* *status-codes*

PARAMETERS

segment-number (–3 to 32767)
The segment for which you want information.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
0	the crosshair cursor
1 to **32767**	a specific segment

status-codes (length 0 to 65535)
An array specifying the information to be returned for the specified segment. Valid array characters are:

A	classes
D	detectability
H	highlighting mode
I	image transform
M	writing mode
P	pivot point
S	display priority
V	visibility
W	matrix transform (4125, 4128 and 4129 only)
X	position
Z	dimensionality (4125, 4128 and 4129 only)

DESCRIPTION

This command causes the terminal to send one or more *segment-status-reports* to the host computer.

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist or if the dimensionality of the specified segment does not match the coordinate mode of the terminal.

Segment 0 refers to the crosshair GIN cursor. Segment –1 refers to all currently defined segments (with numbers from 1 to 32767) whose dimensionality matches the current coordinate mode. Segment –3 refers to all segments that match the current segment matching class and whose dimensionality matches the current coordinate mode of the terminal. Segment –2 refers to future segments. The 4128 and 4129 have two future segment *positions*, one for 2D and one for 3D.

Status-codes specifies which information about the segment (or segments) is to be reported. If it is empty (has a length of zero), then the only information in the *segment-status-report* will be the segment number(s).

For information about the reports the terminal sends to the host in response to this command, see the *Segment-status-report* message type.

If you enter this command in Setup mode, the report is sent to the host or, if the Local key light is also on, to the screen.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — 0

status-codes
as shipped — none
on power-up — none
if omitted — empty array

ERRORS

SQ10	(Level 2):	Segment does not exist.
SQ11	(Level 2):	Invalid *segment-number.*
SQ21	(Level 2):	Invalid *status-codes.*
SQ22	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

Segment-status-report message type

INQUIRY COMMAND

none

REPORT-TERMINAL-SETTINGS Command

Host Syntax

$^{E}_{C}$**IQ** char:*first* char:*second*

Setup Syntax

STATUS $^{S}_{P}$ $^{E}_{C}$ *first* *second*

PARAMETERS

first
Specifies the first inquiry code ASCII character ($^{S}_{P}$ to ~, ADE 32 to 126).

second
Specifies the second inquiry code ASCII character ($^{S}_{P}$ to ~, ADE 32 to 126).

DESCRIPTION

This command causes the terminal to send a *terminal-settings-report* to the host computer.

REFERENCES

Terminal-settings-report message type

INQUIRY COMMAND

none

REPORT-4010-STATUS Command

Host Syntax

```
ECEQ
```

DESCRIPTION

This command causes the terminal to send a *4010-status-report* to the host. If 4010-GIN was enabled, it is disabled and alpha mode is entered.

REFERENCES

4010-status-report message type
ENTER-ALPHA-MODE command

INQUIRY COMMAND

none

RESERVE-COLOR-COPIER Command

4111 or OPTION 19

Host Syntax

```
ECQR    int:reserve-code
```

Setup Syntax

```
HCRESERVE SP reserve-code
```

PARAMETERS

reserve-code (0 or 1)
Specifies whether 4691 or 4692 Color Graphics Copier with OPTION 02 Multiplexer has been told to be reserved. Setup mode parameters are **YES** and **NO**.

0	**NO**	unreserve — color copier is free to poll other connected devices.
1	**YES**	reserve — color copier is restricted from polling other connected devices.

DESCRIPTION

This command reserves the 4691 or 4692 Color Graphics Copier equipped with Option 02 (the four-channel multiplexer). The 4691 or 4692 with Option 02 can poll up to four connected devices and receive copies from any of them. The copier is prevented from polling the other connected devices while copies are being made.

If the 4691 or 4692 does not have Option 02, or if a 4695 is connected to an Option 19 color hardcopy interface or 4111, then this command causes no action.

Reserving the copier can be particularly useful when several copies need to be made sequentially without interruption from another multiplexed device.

If the copier is reserved or in the process of making a copy, then it issues a Busy signal to all other multiplexed devices. When a hard copy request (issued by pressing the HARD COPY key or by the HARDCOPY escape sequence) is sent to a busy copier, then the indicator light in the HARD COPY key blinks, indicating the Busy condition. The light continues to blink until the copier resumes polling and sees the copy request.

If the copier is already reserved by another device and you issue a reserve request, the terminal responds as if the copier is busy; that is, the light in the HARD COPY key blinks until the copier resumes polling and can honor the reserve request.

If the copier is reserved, device polling is restored when:

- The command HCRESERVE No is issued.
- The CANCEL key on the terminal is pressed. Pressing the CANCEL key, however, aborts other operations in progress (see the discussion of CANCEL).
- RESUME (on the multiplexer) is pressed. This does *not* reset the terminal's status setting for this command.

NOTE

To find out the reserve status of the copier, check the copier itself, not the status of the RESERVE command.

DEFAULTS

reserve-code
as shipped — 0
on power-up — 0
if omitted — 1

ERRORS

QR00	(Level 0):	Unrecognized command. (Must be a 4111 or Option 19, version 5 or later, must be installed.)
QR03	(Level 2):	The color hardcopy interface is busy.
QR11	(Level 2):	Invalid *reserve-code.*
QR19	(Level 2):	No color copier is connected.

REFERENCES

CANCEL key
HARDCOPY command
HARD COPY key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

RESET Command

Host Syntax

$^{E}{}_{C}$KV

Setup Syntax

$^{E}{}_{C}$KV

DESCRIPTION

The RESET command initializes the terminal to its power-up condition. It is equivalent to pressing the MASTER RESET key, or to turning the terminal off and then turning it on again.

NOTE

The terminal takes from 10 to 30 seconds to execute the RESET command. During that time, it is performing its power-up reset and self-test routines, and cannot process data coming from the host.

Therefore, if you issue a RESET command from the host, you should wait at least 30 seconds before sending other commands or data to the terminal.

INQUIRY COMMAND

none

RESTORE Key

DESCRIPTION

Pressing this key (the SHIFTed version of the VIEW key) restores the current view's viewing parameters (in 2D mode, set by the SET-WINDOW command; in 3D mode, set by the SET-VIEWING-TRANSFORM command) and Local Viewing function box to the size, shape, and position as stored on the front of the terminal's circular restore-queue, and moves the front pointer to the next entry. If the terminal is not in a Local Viewing function — that is, if neither the ZOOM light nor the PAN light is on — then the "restored" Local Viewing function box will not be visible.

If the current view is part of a view display cluster, then the RESTORE key affects all views in that display cluster. (See the SET-VIEW-DISPLAY-CLUSTER command for details.)

Viewing parameters and Local Viewing function box values are entered into a circular queue each time that the VIEW key is pressed. Up to four sets of "old values" for the viewing parameters and Local Viewing function box can be stored.

If the CTRL key is pressed together with RESTORE, then the original viewing parameters and Local Viewing function box are restored.

The RESTORE key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

PAN key
SET-VIEW-DISPLAY-CLUSTER command
SET-VIEWING-TRANSFORM command
SET-WINDOW command
VIEW key
ZOOM key
4128/4129 Operators Manual

RESTORE-PIXELS-FROM-MEMORY Command

4111, 4115P51, or 4120 SERIES

Host Syntax

$^{E}_{C}$**UK** int:*ID-number* int:*restore-flag*
xy:*lower-left*

Setup Syntax

$^{E}_{C}$**UK** $^{S}_{P}$ *ID-number* *restore-flag*
lower-left

PARAMETERS

ID-number (–1 and 1 to 32767)
Specifies main memory area to be restored.

–1	Release all memory buffers used to save pixels
1 to **32767**	A specific memory buffer

restore-flag (0 to 3).
The two low bits of this integer parameter are two separate flag bits: Bit 0 controls memory release and bit 1 controls use of the *lower-left* parameter.

0	Release memory; use original position.
1	Retain memory; use original position.
2	Release memory; use position specified by *lower-left.*
3	Retain memory; use position specified by *lower-left.*

lower-left (0,0 to 4095,4095)
Specifies the pixel position at which the pixels are to be restored.

DESCRIPTION

This command restores a rectangular area of pixels to the screen from a numbered buffer in main memory identified by *ID-number*. Initially, the SAVE-PIXELS-TO-MEMORY command assigns the number to the memory buffer area. The rectangular area is restored at the starting position specified by *lower-left* or at the position it was saved from, depending on bit 1 of *restore-flag*.

If *restore-flag* is **0** or **1**, the region is restored to the position it was saved from. If *restore-flag* is **2** or**3**, the region is restored at the starting position specified by *lower-left*. If *lower-left* is omitted, the region is restored at position 0,0.

If *restore-flag* is **0** or **2**, the main memory area is deleted (freed). If *restore-flag* is **1** or **3**, the main memory area is retained for future restores.

Pixels which would fall off the screen because of a peculiar *lower-left* are not restored. Memory can be released without restoring the image by specifying a *lower-left* value equal to or greater than 1280,1024 (or, for a 4111: 1024,768).

If *ID-number* is **–1**, all memory buffers used to save pixels are released without restoring any pixels and the restore-flags have no meaning regardless of *restore-flag*.

Pixel Writing Factors. The pixel writing factors and axis, as set by SET-PIXEL-WRITING-FACTORS command, determine the size, starting corner, and direction of the pixels being restored.

DEFAULTS

ID-number
as shipped — none
on power-up — none
if omitted — error UK11

restore-flag
as shipped — 0
on power-up — 0
if omitted — 0

lower-left
as shipped — 0,0
on power-up — 0,0
if omitted — 0,0

ERRORS

UK10	(Level 2):	*ID-number* not previously saved.
UK11	(Level 2):	Invalid *ID-number*.
UK21	(Level 2):	Invalid *restore-flag*.
UK31	(Level 2):	Invalid *lower-left* value.

REFERENCES

SAVE-PIXELS-TO-MEMORY command
SET-PIXEL-WRITING-FACTORS command

INQUIRY COMMAND

none

RUNLENGTH-WRITE Command

Host Syntax

$^{E}_{C}$**RL** int-array:*indices-array*

Setup Syntax

$^{E}_{C}$**RL** $^{S}_{P}$ *indices-array*

PARAMETERS

indices-array (length 0 to 16384)
Specifies the color indices of pixels (0 to $2^{31}-1$).

DESCRIPTION

This command sets the color indices of "runs" in a fill pattern definition or in the pixel viewport. A "run" of pixels is just a contiguous line of pixels which wraps the edge of the area being filled.

If no fill-pattern definition is open, for each specified index all pixels within the pixel rectangle defined by the current pixel writing factors are set to that index. The pixel beam position is then moved according to the current pixel writing factors. See SET-PIXEL-WRITING-FACTORS command for details.

If a fill-pattern definition is open, for each specified index one pixel is to that index, and the next pixel position is moved on to the right. When it goes past the right edge of the pattern, the next pixel position is moved down one row and to the left edge. When it goes past the right edge of the last row, the fill-pattern definition is closed and any remaining indices in the command are discarded.

The indices are packed into *indices-array* by setting the low bits of an element to represent an index and the high bits to represent the number of pixels in the run. Where N is the number of bits per pixel (as set by RASTER-WRITE command), the terminal uses this algorithm to extract color indices from *index-array*:

```
For J = 1 to length of indices-array:
  Set Runcode to indices-array (J)
  Set Index to Runcode AND $2^N-1$
  Shift Runcode right N bits
  For K = 1 to Runcode
    Set Index-of-pixels to Index
    Move pixel-position
```

DEFAULTS

indices-array
as shipped — none
on power-up — none
if omitted — empty array

ERRORS

RL11	(Level 2):	Invalid *indices-array*.
RL12	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

BEGIN-FILL-PATTERN command
BEGIN-PIXEL-OPERATIONS command
SET-PIXEL-WRITING-FACTORS command

INQUIRY COMMAND

none

SAVE Command

Host Syntax

$^{E}{}_{C}$**JV** string:*item-type* int:*item-number*
string:*separator* device:*destination*

Setup Syntax

SAVE $^{S}{}_{P}$ *item-type*
item-number **TO** *destination*

PARAMETERS

item-type (length of 3)
The type of item to be saved. Valid values are:

MAC macro definition

SEG segment definition

RAS pixels with RASTER-WRITE commands

RUN pixels with RUNLENGTH-WRITE commands

item-number
The macro or segment or number of pixels to be saved, Valid values depend on *item-type*:

for MAC, all values valid for DEFINE-MACRO (–1 means all)

for SEG, –4, –3, –1, and 1 to 32767

for RAS and RUN, –2 to 32767

separator
The empty string or **TO**.

destination
The device and/or file to receive the saved item. See *Device* parameter type.

DESCRIPTION

The SAVE command causes the terminal to save an item from its internal memory by sending the series of terminal commands necessary to recreate the item to the specified destination device. The "item" may be:

- A macro definition
- A segment definition
- Pixels from the current pixel viewport

The destination device can be any file device capable of receiving general several-bit ASCII code.

If *item-type* is "MAC", *item-number* is a macro number. Valid macro numbers are the same as for DEFINE-MACRO, except that –1 means "all macro definitions."

If *item-type* is "SEG", *item-number* is a segment number. Segment 1 to 32767 numbers represent specific user-defined segments. Segment number –1 means "all user-defined segments," and segment number –3 means "all segments that match the current segment matching class." Segment –4 means "all segments visible in the current view" and causes an action equivalent to the PLOT command. For segment number –4, all calls to segments are expanded into their primitives; all other segment numbers only save the calls, not the called segment. The coordinate mode is initially set to 0 (smaller range) until an x- or y-coordinate greater than 4095 is to be saved. At this point, a SET-COORDINATE-MODE command is issued to change the coordinate mode to 1. 3D segments cannot be "saved" by this command. However, a functionality similar to SAVE is: copy SG: to *destination*. To bring the 3D segment back to the terminal screen, enter: copy *source* to SG:. (See the *Device* parameter type for details on SG:.)

If *item-number* is "RAS" or "RUN", *item-number* specifies the number of pixels to be saved, starting at the current pixel beam position. The RASTER-WRITE command is saved if *item-number* is "RAS", while RUNLENGTH-WRITE is used for "RUN." The pixel beam position is updated after the SAVE command has been executed to the position following the last pixel saved. If *item-number* is –1, all the pixels in the pixel viewport are saved, in which case the pixel beam position is not changed.

If *item-number* is –2, a BEGIN-PIXEL-OPERATIONS command with current bits-per-pixel (surface and ALU, both), a SET-PIXEL-VIEWPORT command with the current settings, and a SET-PIXEL-WRITING-FACTORS command with the current settings are saved first, followed by the pixels from the entire pixel viewport. The pixel beam position is not changed. However, –2 provides all the information necessary to reconstruct the pixel image except the surface and color map settings.

When a macro definition is saved, a DEFINE-MACRO command is sent to the destination device for each saved macro.

For each segment saved, the following commands are sent to the destination device:

- For segment −4 WINDOW and VIEWPORT commands, specifying the current view's window and viewport.
- A series of commands to set most future segment dynamic and static attributes, such as SET-SEGMENT-HIGHLIGHTING, SET-PIVOT-POINT, SET-SEGMENT-IMAGE-TRANSFORM. Note that SET-SEGMENT-VISIBILITY is not saved.
- A SET-GRAPHTEXT-PRECISION: 1 command, setting the graphtext precision to "string precision."
- A BEGIN-SEGMENT command for the segment specified in the SAVE command's second parameter.
- A series of graphic primitives, and commands to set primitive attributes. These might include ENTER-VECTOR-MODE commands, xy coordinates, MOVE, DRAW, DRAW-MARKER commands, SET-LINE-STYLE commands, etc.
- Alphatext is saved as string-precision graphtext.

When a segment is subsequently loaded, it may change some of the future segment attributes which were in effect before the segment was loaded. To see the commands which are stored in the file created by a SAVE command, put the terminal in snoopy mode and load the file.

DEFAULTS

item-type
as shipped — none
on power-up — none
if omitted — error JV11

item-number
as shipped — none
on power-up — none
if omitted — error JV21

separator
as shipped — none
on power-up — none
if omitted — error JV31

destination
as shipped — none
on power-up — none
if omitted — error JV41

ERRORS

JV02 (Level 3): Out of memory while attempting DMA transfer (4111 or Option 3A or 3C only).

JV10 (Level 2): Invalid *item-type*.

JV12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JV20 (Level 2): The specified macro or segment does not exist, or segment is being defined.

JV21 (Level 2): Invalid *item-number*.

JV31 (Level 2): Invalid *separator* (must be empty string or **TO**).

JV32 (Level 3): Out of memory while parsing the parameter.

JV40 (Level 2): The specified *destination* is not installed.

JV41 (Level 2): Invalid *destination* specifier.

JV42 (Level 2): Out of memory while parsing the parameter, or while executing the command.

JV43 (Level 2): Not a valid *destination* device, device is busy, or existing disk file is write protected or open.

JV49 (Level 2): Device hardware error. (I/O error, drive not ready, hardware write-protect error, or DMA block transfer error.)

REFERENCES

BEGIN-PIXEL-OPERATIONS command
DEFINE-MACRO command
Device parameter type
LOAD command
PLOT command
RASTER-WRITE command
RUNLENGTH-WRITE command
SET-COORDINATE-MODE command
SET-CURRENT-MATCHING-CLASS command
SET-PIXEL-VIEWPORT command
SET-PIXEL-WRITING-FACTORS command

INQUIRY COMMAND

none

SAVE-NONVOLATILE-PARAMETERS

4111

Host Syntax

$^{E}_{C}$KU

Setup Syntax

NVSAVE

DESCRIPTION

This command saves in a nonvolatile memory all parameters that are saved in the 4115/4120 Series CMOS memory. (Key macros are not saved as they are in the 4105, 4107, and 4109.)

Each byte of nonvolatile memory has a lifetime of about 10,000 writes. If you attempt to write to nonvolatile memory after that, the terminal might display an error message (depending on the SET-ERROR-THRESHOLD level setting) that states there is a "nonvolatile memory hardware error." When that occurs, all settings are reset to factory default the next time the terminal is powered up.

ERRORS

KU09 (Level 2): Nonvolatile memory hardware error. Nonvolatile memory did not store data correctly.

REFERENCES

SET-ERROR-THRESHOLD

SAVE-PIXELS-TO-MEMORY Command

4111, 4115P51, or 4120 SERIES

Host Syntax

EcUJ int:*id-number* xy:*first-corner*
xy:*second-corner*

Setup Syntax

EcUJ Sp *id-number* *first-corner*
second-corner

PARAMETERS

id-number (1 to 32767)
An identifying number for the main memory area used.

first-corner (4111: 0,0 to 1023,767; 4115/4120 Series: 0,0 to 1279,1023)
Specifies one corner of region to be saved.

second-corner (4111: 0,0 to 1023,767; 4115/4120 Series: 0,0 to 1279,1023)
Specifies opposite corner of region to be saved.

DESCRIPTION

This command saves a rectangular area of pixels from bit plane memory to a numbered buffer area in the main terminal memory.

If the specified *ID-number* is already in use, its memory buffer area is freed before saving the new area of pixels. If enough main memory is not available, an error is detected and nothing is saved.

All bit planes are saved regardless of surface definitions.

Pixel Writing Factors. The current pixel writing factors and axis, as set by the SET-PIXEL-WRITING-FACTORS command, determine the number, starting corner, and direction of the pixels that are saved. This operation uses these factors differently than the SAVE command (which is exactly the inverse of RUNLENGTH-WRITE or RASTER-WRITE), as follows: The pixel saved for each pixel-rectangle is always the lower-left pixel of the pixel-rectangle. The pixel-rectangle grid is always oriented to the lower left of the pixel region being saved. A pixel is saved for each fractional rectangle at the top and right edges of the saved pixel region.

DEFAULTS

id-number
as shipped — none
on power-up — none
if omitted — error UJ11

first-corner
as shipped — none
on power-up — none
if omitted — 0,0

second-corner
as shipped — none
on power-up — none
if omitted — 0,0

ERRORS

UJ02	(Level 3):	Not enough main memory available to store the specified region.
UJ11	(Level 2):	Invalid *ID-number*.
UJ21	(Level 2):	Invalid *first-corner*.
UJ31	(Level 2):	Invalid *second-corner*.

REFERENCES

RESTORE-PIXELS-FROM-MEMORY command
SET-PIXEL-WRITING-FACTORS command

INQUIRY COMMAND

none

Segment-Definition Syntactic Construct

Syntax

```
segment-definition  =  [set-up-future-segment, ...]
                       open-segment
                       [segment-part, ...]
                       close-segment

set-up-future-segment  =  { SET-PIVOT-POINT
                            SET-SEGMENT-CLASS
                            SET-SEGMENT-DETECTABILITY
                            SET-SEGMENT-DISPLAY-PRIORITY
                            SET-SEGMENT-HIGHLIGHTING
                            SET-SEGMENT-IMAGE-TRANSFORM[1]
                            SET-SEGMENT-MATRIX-TRANSFORM[2]
                            SET-SEGMENT-POSITION
                            SET-SEGMENT-SCALING-ROTATION[1]
                            SET-SEGMENT-VISIBILITY
                            SET-SEGMENT-WRITING-MODE }

open-segment  =  { BEGIN-SEGMENT
                   BEGIN-NEW-SEGMENT
                   BEGIN-HIGHER-SEGMENT[3]
                   BEGIN-LOWER-SEGMENT[3]
                   DELETE-PART-OF-SEGMENT[4], [5]
                   INSERT-INTO-SEGMENT[4]
                   REPLACE-PART-OF-SEGMENT[4] }

segment-part  =  { primitive
                   primitive-attribute
                   SET-PICKID
                   INCLUDE-COPY-OF-SEGMENT[6]
                   CALL-SEGMENT }

primitive  =  { MOVE
                DRAW
                DRAW-MARKER
                DRAW-RECTANGLE[1]
                DRAW-CURVE[1]
                DRAW-FACETS[2]
                BEGIN-PANEL-BOUNDARY
                END-PANEL
                GRAPHIC-TEXT[7]
                DISPLAY-ALPHA-TEXT[8]
                BS[8]
                CR[8]
                HT[8]
                LF[8]
                VT[8] }
```

```
primitive-attributes  =  { SET-LINE-STYLE
                           SET-4014-LINE-STYLE
                           SET-LINE-INDEX
                           SET-BACKGROUND-INDICES
                           SET-MARKER-TYPE
                           SET-TEXT-INDEX
                           SET-GRAPHTEXT-FONT[9]
                           SET-GRAPHTEXT-PRECISION[10]
                           SET-GRAPHTEXT-ROTATION[9]
                           SET-GRAPHTEXT-SIZE[9]
                           SET-GRAPHTEXT-SLANT[9]
                           SET-ALPHATEXT-FONT
                           SET-4014-ALPHATEXT-SIZE
                           SET-GRAPHICS-AREA-WRITING-MODE
                           SELECT-FILL-PATTERN
                           SET-PANEL-FILLING-MODE
                           SET-DRAW-BOUNDARY-MODE
                           SELECT-OBJECT-SURFACE-COLOR[2]
                           SELECT-TRANSLUCENCY-PATTERN[2] }

close-segment  =  { END-SEGMENT
                    BEGIN-NEW-SEGMENT[3]
                    BEGIN-HIGHER-SEGMENT[3]
                    BEGIN-LOWER-SEGMENT[3]
                    SET-COORDINATE-MODE }
```

[1] 2D segments only
[2] 3D segments only
[3] Only if a segment is already open.
[4] Only if the segment already exists.
[5] Segment is closed immediately.
[6] Contents of the named segment are included.
[7] Stroke-precision graphtext is remembered as vectors.
[8] Only if the dialog area is disabled.
[9] Not remembered in the segment.
[10] Only SET-GRAPHTEXT-PRECISION:string is remembered.

DESCRIPTION

A segment is a group of graphic primitives and primitive attributes that is remembered locally in the terminal's memory, and that can be manipulated as an entity. The main advantage of segments is that they reduce communications overhead, since they are automatically redrawn when the view is erased and can be duplicated quickly for repeated images. Another advantage is segments can be used as a GIN cursor, either as the primary cursor or to "drag" the segment with a GIN device. (See the SET-GIN-CURSOR command for details on using segments as GIN cursors.)

A newly defined segment is visible only in the current view (if it is visible at all). To make a segment visible in another view, select that view and send a SET-SEGMENT-VISIBILITY command.

Before the Segment Definition

Before defining a segment, set up the future-segment environment to suit your needs. The pivot-point is important since it establishes the segment's origin, about which the segment is scaled and rotated, and which is mapped into the segment's position. Pivot-point is the only static segment attribute; after a segment is opened, its pivot-point cannot be changed. The dynamic segment attributes of the new segment are those previously set (or left default) for segment –2, the future segment. There are two "future segments" in the 4128 and 4129 terminals, one for 2D and one for 3D.

Starting A Segment Definition

When a segment is opened, the current pivot point, dimensionality, and future-segment dynamic attributes are applied to it. The segment will be defined with its pivot point at the location specified by the most recent SET-PIVOT-POINT command. (See the description of the SET-PIVOT-POINT command for details.) There are two pivot points: a 2D pivot point for 2D segments (coordinate mode 0 or 1) and a 3D pivot point for 3D segments (coordinate mode 2). The pivot point is an unchangeable attribute of a segment; therefore, if the segment's pivot point is to be different from that previously defined, it is important to issue an appropriate SET-PIVOT-POINT command before opening the segment.

On the 4128 and 4129, the segment's dimensionality is determined by the coordinate mode at the time the segment is opened, as specified by the SET-COORDINATE-MODE command. If the coordinate mode is 0 or 1 (2D) then the segment's dimensionality attribute is 0 (2D); if the coordinate mode is 2 (3D) then the segment's dimensionality attribute is 1 (3D).

Defining Segment Contents

When a segment is open, all graphic primitives, primitive attributes, pick-ids, and calls to other segments are remembered in the segment. An INCLUDE-COPY-OF-SEGMENT command may be used to insert the contents of one segment into the currently open segment.

The coordinates of the primitives defined by coordinates (such as MOVE and DRAW-RECTANGLE) are stored in the segment as relative coordinates, relative to the last set position. This allows the coordinates to be scaled, rotated and positioned relative to the segment's pivot point.

Stroke-precision graphtext and arcs created by DRAW-CURVE are remembered as groups of MOVEs and DRAWs, and so may be scaled, rotated and positioned with segment transform commands.

Dot-matrix text, created by DISPLAY-ALPHATEXT and string-precision GRAPHIC-TEXT, are stored as groups of characters which do not alter the graphic position within the segment, but do alter the beam position on the screen when they are displayed. This text is not scaled or rotated by segment transforms, but is positioned. (One noticeable effect is that vectors drawn from the end of dot matrix text are transformed to end at the "right" place relative to the beginning of the text, but start at the end of the text. Another effect is that if a middle group of characters separated by pickIDs is deleted from the segment, the following text "closes up" to the preceding text, regardless of segment editing mode.)

A primitive attribute is stored in the open segment under two conditions:

- If a primitive attribute command is received, the attribute is stored at the current location.
- If such a command has not been received since the segment was opened *and the current setting for the attribute is not the terminal's default*, then when the first primitive affected by the attribute is received with the segment open, the primitive attribute is stored in the segment just before the primitive.

For example, this sequence stores a line-index attribute just before the draw:

```
SET-LINE-INDEX: 5
BEGIN-SEGMENT: 1
MOVE: 0,0
DRAW: 100,100
END-SEGMENT
```

The following sequence, however, stores the attribute before pickID 2:

```
BEGIN-SEGMENT: 2
SET-LINE-INDEX: 6
SET-PICKID: 2
MOVE: 0,0
DRAW: 100,100
END-SEGMENT
```

In the first example, the line-index would have to be edited along with the draw, but in the second example the line-index could be edited by itself.

Internal implementation of the BEGIN-PANEL-BOUNDARY, SELECT-FILL-PATTERN, SET-PANEL-FILLING-MODE and SET-DRAW-BOUNDARY-MODE commands groups five attributes into a cluster called panel style (see the Picture Processor op code PANSTL in Appendix F for details). If any of these panel attributes is non-default when a BEGIN-PANEL-BOUNDARY or DRAW-RECTANGLE command is first received within a segment definition, then a complete PANSTL op code is stored in the segment just before the panel or rectangle op codes. When another panel or rectangle is placed in the segment, a PANSTL opcode is also included whenever any of the five attributes have changed.

In the current version of the 4110 and 4120 series workstations, the primitive attribute marker-type is bound to the marker primitive when it is stored in the segment. This means that the type of a marker cannot be changed separately from the marker by inserting a new marker type or by deleting the old one.

Terminating The Segment Definition

A segment definition is terminated by an END-SEGMENT, BEGIN-HIGHER-SEGMENT, BEGIN-LOWER-SEGMENT, BEGIN-NEW-SEGMENT, or SET-COORDINATE-MODE commands. Changing coordinate mode from 2D (0 or 1) to 3D (2) or vice versa closes an open segment.

Segment-Status-Report Message Type

Syntax

```
segment-status-report    =  [report-for-one-segment . . . ]
                            [term-sig-char]
                            EOM-indicator

report-for-one-segment   =  [EOM-indicator]
                            [sig-char]
                            int-report:segment-number-or-error-code
                            [segment-attribute-report . . . ]
                            [EOM-indicator]

segment-attribute-report =  { segment-class-report      }
                            { detectability-report      }
                            { highlighting-report       }
                            { image-transform-report    }
                            { writing-mode-report       }
                            { pivot-point-report        }
                            { display-priority-report   }
                            { visibility-report         }
                            { matrix-transform-report   }
                            { position-report           }
                            { dimensionality-report     }

segment-class-report     =  char-report:A
                            int-array-report:class-numbers

detectability-report     =  char-report:D
                            int-report:0 or 1

highlighting-report      =  char-report:H
                            int-report:0 or 1

image-transform-report   =  char-report:I
                            real-report:x-scale-factor
                            real-report:y-scale-factor
                            real-report:rotation-in-degrees
                            xy + -report:position

writing-mode-report      =  char-report:M
                            int-report:1 or 2

pivot-point-report       =  char-report:P
                            xy + -report:pivot-point

display-priority-report  =  char-report:S
                            int-report:display-priority

visibility-report        =  char-report:V
                            int-report:0 or 1

matrix-transform-report  =  char-report:W
                            real-array-report:3x3 matrix transform

position-report          =  char-report:X
                            xy + -report:position

dimensionality-report    =  char-report:Z
                            int-report:0 or 1
```

DESCRIPTION

The terminal sends a *segment-status-report* to the host computer in response to the REPORT-SEGMENT-STATUS command. When the terminal sends a report to the host, it enters bypass mode. (See ENTER-BYPASS-MODE.)

Overall Syntax. The *segment-status-report* consists of zero or more *report-for-one-segments*, followed by a *term-sig-char* and an *EOM-indicator*.

Term-Sig-Char. The *term-sig-char* is determined by the most recent SET-REPORT-SIG-CHARS command for non-GIN reports. (See the description of the SET-REPORT-SIG-CHARS command for details.) The *term-sig-char* is provided as a convenience for the host routine which parses the *segment-status-report*; it serves to mark the end of the report.

If the *term-sig-char* is set to NU by the SET-REPORT-SIG-CHARS command, it is omitted from the *segment-status-report*. However, setting the *term-sig-char* to NU would be unwise; the host applications program probably needs the *term-sig-char* to tell when it is done parsing the *segment-status-report*.

Final EOM-Indicator. After sending the *term-sig-char*, the terminal ends the *segment-status-report* with an *EOM-indicator*. This *EOM-indicator* is always sent; it helps to ensure that the host applications program receives the preceding characters in a timely manner. (In most host operating systems, the user program does not actually receive a message from the terminal until the message ends with a CR.)

Report-for-One-Segment. Each *report-for-one-segment* describes the attributes for one segment.

The *report-for-one-segment* may begin with an optional *EOM-indicator*. This *EOM-indicator* is provided because of the terminal's "maximum line length" feature. (See the description of the SET-REPORT-MAX-LINE-LENGTH command for details.) If too many characters have already been sent on the current line, so that sending the remainder of the *report-for-one-segment* would cause the maximum line length to be exceeded, then the terminal ends the current line with an *EOM-indicator*. The *sig-char* that follows would then be the first character of the next line of text.

An *EOM-indicator* is sent following each *report-for-one-segment* if the most recent SET-REPORT-EOM-FREQUENCY command specified "more frequently" rather than "less frequently."

The *sig-char* is provided as a convenience for the host program's parsing routine. It serves to mark the beginning of each *report-for-one-segment*. The *sig-char* is a single ASCII character, determined by the most recent SET-REPORT-SIG-CHARS command for non-GIN reports. (See the SET-REPORT-SIG-CHARS command for details.) If the *sig-char* is NU, it is omitted.

After the *sig-char* comes an *int-report*: the segment number (or error code) for the particular segment whose attributes are being reported. If the *char-array* parameter in the REPORT-SEGMENT-STATUS command was empty, then the segment number is the only item reported in the *report-for-one-segment*. An error code (described later) is substituted for the segment number if an invalid segment number or segment attribute code, or if the segment specified does not exist.

Following the segment number, each *report-for-one-segment* contains zero or more *segment-attribute-reports*. There is one *segment-attribute-report* for each letter in the *char-array* parameter of the REPORT-SEGMENT-STATUS command.

Each *segment-attribute-report* contains information about one of the segment's attributes, and begins with the code letter for that attribute. For instance, a *segment-classes-report* begins with the letter A, *visibility-report* begins with the letter V, and a *position-report* begins with the letter X.

The *pivot-point-report* portion of the *segment-attribute-report* is invalid if the segment was created using the **SG:** pseudo device.

The *dimensionality-report* for a 2D segment contains a 0; for a 3D segment it contains a 1.

The *image-transform-report* for a 3D segment contains all zeros.

The *matrix-transform-report* for a 2D segment is a null (zero length) array.

The *pivot-point-report* and *position-report* are returned as either a 2D or 3D point, depending on the current coordinate mode of the terminal.

ERRORS

If the REPORT-SEGMENT-STATUS command specified an invalid segment number, a segment number for a segment which does not exist, or an invalid code letter for a segment attribute, then the *segment-status-report* sent back to the host includes an error code. In that case, the *report-for-one-segment* has the special error code integer in place of the segment number, and there are no *segment-attribute-reports.* Table 7-17 lists these special error codes.

Table 7-17

ERROR CODES IN SEGMENT-STATUS-REPORTS

segment-number **Error Code**	**Meaning**
–32767	The segment number in the REPORT-SEGMENT-STATUS was invalid.
–32766	The REPORT-SEGMENT-STATUS command specified a segment number for a segment which does not exist
–32765	The REPORT-SEGMENT-STATUS command included (in its char-array) a letter which is not a valid segment attribute code.

In addition to the error information in the *segment-status-report*, type SQ10, SQ11, and SQ21 errors are detected in the terminal. These error codes will be sent to the host if a REPORT-ERRORS command is issued. (For details, see the descriptions of the REPORT-ERRORS command and the *Errors-report* syntactic construct.)

REFERENCES

Char-report message type
ENTER-BYPASS-MODE command
Errors-report syntactic construct
Int-report message type
REPORT-ERRORS command
REPORT-SEGMENT-STATUS command
SET-REPORT-EOM-FREQUENCY command
SET-REPORT-MAX-LINE-LENGTH command
SET-REPORT-SIG-CHARS command
XY+-report message type

SELECT-CODE Command

Host Syntax

$^{E}_{C}$**%!** int:*command-set*

Setup Syntax

CODE $^{S}_{P}$ *command-set*

PARAMETERS

command-set (0, 1, 2, or 3)
Selects the command set to be recognized by the terminal.

- **0** **TEK** command set
- **1** **ANSI** X3.64 command set
- **2** **EDIT** environment, ANSI X3.64 command set (4111 only)
- **3** **VT52** command set (4111 only)

Setup mode parameters are **TEK**, **ANSI**, **EDIT** (4111 only), and **VT52** (4111 only).

DESCRIPTION

This command sets the terminal to recognize the commands of either the TEKTRONIX 4100 System, ANSI X3.64, or the VT52 system. Since there are some syntax commonalities between the different command sets, the terminal can recognize only one command set at a time.

If *command-set* is **0**, the terminal is set so that it understands the TEK 4100 commands, which are described in this section. The terminal is also put into Alpha mode.

If *command-set* is **1**, the terminal is set so that it understands the ANSI X3.64 and ISO 6429 commands, which are described in Section 8.

If *command-set* is **2**, the terminal is set up for running VT100 applications programs. This option has the following effects:

- Sets the terminal to recognize ANSI commands.
- Sets Origin mode to absolute.
- Sets dialog area and dialog buffer to 24 lines.
- If no dialog area is enabled, enables dialog area 1 and makes it visible.
- Defines a scrolling region of 24 lines.
- Disables all expansions of programmed keys. All keys assume their default meanings (the programmed meanings can be enabled from Setup by typing KEY-EXPAND YES).
- Sets Insert/Replace mode to Replace.

If *command-set* is **3**, the terminal is set so that it understands the VT52 commands described in Section 9. Entering VT52 mode causes the default character set (determined by the keyboard attached) to be selected as the G0 and G1 sets.

In the 4111, if no dialog area is currently enabled when the terminal exits TEK mode to enter ANSI, EDIT, or VT52 mode, then dialog area 1 is enabled. In that case, when the terminal returns to TEK mode all dialog areas will be disabled. If the terminal receives a setup DAENABLE NO command while in ANSI, EDIT, or VT52 mode, it defers executing that command until it next enters TEK mode. Commands to enable a specific dialog area, however, have immediate effect, and that effect persists even when the terminal returns to TEK mode.

Setup mode has a separate command parser, and is therefore not affected by this command.

DEFAULTS

command-set
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

%!11 (Level 2): Invalid *command-set* (must be 0, 1, 2, or 3).

REFERENCES

ENTER-ALPHA-MODE command

INQUIRY COMMAND

none

SELECT-COLORHARDCOPY-IMAGE-DENSITY Command

4111 or OPTION 19

Host Syntax

```
EcQU  int:density-code
```

Setup Syntax

```
HCDENSITY Sp  density-code
```

PARAMETERS

density-code (0 and 1)
Selects 4692 density code. Setup mode parameters are **LOW** and **HIGH**.

0 **LOW**; low density of 128 dots per inch.

1 **HIGH**; high density of 156 dots per inch.

DESCRIPTION

This command sets the number of dots per inch for color copies made on the 4692 color copier. The size of the copy is always smaller when the density is *high*. However, because the ink dots are closer together, the image will be brighter when the density is *high*.

This command has no effect on the 4691 or 4695 color copiers.

DEFAULTS

density-code
as shipped — 1
on power-up — remembered
if omitted — 1

ERRORS

QU00 (Level 0): Unrecognized command. (Not a 4111 or Option 19,version 5 or later, must be installed.)

QU11 (Level 2): Invalid *density-code.*

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SELECT-FILL-PATTERN Command

Host Syntax

```
EcMP  int:fill-pattern-number
```

Setup Syntax

```
EcMP Sp  fill-pattern-number
```

PARAMETERS

fill-pattern-number (–32768 to 32767)
Numbers from 1 to 32767 represent specific fill patterns. Of these, patterns 1 to 16 are predefined, while patterns 17 through 32767 exist only if defined by the user. Zero and negative numbers represent fill patterns which consist entirely of the corresponding color index. Numbers from –32767 to –256 cause panels not to be filled.

DESCRIPTION

This command selects the pattern used to fill the interior of panels. The pattern number can range from –32768 to 32767, as follows:

- Fill patterns –255 to 0 fill panels with solid color indices. All other negative numbers cause the interiors of subsequent panels to be left unfilled.
- Patterns 1 through 16 are pre-defined; Figure 7-13 shows examples of these fill patterns. Patterns 1 through 16 may be redefined by the user; but if these patterns are deleted (redefined with a height of zero), they will not revert back to the predefined patterns until the terminal is turned off or reset.
- Patterns 17 through 32767 are reserved for the user to define.

If a panel being filled occurs on a surface which has fewer bit planes than the *bits-per-pixel* value used to define the fill pattern, then the terminal uses only the high-order bits of each color index in the fill pattern. (This is the same rule as that employed by RASTER-WRITE, RUNLENGTH-WRITE, and PIXEL-COPY commands.)

Table 7-18 lists the number of *bits-per-pixel* used for each predefined fill pattern.

Table 7-18

NUMBER OF BITS-PER-PIXEL IN PREDEFINED FILL PATTERNS

Fill Pattern Numbers	Bits-Per-Pixel
16	4
–1 to 15	1
–2 and –3	2
–4 to –7	3
–8 to –15	4
–16 to –31	5
–32 to –63	6
–64 to –127	7
–128 to –255	8

DEFAULTS

fill-pattern-number
as shipped — –1
on power-up — –1
if omitted — 0

ERRORS

MP10 (Level 2): Specified fill pattern does not exist (has not been defined).

MP11 (Level 2): Invalid *fill-pattern-number*.

REFERENCES

BEGIN-FILL-PATTERN command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

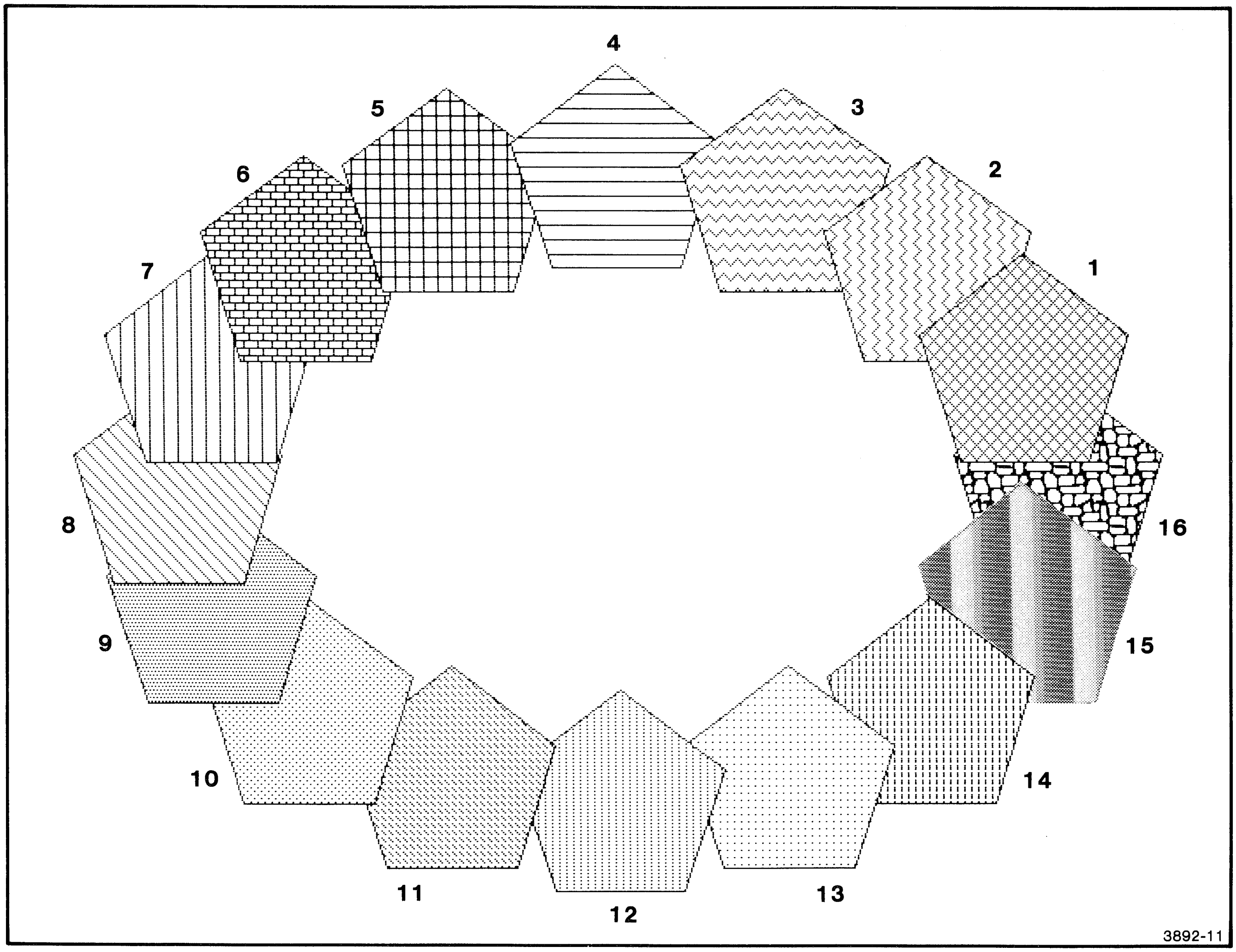


Figure 7-13. Standard Fill Patterns.

SELECT-HARDCOPY-INTERFACE Command

4111 or OPTION 19

Host Syntax

```
EcQD   int:interface
```

Setup Syntax

```
HCINTERFACE Sp   interface
```

PARAMETERS

interface (4111: 1;4115,4120 Series: 0 or 1)
Specifies which hardcopy interface is used when the terminal receives a hardcopy command. Setup mode parameters are **MONO** and **COLOR**.

0 **MONO**: the standard hardcopy interface, to which can be connected TEKTRONIX 4612 and 4632 Video Hard Copy Units.

1 **COLOR**: the Option 19 hardcopy interface, to which can be connected a TEKTRONIX 4691, 4692, or 4695 Color Graphics Copier.

DESCRIPTION

The SELECT-HARDCOPY-INTERFACE command selects the hardcopy interface that is used when the terminal receives a HARDCOPY command or a 4010-HARDCOPY command, or when the HARD COPY key is pressed. This command chooses between the standard hardcopy interface and the Option 19 color hardcopy interface.

CAUTION

While the terminal is making a color hard copy, do not press the HARD COPY key on the monochrome copier. It will lock up the terminal.

DEFAULTS

interface
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

QD00 (Level 0): Unrecognized command. (Not a 4111 or Option 19 is not installed.)

QD11 (Level 2): Invalid *interface* parameter.

REFERENCES

HARDCOPY command
HARD COPY key
4010-HARDCOPY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SELECT-OBJECT-SURFACE-COLOR Command

4128, 4129

Host Syntax

$^{E}{}_{C}$TO int:*object-surface-color*

Setup Syntax

$^{E}{}_{C}$TO *object-surface-color*

PARAMETERS

object-surface-color (–4095 to 255)
Specifies the object-surface-color.

DESCRIPTION

This command sets the *object-surface-color* primitive attribute. This helps determine the Extended Range Color Index (ERCI) of facet vertices within segments when the view is renewed when the view's object-surface-display attribute is "shaded-surface." If the range for the object-surface-color has not been defined or has been deleted, and error is detected and the attribute is not changed.

The effect of the *object-surface-color* attribute depends on its value, the length of the *indices* and *normals* array parameters of each DRAW-FACETS command, and on whether the related object-surface-color-range is defined, as follows:

> When *indices* and *normals* have no elements and *object-surface-color* is –4095 to –1, all facet vertices have an ERCI of 4095 to 1.
>
> When *indices* and *normals* have no elements and *object-surface-color* is 1 to 255, and the object-surface-color range is defined, the Local Lighting Model is used to compute the ERCI of each triangle or quadrilateral facet vertex.
>
> When the *indices* array parameter has one or more elements (and *normals* has none), the elements of *indices* determine the color of the facets. (*Object-surface-color* is not used for those facets.)
>
> When *normals* has one or more elements (and *indices* has one), and when the *object-surface-color* is 1 to 255, and the indicated object-surface-color-range is defined, the Local Lighting Model is used to compute the ERCI of each facet vertex.
>
> In all other cases, ERCI = 0 for all vertices. (See the Local Lighting Model section of Appendix G for details.)

DEFAULTS

object-surface-color
as shipped — –16
on power-up — –16
if omitted — 0

ERRORS

TO00	(Level 0):	Unrecognized command. (Terminal is not a 4128 or 4129.)
TO10	(level 2):	Specified object-surface-color range definition does not exist (has not been defined or has been deleted).
TO11	(Level 2):	Invalid *object-surface-color.*

REFERENCES

Appendix G, Shaded Surfaces
DRAW-FACETS command
SET-OBJECT-SURFACE-COLOR-RANGE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SELECT-TRANSLUCENCY-PATTERN Command

4128, 4129

Host Syntax

EcMO int:*translucency-pattern*

Setup Syntax

EcMO Sp *translucency-pattern*

PARAMETERS

translucency-pattern (–16 to 16)
Specifies a translucency pattern.

DESCRIPTION

This command specifies the translucency pattern for the display of 3D "shaded-surface" and "hidden-line" facets. A translucency pattern determines which pixels are shaded and which are left unshaded, depending on whether the shaded image is "sectioned." (See the SET-SECTIONING-PLANES command for details on sectioning.)

For *translucency-pattern* 0, all pixels are always shaded.

For *translucency-pattern* 1 through 16 (see Figure 7-14 below), the unsectioned shaded image is displayed with the specified pattern. The sectioned image is not displayed (it is entirely transparent).

For *translucency-pattern* –1 through –16, the sectioned shaded image is displayed with patterns 1 through 16. The unsectioned image is displayed with pattern 0 (opaque).

3D Object Surface Translucency Patterns

The low two bits of a pixel's x and y coordinates determine whether the pixel is displayed or not in a particular translucency pattern. The dots indicate which pixels are displayed.

DEFAULTS

translucency-pattern
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MO00	(Level 2):	Unrecognized command. (The terminal is not a 4128 or 4129).
MO11	(Level 2):	Invalid *translucency-pattern*.

REFERENCES

DRAW-FACETS command
SET-SECTIONING-PLANES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

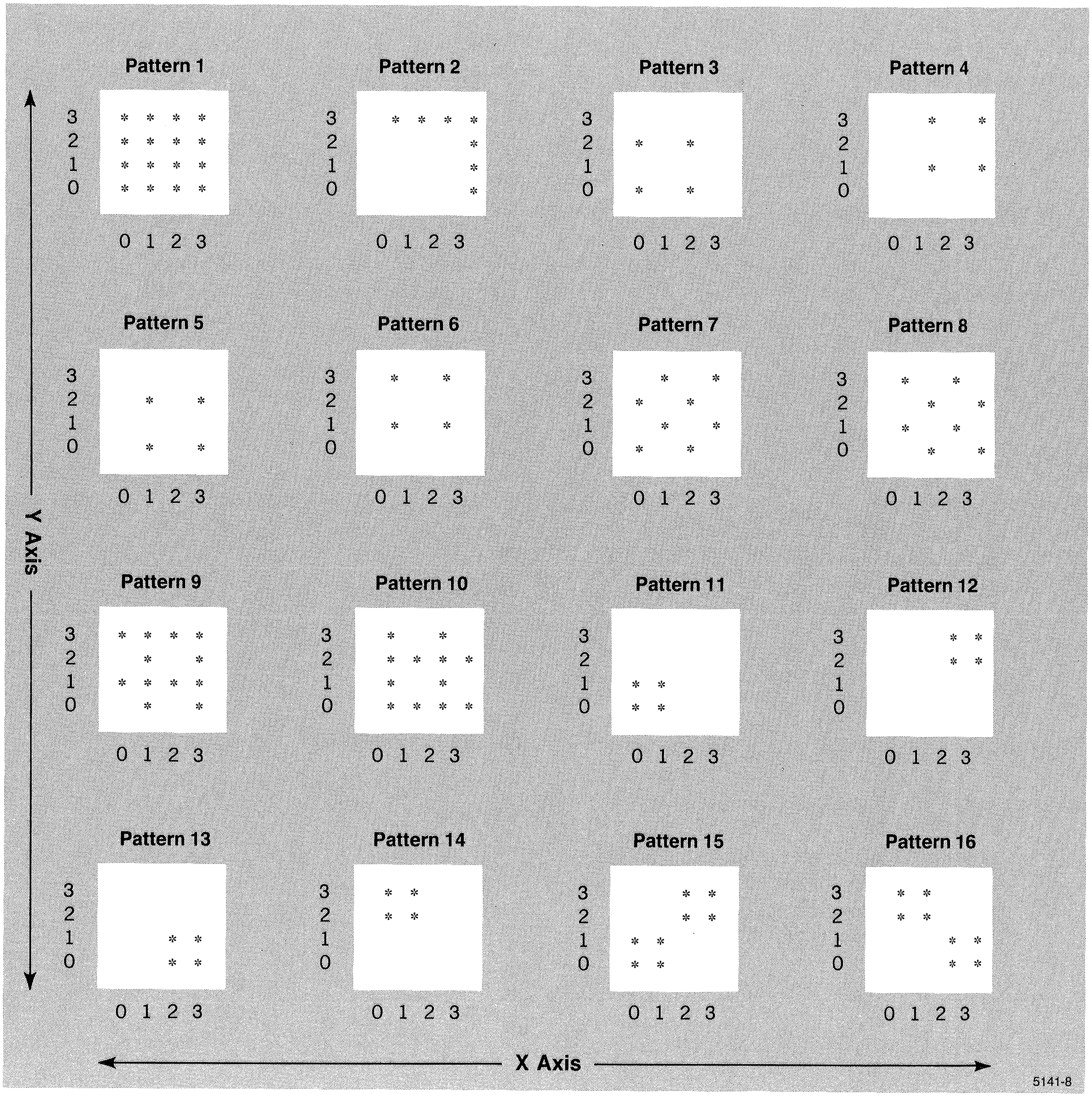


Figure 7-14. Translucency Patterns

SELECT-TRAVERSAL-DIRECTION Command

4128, 4129

Host Syntax

$^{E}{}_{C}$**MH** int:*traversal-direction*

Setup Syntax

$^{E}{}_{C}$**MH** $^{S}{}_{P}$ *traversal-direction*

PARAMETERS

traversal-direction (0 to 7)

0 right-hand rule; display back-facing object-surface; display degenerate triangles

1 right-hand rule; suppress back-facing object-surface; display degenerate triangles

2 left-hand rule; display back-facing object-surface; display degenerate triangles

3 left-hand rule; suppress back-facing object-surface; display degenerate triangles

4 right-hand rule; display back-facing object-surface; do not display degenerate triangles

5 right-hand rule; suppress back-facing object-surface; do not display degenerate triangles

6 left-hand rule; display back-facing object-surface; do not display degenerate triangles

7 left-hand rule; suppress back-facing object-surface; do not display degenerate triangles

DESCRIPTION

This command determines whether back-facing facets are suppressed when a view whose object-surface-display attribute is "shaded-surface" or "hidden-line" is renewed and determines the (right or left) handedness of the system used to determine which direction the facets face. This handedness is also used to determine the direction of normals of facets which have no indices or normals specified for them. (See the DRAW-FACETS command.)

Each triangle in a facet is suppressed or not depending on the "direction" in which the points are displayed on the screen. Under the right-handed rule, if points are displayed in a clockwise manner, the triangle is back-facing. Under the left-handed rule, counterclockwise is back-facing.

Triangles degenerate when the 3 points that are displayed on the screen are colinear.

DEFAULTS

traversal-direction
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MH00 (Level 2): Unrecognized command. (The terminal is not a 4128 or 4129.)

MH11 (Level 2): Invalid *traversal-direction*.

REFERENCES

DRAW-FACETS command
RENEW-VIEW command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SELECT-VIEW Command

Host Syntax

```
EcRC   int:view-number
```

Setup Syntax

```
EcRC Sp  view-number
```

PARAMETERS

view-number (–1 to 64)
Specifies the view to be selected.

–1	the next lower-numbered view which currently exists
0	the next higher-numbered view which currently exists
1 to **64**	a specific view

DESCRIPTION

This command selects a view to be the current view. If the specified view does not exist, it is created. On the 4128 and 4129, the light in the NEXTVIEW key turns on when the view selected (or created) has a pan-dimensionality attribute of "3D." (See the SET-VIEW-DISPLAY-ATTRIBUTES command for details.)

If *view-number* is 0, the next higher-numbered existing view is selected. If no higher-numbered view exists, then the lowest-numbered existing view is selected. This method of selecting the next view is equivalent to pressing the NEXTVIEW key, except that the view's border does not blink.

If *view-number* is –1, the next lower-numbered existing view is selected. If no lower-numbered view exists, then the highest-numbered existing view is selected. This method is equivalent to pressing CTRL-NEXTVIEW, except that the view's border does not blink.

A view is simply a set of viewing parameters. Up to 64 independent views can be defined, but only the one currently selected is used for the display of graphic primitives. Also, only the currently selected view can be added to the list of views in which a segment is visible. At the time a view is created, no segments are visible in it. See the SET-SEGMENT-VISIBILITY command for details.

When a new view is created it inherits the viewing parameters of the previously current view. The global (2D and 3D) viewing parameters and the commands that set them are:

2D and 3D Viewing Parameter	Command
Viewport	SET-VIEWPORT
Surface	SET-VIEW-ATTRIBUTES
Wipe Index	SET-VIEW-ATTRIBUTES
Border Index	SET-VIEW-ATTRIBUTES
Border Visibility	SET-BORDER-VISIBILITY command or the BORDER key

The 2D viewing parameters and the commands that set them are:

2D Viewing Parameter	Command
Window	SET-WINDOW
Full Overview Window	SET-OVERVIEW-WINDOW
Partial Overview Window	SET-OVERVIEW-WINDOW
Function Box Window	SET-OVERVIEW-WINDOW
Alphatext Window	SET-OVERVIEW-WINDOW

The 3D viewing parameters (4128 and 4129 only) and the commands that set them are:

3D Viewing Parameter	Command
View Reference Point	SET-VIEWING-TRANSFORM
View Plane Normal	SET-VIEWING-TRANSFORM
View Up	SET-VIEWING-TRANSFORM
UV Window Size	SET-VIEWING-TRANSFORM
Eye Position	SET-VIEWING-TRANSFORM
Projection Type	SET-VIEWING-TRANSFORM
View Motion Radius	SET-VIEWING-TRANSFORM
Back Plane	SET-VIEWING-TRANSFORM
Front Plane	SET-VIEWING-TRANSFORM
Pan Dimensionality	SET-VIEW-DISPLAY-ATTRIBUTES
Object Surface Display	SET-VIEW-DISPLAY-ATTRIBUTES
Color Smoothing	SET-VIEW-DISPLAY-ATTRIBUTES
Sectioning	SET-SECTIONING-PLANES
Plane Points	SET-SECTIONING-PLANES
Ambient Light Intensity	SET-LIGHTSOURCE

When a new view is created, 3D lightsources in the range 1 to 16 may have already been defined and turned on for that view by the SET-LIGHTSOURCE command.

DEFAULTS

view-number
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

RC02 (Level 3): Insufficient memory to create a view.

RC11 (Level 2): Invalid *view-number.*

REFERENCES

BORDER key
DELETE-VIEW command
NEXTVIEW key
SET-BORDER-VISIBILITY command
SET-LIGHTSOURCE command
SET-OVERVIEW-WINDOW command
SET-SECTIONING-PLANES command
SET-SEGMENT-VISIBILITY command
SET-VIEW-ATTRIBUTES command
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEWING-TRANSFORM command
SET-VIEWPORT command
SET-WINDOW command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-ALPHATEXT-FONT Command

Host

$${}^{E}{}_{C}\begin{Bmatrix}{}^{S}{}_{I}\\ {}^{S}{}_{O}\end{Bmatrix}$$

PARAMETERS

The ${}^{S}{}_{I}$ character selects the standard alphatext font. The ${}^{S}{}_{O}$ character selects the Katakana font, if the terminal is equipped with a Katakana keyboard.

DESCRIPTION

This command selects the font used for displaying alphatext on those terminals equipped with the Katakana keyboard, Option 4K.${}^{E}{}_{C}{}^{S}{}_{I}$ selects the standard ASCII font, while ${}^{E}{}_{C}{}^{S}{}_{O}$ selects the alternate font. The selected font is used for alphatext in the dialog area and on the screen, as well as for "string precision" graphtext. However, only the ASCII font is used in Setup mode.

REFERENCES

GRAPHIC-TEXT command
SET-GRAPHTEXT-PRECISION command

INQUIRY COMMAND

none

SET-ANSWERBACK-MESSAGE Command

4111

Setup Syntax

ANSWERBACK *answerback-message*

PARAMETERS

answerback-message
A character array of length 0 to 20. The characters must be ASCII characters in the range 0 to 127 (decimal). The characters must be delimited, at each end of the string, with identical ASCII characters not included in the message string.

DESCRIPTION

This command defines the answerback message that will be sent from the terminal to the host in response to receipt of an $^{E}{}_{Q}$ (Enquiry) character from the host or when the CTRL-BREAK character is pressed on the keyboard. Except for Local mode, the terminal will respond with the answerback message while in any of its operating modes (TEK, ANSI, Edit, or VT52), but this command is recognized only in Setup mode.

The answerback message is saved in non-volatile memory when the SAVE-NONVOLATILE-PARAMETERS command is issued and remembered when the terminal is powered-down.

DEFAULTS

answerback-message
as shipped — empty array
on power-up — remembered
if omitted — empty array

ERRORS

N011 (Level 2): Invalid answerback-message

REFERENCES

$^{E}{}_{Q}$ Character
BREAK key

SET-BACKGROUND-COLOR Command

Host Syntax

$^{E}{}_{C}$**TB** int:*first-color-coordinate*
int:*second-color-coordinate*
int:*third-color-coordinate*

Setup Syntax

TB $^{S}{}_{P}$ *first-color-coordinate*
second-color-coordinate
third-color-coordinate

PARAMETERS

The three color coordinates are either HLS, RGB, CMY, or Machine RGB coordinates, according to the *color-specifying-mode* in the most recent SET-COLOR-MODE command. A blinking color can be specified by adding 1000 to the value of the *third-color-coordinate* parameter.

The valid ranges for the three parameters are:

first-color-coordinate

HLS	–32768 to 32767
RGB	0 to 100
CMY	0 to 100
Machine RGB	0 to 255

second-color-coordinate

HLS	0 to 100
RGB	0 to 100
CMY	0 to 100
Machine RGB	0 to 255

third-color-coordinate

HLS	0 to 100 or 1000 to 1100
RGB	0 to 100 or 1000 to 1100
CMY	0 to 100 or 1000 to 1100
Machine RGB	0 to 255 or 1000 to 1255

DESCRIPTION

The SET-BACKGROUND-COLOR command sets the color of the background surface which is behind all the transparent writing surfaces.

Whenever the background color is set, the background gray level is set to an equivalent NTSC gray level. The conversion formula that relates gray level to color is:

Gray level = 30% (Red level) + 59% (Green level) +
11% (Blue level)

The resulting gray level is always rounded to the nearest integer.

Two other commands can be used to set the background color are SET-BACKGROUND-GRAY-LEVEL and SET-SURFACE-COLOR-MAP.

Adding 1000 to the value of the *third-color-coordinate* causes the color to blink by alternating between black and the specified color at a rate of 1½ cycles per second. For example, in HLS mode a normal red background is indicated by (120, 50, 100), and a blinking red background is given by (120, 50, 1100).

NOTE

If you specify a SUBTRACTIVE overlay mode in the SET-COLOR-MODE command, then you should also specify a background color of white (or some other light color) with the SET-BACKGROUND-COLOR or SET-BACKGROUND-GRAY-LEVEL command.

DEFAULTS

first-color-coordinate
as shipped — 0
on power-up — 0
if omitted — 0

second-color-coordinate
as shipped — 0
on power-up — 0
if omitted — 0

third-color-coordinate
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

TB11 (Level 2): Invalid first parameter. (If in HLS mode, must range from –32768 to 32767. If in RGB or CMY mode, must range from 0 to 100. If in Machine RGB mode, must range from 0 to 255.)

TB21 (Level 2): Invalid second parameter (HLS, RGB, CMY must range from 0 to 100, Machine RGB must range from 0 to 255).

TB31 (Level 2): Invalid third parameter (HLS, RGB, CMY must range from 0 to 100 and 1000 to 1100, Machine RGB must range from 0 to 255 and 1000 to 1255).

REFERENCES

SET-BACKGROUND-GRAY-LEVEL command
SET-COLOR-MODE command
SET-SURFACE-COLOR-MAP command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BACKGROUND-GRAY-LEVEL Command

Host Syntax

$^{E}{}_{C}$**RB** int:*gray-level*

Setup Syntax

$^{E}{}_{C}$**RB** $^{S}{}_{P}$ *gray-level*

PARAMETERS

gray-level (0 to 100 and 1000 to 1100 in color modes 1, 2, and 3; 0 to 255 and 1000 to 1255 in color mode 4)
The "percent" of lightness to which the background (behind all writing surfaces) is set. 0 represents black, while 100 represents white. Adding 1000 to a gray level blinks the background.

DESCRIPTION

This command determines the gray level used on the screen for background. It is the value of the screen when no object or opaque view objects are present.

Initial (minimum) value is "0% lightness," or black. Maximum value is "100% lightness," or white.

Please read the discussion of the gray levels and gray indices and their meanings and assignments, included in the description of the SET-SURFACE-GRAY-LEVELS command.

If you add 1000 to the gray level number, the background blinks by alternating between black and the specified *gray-level.*

Whenever the background gray level is set, the background color is set to an equivalent gray level.

Two other commands set the background gray level: SET-BACKGROUND-COLOR and SET-SURFACE-COLOR-MAP.

DEFAULTS

gray-level
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

RB11 (Level 2): Invalid *gray-level.*

REFERENCES

SET-BACKGROUND-COLOR command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-GRAY-LEVELS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BACKGROUND-INDICES Command

Host Syntax

```
EcMB  int:text-background-index
      int:dash-gap-index
```

Setup Syntax

```
EcMB Sp  text-background-index
         dash-gap-index
```

PARAMETERS

text-background-index (–2 to 32767)
Specifies the background index for string-precision graphtext and alphatext which is not displayed in the dialog area.

–2	the wipe index for the current viewport
–1	no index; the pixels in character backgrounds are left unchanged
0 to **32767**	a specific color index

dash-gap-index (–2 to 32767)
Determines the color index for the "gaps" in dashed lines.

–2	the wipe index for the current viewport
–1	no index; the pixels in the gaps in dashed lines are left unchanged
0 to **32767**	a specific color index

DESCRIPTION

The SET-BACKGROUND-INDICES command specifies the color indices used for the backgrounds of string-precision graphtext (and alphatext outside the dialog area). It also specifies the color index used for the "gaps" in dashed lines.

Index –2. In this command, an index of –2 represents the wipe index for the current viewport. Specifying index –2 for the text background index is like specifying "replace mode" in the SET-GRAPHICS-AREA-WRITING-MODE command.

Index −1. In the SET-BACKGROUND-INDICES command, an index of −1 means "no index." That is, it specifies that the character background (or dash gap) pixels are to be left unchanged. Specifying −1 for the character background index is like specifying "overstrike mode" in the SET-GRAPHICS-AREA-WRITING-MODE command.

NOTE

> *The SET-GRAPHICS-AREA-WRITING-MODE and SET-BACKGROUND-INDICES commands both affect how alphatext is displayed in the graphics area. Thus, each of these commands supersedes the effect of the other.*

DEFAULTS

text-background-index
as shipped — −2
on power-up — agrees with the remembered value for the SET-GRAPHICS-AREA-WRITING-MODE command (GAMODE command). If the GAMODE is REPLACE, the text-background index is −2; if GAMODE is OVERSTRIKE, the text text-background-index is −1.
if omitted — 0

dash-gap-index
as shipped — −1
on power-up — −1
if omitted — 0

ERRORS

MB11 (Level 2): Invalid *text-background-index.*

MB21 (Level 2): Invalid *dash-gap-index.*

REFERENCES

SET-GRAPHICS-AREA-WRITING-MODE command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-GRAY-LEVELS command
SET-VIEW-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BAUD-RATES Command

Host Syntax

$^{E}{}_{C}$**NR** int:*transmit-data-rate*
int:*receive-data-rate*

Setup Syntax

BAUDRATE $^{S}{}_{P}$ *transmit-data-rate*
receive-data-rate

PARAMETERS

transmit-data-rate
The rate, in bits per second, at which the terminal transmits characters to the host computer. Valid values are 1 (which means "external clock"), 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, and 38400.

receive-data-rate
The rate at which the terminal receives characters from the host computer. Valid values are 0 (which means "same as transmit-rate", 1 (which means "external clock"), 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, and 38400.

DESCRIPTION

Specifies the line data rates to be used for all subsequent communications. Split baud rates (different speeds for receiving and transmission) are allowed.

The receiving rate is the rate at which the terminal expects to receive data. The transmission rate is the rate at which individual characters are clocked out of the terminal.

Valid values of the data rate parameters are: 0, 1, 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600, 19200, and 38400 bits/second.

A receive baud rate of zero means that the terminal is to use the same receive speed as the transmission rate. A transmit rate of zero is invalid.

A receive or transmit rate of one means that the terminal is to use an external clock to determine its data rate. (The external clock would be provided on the RECEIVE CLOCK or TRANSMIT CLOCK input of the RS-232 connector.)

In addition, you can specify a "transmit data rate limit" with the SET-TRANSMIT-RATE-LIMIT command, and a transmission delay time with the SET-TRANSMIT-DELAY command. These commands control the effective maximum speed for the terminal-to-host communications, which may be less than the rate at which the terminal sends each individual character.

The baud rates may be set by the operator in Setup mode, or they may be included as a SET-BAUD-RATES command in a file to be loaded from the terminal's disk drive. It is usually unwise to attempt to change the terminal's baud rates by command from the host computer.

NOTE

The 4125, 4128, and 4129 can display simple alphanumerics and graphics only up to a maximum continuous data rate of 9600 bits/second. (This does not include commands which require more than routine processing, such as the INCLUDE-COPY-OF-SEGMENT or LOAD commands.) At higher data rates, some "handshaking" protocol must be used to prevent the terminal's communications input queue from overflowing.

Moreover, even at slow data rates, it is prudent to use a handshaking protocol. The terminal can take an appreciable amount of time to execute some commands which can be issued using only a very few characters. If a handshaking protocol is not used, the terminal's input queue may overflow while executing such commands.

Such a handshaking protocol might be as simple as issuing a REPORT-4010-STATUS command from time to time, and waiting to receive the reply before issuing more commands to the terminal. Alternatively, any of several data communications protocols may be used: Flagging mode, Prompt mode, or Block mode. Any of these communications modes will prevent the input queue from overflowing.

DEFAULTS

transmit-data-rate
as shipped — 2400
on power-up — remembered
if omitted — error NP11

receive-data-rate
as shipped — 2400
on power-up — remembered
if omitted — 0

ERRORS

NR11	(Level 2):	Invalid transmit (terminal-to-host) data rate.
NR21	(Level 2):	Invalid receive (host-to-terminal) data rate.

REFERENCES

SET-TRANSMIT-RATE-LIMIT command
SET-TRANSMIT-DELAY command
SET-QUEUE-SIZE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-CONTINUE-CHARS Command

4111 or OPTION 01

Host Syntax

${}^{E}{}_{C}$OC int:*transmit-continue-char*
int:*receive-continue-char*

Setup Syntax

BCONTINUECHARS ${}^{S}{}_{P}$ *transmit-continue-char*
receive-continue-char

PARAMETERS

transmit-continue-char (0 to 127)
The numeric equivalent of the continue-char for blocks transmitted from the terminal to the host.

receive-continue-char (0 to 127)
The numeric equivalent of the continue-char for blocks received by the terminal from the host.

DESCRIPTION

Sets the *block-continue-chars* which signal the end of lines within a block (other than the last line, which uses the block-end-char).

This command is invalid if the terminal is armed for block mode.

The *block-continue-char* signals the end of a line in a block, and indicates that there are more lines of the block to come.

DEFAULTS

transmit-continue-char
as shipped — 38
on power-up — remembered
if omitted — 0

receive-continue-char
as shipped — 38
on power-up — remembered
if omitted — 0

ERRORS

OC00	(Level 2):	Unrecognized command. (Not a 4111 or Option 1 is not installed.)
OC03	(Level 2):	Command is invalid at this time. (Terminal must not be armed for block mode.)
OC11	(Level 2):	Invalid *transmit-continue-char.*
OC13	(Level 2):	*Transmit-continue-char* must be different from *block-master-char* and *block-end-char.*
OC21	(Level 2):	Invalid *receive-continue-char.*
OC23	(Level 2):	*Receive-continue-char* must be different from *block-master-char* and *block-end-char.*

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-END-CHARS Command

4111 or OPTION 01

Host Syntax

$^{E}{}_{C}$OE int:*transmit-end-char*
int:*receive-end-char*

Setup Syntax

BENDCHARS $^{S}{}_{P}$ *transmit-end-char*
receive-end-char

PARAMETERS

transmit-end-char (0 to 127)
Numeric equivalent of the *block-end-char* for blocks sent from the terminal to the host.

receive-end-char (0 to 127)
Numeric equivalent of the *block-end-char* for blocks received by the terminal from the host.

DESCRIPTION

This command, for use with Block mode, sets the *block-end-chars* for blocks sent to and from the terminal.

This command is invalid if the terminal is armed for Block mode.

The *block-end-char* signals that there are no more lines in the block.

DEFAULTS

transmit-end-char
as shipped — 36
on power-up — remembered
if omitted — 0

receive-end-char
as shipped — 36
on power-up — remembered
if omitted — 0

ERRORS

OE00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OE03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OE11 (Level 2): Invalid *transmit-end-char*.

OE13 (Level 2): *Transmit-end-char* must be different from *block-master-char* and *block-continue-char*.

OE21 (Level 2): Invalid *receive-end-char*.

OE23 (Level 2): *Receive-end-char* must be different from *block-master-char* and *block-continue-char*.

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-HEADERS Command

4111 or OPTION 01

Host Syntax

```
EcOH  int-array:transmit-header
      int-array:receive-header
```

Setup Syntax

```
BHEADERS Sp  transmit-header  receive-header
```

PARAMETERS

transmit-header
Numeric equivalents (range 0 to 127) of one to ten characters. These characters comprise the *block-header* for blocks sent from the terminal to the host.

receive-header
Similar to the *transmit-header*, but for blocks received from the host.

DESCRIPTION

Sets the transmit and receive header sequences for block mode. (Here, *transmit-header* means the header sequence for blocks sent from the terminal to the host computer; *receive-header* means the header for blocks sent from the host to the terminal.) Each header is specified as an int-array, where the ints in the array are the numeric equivalents of the ASCII characters in the header. There may be one to ten characters in each header.

This command is invalid if the terminal is armed for block mode.

NOTE

You should use different strings for the transmit and receive block headers. Otherwise, echoes from the host of blocks which the terminal transmits would be interpreted by the terminal as blocks coming from the host program.

DEFAULTS

transmit-header
as shipped — **6HEADTX**
on power-up — remembered
if omitted — error OH11

receive-header
as shipped — **6HEADRX**
on power-up — remembered
if omitted — error OH21

ERRORS

OH00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OH02 (Level 3): Out of memory while performing command.

OH03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OH11 (Level 2): Invalid *char* (must be 0 to 127) or array count (must be in range 1 to 10) in *transmit-header*.

OH12 (Level 3): Out of memory while parsing the parameter.

OH21 (Level 2): Invalid *char* (must be 0 to 127) or array count (must be in range 1 to 10) in *receive-header*.

OH22 (Level 3): Out of memory while parsing the parameter.

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-LENGTH Command

4111 or OPTION 01

Host Syntax

$^{E}{}_{C}$**OS** int:*transmit-block-length*
int:*receive-block-length*

Setup Syntax

BLENGTH $^{S}{}_{P}$ *transmit-block-length*
receive-block-length

PARAMETERS

transmit-block-length (5 to 65535)
Maximum length in bytes of unpacked data in blocks which the terminal sends to the host.

receive-block-length (5 to 65535)
Maximum length in bytes of unpacked data in blocks received from the host.

DESCRIPTION

Sets the block lengths for transmitted and received blocks in block mode. (Here, "transmitted block" means a block which the terminal sends to the host; "received block" means a block which the host sends to the terminal.)

The specified block length is the number of data bytes, including the four control bytes, before packing. The actual number of characters transmitted over the line will be greater because of the packing and formatting overhead. (For more information on the packing of data, see the description of the SET-BLOCK-PACKING command.)

This command is invalid if the terminal is already in block mode, or if it is already armed for block mode.

DEFAULTS

transmit-block-length
as shipped — 256
on power-up — remembered
if omitted — error OS11

receive-block-length
as shipped — 256
on power-up — remembered
if omitted — error OS21

ERRORS

OS00 (Level 2): Unrecognized command. (Not a 4111 or 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 *transmit-block-length.*

OS21 (Level 2): Invalid *receive-block-length.*

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-LINE-LENGTH Command

4111 or OPTION 01

Host Syntax

```
EcOL  int:maximum-line-length
```

Setup Syntax

```
BLINELENGTH Sp  maximum-line-length
```

PARAMETERS

maximum-line-length (12 to 65535)
The maximum number of characters in each line of a block which the terminal sends to the host.

DESCRIPTION

The SET-BLOCK-LINE-LENGTH command sets the maximum number of characters in each "line" of a block which the terminal sends to the host in block mode.

When comparing the length of a line with this maximum line length, the "line length" includes the header characters, the characters of packed data, and the *block-continue-char* or *block-end-char*. Not included are the characters in the *EOL-string*. (See block for details.)

You should choose a maximum line length which does not exceed the capacity of the host computer's input buffer. That is, the *maximum-line-length* in the SET-BLOCK-LINE-LENGTH command, plus the characters in the *EOL-string*, should not exceed the host's input buffer size.

When the terminal sends characters in block mode, it counts the characters of packed data on each line as it sends them. When the character count reaches one less than the current *maximum-line-length* setting, the terminal ends the line by sending the *block-continue-char* and the current *EOL-string*.

This command is invalid if the terminal is already in block mode, or if it is already armed for block mode.

DEFAULTS

maximum-line-length
as shipped — 70
on power-up — remembered
if omitted — error OL11

ERRORS

OL00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 not installed.)

OL03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OL11 (Level 2): Invalid *maximum-line-length.*

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-MASTER-CHARS Command

4111 or OPTION 01

Host Syntax

$^{E}{}_{C}$OM int:*transmit-master-char*
int:*receive-master-char*

Setup Syntax

BMASTERCHARS $^{S}{}_{P}$ *transmit-master-char*
receive-master-char

PARAMETERS

transmit-master-char (0 to 127)
The numeric equivalent of the *block-master-char* for blocks sent from the terminal to the host.

receive-master-char (0 to 127)
The numeric equivalent of the *block-master-char* for blocks received from the host.

DESCRIPTION

Sets the master characters for block mode transmission and reception.

This command is used in conjunction with the SET-BLOCK-NON-XMT-CHARS command. When the host or terminal would otherwise have occasion to send one of the non-transmittable characters, it sends instead the "master character," followed by another character. This two-character combination substitutes for the non-transmittable character.

This command is invalid if the terminal is armed for block mode.

For the terminal to properly recognize the *block-master-chars* they must be different than the *block-continue-char* and the *block-end-char.*

DEFAULTS

transmit-master-char
as shipped — 35
on power-up — remembered
if omitted — 0

receive-master-char
as shipped — 35
on power-up — remembered
if omitted — 0

ERRORS

OM00	(Level 2):	Unrecognized command. (Not a 4111 or Option 01 is not installed.)
OM03	(Level 2):	Command invalid at this time. (Terminal must not be armed for block mode.)
OM11	(Level 2):	Invalid *transmit-master-char.*
OM13	(Level 2):	*Transmit-master-char* must be different from *block-end-char* and *block-continue-char.*
OM21	(Level 2):	Invalid *receive-master-char.*
OM23	(Level 2):	*Receive-master-char* must be different from *block-end-char* and *block-continue-char.*

REFERENCES

SET-BLOCK-NON-XMT-CHARS command
SET-BLOCK-PACKING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-NON-XMT-CHARS Command

4111 or OPTION 01

Host Syntax

$^{E}_{C}$ON int-array:*transmit-chars*
int-array:*receive-chars*

Setup Syntax

BNONXMTCHARS $^{S}_{P}$ *transmit-chars*
receive-chars

PARAMETERS

transmit-chars
An int-array in which each int represents an ASCII character which may not appear in the packed-data of a block sent from the terminal to the host. Each int in the array must be in the range from 0 to 127.

receive-chars
Similar to the first parameter, but for blocks received by the terminal from the host.

DESCRIPTION

Sets the non-transmittable characters for the terminal to transmit and receive (to and from the host) while in block mode. Whenever the terminal or host computer would otherwise send one of these characters within the characters of packed data, it (the terminal or host) substitutes a two-character sequence : the "master character," followed by a character which substitutes for the non-transmittable character. The substitution characters are assigned as follows: for the first non-transmittable character, the letter (A); for the second non-transmittable character, the letter (B); etc.

The master character, block-continue character, and block-end character must not be allowed to occur within the characters of packed data. If the packing scheme (chosen with the set-block-packing command) permits this to occur, then those characters must be designated as "non-transmittable" characters.

This command is invalid if the terminal is already in block mode, or if it is armed for block mode.

DEFAULTS

transmit-chars
as shipped — 35, 36, 38
on power-up — remembered
if omitted — empty array

receive-chars
as shipped — 35, 36, 38
on power-up — remembered
if omitted — empty array

ERRORS

ON00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

ON03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

ON11 (Level 2): Invalid character code or array count in *transmit-chars* array. (The array count must range from 0 to 20, and the character codes must range from 0 to 127.)

ON12 (Level 3): Out of memory while parsing the parameter.

ON21 (Level 2): Invalid character code or array count in *receive-chars* array. (The array count must range from 0 to 20, and the character code must range from 0 to 127.)

ON22 (Level 3): Out of memory while parsing the parameter.

REFERENCES

SET-BLOCK-MASTER-CHARS command
SET-BLOCK-PACKING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-PACKING Command

4111 or OPTION 01

Host Syntax

$^{E}{}_{C}$OP int:*transmit-unpacked-bits*
int:*transmit-packed-bits*
int:*receive-unpacked-bits*
int:*receive-packed-bits*

Setup Syntax

BPACKING $^{S}{}_{P}$ *transmit-unpacked-bits*
transmit-packed-bits
receive-unpacked-bits
receive-packed-bits

PARAMETERS

transmit-unpacked-bits (7 or 8)
The number of bits per byte of unpacked-data in blocks sent from the terminal to the host.

transmit-packed-bits (6, 7, or 8)
The number bits per "pseudo-byte" in the packed-data of blocks sent from the terminal to the host.

receive-unpacked-bits (7 or 8)
Like *transmit-unpacked-bits*, but for blocks received by the terminal from the host.

receive-packed-bits (6, 7, or 8)
Like *transmit-packed-bits*, but for blocks received from the host.

DESCRIPTION

This command determines how characters to be sent in block mode are packed before being included among the "packed data" in a block.

The command is invalid if the terminal is already in block mode, or if it is armed for block mode.

The command has four int parameters; the first two govern block mode transmissions from the terminal to the host computer, while the second two govern transmissions from the host to the terminal.

The purpose of packing is to convert the full seven-bit ASCII character set (or a set of eight-bit full binary bytes) into a reduced character set for transmission over a host/communications system having limited transmission capability.

Converting to a Stream of Binary Bits. The data to be transmitted is considered to be a long string of 7-bit or 8-bit bytes layed "end to end," forming one long string of binary bits. The first bit is the high-order bit of the first byte; the last bit is the low-order bit of the last byte.

Composing Pseudo-Bytes and then the Actual Characters To Be Transmitted. Next the stream is divided into a series of "pseudo-bytes" of 6, 7, or 8 bits each. An offset is added to each pseudo-byte, thereby converting it into a standard ASCII character; Table 7-19 shows the offset which is added for each allowable pseudo-byte size.

Table 7-19

PACKED PSEUDO-BYTE CHARACTERISTICS

Number of meaningful data bits per pseudo-byte	Offset added to make a standard ASCII character	Range of possible ASCII decimal equivalents for the characters transmitted
6	32	32 to 95 ASCII characters from $^{S}{}_{P}$ to _
7	0	0 to 127 Full ASCII character set
8	0	0 to 255 Full eight-bit data bytes

When the end of a block is processed, if there are not enough bits to fill out the last pseudo-byte, an appropriate number of zeroes are appended to the end of the stream of bits. On input, this padding is ignored. Note that padding is inserted only at the end of a block and not at the end of a line within a block.

DEFAULTS

transmit-unpacked-bits
as shipped — 7
on power-up — remembered
if omitted — error OP11

transmit-packed-bits
as shipped — 6
on power-up — remembered
if omitted — error OP21

receive-unpacked-bits
as shipped — 7
on power-up — remembered
if omitted — error OP31

receive-packed-bits
as shipped — 6
on power-up — remembered
if omitted — error OP41

ERRORS

OP00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OP03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OP11 (Level 2): Invalid *transmit-unpacked-bits.*

OP21 (Level 2): Invalid *transmit-packed-bits.*

OP31 (Level 2): Invalid *receive-unpacked-bits.*

OP41 (Level 2): Invalid *receive-packed-bits.*

REFERENCES

Block syntactic construct
Block-control-bytes syntactic construct
SET-BLOCK-CONTINUE-CHARS command
SET-BLOCK-END-CHARS command
SET-BLOCK-HEADERS command
SET-BLOCK-MASTER-CHARS command
SET-BLOCK-NON-XMT-CHARS command
SET-EOM-CHARS command
SET-EOL-STRING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BLOCK-TIMEOUT Command

4111 or OPTION 01

Host Syntax

$^{E}{}_{C}$**OT** int:*number-of-seconds*

Setup Syntax

BTIMEOUT $^{S}{}_{P}$ *number-of-seconds*

PARAMETERS

number-of-seconds (0 to 65535)
Duration of timeout period in seconds. 0 disables the timeout feature.

DESCRIPTION

In block mode, when the terminal sends a block to the host, it expects an "ACK" block to come back from the host. If the terminal does not receive that ACK within a certain period of time, it re-transmits the block. The SET-BLOCK-TIMEOUT command determines how long the terminal waits before retransmitted the block.

If the int parameter in this command is 0, then the "retransmit on timeout" feature is disabled.

The timeout parameter should be set to a value which is longer than the maximum expected host response time.

DEFAULTS

number-of-seconds
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

OT00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OT11 (Level 2): Invalid *number-of-seconds.*

REFERENCES

Block syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BORDER-VISIBILITY Command

Host Syntax

EcRE int:*border-visibility-mode*

Setup Syntax

EcRE Sp *border-visibility-mode*

PARAMETERS

border-visibility-mode (0, 1, or 2)
Specifies whether the border of the current view is visible or invisible.

0 invisible

1 visible

2 toggles the border visibility: if visible, it becomes invisible; if invisible, it becomes visible.

DESCRIPTION

The command controls the visibility of a border drawn around the current view's viewport.

If the parameter is 1, a border is drawn around the current view's viewport. The border is drawn as a solid line, just within the viewport. It is drawn in the color index specified by the SET-VIEW-ATTRIBUTES command for that view.

The border is always drawn in Set mode (ALU mode 11) so that it overwrites the pixels below it. The border is removed by writing over the border, again in Set mode, with the background wipe index for the viewport. Thus turning the border on and off will erase any pixels on the border of the viewport. (For a description of ALU mode 11, see the BEGIN-PIXEL-OPERATIONS command. For a description of the background wipe index, see SET-VIEW-ATTRIBUTES.)

If the parameter is 0, the border around the current viewport is made invisible.

If the parameter is 2, the border is toggled. That is to say, if the border is visible, it is made invisible; and if it is invisible, it is made visible.

DEFAULTS

border-visibility-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

RE11 (Level 2): Invalid *border-visibility-mode* parameter.

REFERENCES

BEGIN-PIXEL-OPERATIONS command
BORDER key
SET-VIEW-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BREAK-TIME Command

Host Syntax

$^{E}_{C}$**NK** int:*break-time-in-milliseconds*

Setup Syntax

BREAKTIME $^{S}_{P}$ *break-time-in-milliseconds*

PARAMETERS

break-time-in-milliseconds (0 to 65535)
Approximate duration of a "break" signal. Zero causes no break signal to be sent.

DESCRIPTION

This command sets the length of the BREAK function in milliseconds. The actual break time is as close as possible (that is, within 25 ms) to the value specified, determined by the terminal clock resolution. The default break time delay is 200 ms.

When the BREAK Key is pressed, the terminal sends a "space" (as opposed to a "mark," in telegraph terminology) to the host computer. The "space" (a positive voltage on the RS-232 connector's TDATA line) lasts for a sufficient time that the host computer (or the data communications equipment) will recognize that the terminal is not sending a valid ASCII character.

(In Half Duplex Supervisor mode, the BREAK Key causes the modem to stop sending its secondary carrier for the designated length of time.)

On almost all systems, a break time of 200 ms will work well. Use the SET-BREAK-TIME command to change the default setting only if the 200 ms break time does not work well on your system.

The terminal remembers its break time setting even when turned off; thus, the SET-BREAK-TIME command need only be given when the terminal is installed. (In the vast majority of cases, the command need not even be given then, as the default 200 ms break time will suffice.)

Setting a break time of zero effectively disables the BREAK key; this may be useful for host systems which do not tolerate breaks.

DEFAULTS

break-time-in-milliseconds
as shipped — 200
on power-up — remembered
if omitted — 0

ERRORS

NK11 (Level 2): Invalid *break-time-in-milliseconds.*

REFERENCES

BREAK key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-BYPASS-CANCEL-CHAR Command

Host Syntax

${}^{E}{}_{C}$NU int:*bypass-cancel-char*

Setup Syntax

BYPASSCANCEL ${}^{S}{}_{P}$ *bypass-cancel-char*

PARAMETERS

bypass-cancel-char (0 to 127)
ASCII decimal equivalent of the bypass cancel character.

DESCRIPTION

The SET-BYPASS-CANCEL-CHAR command defines which ASCII character is to serve as the *bypass-cancel-character* for removing the terminal from bypass mode.

The *bypass-cancel-character* is the character that removes the terminal from Bypass mode when it is received by the terminal.

If the character is set to ${}^{N}{}_{U}$ — ASCII decimal equivalent (ADE) of 0 — then the Bypass mode is disabled and cannot be entered.

If your host does not echo any characters sent to it, set the *bypass-cancel-character* to ${}^{N}{}_{U}$ (ADE 0). If your host echoes any characters, set the *bypass-cancel-character* to the last character which the host echoes upon receiving an *EOL-string*.

For instance, suppose that the current *EOL-string* consists of the single character, ${}^{C}{}_{R}$, and that the host computer echoes each ${}^{C}{}_{R}$ as ${}^{C}{}_{R}{}^{L}{}_{F}$. Then, as the terminal sends reports and files to the host, the last character in each line sent to the host is ${}^{C}{}_{R}$, and the last character in the echo of each such line is ${}^{L}{}_{F}$. The ${}^{L}{}_{F}$ character, then, should be selected as the bypass-cancel-char. Since the ASCII decimal equivalent of ${}^{L}{}_{F}$ is 10, this can be done with a SET-BYPASS-CANCEL-CHAR : 10 command:

```
set-bypass-cancel-char : 10
= EcNu int : 10
= EcNU:
```

NOTE

Unlike earlier Tektronix terminals, there can be only one bypass-cancel-character *at a time on the 4110/4120-series terminals. Also, unlike the 4014, the* bypass-cancel-char *is not executed upon receipt.*

For more information on bypass mode, see the ENTER-BYPASS-MODE command.

DEFAULTS

bypass-cancel-char
as shipped — 10
on power-up — remembered
if omitted — 0

ERRORS

NU11	(Level 2):	Invalid numeric equivalent of bypass-cancel character.

REFERENCES

ENTER-BYPASS-MODE command
SET-EOL-STRING command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-COLOR-COPIER-DATA-RESOLUTION Command

4111 or OPTION 19

Host Syntax

```
ECQB   int:number-of-bytes
```

Setup Syntax

```
HCDATARES SP   number-of-bytes
```

PARAMETERS

number-of-bytes (1 or 2)
Specifies how many bytes of color resolution the data sent to a color copier will have.

1	one byte color resolution
2	two bytes color resolution

DESCRIPTION

This command sets the precision of the color resolution of the data sent from the terminal to the copier for each of three colors: red, green, and blue (RGB). One byte of color resolution contains two bits of information for each of the three colors. Two bytes of color precision contain four bits of information for each of the three colors.

4111

A SET-COLOR-COPIER-DATA-RESOLUTION setting of one byte causes the 4692 to use 64 colors for shading. A setting of two bytes causes the 4692 to use 216 colors for shading.

The action of the SET-COLOR-COPIER-DATA-RESOLUTION command on the 4691 and 4695 Copiers depends on the setting of the SET-COLORHARDCOPY-DITHERING command. With dithering enabled, a setting of one byte causes the 4691 or 4695 Copier to use 64 colors for shading. One byte of color resolution results in faster data transfers when the information is transferred using the COPY command with a source string parameter of SC:, or when a hardcopy is performed (with a HARDCOPY command, the HARD COPY key, or when the destination device is HC: for either the COPY or SPOOL commands).

With dithering enabled, a setting of two bytes causes the 4691 or 4695 Copier to use 256 colors for shading. Two bytes of color resolution result in the color being more precisely copied from the actual terminal color when a hardcopy is performed.

4115/4120 Series

The action of the SET-COLOR-COPIER-DATA-RESOLUTION command depends on the setting of the SET-COLORHARDCOPY-DITHERING command, and on the source device SC: parameter for the COPY command.

If dithering is not enabled, SET-COLOR-COPIER-DATA-RESOLUTION designates the color resolution only for the 4692 Color Copier. In this case, the 4692 uses its built-in dithering. A SET-COLOR-COPIER-DATA RESOLUTION setting of one byte causes the 4692 to use 64 colors to shade the image; a setting of two bytes causes the 4692 to use 4096 colors to shade the image. (Without dithering enabled, the 4691 and 4695 Copiers are restricted to eight colors.)

When dithering is enabled, SET-COLOR-COPIER-DATA-RESOLUTION designates the color resolution only if the image is transferred using the COPY command with SC: or SC:0 as the source device.

When dithering is enabled and the image is transferred using the COPY command with a source string parameter of SC:0 or SC:, a SET-COLOR-COPIER-DATA-RESOLUTION setting of one byte causes the copiers to use 64 colors to shade the image. The transfer is faster with one byte of color resolution. One byte of color resolution also saves disk storage space when storing data formatted by SC:0 or SC:.

When dithering is enabled and the image is transferred using the COPY command with a source string parameter of SC:0 or SC:, a SET-COLOR-COPIER-DATA-RESOLUTION setting of two bytes causes the copiers to use 4096 colors to shade the image.

DEFAULTS

number-of-bytes
as shipped — 1
on power-up — remembered
if omitted — error QB11

ERRORS

QB00	(Level 0):	Unrecognized command. (Not a 4111 or Option 19 is not installed.)
QB11	(Level 2):	Invalid *number-of-bytes* (must be 1 or 2).

REFERENCES

COPY command
HARDCOPY command
HARD COPY key
SET-HARDCOPY-INTERFACE command
SPOOL command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-COLORHARDCOPY-DITHERING Command

4111 or OPTION 19

Host Syntax

$^{E}_{C}$**QM** int:*dither-code*

Setup Syntax

HCDITHER $^{S}_{P}$ *dither-code*

PARAMETERS

dither-code (4111: 0 or 1; 4115/4120 Series: 0, 1, or 2)

Setup mode parameters are **YES**, **NO**, and **MATCH**.

0	**NO**	disabled.
1	**YES**	dithering only enabled.
2	**MATCH**	color matching and dithering enabled (not available in 4111).

DESCRIPTION

4111

This command selects dithering for copies made on the 4691 and 4695 color copiers.

With dithering enabled, a color data resolution of one byte per pixel emulates up to 64 colors and a color resolution of two bytes per pixel emulates up to 256 colors. Color data resolution is set by the SET-COLOR-COPIER-DATA-RESOLUTION command. With dithering disabled, the 4691 and 4695 are limited to eight colors.

This command has no effect on copies made on the 4692 color copier.

4115/4120 Series

This command selects dithering or color matching for copies made on the color copiers.

With dithering enabled, the color copier can emulate approximately 16 million colors. If the image is sent to the copier using the COPY command with the SC: or SC:0 parameter, the number of colors are restricted, depending on the number of color data resolution bytes. Color data resolution is set by the SET-COLOR-COPIER-DATA-RESOLUTION command.

With color matching enabled, a conversion algorithm more closely matches the 4692 Copier inks to the terminal colors. Dithering is performed automatically. The COPY command with the SC: or SC:0 parameter cannot be used with color matching.

With dithering disabled, the 4692 Copier uses its own dithering, and the SET-COLOR-COPIER-DATA-RESOLUTION command designates the color resolution. With dithering disabled, the 4691 and 4695 are limited to eight colors.

DEFAULTS

dither-code
as shipped — 0
on power-up — remembered
if omitted — 1

ERRORS

M00 (Level 0): Unrecognized command. (Must be a 4111 or Option 19, version 5 or later, must be installed.)

QM11 (Level 2): Invalid *dither-code.*

REFERENCES

COPY command
SET-COLOR-COPIER-DATA-RESOLUTION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-COLORHARDCOPY-FORM-WIDTH Command

4111 or OPTION 19

Host Syntax

```
EcQF   int:form-width
```

Setup Syntax

```
HCFORMWIDTH Sp   form-width
```

PARAMETERS

form-width (0 and 1)

0	Color hard copy is 960 dots wide. 8 1/2 inch wide media
1	Color hard copy is 1024 dots wide. 10 inch wide media

DESCRIPTION

This command sets the maximum width of the image on a 4695 color copier.

If the media in the 4695 is narrower than the image, ink may be squirted onto the copier's platen. Since the pixel area of 4125, 4128 and 4129 terminals is 1024 high, the form width must be set to 1 for a full-screen copy.

DEFAULTS

form-width
as shipped — 1
on power-up — remembered
if omitted — 1

ERRORS

QF00	(Level 0):	Unrecognized command. (Must be a 4111 or Option 19, version 5 or later, must be installed.)
QF11	(Level 2):	Invalid *form-width.*

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-COLOR-MODE Command

Host Syntax

```
EcTM   int:color-specifying-mode
       int:color-overlay-mode   int:gray-mode
```

Setup Syntax

```
CMODE Sp   color-specifying-mode
           color-overlay-mode   gray-mode
```

PARAMETERS

color-specifying-mode (0 to 4)
Determines which system of color coordinates is used to specify color mixtures in subsequent color operations.

0	no change
1	RGB (red, green, blue)
2	CMY (cyan, magenta, yellow)
3	HLS (hue, lightness, saturation; default)
4	Machine RGB

color-overlay-mode (0 to 3)
Specifies which mode is used when colors are placed on top of each other.

0	no change
1	OPAQUE
2	SUBTRACTIVE
3	ADDITIVE

gray-mode (0 to 2)
Specifies whether operation is color or black and white.

0	no change
1	COL; normal color operation
2	BW; displays color images in black and white

DESCRIPTION

This command sets three "color mode" parameters.

Color-Specifying-Mode. The *color-specifying-mode* parameter determines which of the three systems of color coordinates is used for specifying color mixtures in subsequent SET-SURFACE-COLOR-MAP and SET-BACKGROUND-COLOR commands. If this parameter is 0, or is omitted, the color specifying mode is left unchanged.

The default on power-up is "HLS." In the HLS system, colors are specified by hue, lightness, and saturation coordinates on the color cone. See Appendix E for details about the HLS color cone.

The RGB system determines a color mixture by adding together different proportions of the additive primary colors: red, green, and blue light sources. The RGB system closely resembles the actual operation of the terminal's color display hardware.

The CMY system mixes different proportions of the subtractive primary colors: cyan, magenta, and yellow inks. The terminal's CMY mode emulates the mixing of the cyan, magenta, and yellow pigments used in many printing processes.

Machine RGB mode increases the resolution in red, green, and blue from that available with the regular RGB mode (0 to 100) to the number of bits in each color that are available at the output of the color map. The valid range is 0 to 255.

Color-Overlay-Mode. The *color-overlay-mode* parameter specifies the behavior of the terminal's writing surfaces. If this parameter is zero, the color overlay mode is left unchanged.

In OPAQUE mode (mode 1), pictures drawn on a surface are deemed to be opaque; they obscure pictures drawn on surfaces behind them. When the terminal is turned on, it is in OPAQUE mode.

In SUBTRACTIVE mode (mode 2), pictures are drawn using transparent inks. The terminal behaves like a "light table," in which transparent overlays are placed on top of a diffusing light source.

NOTE

> *If you specify the SUBTRACTIVE* color-overlay-mode *in the SET-COLOR-MODE command, then you should also specify a background color of white (or some other light color) with the SET-BACKGROUND-COLOR or SET-BACKGROUND-GRAY-LEVEL command.*

In ADDITIVE mode (mode 3), the images drawn on different surfaces act as if their colored inks were comprised of many small point light sources. Where colors on one surface overlap with colors on another surface, the light from the two surface's light sources combine. For instance, a red object on one surface and a green object on another surface would combine to produce a yellow color where the two objects overlap.

Gray-Mode. The *gray-mode* parameter determines whether colors are displayed in color or in black and white. If this parameter is 0, or is omitted, the gray mode is left unchanged. COL mode (mode 1). BW mode (mode 2) causes colors to appear as shades of gray, according to the NTSC transform:

Gray level = 30% (Red level) + 59% (Green level) +
11% (Blue level)

DEFAULTS

color-specifying-mode
as shipped — 3
on power-up — 3
if omitted — 0

color-overlay-mode
as shipped — 1
on power-up — 1
if omitted — 0

gray-mode
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

TM11 (Level 2): Invalid *color-specifying-mode*

TM21 (Level 2): Invalid *color-overlay-mode.*

TM31 (Level 2): Invalid *gray-mode.*

REFERENCES

Appendix E, "Color Coordinate Systems"
SET-BACKGROUND-COLOR command
SET-BACKGROUND-GRAY-LEVEL command
SET-SURFACE-DEFINITIONS command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-PRIORITIES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-COORDINATE-MODE Command

Host Syntax

```
EcUX   int:coordinate-mode   int:intc-report-size
```

Setup Syntax

```
EcUX SP   coordinate-mode   intc-report-size
```

PARAMETERS

coordinate-mode (0 or 1: 4125, 4128, 4129; 2: 4128, 4129)
Specifies the format of *xy* and *xy+* parameters and *xy-report* and *xy+-report* message types.

intc-report-size (0, 2 to 6)
Specifies the number of characters in succeeding *intc-reports* sent from the terminal to the host.

DESCRIPTION

This command specifies the format for *xy* and *xy+* parameters, *xy-reports* and *xy+-reports*, and the length of *intc-reports*.

In the 4128 and 4129, the *coordinate-mode* also determines the dimensionality of a segment when it is opened. If the coordinate mode is changed from 2D (0 or 1) to 3D (2) or from 3D to 2D while a segment, panel, or graphtext character is open, then it is closed before the coordinate mode change takes effect.

Coordinate Mode 0. In coordinate mode 0 (the default), the terminal recognizes *xy* parameters in the 12-bit format. See the description of the *xy* parameter type for details.

All coordinates are absolute, explicitly defining points with values from 0 to 4095. *XY-reports* are in either 12-bit or 10-bit format, depending on the command that caused them.

Segments opened in coordinate mode 0 are 2D segments.

Coordinate Mode 1. Coordinate mode 1 allows addressing the entire 32-bit 2D terminal space. *XY* (including *xy+*) and *xy-report* (including *xy+-report*) parameters take the following forms:

xy+ = *xy* = int:*x-coord* int:*y-coord*
xy+report = *xy-report* = intc-report:*x-coord*
intc-report:*y-coord*

In coordinate mode 1, *xy*'s are sometimes absolute and sometimes relative. In all escape sequence commands, all *xy*s are absolute, and all *xy-arrays* start with an absolute coordinate, followed by points relative to the previous accumulated *xy* values within the context of the array. After an ENTER-VECTOR-MODE or ENTER-MARKER-MODE command, the first *xy* is absolute, and the following *xy*s are relative to the accumulated *xy* values within the context of the current beam position.

Segments opened in coordinate mode 1 are 2D segments.

Coordinate Mode 2. In coordinate mode 2, *xy+* parameters are encoded as triples of *int* parameters which define xyz points in 3D terminal space. *XY+-reports* consist of a triple of *intc-reports*. *XY* parameters and *xy-reports* are encoded just as they are in coordinate mode 1. Thus, *xy* parameters, *xy+* parameters, *xy-report* parameters, and *xy+-report* messages take the following forms:

xy = int:*x-coord* int:*y-coord*
xy+ = int:*x-coord* int:*y-coord* int:*z-coord*
xy-report = intc-report:*x-coord* intc-report:*y-coord*
xy+-report = intc-report:*x-coord*
intc-report:*y-coord*
intc-report:*z-coord*

In coordinate mode 2, *xy+*s are sometimes absolute and sometimes relative. In all escape sequence commands, all *xy+*s are absolute, and all *xy+-arrays* start with an absolute point, followed by points relative to the previous accumulated xyz values within the context of the array. After an ENTER-VECTOR-MODE or ENTER-MARKER-MODE command, the first *xy+* is absolute, and the following *xy+*s are relative to the accumulated xyz values within the context of the current 3D beam position.

Segments opened in coordinate mode 2 are 3D segments.

Intc-Report-Size. This parameter specifies the length of *intc-reports* that the terminal sends to the host when it sends *xy+-report*s in coordinate modes 1 and 2, and for the *terminal-settings-report*s for certain commands whose parameters have terminal-space integer ranges. These commands include SET-GRAPHTEXT-SIZE, SET-VIEWING-TRANSFORM, and SET-VIEW-DISPLAY-ATTRIBUTES. If *intc-report-size* is 0, the length of *intc-reports* is unchanged.

DEFAULTS

coordinate-mode
as shipped — 0
on power-up — 0
if omitted — 0

intc-report-size
as shipped — 3
on power-up — 3
if omitted — 0

ERRORS

UX11	(Level 2):	Invalid *coordinate-mode.*
UX21	(Level 2):	Invalid *intc-report-size.*

REFERENCES

ENTER-MARKER-MODE command
ENTER-VECTOR-MODE command
Int parameter type
Intc-report message type
SET-GRAPHTEXT-SIZE command
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEWING-TRANSFORM command
Terminal-settings-report message type
XY parameter type
XY+ parameter type
XY-report message type
XY+-report message type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-CURRENT-MATCHING-CLASS Command

Host Syntax

```
EcSL   int-array:inclusion-set
       int-array:exclusion-set
```

Setup Syntax

```
EcSL SP   inclusion-set   exclusion-set
```

PARAMETERS

inclusion-set (array length 0 to 32768;–1, 1 to 64)
The set of classes used in the inclusion part of a matching operation.

–1	all classes
1 to **64**	a specific class

exclusion-set (array length 0 to 32768;–1, 1 to 64)
The set of classes used in the exclusion part of a matching operation.

–1	all classes
1 to **64**	a specific class

DESCRIPTION

This command establishes the inclusion and exclusion sets used in matching operations.

A matching operation is done for each defined segment (except segment 0, the crosshair cursor) when segment-number –3 is specified in a segment command.

The matching operation is:

IF The segment's class set intersected with the inclusion set equals the inclusion set.

AND The segment's class set intersected with the exclusion set equals the empty set.

THEN The command is performed on the segment.

Otherwise, the command is not performed on the segment.

The elements in the *inclusion-set* and *exclusion-set* parameters represent user-selected classes to be included or excluded in the current matching class. There are 64 possible classes (1 to 64), and all combinations are valid. If you assign -1 as a member of either set, that set includes all classes. For a detailed discussion of segment matching classes see the *4110/4120 Series Host Programmers Manual.*

DEFAULTS

inclusion-set
as shipped — empty array
on power-up — empty array
if omitted — empty array

exclusion-set
as shipped — empty array
on power-up — empty array
if omitted — empty array

ERRORS

SL11	(Level 2):	Invalid *inclusion-set* array.
SL12	(Level 3):	Out of memory while parsing the parameter.
SL21	(Level 2):	Invalid *exclusion-set.*
SL22	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

4110/4120 Series Host Programmers Manual
SET-SEGMENT-CLASS command

INQUIRY COMMAND

none

SET-CURVE-SMOOTHNESS Command

Host Syntax

$^{E}_{C}$**UG** real:*smoothness*

Setup Syntax

$^{E}_{C}$**UG** $^{S}_{P}$ *smoothness*

PARAMETERS

smoothness (0.0 to 1.0)

0.0 Smoothest (1 degree per vector).
1.0 Roughest (45 degrees per vector).

DESCRIPTION

This command determines smoothness of an arc drawn with the DRAW-CURVE command.

A *smoothness* of **0** results in one degree per vector, or 360 vectors per circle. A *smoothness* of **1** results in 45 degrees per vector or 8 vectors per circle. Values between 0 and 1 are linearly interpolated by this equation:

degrees per vector = 1 + (44 • *smoothness*)

The default smoothness is 1/11, resulting in 5 degrees per vector.

NOTE

Since arcs are remembered in segments as vectors, smoother arcs take more terminal memory space than rougher arcs.

DEFAULTS

smoothness
as shipped — 23831, –18 (1/11)
on power-up — 23831, –18 (1/11)
if omitted — 0,0

ERRORS

UG11 (Level 2): Invalid value for *smoothness*.

REFERENCES

DRAW-CURVE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-ALTERNATE-INDEX Command

Host Syntax

$^{E}_{C}$**LJ** int:*color-index*

Setup Syntax

DA2INDEX $^{S}_{P}$ *color-index*

PARAMETERS

color-index (0 through 32767)
Specifies the color index you want to represent boldface.

DESCRIPTION

This command specifies the color index to be used for characters in "boldface," set by a SELECT-GRAPHIC-RENDITION command.

Even though this command is part of the Tek 4110/4120 command set, it affects the action of a command that is part of the ANSI command set.

If you do not set the color index for boldface characters with this command, the color defaults to 2 (4111) or 7 (4115/4120 Series).

There is a maximum color index for the surface on which the dialog area is located: one less than 2^M, where M is the number of bit planes assigned to that surface. (See the description of the SET-SURFACE-DEFINITIONS command for more information on assigning bit planes to surfaces.)

Color indices greater than the maximum are treated as if they were equal to that maximum. Surface –1, the "Super Surface" is an exception; it corresponds to all bit planes in all defined surfaces. (See the *4110/4120 Series Host Programmers Manual* for details on the Super Surface.)

For the 4111, this command sets the foreground index of color-pair 2. This is compatible with the 4120 Series since the SET-GRAPHIC-RENDITION command, with a rendition value of 1 (alternate foreground index, bold), selects color-pair 2. The foreground index of color-pair 2 may also be set with the SET-DIALOG-AREA-INDEX command.

DEFAULTS

color-index
  as shipped — 7
  on power-up — remembered
  if omitted — 0

ERRORS

LJ11 (Level 2): Invalid *color-index.*

REFERENCES

SELECT-GRAPHIC-RENDITION command
  (ANSI command set)
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-BUFFER-SIZE Command

Host Syntax

```
EcLB   int:number-of-lines
```

Setup Syntax

```
DABUFFER Sp  number-of-lines
```

PARAMETERS

number-of-lines (2 to 32767)
  Specifies the number of lines in the dialog area scroll buffer.

DESCRIPTION

This command sets the size of the scroll buffer for the currently enabled dialog area and takes effect the next time the dialog area is made visible. The width of the lines is set by the SET-DIALOG-AREA-CHARS command. If *dialog-area* –2 is currently enabled on the 4128 or 4129, an error is detected and no action is taken.

When the dialog area is next made visible, if the buffer size is less than the number of lines in the dialog area viewport (as set by the SET-DIALOG-AREA-LINES command), error LV03 is generated, and the buffer size is increased to equal the number of lines in the viewport. If the buffer size cannot be increased (e.g., lack of memory space), the size of the viewport is decreased to equal the buffer size.

DEFAULTS

number-of-lines
  as shipped — 48 (4111) or 34 (4115/4120 Series)
  on power-up — remembered
  if omitted — error LB11

ERRORS

LB03 (Level 2): Context error. Buffer size cannot be set when *dialog-area* –2 is enabled.

LB11 (Level 2): Invalid *number-of-lines.*

REFERENCES

DIALOG key
SET-DIALOG-AREA-CHARS command
SET-DIALOG-AREA-LINES command
SET-DIALOG-AREA-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-CHARS Command

Host Syntax

```
EcLC   int:number-of-chars
```

Setup Syntax

```
DACHARS Sp   number-of-chars
```

PARAMETERS

number-of-chars (4111: 5 to 132;
4115/4120 Series: 5 to 160)
Specifies number of characters per line in the dialog area.

DESCRIPTION

This command sets the number of characters per line of the currently enabled dialog area and takes effect when the dialog area is next made visible. This controls the size of both the dialog area viewport and the dialog area scroll buffer.

On the 4115/4120 Series, if *number-of-chars* is in the range 81 to 160, and the character size is large (as set by the SET-4014-ALPHATEXT-SIZE command), when the dialog area is made visible the size of the dialog area viewport and the scroll buffer are both reduced to 80 and error LV03 is detected. If the character size is changed to small, the dialog area viewport and scroll buffer sizes are increased to *number-of-chars.*

On the 4111, "small" is the only size of alphatext.

If *dialog-area* –2 is currently enabled on the 4128 or 4129, an error is detected and no action is taken.

DEFAULTS

number-of-chars
as shipped — 132 (4111), 80 (4115/4120 Series), or 60 (dialog area –2)
on power-up — remembered
if omitted — error LC11

ERRORS

LC03 (Level 2): Context error. *Number-of-chars* cannot be set when *dialog-area* –2 is enabled.

LC11 (Level 2): Invalid *number-of-chars.*

REFERENCES

DIALOG key
SET-DIALOG-AREA-BUFFER-SIZE command
SET-DIALOG-AREA-VISIBILITY command
SET-4014-ALPHATEXT-SIZE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-INDEX Command

Host Syntax

```
EcLI  int:character-index
      int:char-background-index
      int:color-pair
```

Setup Syntax

```
DAINDEX Sp  character-index
            char-background-index
            color-pair
```

PARAMETERS

character-index (0 to 32767)
The color index with which characters are displayed in the dialog area.

char-background-index (0 to 32767)
The color index with which the backgrounds of characters are displayed.

color-pair (4111: –8 to 32767; 4115/4120 Series: 0 to 32767)
The color index used when erasing the dialog area.

On the 4111, color-pair specifies which of the eight color-pairs is being defined. This parameter is ignored by the 4115/4120 Series.

–1 to 32767 color pair 1 (default)
–2 color pair 2 (bold)
–3 color pair 3
–4 color pair 4
–5 color pair 5
–6 color pair 6
–7 color pair 7
–8 color pair 8

DESCRIPTION

This command sets the color indices used in the dialog area. The first parameter specifies the color index used to write characters. The second parameter specifies the color index used to write the backgrounds of those characters. Unoccupied character cells are always transparent.

There is a maximum color index for the surface on which the dialog area is located: one less than 2^M, where M is the number of bit planes assigned to that surface. (See the description of the SET-SURFACE-DEFINITIONS command for more information on assigning bit planes to surfaces.)

Color indices greater than the maximum are treated as if they were equal to that maximum. Surface −1, the "Super Surface" is an exception; it corresponds to all bit planes in all defined surfaces. (See the *4110/4120 Series Host Programmers Manual* for details on the Super Surface.)

Example. One way to show the operator the boundaries of the dialog area is to make characters typed there appear on a light gray background.

Assume the following: (a) There is only one surface, which has three bit planes. That is, color indices can range from 0 to 7 on that surface. (b) The background gray level is "black" 0% lightness; thus, pixels written in color index 0 ("transparent") will appear black. (c) color index 7 is set to "100% lightness," and color index 4 to "50% lightness."

Under those assumptions, you can make the dialog area appear on an enhanced background by issuing the following command:

SET-DIALOG-AREA-INDEX : 7, 4, 4
= ^{E}c**LI** int:7 int:4 int:4
= ^{E}c**LI744**

The command's three parameters (7, 4, and 4) have the following effects. Characters are displayed in color index 7 (white). The background for each character cell is color index 4 (50% lightness). When the dialog scroll is erased, all its pixels are set to color index 4 (50% lightness).

Color-pair. On the 4111, this parameter specifies which color-pair is being defined. Color-pairs are selected with the SELECT-GRAPHIC RENDITION command. On the 4115/4120 Series, this parameter is ignored.

Character-Index, Char-Background-Index. When a character is typed into the dialog area, the character is written in the current dialog *character-index* (the first parameter in the SET-DIALOG-INDEX command. The other pixels in that character cell are written in the current *char-background-index*. (The second parameter in the command.)

Interaction with SET-DIALOG-WRITING-MODE command. How characters are written in the dialog area does not depend only on this command; the SET-DIALOG-AREA-WRITING-MODE command also has an effect.

Errors. Errors generated by this command are not generated until the next time the dialog area is made visible. If the dialog area is visible when the terminal receives the command, the dialog area must be made visible again (either with the DIALOG key or with a SET-DIALOG-AREA-VISIBILITY command) before errors are generated (or before the dialog area indices are changed).

DEFAULTS

character-index
as shipped — 1
on power-up — remembered
if omitted — 0

char-background-index
as shipped — 0
on power-up — remembered
if omitted — 0

color-pair
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

LI11	(Level 2):	Invalid *character-index.*
LI21	(Level 2):	Invalid *character-background-index.*
LI31	(Level 2):	Invalid *color-pair.*

REFERENCES

ERASE-DIALOG-AREA command
SELECT-GRAPHIC-RENDITION command
SET-DIALOG-AREA-VISIBILITY command
SET-DIALOG-AREA-SURFACE command
SET-DIALOG-AREA-WRITING-MODE command
SET-SURFACE-DEFINITIONS command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-LINES Command

Host Syntax

EcLL int:*number-of-lines*

Setup Syntax

DALINES Sp *number-of-lines*

PARAMETERS

number-of-lines (4111: 2 to 48; 4115/4120 Series: 2 to 64)
Specifies the number of lines in the dialog area viewport.

DESCRIPTION

This command sets the number of lines in the dialog area viewport and takes effect when the dialog area is next made visible.

When the dialog area is next made visible, if the buffer size (as set by the SET-DIALOG-AREA-BUFFER-SIZE command) is less than the number of lines in the dialog area viewport, error LV03 is generated, and the buffer size is increased to equal the number of lines in the viewport. If the buffer size cannot be increased (e.g., lack of memory space), the size of the viewport is decreased to equal the buffer size.

If *number-of-lines* is in the range 35 to 64, and the character size is large (as set by the SET-4014-ALPHATEXT-SIZE command), when the dialog area is made visible the size of the dialog area viewport and the scroll buffer are both reduced to 34 and error LV03 is detected. If the character size is changed to small, the dialog area viewport and scroll buffer sizes are increased to *number-of-lines.*

If *dialog-area* –2 is currently enabled on the 4128 or 4129, an error is detected and no action is taken.

DEFAULTS

number-of-lines
as shipped — 5; 4 for dialog area –2
on power-up — remembered
if omitted — error LL11

ERRORS

LL03 (Level 2): Context error. *Number-of-lines* cannot be set when *dialog-area* –2 is enabled.

LL11 (Level 2): Invalid *number-of-lines*.

REFERENCES

DIALOG key
SET-DIALOG-AREA-BUFFER-SIZE command
SET-DIALOG-AREA-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-POSITION Command

Host Syntax

```
EcLX   xy:lower-left-corner
```

Setup Syntax

```
DAPOSITION Sp   lower-left-corner
```

PARAMETERS

lower-left-corner (X = 0 to 4095, Y = 0 to 4095)
Specifies the position on the screen of the dialog area's lower-left corner.

DESCRIPTION

This command sets the position of the lower left corner of the currently enabled dialog area and takes effect the next time the dialog area is made visible. The actual position set is the nearest lower-left corner of a character cell in the dialog area grid.

The terminal always fits the number of lines and characters per line (as set by the SET-DIALOG-AREA-LINES and SET-DIALOG-AREA-CHARS commands) on the screen. If necessary, the dialog area position is moved down and/or to the left to accommodate this; when the dialog area is made visible, error LV03 level 0 is detected. See the SET-DIALOG-AREA-VISIBILITY command description for details.

DEFAULTS

lower-left-corner
as shipped — (0,0)
on power-up — remembered
if omitted — (0,0)

ERRORS

LX11 (Level 2): Invalid *lower-left-corner.*

REFERENCES

SET-DIALOG-AREA-CHARS command
SET-DIALOG-AREA-LINES command
SET-DIALOG-AREA-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-SURFACE Command

Host Syntax

```
EcLS   int:surface-number
```

Setup Syntax

```
DASURFACE Sp   surface-number
```

PARAMETERS

surface-number (1 to 8)
Specifies which surface's color-to-index mapping the dialog area uses for its color indices. The value −1 is valid on all terminals and indicates the super surface, which consists of all bit-planes of all currently defined surfaces. (See the *4110/4120 Series Host Programmers Manual* for more information about the super surface.)

DESCRIPTION

This command specifies which surface's color map the dialog area uses to assign its own color map. Other aspects of the surface (such as visibility and priority) do not affect the appearance of the dialog area.

This command takes effect the next time the dialog area is made visible.

DEFAULTS

surface-number
as shipped — 1
on power-up — remembered
if omitted — error LS11

ERRORS

LS11 (Level 2): Invalid *surface-number.*

REFERENCES

SET-DIALOG-AREA-INDEX command
SET-DIALOG-AREA-VISIBILITY command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-VISIBILITY Command

Host Syntax

EcLV int:*visibility*

Setup Syntax

DAVIS Sp *visibility*

PARAMETERS

visibility (0 or 1)
Specifies whether the dialog area is visible or invisible. Setup mode parameters are **YES** and **NO**.

0	**NO**; makes the dialog area invisible.
1	**YES**; makes the dialog area visible.

DESCRIPTION

This command makes a dialog area visible or invisible.

Making a Dialog Area Visible. If visibility is one and dialog area 1 to 64 is currently enabled, the contents of the dialog area scroll are displayed on the screen. If no dialog area is enabled (all are disabled), dialog area 1 is made visible. If dialog area –2 is currently enabled, its visibility attribute is set to ON.

The most recent settings for the following commands take effect for all dialog areas when any dialog area, except dialog area –2, is made visible:

SET-DIALOG-AREA-INDEX
SET-DIALOG-AREA-SURFACE
SET-4014-ALPHATEXT-SIZE

The most recent settings for the following commands take effect for only the currently enabled dialog area, except dialog area –2, when it is made visible:

SET-DIALOG-AREA-BUFFER-SIZE
SET-DIALOG-AREA-CHARS
SET-DIALOG-AREA-LINES
SET-DIALOG-AREA-POSITION
SET-DIALOG-AREA-WRITING-MODE

When dialog area –2 is made visible, the most recent setting of only SET-DIALOG-AREA-POSITION takes effect.

If the dialog area position is such that the dialog viewport will not fit on the screen, or if the dialog area buffer is smaller than will fit in the terminal's memory or is smaller than dialog area lines, then error LV03 level 0 occurs. So long as the dialog area remains visible, the REPORT-TERMINAL-SETTINGS command and the setup mode STATUS command will report the altered settings. However, the old settings remain in the terminal's battery-powered backup memory. If the dialog area is made invisible again, then the REPORT-TERMINAL-SETTINGS and STATUS command will report the settings stored in the backup memory.

Dialog area 1 is visible on power-up (or reset) if any dialog area (other than –2) was enabled at power-down (or before the reset). Dialog area –2 has its visibility attribute set to ON at Power-up or reset.

Making the Dialog Area Invisible. If *visibility* is zero and dialog area 1 to 64 is currently enabled, the contents of the dialog area are removed from the screen. If no dialog area is currently enabled, dialog area 1 is made not visible. If dialog area –2 is currently enabled, its visibility attribute is set to OFF.

Dialog Area Priority. If more than one dialog area is visible at the same time, the most recently made visible is displayed in front of all the others. Dialog area –2 is an exception to this priority rule; it is always displayed in front of all the others.

DEFAULTS

visibility
as shipped — 0 for dialog area 1; 1 for dialog area –2
on power-up — for dialog area 1, same as remembered setting for ENABLE-DIALOG-AREA 1; for dialog area –2, always 1
if omitted — 1

ERRORS

LV03	(Level 0):	One or more of the dialog area parameters was altered when the dialog area was made visible.
LV11	(Level 2):	Invalid *visibility*.

REFERENCES

CLEAR-DIALOG-SCROLL command
DIALOG key
ENABLE-DIALOG-AREA command
PAN key
REPORT-TERMINAL-SETTINGS command
SET-DIALOG-AREA-BUFFER-SIZE command
SET-DIALOG-AREA-CHARS command
SET-DIALOG-AREA-INDEX command
SET-DIALOG-AREA-LINES command
SET-DIALOG-AREA-POSITION command
SET-DIALOG-AREA-SURFACE command
SET-DIALOG-AREA-WRITING-MODE command
SET-4014-ALPHATEXT-SIZE command
STATUS command
ZOOM key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-AREA-WRITING-MODE Command

Host Syntax

```
EcLM   int:writing-mode
```

Setup Syntax

```
DAMODE Sp   writing-mode
```

PARAMETERS

writing-mode (0 or 1)
Specifies the writing mode used for text going to the dialog area. Setup mode parameters are **REPLACE** and **OVERSTRIKE**.

0	**REPLACE**
1	**OVERSTRIKE**

DESCRIPTION

This command sets the writing mode for dialog area characters with respect to characters already present in the dialog area.

If *writing-mode* is 1 (overstrike), Sp and _ (underscore) characters are written over old characters; other characters act like the terminal is in Replace mode. You can underline characters by backspacing and typing over them with the "underscore" character (_).

If *writing-mode* is 0 (Replace), dialog area characters completely replace the old characters in a given character space. In this mode, a character typed in the dialog area erases any characters formerly at that character position. This mode is useful with the "line editing" features of some host operating systems.

DEFAULTS

writing-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

LM11 (Level 2): Invalid *writing-mode.*

REFERENCES

SET-DIALOG-AREA-INDEX command
SET-GRAPHICS-AREA-WRITING-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DIALOG-HARDCOPY-ATTRIBUTES Command

4111 or OPTION 19

Host Syntax

```
EcQL  int:number-of-pages   int:page-origin
      int:page-definition
```

Setup Syntax

```
HCDAATTRIBUTES Sp  number-of-pages
                   page-origin
                   page-definition
```

PARAMETERS

number-of-pages (0 to 32767)
The number of pages to be printed starting at the origin.

0 No change.

1 to **32767** Number of pages to be printed.

page-origin (0, 1, and 2)
Starting position for the number of pages to be printed.

0 Start at the first line in the visible dialog area.

1 Start at first line in dialog area scroll buffer.

2 Print last n pages, where n is *number-of-pages.*

page-definition (0, 1, and 2)
Defines the length of the page on the copier.

0 Page is 60 text lines plus three blank lines at the top and three blank lines at the bottom. Formfeed characters are ignored.

1 Page is 66 text lines long or until a formfeed character appears in the text, whichever occurs first.

2 Page ends whenever a formfeed character appears in the text.

DESCRIPTION

This command sets values for dialog area copies printed on the 4695 Color Copier by HARDCOPY:3 or by pressing the CTRL-HARDCOPY keys. If the *number-of-pages* parameter is larger than the number of pages in the dialog area scroll buffer, only the available pages are printed. Control characters other than formfeed are sent to the 4695 as upside down question marks.

This command has no effect on a 4691 or 4692 color copier connected to the Option 19 color copier interface.

NOTE

> *The terminals do not store formfeed characters in the dialog area except as a SNOOPY character. The formfeed is displayed as a SNOOPY character if entered when the terminal is in Setup mode or with SNOOPY mode turned on.*

DEFAULTS

number-of-pages
as shipped — 1
on power-up — 1
if omitted — 1

page-origin
as shipped — 0
on power-up — 0
if omitted — 0

page-definition
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

QL00	(Level 0):	Unrecognized command. (Must be a 4111 or Option 19 must be installed.)
QL11	(Level 2):	Invalid *number-of-pages*.
QL21	(Level 2):	Invalid *page-origin*.
QL31	(Level 2):	Invalid *page-definition*.

REFERENCES

COPY command
HARDCOPY command
HARD COPY key
SELECT-HARDCOPY-INTERFACE command
SET-SNOOPY-MODE command
SPOOL command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DMA-BLOCK-SIZE Command

4111 or OPTIONS 3A, 3C, 3Q

Host Syntax

$^{E}{}_{C}$**JH** int:*DMA-block-size*

Setup Syntax

$^{E}{}_{C}$**JH** $^{S}{}_{P}$ *DMA-block-size*

PARAMETERS

DMA-block-size (1 to 65504)
Specifies the maximum number of bytes that are transferred by DMA in a single block.

DESCRIPTION

When DMA is the *source* in a file transfer operation, the block size specifies the number of bytes that the terminal reserves for a single DMA block. The block the DMA sends to the terminal may be smaller than or equal to the *DMA-block-size*. Each block transfer from the DMA is completed before the terminal sends the block to the *destination*. If the block is larger than the specified block size, an error is detected and the transfer is aborted.

When DMA is the *destination* in a file transfer operation, the block size specifies the maximum number of bytes that the terminal sends to the DMA during a single data transfer. If the terminal has more data to send than fits in one block, the data is broken into multiple blocks, each of which is smaller or equal in size to *DMA-block-size*.

DEFAULTS

DMA-block-size
as shipped — none
on power-up — 512
if omitted — error JH11

ERRORS

JH00 (Level 0): Unrecognized command. (Not a 4111 or Option 3A, 3C, 3Q is not installed.)

JH03 (Level 2): SET-DMA-BLOCK-SIZE command received after DMA failed to power up.

JH11 (Level 2): Invalid *DMA-block-size*.

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DRAW-BOUNDARY-MODE Command

Host Syntax

```
EcUB   int:draw-boundary-mode
```

Setup Syntax

```
EcUB   draw-boundary-mode
```

PARAMETERS

draw-boundary-mode (0 or 1)
Specifies whether boundaries of panels defined by the DRAW-RECTANGLE command are drawn.

0 boundaries are not drawn

1 boundaries are drawn

DESCRIPTION

This command specifies whether the boundaries of rectangles defined by the DRAW-RECTANGLE command will be drawn or not.

DEFAULTS

draw-boundary-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

UB11 (Level 2): Invalid *draw-boundary-mode.*

REFERENCES

DRAW-RECTANGLE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-DUPLEX-MODE Command

OPTION 01

Host Syntax

```
EcOD   int:duplex-mode
```

Setup Syntax

```
DUPLEX Sp   duplex-mode
```

PARAMETERS

duplex-mode (0 to 3)
Specifies which sort of Duplex mode is active. Setup mode parameters are **FULL**, **NORMAL**, **ARTS**, and **SUPER**.

0 **FULL**; full duplex data communications protocol

1 **NORMAL**; half duplex normal protocol

2 **ARTS**; half duplex with automatic request to send

3 **SUPER**; half duplex with supervisor

DESCRIPTION

The SET-DUPLEX command allows the host computer to set the duplex mode for the terminal's communication with the host. An int of zero places the terminal in normal full duplex mode. Mode 1 is the normal half duplex mode. Mode 2 places the terminal in half duplex with an automatic request to send. Mode 3 puts the terminal in half duplex mode with a supervisor function operating from the host.

Normally, the terminal's duplex mode is set by the operator, using the "setup mode" DUPLEX command; see the appropriate operator's manual for details. However, the SET-DUPLEX "escape sequence" command is also provided for possible use by the host computer.

Programming Considerations. This command (like all commands) does not take effect until the terminal has processed it. After the host sends this command, it should wait for the command to be completely processed through the terminal's input queue before sending additional data that relies on a mode set by the command. After sending the SET-DUPLEX command, the host might issue a REPORT-TERMINAL-SETTINGS command to inquire the terminal's duplex setting. The host would then read the *terminal-settings-report* which the terminal returns. If the terminal reports the correct duplex mode, then the host can proceed with the rest of its program; otherwise, it can issue another REPORT-TERMINAL-SETTINGS command and try again.

DEFAULTS

duplex-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

OD00 (Level 2): Unrecognized command. (Option 1 is not installed.)

OD01 (Level 2): Invalid *duplex-mode.*

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-ECHO Command

Host Syntax

$^{E}{}_{C}$KE int:*echo-mode*

Setup Syntax

ECHO $^{S}{}_{P}$ *echo-mode*

PARAMETERS

echo-mode (0 or 1)
Specifies whether the terminal echos characters typed on the keyboard (local echo). Setup mode parameters are **YES** and **NO**.

0 **NO**; no local echo

1 **YES**; local echo of typed characters

DESCRIPTION

If *echo-mode* is 1, then characters which the operator types to the host are echoed locally by the terminal.

If *echo-mode* is 0, all local echoing is stopped.

NOTE

If the optional block mode communications protocol is used, then local echo should be enabled.

NOTE

While in full duplex and echo Y modes, if the terminal is busy receiving a graphtext command and the user types on the keyboard, the text displayed on the screen contains spurious lines and graphtext characters can be changed.

DEFAULTS

echo-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

KE11 (Level 2): Invalid *echo-mode.*

REFERENCES

ARM-FOR-BLOCK-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-EDIT-CHARS Command

Host Syntax

```
EcKZ  int:char-delete
      int:line-delete
      int:take-literally
```

Setup Syntax

```
EDITCHARS SP  char-delete
              line-delete
              take-literally
```

PARAMETERS

char-delete (0 to 127)
Numeric equivalent of the Setup mode char-delete character.

line-delete (0 to 127)
Numeric equivalent of the Setup mode line-delete character.

take-literally (0 to 127)
Numeric equivalent of the Setup mode take-literally character.

DESCRIPTION

The SET-EDIT-CHARS command sets the values of the *char-delete*, *line-delete*, and *take-literally* characters which the operator can use for line editing while in SETUP mode.

Specifying NU (ADE 0) for any of these characters causes the present value of that character to be left unchanged. Specifying CR for any of these is allowed, but CR remains as a line terminator, and does not perform editing functions.

Char-Delete Character. The *char-delete* character deletes the character just typed from the Setup mode command line being typed. The cursor backs up one character position, so that an operator can retype the character correctly.

Line-Delete Character. The *line-delete-character* deletes the current Setup mode command line. The terminal displays an asterisk to prompt an operator for another Setup mode command line.

Take-Literally Character. The *take-literally* character allows an operator to type special characters as part of a command. For instance, the operator may be typing a command which takes a delimited sequence of ASCII characters as a parameter, and want to include CR as part of that sequence. To do this, the operator can type the current *take-literally* character, followed by CR.

The *take-literally* character causes the terminal to interpret the following character as just another data item. This lets the operator enter parameters which include special characters like CR, the current *char-delete* character, the current *line-delete* character, or even the current *take-literally* character.

DEFAULTS

char-delete
as shipped — 127 (DT)
on power-up — remembered
if omitted — unchanged

line-delete
as shipped — 24 (CN)
on power-up — remembered
if omitted — unchanged

take-literally
as shipped — 126 (~)
on power-up — remembered
if omitted — unchanged

ERRORS

KZ11 (Level 2): Invalid *char-delete* character.

KZ21 (Level 2): Invalid *line-delete* character.

KZ31 (Level 2): Invalid *take-literally* character.

REFERENCES

See the description of Setup mode in the Operator's Manual for the particular TEKTRONIX 4110/4120 Series Computer Display Terminal.

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-EOF-STRING Command

Host Syntax

```
ECNE   int-array:EOF-string
```

Setup Syntax

```
EOFSTRING SP  EOF-string
```

PARAMETERS

EOF-string
This int-array holds numeric equivalents of up to ten ASCII characters, which comprise the *EOF-string*. Each numeric equivalent must be in the range from 0 to 127.

DESCRIPTION

This command sets the terminal's *EOF-string* (end-of-file string).

The *EOF-string* is used (when the terminal is not in block mode) to mark the end of a file being transferred between the terminal and the host computer. (In block mode, a bit in one of the control bytes at the end of the block serves a similar purpose.)

If the terminal is not in block mode, it appends the current *EOF-string* to the end of each file it sends to the host in response to a copy command. When the host is sending a file to the terminal, it should append the current *EOF-string* at the end of its transmission. (This lets the terminal know when the end of the file has been reached.)

When the terminal is in block mode, the end-of-file bit in the *block-control-bytes* takes the place of the *EOF-string*. When the terminal is in Block mode, or is armed for Block mode, it is incapable of recognizing the *EOF-string* in data coming from the host computer.

NOTE

If you will not be using Block mode, do not arm the terminal for Block mode. If the terminal is armed for Block mode, but is not yet actually in Block mode, the only way to terminate a copy from the host is with the CANCEL key.

When not in block mode, the terminal intercepts *EOF-strings* in data coming from the host, and deletes them from the incoming data stream. Because of this, it may be prudent to set the *EOF-string* to the desired value only just before each copy operation, and to set the *EOF-string* to the empty string after the copy.

DEFAULTS

EOF-string
as shipped — empty array
on power-up — remembered
if omitted — empty array

ERRORS

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): Out of memory while parsing the parameter.

REFERENCES

ARM-FOR-BLOCK-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-EOL-STRING Command

Host Syntax

```
EcNT   int-array:EOL-string
```

Setup Syntax

```
EOLSTRING SP
```

PARAMETERS

EOL-string
An array of from 0 to 2 *int*s. Each *int* is the numeric equivalent of an ASCII character, and so must be in the range from 0 to 127.

DESCRIPTION

This command sets the end-of-line (EOL) string. This string is typically the single character, CR; the SET-EOL-STRING command lets you set it to other combinations, such as, say, CRLF.

In Block mode, the terminal sends the *EOL-string* at the end of each line it sends to the host.

When not in Block mode, the *EOL-string* is sent at the end of reports which the terminal sends the host; see the syntax description for the particular report type for details.

DEFAULTS

EOL-string
as shipped — 13 (CR)
on power-up — remembered
if omitted — empty array

ERRORS

NT11 (Level 2): Invalid *EOL-string* (contents: from 0 to 2 *int*s; each in the range from 0 to 127).

NT12 (Level 3): Out of memory while parsing the parameter.

REFERENCES

ARM-FOR-BLOCK-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-EOM-CHARS Command

Host Syntax

```
EcNC   int:EOM-char-1   int:EOM-char-2
```

Setup Syntax

```
EOMCHARS Sp   EOM-char-1   EOM-char-2
```

PARAMETERS

EOM-char-1 (0 to 127)
Numeric equivalent of an EOM-char: an ASCII character that marks the end of a line of text in data sent to the host computer. 0 means "no character".

EOM-char-2 (0 to 127)
Numeric equivalent of another EOM-char. (If only one EOM-char is desired, set one of the two parameters in the SET-EOM-CHARS command to 0. If no EOM-chars are desired, set both parameters to 0.)

DESCRIPTION

Specifies one or two characters to be used as "turn-around", or EOM-(end-of-message) characters.

The ASCII Nu character (whose numeric equivalent is zero) may not be used as an *EOM-character*. You can specify any other ASCII character as an *EOM-character* by including its numeric equivalent as one of the two parameters in this command. To specify only one *EOM-character*, give its numeric equivalent in one of the parameters, and set the other parameter to 0. To designate no *EOM-character*s, set both parameters to zero.

See the *4110/4120 Series Host Programmers Manual* for a description of the effects of *EOM-chars*.

DEFAULTS

EOM-char-1
as shipped — 13 (Cr)
on power-up — remembered
if omitted — 0

EOM-char-2
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

NC11 (Level 2): Invalid *EOM-char-1.*
NC21 (Level 2): Invalid *EOM-char-2.*

REFERENCES

4110/4120/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-ERROR-THRESHOLD Command

Host Syntax

```
EcKT   int:error-threshold-level
```

Setup Syntax

```
ERRORLEVEL Sp   error-threshold-level
```

PARAMETERS

error-threshold-level (0 to 4; 0 to 5 on 4128 and 4129)
The minimum error severity for an error message to be displayed. 0 causes all errors to be displayed, while 4 (5) suppresses all error messages.

DESCRIPTION

Sets the minimum severity level for errors whose messages are to be displayed.

Errors are always recorded in the terminal's error queue, and the report may be obtained with the REPORT-ERRORS command.

DEFAULTS

error-threshold-level
as shipped — 2
on power-up — 2
if omitted — 0

ERRORS

KT11 (Level 2): Invalid *error-threshold-level.*

REFERENCES

Appendix C, *Error Codes*
REPORT-ERRORS command

INQUIRY COMMAND

REPORT TERMINAL-SETTINGS command

SET-FIXUP-LEVEL Command

Host Syntax

$^{E}_{C}$**RF** int:*fixup-level*

Setup Syntax

FIXUP $^{S}_{P}$ *fixup-level*

PARAMETERS

fixup-level (0 to 32767)
A number specifying how frequently the terminal updates the current viewport in its display.

DESCRIPTION

This command controls the amount of time spent updating the current viewport on the terminal when changes are made that affect the current view. Although this could be done with every change, this might be a time-consuming task for complex views. The higher the *fixup-level*, the more effort will be spent updating the viewport. A complete update occurs on a RENEW-VIEW command, a PAGE command, or a depression of the PAGE key.

Table 7-20 lists the *fixup-level*s and their meanings. A positive *fixup-level* not listed in the table has the same effect as the next lower *fixup-level* which is listed. For instance, *fixup-level* 3 has the same effect as *fixup-level* 2; *fixup-level* 100 has the same effect as *fixup-level* 6. A negative *fixup-level* causes the *fixup-level* to be set to 0, and error RF11 (Level 1) to be generated.The default *fixup-level* is six.

DEFAULTS

fixup-level
as shipped — 6
on power-up — 6
if omitted — 0

ERRORS

RF11 (Level 1): Invalid *fixup-level.*

REFERENCES

SET-SEGMENT-WRITING-MODE command
SET-SEGMENT-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

Table 7-20

FIXUP LEVELS

Fixup Level	Meaning
0	The screen is updated on a RENEW-VIEW or PAGE command, or when the PAGE or VIEW key is pressed. (At fixup level 0, these are the only times the viewport contents are altered.)
2	The action listed above for fixup level 0 is performed. Besides this, the display is updated to show any additions to the current view, as those additions are made. (For instance, whenever a line is drawn in the current view, the display is updated to show that line. Also, whenever a segment is made visible, the display is updated to show that segment.) (At fixup level 2, when a segment is moved, it is drawn in the new position, but not erased from its old position. The display is not updated when a segment is made invisible or deleted.)
4	The actions listed above for fixup levels 0 and two are performed. In addition, segments displayed in XOR mode are erased (by drawing them again in XOR mode) in response to commands that change the segment's visibility or position in terminal space. Segments displayed in Set mode are treated as in fixup level two.
6	The actions described above for fixup levels 0 through 4 are performed. In addition, whenever a segment is made invisible or is deleted, that segment is erased from the viewport. (For segments drawn in Set mode, this is done by redrawing the segment in the current wipe index. For segments drawn in XOR mode, this is done by redrawing the segment in XOR mode.)

SET-FLAGGING-MODE Command

Host Syntax

${}^{E}{}_{C}$**NF** int:*flagging-mode*

Setup Syntax

FLAGGING ${}^{S}{}_{P}$ *flagging-mode*

PARAMETERS

flagging-mode (0 to 4)

Specifies whether flagging occurs between the terminal and the host, and if flagging occurs, what type. Setup mode parameters are **NONE**, **INPUT**, **OUTPUT**, **IN/OUT**, and **DTR/CTS**.

0	**NONE**; no flagging
1	**INPUT**; DC1/DC3 flagging when receiving data from the host
2	**OUTPUT**; DC1/DC3 flagging when transmitting to the host
3	**IN/OUT**; DC1/DC3 flagging both when transmitting and when receiving
4	**DTR/CTS**; flagging with the data terminal ready and clear to send RS-232 signal lines.

DESCRIPTION

This command sets the "flagging," or "handshaking," protocol between the terminal and its host computer. (The SET-PORT-FLAGGING-MODE command controls similar protocols between the terminal and its RS-232 peripheral devices.)

Mode 0 (No Flagging). DC1/DC3 and DTR/CTS flagging are both disabled.

NOTE

When DC1/DC3 flagging (modes 1, 2, and 3) is enabled, the terminal's and host's flag recognition mechanism absorb the ${}^{D}{}_{1}$ and ${}^{D}{}_{3}$ characters.

Mode 1 (INPUT). The terminal uses the "DC1/DC3" flagging protocol when receiving characters from the host. If the host is sending characters to the terminal faster than the terminal can process them, so that the terminal's input buffer is in danger of overflowing, then the terminal sends the host a ${}^{D}{}_{3}$ character. The host is then expected to suspend transmission of characters to the terminal. When the terminal is ready for more characters, it sends the host a ${}^{D}{}_{1}$. The host is then expected to resume transmission of characters to the terminal.

Mode 2 (OUTPUT). The terminal uses the "DC1/DC3" flagging protocol when transmitting characters to the host. The host can send the terminal a ${}^{D}{}_{3}$ when its input buffer is in danger of overflowing. The terminal sends at most 1 or 2 more characters, and then stops transmitting to the host. When it receives a ${}^{D}{}_{1}$, the terminal resumes its transmission to the host.

Mode 3 (IN/OUT). The terminal uses the "DC1/DC3" flagging protocol both when receiving characters from the host and when transmitting characters to the host.

Mode 4 (DTR/CTS). In DTR/CTS flagging, two signal lines at the RS-232 connector are used to regulate the flow of data between the terminal and the host computer. These lines are DTR (data terminal ready) and CTS (clear to send).

NOTE

DTR/CTS flagging is usually not practical when the host is connected to the terminal over telephone lines by the use of modems. (In such a circumstance, the host does not have direct access to the DTR and CTS signal lines.) This flagging mode is only practical if the host is connected directly to the terminal.

The terminal indicates that it wishes to transmit data by asserting DTR (placing a positive voltage on the DTR signal line). If the host is ready to receive the data, it asserts CTS. The terminal is only allowed to transmit when CTS is asserted. Should the terminal be transmitting characters faster than the host can process them, so that the host's input buffer is in danger of overflowing, the host can drop CTS (place a negative voltage on the CTS signal line). With CTS negative, the terminal automatically stops transmitting. When the host is ready to receive more characters, it asserts CTS again, and the terminal resumes its transmission.

If DTR/CTS flagging is enabled, the terminal uses the DTR (data terminal ready) signal line in the same way that the host uses the CTS line. If the host is sending characters faster than the terminal can process them, so that the terminal's input buffer is in danger of overflowing, then the terminal drops DTR (places a negative voltage on the DTR signal line). The host is then expected to stop transmitting to the terminal. When the terminal is ready for more characters, it asserts DTR (places a positive voltage on the DTR line), and the host resumes its transmission to the terminal.

DEFAULTS

flagging-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

NF11 (Level 2): Invalid *flagging-mode.*

REFERENCES

SET-BLOCK-NON-XMT-CHARS command
SET-PORT-FLAGGING-MODE command

INQUIRY COMMAND

REPORT TERMINAL-SETTINGS command

SET-GIN-AREA Command

Host Syntax

$^{E}{}_{C}$**IV** int:*device-function*
int:*window-type*
xy:*first-corner*
xy:*second-corner*

Setup Syntax

GINAREA $^{S}{}_{P}$ *device-function*
window-type
first-corner
second-corner

PARAMETERS

device-function
The GIN *device-function.* See the ENABLE-GIN command for details.

window-type (–1 to 64)
The type of window you are specifying.

–1	the window set by the SET-GIN-WINDOW command
0	the window of the current view
1 to 64	the window of view 1 to 64, respectively

first-corner (X = 0 to 4095, Y = 0 to 4095)
The first corner of the area you are specifying on the specified GIN device.

second-corner (X = 0 to 4095, Y = 0 to 4095)
The second corner of the area you are specifying on the specified GIN device.

DESCRIPTION

This command maps a specified rectangular region on a GIN device (a GIN area) onto a specified rectangular window in terminal coordinate space.

Device-Function. When you assign a GIN area to a *device-function*, the GIN area is assigned to all functions of the specified device. For example, if you assign a GIN area to the tablet-locator *device-function*, and then invoke the tablet-pick *device-function*, the same GIN area is used.

Window-type. If you specify a *window-type* of −1, the GIN device area (specified by the third and fourth parameters) is mapped into the terminal space region specified by the most recent SET-GIN-WINDOW command. The specified GIN device area remains mapped into the specified terminal space window until another SET-GIN-AREA command remaps it by enclosing the former device area within its own.

If you specify a *window-type* of 0, the GIN device area is mapped into the window of the current view for 2D GIN or the U-V window of the current view for 3D GIN. These windows change when you change views. These windows may also be changed by the Local Viewing System.

If you specify a *window-type* of 1 through 64, the GIN area is mapped into the window associated with the view of that number for 2D GIN or the U-V window associated with the view of that number for 3D GIN. If you specify the number of a view that is not defined, then when the *device-function* is invoked, it acts as though there has not been a GIN area defined.

If the current view is different from the view specified by this parameter, the specified view is used for GIN action. Once the specified *device-function* stops its GIN action, the previous current view (the one in effect before the GIN action) is again the current view.

Coordinates. This command assumes that the GIN device is 4096-by-4096 units square. If the device is rectangular, the command assigns 4096 units to the long axis of the device, and a proportionate number of units to the short axis.

The x- and y-coordinates you assign are sorted so that any two corners specify the rectangle.

If you specify the two x-coordinates the same or the two y-coordinates the same (if the GIN area is zero in either direction) then an error is generated, except when you specify both corners to be (0,0), in which case, the GIN area defaults to from (0,0) to (4095,4095).

You can define multiple GIN areas for tablet and plotter devices, but only one for thumbwheels. If the device specified in the *device-function* parameter is the terminal thumbwheels, the GIN area must be from (0,0) to (4095,4095). If you define overlapping areas on non-thumbwheel devices, and the device position is inside more than one GIN area, the most recently defined GIN area and *window-type* values are used.

GIN coordinate data for the device is mapped from the GIN area on the associated window. If the window is that of the current view (*window-type* = 0), the movement rate remains in proper scale to the window, regardless of the zoom scale factor.

To delete a GIN area, define a GIN area which totally covers the GIN area you want to delete. To delete all GIN areas, define both the GIN window and the GIN area to be from (0,0) to (4095,4095).

DEFAULTS

device-function
as shipped — none
on power-up — none
if omitted — 0

window-type
as shipped — −1 for 2D GIN; 0 for 3D GIN
on power-up — −1 for 2D GIN; 0 for 3D GIN
if omitted — −1 for 2D GIN; 0 for 3D GIN1

first-corner
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

second-corner
as shipped — (4095,4095)
on power-up — (4095,4095)
if omitted — (4095,4095)

ERRORS

IV03	(Level 3):	Out of memory while processing command.
IV11	(Level 2):	Invalid *device-function* (see ENABLE-GIN).
IV21	(Level 2):	Invalid *window-specifier.*
IV31	(Level 2):	Invalid *first-corner.*
IV41	(Level 2):	Invalid *second-corner* or zero width or height GIN area.

REFERENCES

ENABLE-GIN command
SET-GIN-WINDOW command

INQUIRY COMMAND

none

SET-GIN-CURSOR Command

Host Syntax

EcIC int:*device-function* int:*segment-number*

Setup Syntax

EcIC SP *device-function* *segment-number*

PARAMETERS

device-function
Specifies the GIN device code and the GIN function to which the cursor segment is assigned. (See the ENABLE-GIN command for details.)

segment-number (0 to 32767)
Specifies the segment to be used as a GIN cursor. Segment 0 is the crosshair cursor.

DESCRIPTION

This command specifies the segment to be used as the GIN cursor for subsequent GIN operations with the specified GIN *device-function*. Segment 0, the standard crosshair cursor, is the default GIN cursor if no other has been assigned.

If this command is received while the specified GIN device-function is not enabled, no immediate action occurs. The next time the specified GIN device-function is enabled, the specified segment will serve as its cursor.

If this command is received while the specified GIN device-function is enabled, the previous cursor segment is disabled as the GIN cursor and the segment specified becomes the new GIN cursor. When this is done, the new cursor is moved to the current GIN position.

A segment is a GIN cursor only when (1) it has been selected as the cursor for a GIN device-function with this command, and (2) that GIN device-function has been enabled with the ENABLE-GIN command.

The segment selected for *device-function* 0 (thumbwheel-locate) is also used for 4010-GIN. The segments selected for *device-functions* 8 and 10 (tablet-locate and tablet-stroke) are also used for 4953-GIN.

When a segment becomes a GIN cursor, its writing mode, detectability, and visibility are remembered and then changed as follows:

- writing mode is set to XOR mode
- detectability is turned off
- visibility is turned on

As the enabled device is manipulated, the cursor segment's position is continuously updated to the current GIN position. This may be outside the current view's window, and the segment may be clipped partially or entirely. Segment 0 (the default crosshair cursor) is displayed by special hardware, so it is never clipped and is displayed within the current view; if its position is outside the current view's window, it is displayed at the closest point on the edge of the view.

After GIN is enabled, you can change the cursor's writing mode, detectability, visibility, and position with the appropriate commands.

- For graphic tablets used in Absolute mode and for plotters (that is, for Device Codes 1, 3, 4, and 5), the segment's position will be reset to the GIN position the next time the device is manipulated.
- For Relative mode devices (that is, for Device Codes 0, 6, 8, and 9), the segment's new position will become the GIN position the next time the device is manipulated. That is, the cursor segment will begin to move from the position set by the command.

(For details, see the SET-SEGMENT-POSITION, SET-SEGMENT-IMAGE-TRANSFORM, and SET-SEGMENT-MATRIX-TRANSFORM commands.)

When a segment ceases to function as the GIN cursor (that is, when GIN is disabled or another segment is selected to be the cursor), its previous writing mode, detectability, and visibility attributes are restored. The segment's position attribute, however, is not restored.

Several *device-functions* can have the same cursor assigned at the same time. The position of the cursor is determined by the last device to change position. Each device maintains its own "GIN-position" which is unaffected by other devices, so that a shared cursor segment flickers back and forth between the devices' GIN-positions as the devices are operated.

If the specified segment does not exist or is currently being defined, or if a 3D segment is specified as a cursor for a pick or stroke operation, then an error is detected.

DEFAULTS

device-function
- as shipped — none
- on power-up — none
- if omitted — 0

segment-number
- as shipped — 0
- on power-up — 0
- if omitted — 0

ERRORS

IC13	(Level 2):	GIN 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*.
IC23	(Level 2):	Context error. A 3D segment cannot be used as a GIN cursor for a pick or stroke function.

REFERENCES

ENABLE-GIN command
SET-SEGMENT-DETECTABILITY command
SET-SEGMENT-IMAGE-TRANSFORM command
SET-SEGMENT-MATRIX-TRANSFORM command
SET-SEGMENT-POSITION command
SET-SEGMENT-VISIBILITY command
SET-SEGMENT-WRITING-MODE command

INQUIRY COMMAND

none

SET-GIN-DISPLAY-START-POINT Command

Host Syntax

```
EcIX  int:device-function
      xy:start-point
```

Setup Syntax

```
GINSTARTPOINT Sp  device-function
                  start-point
```

PARAMETERS

device-function
The GIN device and function. See the ENABLE-GIN command for details.

start-point
Specifies the point at which GIN-inking and GIN-rubberbanding will start. The valid ranges for the different coordinate modes are:

coordinate mode 0:	X,Y = 0 to 4095
coordinate mode 1:	X,Y = -2^{31} to $2^{31}-1$

DESCRIPTION

This command sets the *start-point* that is used by the SET-GIN-RUBBERBANDING and SET-GIN-INKING commands when they are enabled with a parameter of 2.

This command is not valid when the 4128 or 4129 terminals are in 3D mode (coordinate mode 2).

DEFAULTS

device-function
as shipped — none
on power-up — none
if omitted — 0

start-point
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

ERRORS

IX03	(Level 0):	Context error. The terminal must be in coordinate mode 0 or 1.
IX11	(Level 2):	Invalid *device-function* parameter.
IX21	(Level 2):	Invalid *start-point*.

REFERENCES

ENABLE-GIN command
SET-GIN-INKING command
SET-GIN-RUBBERBANDING command

INQUIRY COMMAND

none

SET-GIN-GRIDDING Command

Host Syntax

```
EcIG   int:device-function
       int:x-grid-spacing
       int:y-grid-spacing
```

Setup Syntax

```
GINGRIDDING Sp  device-function
                x-grid-spacing
                y-grid-spacing
```

PARAMETERS

device-function
A graphic input *device-function* code. (See ENABLE-GIN command for details.)

x-grid-spacing (0 to $2^{31}-1$)
Horizontal (x-direction) grid spacing.

y-grid-spacing (0 to $2^{31}-1$)
Vertical (y-direction) grid spacing.

DESCRIPTION

This command sets the size of the grid for all subsequent operations of the specified locator or pick GIN functions. The grid starts at 0,0 and consists of points spaced *x-grid-spacing* apart horizontally and *y-grid-spacing* apart vertically. GIN cursor movement is constrained so that the cursor will always lie on one of the grid points.

If *x-grid-spacing* is 0, the grid points are spaced one apart horizontally. If *y-grid-spacing* is 0, the grid points are spaced one apart vertically.

Gridding applies only to the locator and pick functions; gridding is not permitted for the stroke function.

DEFAULTS

device-function
as shipped — none
on power-up — none
if omitted — 0

x-grid-spacing
as shipped — 0
on power-up — 0
if omitted — 0

y-grid-spacing
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

I011	(Level 2):	Invalid *device-function* code. (See the description of the ENABLE-GIN command for a table of *device-function* codes.)
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-grid-spacing*
IG31	(Level 2):	Invalid *y-grid-spacing*

REFERENCES

ENABLE-GIN command

INQUIRY COMMAND

none

SET-GIN-INKING Command

Host Syntax

```
EcII  int:device-function   int:inking-mode
```

Setup Syntax

```
GININKING Sp  device-function   inking-mode
```

PARAMETERS

device-function
See the ENABLE-GIN command for details.

inking-mode (0, 1, or 2).
Specifies whether *inking-mode* is on or off, and if on, where inking starts.

0 turn inking off
1 turn inking on
2 turn inking on (start from *GIN-display-start-point*)

DESCRIPTION

This command turns inking on or off for all subsequent operations of the specified locator or stroke function. Inking is not allowed with the pick function and with locator functions which have a 3D segment for a GIN cursor.

When inking is turned on, each locator event after the first causes a line to be drawn to the point selected by the locator event. During stroke functions, a line is drawn between each point in the stroke.

The line is drawn in SET mode using the current line style and line index. The line is *not* remembered in segments.

If *inking-mode* is 2, each time GIN is enabled the first GIN point is "inked" from the *GIN-display-start-point*, as set by the SET-GIN-DISPLAY-START-POINT command. After the first point, it is identical to *inking-mode* 1. If *inking-mode* is 1, and *rubberbanding-mode* is set to 2 by the SET-RUBBER-BANDING-MODE command, then GIN inking behaves as if *inking-mode* was set to 2.

Inking is not allowed in key release mode.

DEFAULT

device-function
as shipped — none
on power-up — none
if omitted — 0

inking-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

I002 (Level 2): Insufficient memory available for GIN functions.

I011 (Level 2): Invalid *device-function* code.

II11 (Level 2): Inking does not apply to the specified *device-function* code. (Inking is not allowed for the pick function.)

II13 (Level 2): Context error. Inking is not allowed with locator functions which have a 3D segment as the GIN cursor for currently enabled GIN.

II21 (Level 2): Invalid *inking-mode* (must be 0, 1, or 2; 0 or 1 if firmware is Version 3 or earlier).

REFERENCES

ENABLE-GIN command
SET-GIN-DISPLAY-START-POINT command
SET-GIN-RUBBERBANDING command
SET-LINE-INDEX command
SET-LINE-STYLE command

INQUIRY COMMAND

none

SET-GIN-LOGICAL-FUNCTION Command

4128, 4129

Host Syntax

$^{E}{}_{c}$**IO** int-array:*functions*
real-array:*reals*
xy + -array:*points*
char-array:*chars*

Setup Syntax

$^{E}{}_{c}$**IO** $^{S}{}_{P}$ *functions* *reals* *points* *chars*

PARAMETERS

functions (length 0, 1, or 2)
Specifies the device-function and logical function.

first element: *device-function* (0 or locator)

The device-function must be 0 for *logical-function* –1, and must be a locator GIN function for logical function 0 to 6. See the ENABLE-GIN command for valid locator device-functions.

not present 0

second element: *logical-function* (–1 to 6)
If 0 to 6, then the logical function code specifies which two of the 3D axes are controlled by the GIN device's two degrees of freedom.

not present 0

If –1, *reals* and *chars* are used to set up a multiplier and format for distance parameters displayed in dialog area –2 when the terminal is in a 3D Local Viewing function.

reals (if *logical-function* other than –1, length 0; if *logical-function* –1, length 0 or 1)
Specifies the multiplier for values displayed when the terminal is in a 3D Local Viewing function.

first element: multiplier (both integers representing the real must be in the range –32768 to 32767)

not present multiplier not changed

points (length must be 0)

chars (if *logical-function* other than –1, length 0; if *logical-function* –1, length 0 to 52)

Specifies a format for the display of values in dialog area –2 when the terminal is in a 3D Local Viewing function.

not present format set to: ##\###\##%

DESCRIPTION

When logical-function is 0 to 6, this command establishes the mapping from the 2D horizontal and vertical axes of a GIN device onto the U, V, and W axes when the GIN cursor is a user-defined 3D segment for locator functions.

When a GIN locator function is enabled while the GIN cursor is a 3D segment, the XYZ position of the cursor segment is transformed into UVW values, which are remembered as the initial 3D GIN position. As the device inputs new points, the XY values from the device are transformed through GIN-area/GIN-window mapping onto the U, V, or W axes of the 3D GIN position, according to the logical function code. The new UVW values are then transformed into XYZ coordinates to update the position of the cursor segment and perhaps to be sent to the host.

Table 7-21

LOGICAL FUNCTION CODE

log-fcn	horizontal	vertical
0	U	V
1	U	W
2	W	V
3	U	none
4	none	V
5	W	none
6	none	W

When logical-function is –1, this command sets up a multiplier and format for values displayed in dialog area –2 when the terminal is in a 3D Local Viewing function.

When a value is displayed in dialog area –2, the actual value in the terminal is first multiplied by *multiplier*.

The format of the distance parameters displayed for six of the 3D functions (ZOOM, PAN, RADIUS, DEPTH, FOCUS, and CLIP) is set by the *chars* parameter. The maximum number of characters from *chars* actually displayed depends on the 3D Local Viewing function: ZOOM = 20, PAN = 20, CLIP = 15, FOCUS = 35, RADIUS = 41, and DEPTH = 52. The characters in the format are displayed as themselves except for the following characters which control the format of the number itself. The number formatting characters must be contiguous (no non-formatting characters can be placed between formatting characters).

1. " + " causes a plus or minus sign (depending on the value of the number) to be displayed immediately to the left of the first displayed digit. Only one plus (or minus) sign can be in the *chars* array; the sign must be before the radix point. If there are no " + "s and the number is negative, a "–" is displayed in the first "#" or "%" position. If the " + " is specified, it must be the first of the special characters.

2. "%" causes a digit to be displayed, including all leading and trailing zeros. If the number is too large to be displayed in the given number of "%"s and "#"s, then "∗"s are displayed in all "%" positions. The total number of "%"s and "#"s (see 3 below) must not exceed 18.

3. "#" causes a digit to be displayed, except that leading zeros and trailing zeros to the right of the radix point are displayed as spaces. If the number is too large to be displayed in the given number of "%"s and "#"s, then "∗"s are displayed in all "#" positions. The total number of "%"s and "#"s must not exceed 18.

4. "." and "," are used as radix points (decimal points). Only one of either may be in the *chars* array.

5. " \ " is displayed as a space.

6. Trailing characters that do not fit into the space available in dialog area –2 (which is 40 character positions wide) are truncated.

The format is valid for both displayed values when there are two.

Table 7-22

FORMAT EXAMPLES

Format			
Values	####.##	**+ #%#,%% m**	**+ #%.# feet**
1.234	1.23	+ 01,23 m	+ 1.2 feet
–536.8	–536.80	–536,80 m	–∗6.8 feet
8137.96	8137.96	+ ∗37,96 m	+ ∗8.0 feet

DEFAULTS

functions
as shipped — <0,0>
on power-up — <0,0>
if omitted — <0,0>

reals
as shipped — < > (empty array) for GIN functions;
<1.0> for format function
on power-up — < > (empty array) for GIN functions;
<1.0> for format function
if omitted — < > (empty array) for GIN functions;
<1.0> for format function

points
as shipped — < > (empty array)
on power-up — < > (empty array)
if omitted — < > (empty array)

chars
as shipped — < > (empty array) for GIN functions;
\ ### \ ##% for format function
on power-up — < > (empty array) for GIN functions;
\ ### \ ##% for format function
if omitted — < > (empty array for GIN functions;
\ ### \ ##% for format function

ERRORS

IO00	(Level 0):	Unrecognized command (the terminal is not a 4128 or 4129).
I011	(Level 2):	Invalid first element in the *functions* array.
IO11	(Level 2):	Invalid *functions* array.
IO12	(Level 3):	Out of memory while parsing parameter.
IO21	(Level 2):	Invalid *reals* array.
IO22	(Level 3):	Out of memory while parsing parameter.
IO31	(Level 2):	Invalid *points* array.
IO32	(Level 3):	Out of memory while parsing parameter.
IO41	(Level 2):	Invalid *chars* array.
IO42	(Level 3):	Out of memory while parsing parameter.

REFERENCES

ENABLE-GIN command

INQUIRY COMMAND

none

SET-GIN-RATES Command

HOST Syntax

$^{E}_{C}$**IU** **int-array:** *rate-table*

SETUP Syntax

$^{E}_{C}$**IU**$^{S}_{P}$ *rate-table*

PARAMETERS

rate-table
An int-array with 0 to 14 elements. Elements 1 and 2 are device-code and sub-device-code, respectively. Device-code specifies the GIN device (0 for thumbwheels, 8 for mouse, and 9 for joystick). Sub-device-code specifies which rate table of the GIN device is to be set (currently only one rate table exists for each device; therefore, 0 is the only valid entry). Elements 3 through 14 are GIN-space units in the range of 0 to 4095. There may be 0 to 12 table entries for the thumbwheels, 0 to 8 table entries for the mouse, and 0 to 2 table entries for the joystick.

DESCRIPTION

This command alters or initializes the rate-table for keyboard GIN devices. The rate-table determines how far the GIN-position is moved in GIN-space units (see by SET-GIN-AREA command) for a given movement of the GIN device.

There are two special cases for elements 3 through 14:

- If no values are entered, the rate-table is set to the default values.
- If some values are not entered, the rate-table is completed with the last specified entry. For example, the setup command $^{E}_{C}$**IU 0 0 25 50 100 200** sets the thumbwheel rate-table to 25 50 100 200 200 200 200 200 200 200 200 200.

Use With The Thumbwheels. As you turn the thumbwheels counts are accumulated by the terminal. The number of counts accumulated depends on how far the thumbwheels are turned between sampling by the terminal. A complete revolution of a thumbwheel causes 44 counts to be generated and the terminal samples the thumbwheel counts 22 times a second. The GIN position is moved a distance determined by the number of counts accumulated and the entries in the rate-table. If the count is one, the first value in the rate-table is used. If the count is four, the fourth entry in the rate-table is used, etc. Up and right thumbwheel movements cause positive count changes. Down and left cause negative changes. If the count is a negative, the negative of the respective rate-table entry is used. If more than twelve counts are accumulated, the 12th entry is used. Note that entries above 12 are not necessary because you can not turn the thumbwheels that fast.

Pressing the SHIFT key while moving the thumbwheels causes rate-table values to be ignored and the distance to move is equal to the number of counts accumulated.

4014 Terminal "Feel". If you want to approximate the feel of a 4014 terminal enter these values for a *rate-table*: 0 0 80 160 240 320 400 480 560 640 720 800 880 and 960.

Use With The Mouse. As with the thumbwheels, the terminal samples the mouse 22 times a second to determine the number of accumulated counts. Up and right are positive counts. Count values may range from –128 to + 127. These count values correspond to absolute velocities from 0 to ± 15.5 inches/second. Mouse rate table entries determine the distance in GIN space coordinates that the mouse GIN position is moved each sample.

The eight mouse rate table entries correspond to sample counts of 16, 32, 48, 64, 80, 96, 112, and 128. For example, if 48 counts accumulate in the sampling time, the third entry in the rate table is used. Count values other than 16, 32, 48, 64, 80, 96, 112, and 128 are linearly interpolated between the table entries.

Use With The Joystick. Joystick switch closures are translated into joystick GIN movement based on the joystick rate table entries. There are two joystick rate table entries, each is in GIN space coordinates per second. The first entry sets the unshifted joystick GIN speed and the second entry sets the shifted joystick GIN speed. The default rate table entry for unshifted joystick speed (2048), moves the cursor across the screen in two seconds (with the default GIN area and GIN window).

DEFAULTS

rate-table (for thumbwheels)
as shipped — 0 0 16 32 64 128 192 256 320 384 448 512 576 640
on power-up — 0 0 16 32 64 128 192 256 320 384 448 512 576 640
if omitted — 0 0 16 32 64 128 192 256 320 384 448 512 576 640

rate-table (for mouse)
as shipped — 8 0 75 150 225 300 375 450 525 600
on power-up — 8 0 75 150 225 300 375 450 525 600
if omitted — set thumbwheels

rate-table (for joystick)
as shipped — 9 0 2048 128
on power-up — 9 0 2048 128
if omitted — set thumbwheels

ERRORS

IU10 (Level 2): Device code or sub-device code not present (0, 0 for thumbwheels, 8,0 for mouse, 9,0 for joystick).

IU11 (Level 2): Invalid *rate-table* entries (array length must be 0 to 14 for thumbwheels, 1 to 10 for mouse, and 1 to 4 for joystick. Array elements 1 and 2 must be –32768 to 32767, array elements 3 and after must be 0 to 4095).

IU12 (Level 3): Out of memory while parsing array.

REFERENCES

SET-GIN-AREA command

INQUIRY COMMAND

none

SET-GIN-REPORT-FORMAT Command

4111, 4115P51, or 4120 SERIES

Host Syntax

```
EcIK   int:GIN-report-format
```

Setup Syntax

```
EcIK SP  GIN-report-format
```

PARAMETERS

GIN-report-format (0 to 7)
Selects the format of GIN-reports. Bit 0 controls pick report format. Bit 1 controls the number of pick reports per pick event. Bit 2 controls inclusion of view number in a GIN report (locator, pick, or stroke).

DESCRIPTION

This command specifies the format and number of reports returned to the host for each GIN event. See the *GIN-report* message type for details on report formats.

GIN-Pick-Report Format

If *GIN-report-format* is 0, 2, 4, or 6, each *GIN-pick-report* includes two *int-reports* for segment and pickID. If it is 1, 3, 5, or 7, each *GIN-pick-report* includes an *int-array-report* with segment-number/pickID pairs for the segment calling tree.

Number of GIN-Pick-Reports

If *GIN-report-format* is 0, 1, 4, or 5, one *GIN-pick-report* is sent for each pick event. If it is 2, 3, 6, or 7, one *GIN-pick-report* is sent for each pickable segment within the pick aperture per pick event, including a segment-number/pickID of 0,0 in the last report.

View Number

If *GIN-report-format* is 0, 1, 2, or 3, no GIN-report includes a view-number *int-report*. If it is 4, 5, 6, or 7, each type of GIN-report includes an *int-report* for the view-number after the *xy + -report*.

DEFAULTS

GIN-report-format
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IK11 (Level 2): Invalid *GIN-report-format.*

REFERENCES

GIN-report message type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GIN-RUBBERBANDING Command

Host Syntax

```
EcIR  int:device-function
      int:rubberbanding-mode
```

Setup Syntax

```
GINRUBBERBAND Sp  device-function
                  rubberbanding-mode
```

PARAMETERS

device-function
See the ENABLE-GIN command for details.

rubberbanding-mode (0, 1, or 2)
Specifies whether *rubberbanding-mode* is on or off, and if on, where rubberbanding starts.

0 turn rubberbanding off

1 turn rubberbanding on

2 turn rubberbanding on (start at *GIN-display-start-point*)

DESCRIPTION

This command turns rubberbanding on or off for all subsequent operations of the specified locator function. Rubberbanding is not allowed for the pick and stroke functions and for locator functions which have a 3D segment for a GIN cursor.

With rubberbanding turned on and the specified *device-function* enabled, a line is drawn from the most recently selected point to the current cursor position. The line is drawn in XOR mode, using the current line style and line index. If dashed lines are used, the pattern of dashes begins at the fixed end of the line.

If *rubberbanding-mode* is 2, the first GIN point after every GIN-ENABLE is "rubberbanded" from the *GIN-display-start-point*, as set by the SET-GIN-DISPLAY-START-POINT command. After the first point, it is identical to *rubberbanding-mode* 1. If *inking-mode*, as set by the SET-INKING-MODE command, is 1, and *rubberbanding-mode* is 2, then GIN inking behaves as if *inking-mode* was set to 2.

Rubberbanding is not allowed in key release mode.

If rubberbanding is turned on or off while the GIN device is enabled, only subsequent points are affected.

The beam position is not affected by rubberbanding.

When more than one device has rubberbanding enabled, and these devices are being used simultaneously, only one rubberband is active at one time — the one for the device last moved.

DEFAULT

device-function
as shipped — none
on power-up — none
if omitted — 0

rubberbanding-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IO11 (Level 2): Invalid *device-function* parameter. (See the *device-function* code table in the ENABLE-GIN command description for details.)

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.)

IR13 (Level 2): Context error. Rubberbanding is not allowed with locator functions which have a 3D segment as the GIN cursor for currently enabled GIN.

IR21 (Level 2): Invalid *rubberbanding-mode* (must be 0, 1, or 2; 0 or 1 if firmware is Version 3 or earlier).

REFERENCES

ENABLE-GIN command
SET-GIN-DISPLAY-START-POINT command
SET-GIN-INKING command
SET-LINE-INDEX command
SET-LINE-STYLE command

INQUIRY COMMAND

none

SET-GIN-STROKE-FILTERING Command

Host Syntax

```
EcIF   int:device-function
       int:distance-filter
       int:time-filter
```

Setup Syntax

```
TBFILTER Sp   device-function
              distance-filter
              time-filter
```

PARAMETERS

device-function
See ENABLE-GIN command for details.

distance-filter (0 to 4095)
Minimum change in 0 to 4095 GIN device x-coordinate or y-coordinate in order for the terminal to send another *GIN-stroke-report* to the host computer.

time-filter (0 to 32767)
Minimum time in milliseconds between stroke events.

DESCRIPTION

The specified stroke filtering parameters are applied to all subsequent operations of the specified stroke function.

Device-Function. The *device-function* code specifies the GIN device (which must be the tablet or mouse) and the GIN function (which must be stroke function). See the description of the ENABLE-GIN command for details.

Distance-Filter. The *distance-filter* parameter specifies the minimum distance the tablet pen or cursor must move in either x- or y-direction before a new point is output. This distance is expressed in 0 to 4095 GIN device coordinates; if either coordinate changes by more than this distance, then the terminal will send a new *GIN-stroke-report* to the host.

Time-Filter. The *time-filter* parameter specifies the minimum time in milliseconds that will elapse between successive points. As the operator moves the tablet pen or four-button cursor, *GIN-stroke-reports* are sent to the host only at intervals of "time" milliseconds.

The terminal measures time in increments of about 25 milliseconds. Thus, specifying a time of 10 ms is, for practical purposes, the same as specifying a time of zero. Again, for practical purposes, 27 ms is the same as 25 ms.

If both the *distance-filter* and *time-filter* parameters are nonzero, then all criteria of both parameters must be met for a stroke report to be sent to the host.

The first and last points of a stroke are not subject to stroke filtering.

Note that if either the distance or the time parameter is zero, then that type of filtering is absent. With no filtering, points are output at the maximum speed of the tablet interface.

Filtering does not affect cursor movement, but does affect the image formed by inking: inking occurs only between those points whose coordinates are reported to the host.

Locator and pick functions are not affected. Default is no filtering.

DEFAULTS

device-function
as shipped — none
on power-up — none
if omitted — error IF10

distance-filter
as shipped — 0
on power-up — 0
if omitted — 0

time-filter
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IO11 (Level 2): Invalid *device-function.* (See ENABLE-GIN.)

IF10 (Level 2): Stroke filtering not valid for specified *device-function.*

IF21 (Level 2): Invalid *distance-filter.*

IF31 (Level 2): Invalid *time-filter.*

REFERENCES

ENABLE-GIN command
GIN-report message type

INQUIRY COMMAND

none

SET-GIN-WINDOW Command

Host Syntax

$^{E}_{C}$**IW** xy:*first-corner* xy:*second-corner*

Setup Syntax

GINWINDOW $^{S}_{P}$ *first-corner* *second-corner*

PARAMETERS

first-corner (X = -2^{31} to $2^{31}-1$, Y = -2^{31} to $2^{31}-1$)
A corner of the window in terminal space into which a GIN device area is mapped.

second-corner (X = -2^{31} to $2^{31}-1$, Y = -2^{31} to $2^{31}-1$)
Another corner of the window in terminal space into which a GIN device area is mapped.

DESCRIPTION

This command sets a window in terminal space which can be used by the SET-GIN-AREA command. If SET-GIN-AREA specifies a *window-type* of –1, then the GIN area also specified by that command is mapped into the window in terminal space specified by the latest SET-GIN-WINDOW command.

The x- and y-coordinates you assign are sorted so that any two corners specify the rectangle.

DEFAULTS

first-corner
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

second-corner
as shipped — (4095,4095)
on power-up — (4095,4095)
if omitted — (0,0)

ERRORS

IW00 (Level 0): Unrecognized command; firmware is Version 3 or earlier.

IW11 (Level 2): Invalid *first-corner.*

IW21 (Level 2): Invalid *second-corner.*

REFERENCES

SET-GIN-AREA command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHICS-AREA-WRITING-MODE Command

Host Syntax

$^{E}_{C}$**MG** int:*writing-mode*

Setup Syntax

GAMODE $^{S}_{P}$ *writing-mode*

PARAMETERS

writing-mode (0 or 1)
Specifies the Writing mode for the graphics area. Setup mode parameters are **REPLACE** and **OVERSTRIKE**.

0 **REPLACE**

1 **OVERSTRIKE**

DESCRIPTION

This command sets the writing mode for alphatext which is not displayed in the dialog area. (The SET-DIALOG-AREA-WRITING-MODE command determines the writing mode for dialog area text.)

The GRAPHICS-AREA-WRITING-MODE command determines whether a character's background pixels are set to with the current *text-background-index*. In Replace mode (*writing-mode* = 0) the pixels are set; in Overstrike mode (*writing-mode* = 1) they are not set.

Interaction with SET-BACKGROUND-INDICES Command. The SET-GRAPHICS-AREA-WRITING-MODE and SET-BACKGROUND-INDICES commands both affect how alphatext is displayed in the graphic area. Thus, either of these commands can supersede the effect of the other.

Thus, if you set a terminal's graphic area writing mode to overstrike, the *text-background-index* of the *set-background-indices* command is set to –1. If you change the graphic area writing mode from overstrike to replace, then the *text-background-index* is set to –2.

Likewise, a SET-BACKGROUND-INDEX command which changes the *text-background-index* to –1 also changes the graphic area writing mode to overstrike. Changing the *text-background-index* to –2 also changes the graphic area writing mode to replace.

DEFAULTS

writing-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

MG11 (Level 2): Invalid *writing-mode.* (Must be 0 or 1; in Setup mode, must be OVERSTRIKE or REPLACE.)

REFERENCES

SET-BACKGROUND-INDICES command
SET-DIALOG-AREA-WRITING-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHTEXT-FONT Command

Host Syntax

$^{E}{}_{C}$**MF** int:*font-number*

Setup Syntax

$^{E}{}_{C}$**MF** $^{S}{}_{P}$ *font-number*

PARAMETERS

font-number (0 to 32767)
The character font that will be used to display subsequent graphtext.

DESCRIPTION

This command determines which character font will be used to display subsequent "stroke precision" graphtext. Graphtext is text occurring within a GRAPHIC-TEXT command; it can be used within graphic displays and picture segments, and (unlike alphatext) can be scaled, rotated, and slanted.

The default graphtext font is font zero, the standard ASCII font. If an optional keyboard is installed, then other predefined graphtext fonts are available; Table 7-23 lists them.

If the terminal has the Katakana keyboard, then fonts 10 and 11, as well as font 0, are predefined.

The user can define other graphtext fonts, numbered from 0 to 32767. (See *Graphtext-font-definition* syntactic construct.)

Table 7-23

PREDEFINED GRAPHTEXT FONTS

Font Number	Character Set
0	ASCII
1	Swedish
3	U.K.
7	APL
9	Danish/Norwegian
10	JIS Roman
11	JIS Katakana

User-defined graphtext characters supersede predefined ones. Thus, you can re-define some or all of the characters of font 0. (If you later delete these user-defined characters, they are superseded by the corresponding predefined characters.)

If you select a font which has no user-defined characters, then characters from the predefined font with the same font number are used. If there is no pre-defined font with that number, then characters from font zero (the ASCII font) are used.

Any undefined characters in a user-defined font default to the corresponding characters of the predefined font with the same font number. If there is no predefined font with that number, the corresponding characters of font zero are used.

DEFAULTS

font-number
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MF10 (Level 2): Font does not exist.

MF11 (Level 2): Invalid *font-number*.

REFERENCES

GRAPHIC-TEXT command
Graphtext-font-definition syntactic construct
SET-GRAPHTEXT-PRECISION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHTEXT-FONT-GRID Command

Host Syntax

$^{E}_{C}$SG int:*font-number* int:*grid-width*
int:*grid-height*

Setup Syntax

$^{E}_{C}$SG $^{S}_{P}$ *font-number grid-width*
grid-height

PARAMETERS

font-number (0 to 32767)
Names the graphtext font for which a font grid is being defined.

grid-width (1 to 4095)
Specifies the width of each grid unit.

grid-height (1 to 4095)
Specifies the height of each grid unit.

DESCRIPTION

The SET-GRAPHTEXT-FONT-GRID command creates a graphtext font and specifies the dimensions of the "grid" (character cell) used for characters in the font. Individual characters can be defined using the BEGIN-GRAPHTEXT-CHARACTER command.

The font grid is only used in scaling characters to fit the current graphtext size. Characters which are defined to exactly fit the font grid (see Figure 7-15) will be scaled up to the full current graphtext size.

This command must be given before any characters are defined in the specified font. (An error occurs if any user-defined characters exist in the specified font.)

Fonts and their associated grids can be deleted with the DELETE-GRAPHTEXT-CHARACTER command.

DEFAULTS

font-number
as shipped — none
on power-up — none
if omitted — 0

grid-width
as shipped — none
on power-up — none
if omitted — error SG21

grid-height
as shipped — none
on power-up — none
if omitted — error SG31

ERRORS

SG02 (Level 3): Out of memory while defining font grid.

SG10 (Level 2): Font already exists.

SG11 (Level 2): Invalid *font-number.*

SG21 (Level 2): Invalid *grid-width.*

SG31 (Level 2): Invalid *grid-height.*

REFERENCES

BEGIN-GRAPHTEXT-CHARACTER command
DELETE-GRAPHTEXT-CHARACTER command
GRAPHIC-TEXT command
Graphtext-font-definition syntactic construct
SET-GRAPHTEXT-SIZE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

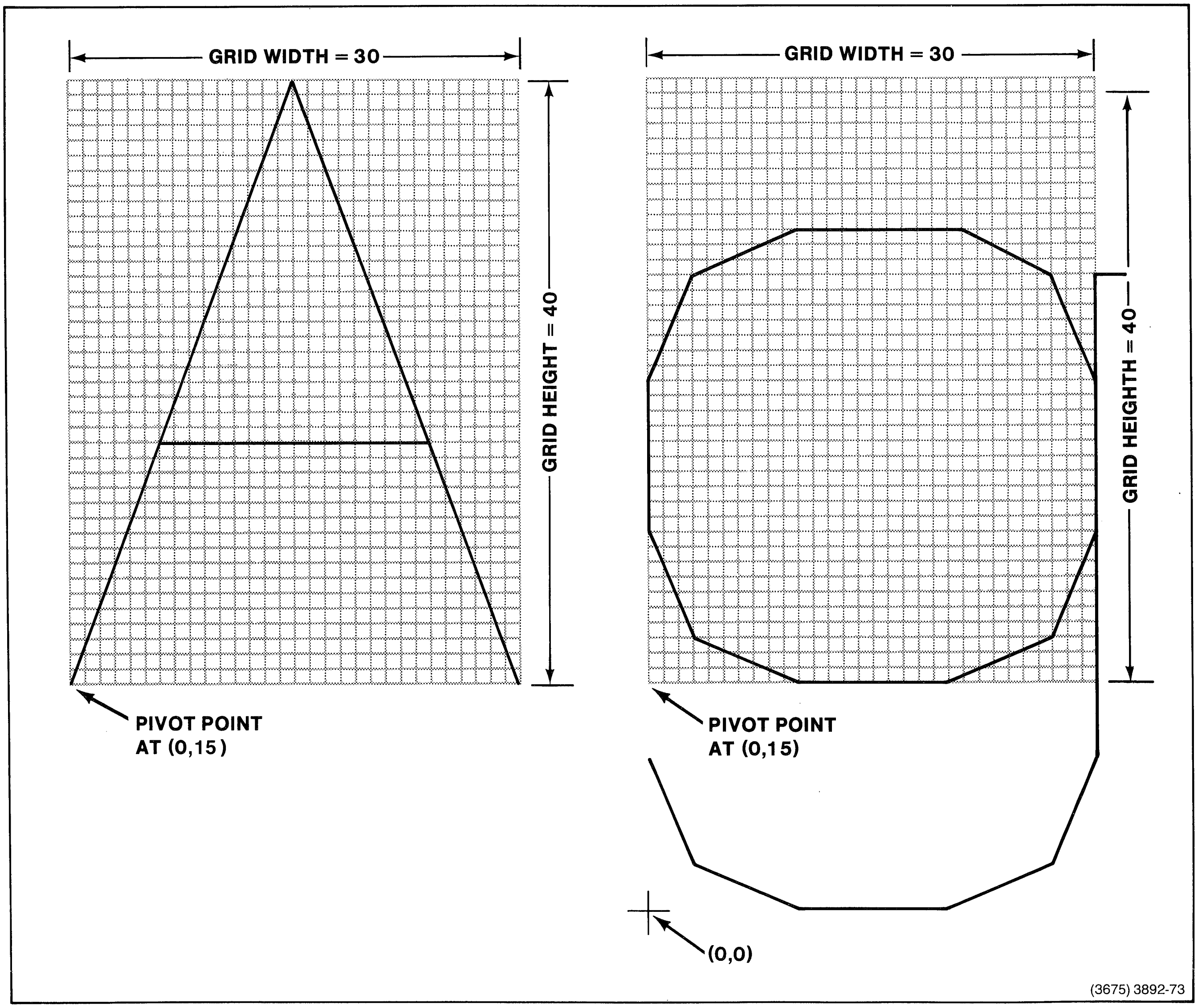


Figure 7-15. A Font Grid, With Two Characters Defined on That Grid.

SET-GRAPHTEXT-PRECISION Command

Host Syntax

```
EcMQ   int:precision
```

Setup Syntax

```
EcMQ Sp  precision
```

PARAMETERS

precision (1 or 2)
Specifies the precision type used for graphtext characters.

1 string precision

2 stroke precision

DESCRIPTION

This command specifies whether string precision or stroke precision is used to draw graphtext characters.

String precision (precision mode 1) specifies that graphtext is displayed exactly as is alphatext. See the DISPLAY-ALPHA-TEXT command for details.

Stroke precision (precision mode 2) specifies that graphtext is displayed according to the settings of the SET-GRAPHTEXT-FONT, SET-GRAPHTEXT-SIZE, SET-GRAPHTEXT-SLANT, and SET-GRAPHTEXT-ROTATION commands.

DEFAULTS

precision
as shipped — 2
on power-up — 2
if omitted — error MQ11

ERRORS

MQ11 (Level 2): Invalid *precision.*

REFERENCES

DISPLAY-ALPHA-TEXT command
GRAPHIC-TEXT command
SET-GRAPHTEXT-FONT command
SET-GRAPHTEXT-SIZE command
SET-GRAPHTEXT-SLANT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHTEXT-ROTATION Command

Host Syntax

$^{E}_{C}$**MR** real:*angle-in-degrees*

Setup Syntax

$^{E}_{C}$**MR** $^{S}_{P}$ *angle-in-degrees*

PARAMETERS

angle-in-degrees (-32767.0 to 32767.0)
The rotation angle in degrees, from the direction of the positive x-axis. Positive angles represent counterclockwise rotations, while negative angles represent clockwise rotations.

DESCRIPTION

Specifies the counterclockwise rotation angle (in degrees) for all subsequent graphtext strings displayed in "stroke precision" mode.

Default is 0.0 degrees.

DEFAULTS

angle-in-degrees
as shipped — 0.0
on power-up — 0.0
if omitted — 0.0

ERRORS

MR11 (Level 2): Invalid *angle-in-degrees*.

REFERENCES

GRAPHIC-TEXT command
SET-GRAPHTEXT-PRECISION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHTEXT-SIZE Command

Host Syntax

```
EcMC  int:cell-width   int:cell-height
      int:inter-character-spacing
```

Setup Syntax

```
EcMC Sp  cell-width   cell-height
         inter-character-spacing
```

PARAMETERS

cell-width (1 to $2^{31}-1$)
Width of a graphtext character cell.

cell-height (1 to $2^{31}-1$)
Height of a graphtext character cell.

inter-character-spacing (0 to $2^{31}-1$)
Spacing between adjacent character cells.

DESCRIPTION

This command specifies the size of the character cell for subsequent stroke-precision graphtext, in terminal space units.

When a stroke graphtext character is displayed, the lower left of the character font grid is mapped to the current beam position, and the upper right of the grid is mapped to the current beam position plus the (rotated and slanted) cell width and height. After the character is displayed, the beam position is moved in the (rotated) x-direction by the cell width plus inter-character-spacing.

The *terminal-settings-report* for this command uses *intc-reports*, whose size is governed by the most recent SET-COORDINATE-MODE command.

DEFAULTS

cell-width
as shipped — 39
on power-up — 39
if omitted — error MC11

cell-height
as shipped — 52
on power-up — 52
if omitted — error MC21

character-spacing
as shipped — 13
on power-up — 13
if omitted — 0

ERRORS

MC11 (Level 2): Invalid *cell-width.*

MC21 (Level 2): Invalid *cell-height.*

MC31 (Level 2): Invalid *character-spacing.*

REFERENCES

GRAPHIC-TEXT command
SET-GRAPHTEXT-FONT-GRID command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-GRAPHTEXT-SLANT Command

Host Syntax

EcMA real:*slant-angle*

Setup Syntax

EcMA Sp *slant-angle*

PARAMETERS

slant-angle (-32767.0 to 32767.0)
The angle each stroke-precision graphtext character is slanted, in degrees clockwise from the vertical.

DESCRIPTION

This command specifies the slant each stroke precision Graphtext character has from the vertical. The *slant-angle* parameter specifies the slant in degrees clockwise from the vertical. If you specify a negative slant, the character is slanted counter-clockwise.

DEFAULTS

slant-angle
as shipped — 0.0
on power-up — 0.0
if omitted — 0.0

ERRORS

MA11 (Level 2): Invalid *slant-angle.*

REFERENCES

GRAPHIC-TEXT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-IMAGE-ORIENTATION Command

4111 or OPTION 19

Host Syntax

EcQO int:*orientation*

Setup Syntax

HCORIENT Sp *orientation*

PARAMETERS

orientation (0 to 3)
Selects the orientation of the hardcopy image with respect to the hardcopy media. Setup mode parameters are **HORIZONTAL**,**VBOTTOM**, **VCENTER**, and **VTOP**.

0 **HORIZONTAL**: long axis of image on long axis of media

1 **VBOTTOM**: long axis of image on short axis of media, at bottom

2 **VCENTER**: long axis of image on short axis of media, centered

3 **VTOP**: long axis of image on short axis of media, at top

DESCRIPTION

The SET-IMAGE-ORIENTATION command sets the orientation of the hardcopy image with respect to the hardcopy media (the paper or transparency on the hardcopy machine). Figure 7-16 shows the result of assigning orientations **0** through **3**.

The orientation of an image becomes apparent when the HARDCOPY command, the HARD COPY key, or the 4010-HARDCOPY command is issued after a SELECT-HARDCOPY-INTERFACE command or COPY or SPOOL to HC: has chosen the color hardcopy interface.

The maximum image sizes for the different orientations are:

	Horizontal			Vertical		
	X		Y	X		Y
4691A size	1559	×	1199	1199	×	1559
4691B size	2459	×	1559	1559	×	2459
4692	1536	×	1152	1152	×	1536
4695width 0	32767	×	960	960	×	32767
4695width 1	32767	×	1024	1024	×	32767

With the 4695, all orientations start the copy at the current paper position.

DEFAULTS

orientation
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

QO00 (Level 0): Unrecognized command. (Not a 4111 or Option 19 is not installed.)

QO11 (Level 2): Invalid *orientation*.

REFERENCES

HARDCOPY command
HARD COPY key
SELECT-HARDCOPY-INTERFACE command
4010-HARDCOPY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

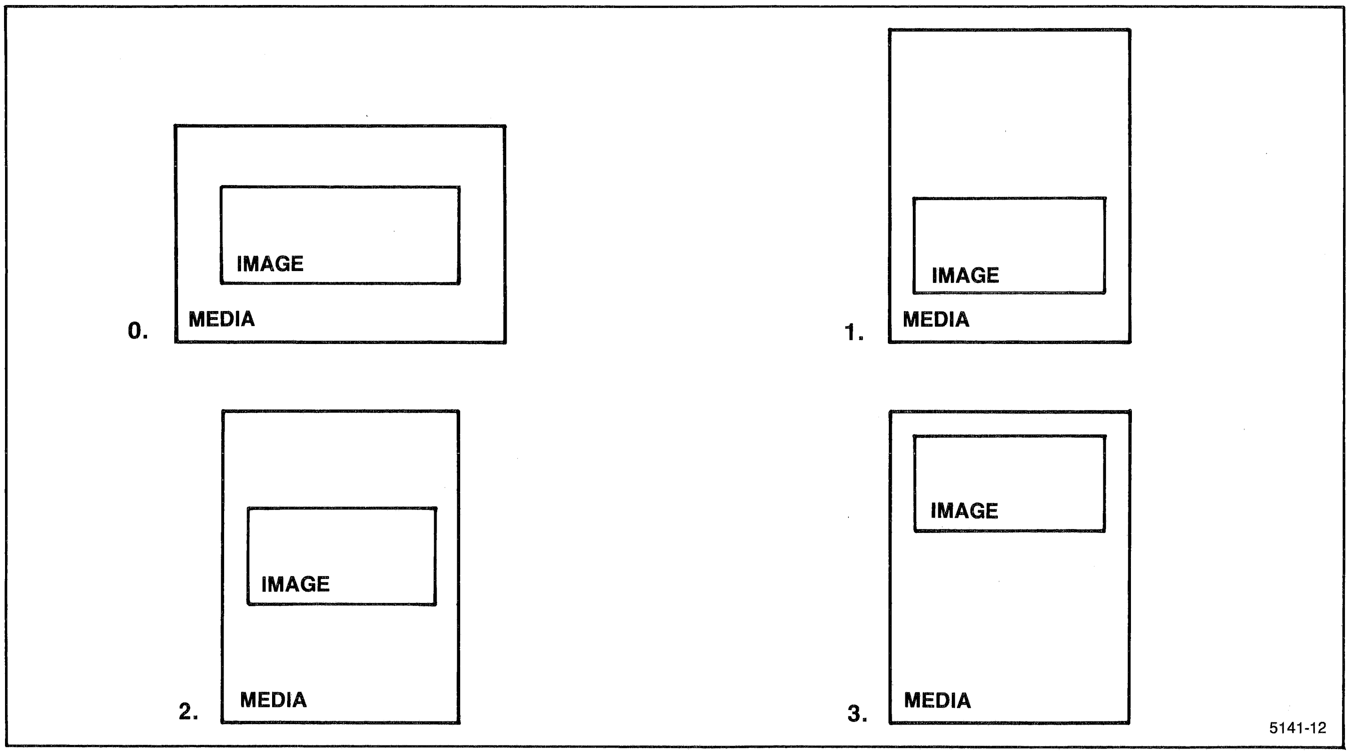


Figure 7-16. Orientation of Copier Images to Media.

SET-KEY-EXECUTE-CHARACTER Command

Host Syntax

$^{E}_{C}$**KY** int:*key-execute-char*

Setup Syntax

KEYEXCHAR $^{S}_{P}$ *key-execute-char*

PARAMETERS

key-execute-char (0 to 127)
Numeric equivalent of the ASCII character which delimits the "execute locally" part of a macro definition.

DESCRIPTION

This command sets the value of the *key-execute-character*, used with the DEFINE-MACRO command.

Normally, when the operator presses a key which has been programmed (with the DEFINE-MACRO command), the characters programmed into the key are sent to the host computer, just as if the operator had typed those characters manually. This includes characters which comprise an "escape sequence" command for the terminal; the terminal, instead of executing such a command, sends the characters which comprise it to the host.

The *key-execute-character* provides a way around this problem. Within a key definition, the *key-execute-character* marks the beginning and end of a sequence of characters which the terminal is to execute locally rather than send to the host.

The *key-execute-character* has this special effect only when the macro containing it is invoked by pressing a key. If, instead, the macro is invoked with an EXPAND-MACRO command, then the *key-execute-character* is treated like any other character in the macro definition.

When a key-macro is expanded by the operator pressing the associated key, the contents of the macro are initially routed to the host computer. When a *key-execute-character* is encountered in the macro, it is discarded and the macro contents are routed to the terminal. Each subsequent *key-execute-character* encountered "toggles" the destination of the macro contents (from terminal to host, or from host to terminal) and is discarded (not displayed or transmitted).

DEFAULTS

key-execute-char
as shipped — 16
on power-up — remembered
if omitted — 0

ERRORS

KY11 (Level 2): Invalid *key-execute-char.*

REFERENCES

DEFINE-MACRO command
EXPAND-MACRO command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-LIGHTSOURCE Command

4128, 4129

Host Syntax

$^{E}{}_{C}$TL int-array: *source-id*
int-array: *views-switch*
xy + -array: *direction*
real-array: *intensity*

Setup Syntax

$^{E}{}_{C}$TL *source-id* *views-switch* *direction* *intensity*

PARAMETERS

source-id (length 1 or 2)
Specifies lightsource number and coordinate system of *direction.*

first element: *lightsource-number* (-1 to 16)

-1	"all" previously specified lightsources except ambient light
0	ambient light
1 to **16**	individual lightsources

second element: *coordinate-system* (0, 1, or 2)

0 or **not present**	no change
1	terminal (XYZ) space coordinate system
2	viewing (UVW) space coordinate system

views-switch (length 0 or 2 to 32768)
Specifies whether lightsources 1 to 16 are on or off, and in which views. If length is 0, there is no change.

first element: *on-off* (0 or 1)

0	turn lightsource OFF
1	turn lightsource ON

second element: *view-1* (-1 to 64)

-1	all views
0	the current view
1 to **64**	a specific view

succeeding elements: *view-n* (1 to 64)

1 to **64**	a specific view

direction (length 0 or 1)
Specifies direction of the lightsources 1 to 16.

first element (xyz = -2^{23} to 2^{23}-1)

not present	no change
present	vector pointing at infinitely distant lightsource. (0,0,0) is not valid for lightsource numbers 1 to 16.

intensity (length 0 or 1)
Specifies the intensity of the lightsource.

first element (–1.0 to 1.0)

not present	no change
present	a positive intensity adds light to a surface; a negative intensity subtracts light from a surface

DESCRIPTION

This command controls terminal lightsources. There are 16 "directed" lightsources and a separate "ambient" lightsource for each view. This command sets intensity, direction, and view-on/off switches for the directed lightsources and the intensity of the ambient lightsource for the current view. A view does not have to exist to be specified by the *views-switch* parameter.

If the first element of *source-id* is 0, the intensity of ambient light in the specified views is set. Different views can have different intensities of ambient light. Since there is no "direction" of ambient light, the second element of *source-id* and the *direction* parameter are ignored.

If the first element of *source-id* is 1 to 16, the intensity and direction of the specified lightsource is set for all views and the lightsource is turned on or off in the specified views. The direction of the lightsource is set according to the *direction* parameter in either XYZ terminal space or UVW viewing space according to the second element of *source-id.*

See Appendix G for details on how lightsources are used in the Local Lighting Model.

The *coordinate-system*, *view-switch*, and *direction* parameters are not used for ambient light although they are checked for valid values.

DEFAULTS

source-id
as shipped — <0,1>
on power-up — <0,1>
if omitted — no change

views-switch
as shipped — < >
on power-up — < >
if omitted — no change

direction
as shipped — < (0,0,1) >
on power-up — < (0,0,1) >
if omitted — no change

intensity
as shipped — <1>
on power-up — <1>
if omitted — no change

ERRORS

TL00 (Level 0): Unrecognized command. (Terminal is not a 4128 or 4129.)

TL02 (Level 3): Insufficient memory to save the light-source parameters.

TL03 (Level 2): Command is invalid in coordinate mode 0 or 1.

TL11 (Level 2): Invalid *source-id.*

TL12 (Level 3): Out of memory while parsing *source-id* array.

TL21 (Level 2): Invalid *views-switch.*

TL22 (Level 3): Out of memory while parsing *views-switch* array.

TL31 (Level 2): Invalid *direction.*

TL32 (Level 3): Out of memory while parsing *direction* array.

TL41 (Level 2): Invalid *intensity* array.

TL42 (Level 3): Out of memory while parsing *intensity* array.

REFERENCES

Appendix G, Shaded Surfaces

INQUIRY COMMAND

none

SET-LINE-INDEX Command

Host Syntax

ᴱcML int:*line-index*

Setup Syntax

ᴱcML ˢᴘ *line-index*

PARAMETERS

line-index (0 to 32767)
Specifies the color index in which subsequent lines are drawn.

DESCRIPTION

This command specifies the color index with which subsequent lines, panel boundaries, markers, and facet edges are to be drawn. Line index is a graphic primitive attribute.

There is a maximum color index for any particular surface — one less than 2^M, where M is the number of bit planes assigned to that surface. If the color index in effect when a line is drawn is greater than the maximum index for the surface, the line is drawn using the maximum index.

The actual color for each index depends on the current color mixture for the particular color index. For details, see the descriptions of the SET-SURFACE-GRAY-LEVELS and SET-SURFACE-COLOR-MAP commands.

DEFAULTS

line-index
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

ML11 (Level 2): Invalid *line-index*.

REFERENCES

BEGIN-PANEL-BOUNDARY command
DRAW command
DRAW-FACETS command
DRAW-MARKER command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-GRAY-LEVELS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-LINE-STYLE Command

Host Syntax

$^{E}{}_{C}$**MV** int:*line-style*

Setup Syntax

$^{E}{}_{C}$**MV** $^{S}{}_{P}$ *line-style*

PARAMETERS

line-style (0 to 7)
Specifies the line style for vectors drawn on the screen.

0	solid
1 to 7	a non-solid dash pattern

DESCRIPTION

This command selects one of eight different line styles (dash patterns) for subsequent vectors.

This command also resets the dashed line generator to the beginning of the selected line style pattern.

A line's style is a graphic primitive attribute.

On a 4128 and 4129, when a view whose object-surface-display attribute (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is "hidden-line" or "shaded-surface" is renewed, all vectors and facet edges are displayed as a solid line (line style 0) regardless of the primitive's line style.

Figure 7-17 shows the eight line styles.

DEFAULTS

line-style
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MV11 (Level 2): Invalid *line-style*.

REFERENCES

DRAW command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

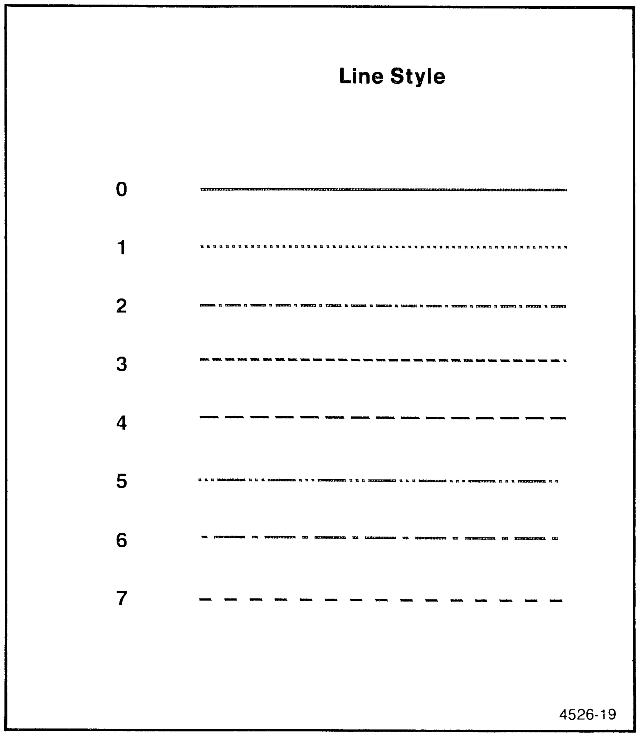


Figure 7-17. Line Styles.

SET-MARKER-TYPE Command

Host Syntax

```
EcMM   int:marker-number
```

Setup Syntax

```
EcMM Sp   marker-number
```

PARAMETERS

marker-number (0 to 10)
Specifies which marker type is used when the terminal draws a marker. See Figure 7-18 for the list of marker types.

DESCRIPTION

This command specifies which marker type appears when the terminal draws a marker. Marker type is a graphic primitive attribute.

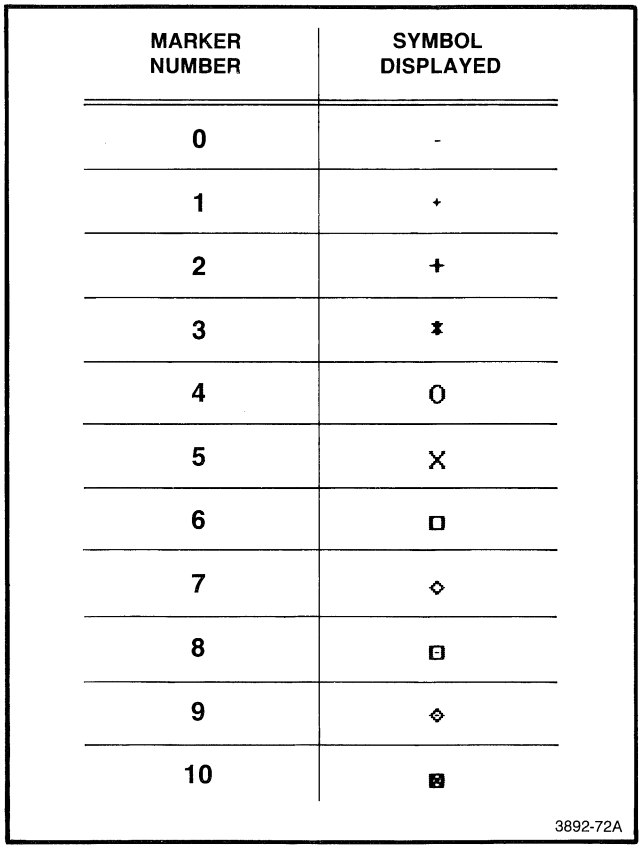
Markers are drawn approximately the same size as similar characters in the default alphatext size.

DEFAULTS

marker-number
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MM11 (Level 2): Invalid *marker number*.



MARKER NUMBER	SYMBOL DISPLAYED
0	-
1	+
2	+
3	✱
4	O
5	X
6	□
7	◇
8	⊡
9	◈
10	▣

3892-72A

Figure 7-18. Appearance of Markers.

REFERENCES

DRAW-MARKER command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-NUMBER-OF-COPIES Command

4111 or OPTION 19

Host Syntax

$^{E}_{C}$**QN** int:*number-of-copies*

Setup Syntax

HCCOPIES $^{S}_{P}$ *number-of-copies*

PARAMETERS

number-of-copies (0 to 64)
Selects the number of copies that are produced when a color hardcopy operation is invoked.

DESCRIPTION

The SET-NUMBER-OF-COPIES command sets the number of copies that are produced each time a HARDCOPY or 4010-HARDCOPY command is received or the HARD COPY key is pressed, when the color copier interface has been selected with the SELECT-HARDCOPY-INTERFACE command.

If you assign 0 as the parameter, one copy is produced. If you assign a positive number greater than 64, the *number-of-copies* parameter is set to 64.

The *number-of-copies* parameter is decremented after each copy; the final value of the parameter is 1. If you cancel the copy operation before all the copies are made, the *number-of-copies* parameter retains its value.

This command sets the number of hardcopies that are produced only for the 4111 or for terminals with the Option 19 color hardcopy interface installed.

DEFAULTS

number-of-copies
as shipped — 1
on power-up — 1
if omitted — 1

ERRORS

QN00 (Level 0): Unrecognized command. (Not a 4111 or Option 19 is not installed.)

QN11 (Level 2): Invalid *number-of-copies* parameter.

REFERENCES

HARDCOPY command
HARD COPY key
SELECT-HARDCOPY-INTERFACE command
4010-HARDCOPY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-OBJECT-SURFACE-COLOR-RANGE Command

4128, 4129

Host Syntax

$^{E}{}_{C}$TR int: *object-surface-color*
int-array: *indices*
real-array: *reflectivity-coefficients*

Setup Syntax

$^{E}{}_{C}$TR *object-surface-color* *indices*
reflectivity-coefficients

PARAMETERS

object-surface-color (1 to 255)
Specifies the object-surface-color range being defined.

indices (length 0 or 2)
Specifies the start-index and end-index for the range.

first element: *start-index* (0 to 255)

not present	no change if currently defined, 1 otherwise

second element: *end-index* (0 to 255)

not present	no change if currently defined, 255 otherwise

reflectivity-coefficients (length 0, 1, or 2)
Specifies the ambient and diffuse reflectivity.

first element: *ambient-reflectivity* (0.0 to 1.0)

not present	no change if currently defined, 1.0 otherwise

second element: *diffuse-reflectivity* (0.0 to 1.0)

not present	no change if currently defined, 1.0 otherwise

DESCRIPTION

This command defines an object-surface-color range. Start-index is the smaller of the two elements of indices and end-index is the larger. (If the two elements are the same, so are start-index and end-index.)

If both *indices* and *reflectivity-coefficients* have length 0, the range is deleted.

See the Local Lighting Model section of Appendix G for details on how an object-surface-color-range is used.

DEFAULTS

object-surface-color
as shipped — none
on power-up — none
if omitted — 0 (error TR11)

indices
as shipped — none
on power-up — none
if omitted — no change if currently set, otherwise <0,255>

reflectivity-coefficients
as shipped — none
on power-up — none
if omitted — no change if currently set, otherwise <1,0>

ERRORS

TR00 (Level 0): Unrecognized command. (Terminal is not a 4128 or 4129.)

TR11 (Level 2): Invalid *object-surface-color*.

TR21 (Level 2): Invalid *indices* array.

TR22 (Level 3): Out of memory while parsing *indices* array.

TR31 (Level 2): Invalid *reflectivity-coefficients* array.

TR32 (Level 3): Out of memory while parsing *reflectivity-coefficients* array.

REFERENCES

Appendix G, Shaded Surfaces
SELECT-OBJECT-SURFACE-COLOR command

INQUIRY COMMAND

none

SET-OVERVIEW-WINDOW Command

Host Syntax

$^{E}_{C}$UW xy:*first-corner*
xy:*second-corner*

Setup Syntax

$^{E}_{C}$UW *first-corner*
second-corner

PARAMETERS

first-corner (X = -2^{31} to $2^{31}-1$, Y = -2^{31} to $2^{31}-1$)
Specifies the first corner of the overview window.

second-corner (X = -2^{31} to $2^{31}-1$, Y = -2^{31} to $2^{31}-1$)
Specifies the second corner of the overview window.

DESCRIPTION

This command specifies the full overview window and partial overview window for the current view. These two windows are used when the OVERVIEW and CTRL-OVERVIEW keys are pressed. It also defines the alphatext window, and the home position for the current view. The x- and y-coordinates you specify are sorted so as to give the lower-left and upper-right corners of the full overview window.

When the OVERVIEW key is pressed, the window is set to the most recent partial overview window set by this command in this view and the view is renewed. When the CTRL-OVERVIEW key is pressed, the window is set to the most recent full overview window set by this command in this view and the view is renewed.

Partial Overview Window. The partial overview window is the bottom four-fifths of the full overview window set by this command. See the *4110/4120/4120 Series Host Programmers Manual* for details on the interaction between the partial overview window and the full overview window.

Home Position. The home position is the beam position that results when the PAGE command is executed or the PAGE key is pressed when the dialog area is disabled. The home position is in the upper-left corner of the partial overview window.

The home position (HX,HY) is calculated as

HX = the left edge of the full overview window
HY = $\text{YTOP} - (27/1280) \bullet (M + 1) - 1$ if $M \geqslant 1280$
$\text{YTOP} - (27/1279) \bullet M$ if $M < 1280$

where

YTOP = the top of the partial overview window
M = the height of the full overview window

Alphatext Window. The alphatext window has the same right, left, and bottom edges as the full overview window, but its top is the home position's y-coordinate. Alphatext cannot be displayed outside of the alphatext window. If the beam position is outside the alphatext window when a $^{U}{}_{S}$ character is received, it is moved to the closest point on the edge of the alphatext window.

Alphatext is printed from left to right and the lines are filled from top to bottom. When alphatext reaches the right edge of the alphatext window, it wraps to the left edge of the following line. When the alphatext reaches the bottom of the alphatext window, it wraps to the left edge of the top of the alphatext window.

The home position and alphatext window are automatically set when the terminal receives this command.

Local Viewing Function Box Window. The Local Viewing function box is limited to the overview window. When the Local Viewing function box is first made visible it is the same size as the current window, but as you move the Local Viewing function box, it cannot go outside the overview window.

DEFAULTS

first-corner
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

second-corner
as shipped — (4095,4095)
on power-up — (4095,4095)
if omitted — (4095,4095)

ERRORS

UW11 (Level 2): Invalid *first-corner* parameter.

UW21 (Level 2): Invalid *second-corner* parameter.

REFERENCES

DISPLAY-ALPHA-TEXT command
OVERVIEW Key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PAGE-FULL-ACTION Command

Host Syntax

```
EcKP   int:page-full-action
```

Setup Syntax

```
PAGEFULL Sp   page-full-action
```

PARAMETERS

page-full-action (0 to 7)
Sets the page-full action, as follows:

0	No action. (Setup mode keyword: **NONE**)
1	Wait for key push. (Setup mode keyword: **STOP**)
2	Perform hardcopy, then page. (Setup mode keyword: **AUTOCOPY**)
3	Hardcopy, then wait for key push, then page. (Setup mode keyword: **3**)
4	Send break signal to host. (Setup mode keyword: **BREAK**)
5	Send break, then wait for key push. (Setup mode keyword: **5**)
6	Send break, then do hardcopy and page. (Setup mode keyword: **6**)
7	Send break, then do hardcopy then wait for key push, then page. (Setup mode keyword: **7**)

DESCRIPTION

This command specifies the action (if any) which the terminal performs when a page-full condition occurs. A page-full condition occurs when the dialog area is disabled and the alpha cursor "wraps" off the bottom of the last margin in the graphics area. "Wrapping" occurs when the cursor is on the bottom line and the terminal receives a linefeed, or when the cursor is at the end of the bottom line and the terminal receives a printable character. The terminal has only one margin, as set by the SET-MARGINS command.

If the parameter is 0, there is no page-full action.

If the parameter is 1, 3, 5, or 7, output to the display is stopped until a key is pressed.

If the parameter is 2, 3, 6, or 7, when a page-full condition occurs, an automatic hard copy is generated. If int = 2 or 6, then the screen is erased after the hard copy. If int = 3 or 7, the screen is erased after the hard copy and after a key is pressed.

If the parameter is 4, 5, 6, or 7, a "break" is sent to the host, just as if the operator had pressed the BREAK key.

Once set in a given mode, the page-full-action continues in that mode until it is specifically changed with another SET-PAGE-FULL-ACTION command.

DEFAULTS

page-full-action
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

KP11 (Level 2): Invalid *page-full-action.*

REFERENCES

HARDCOPY command
PAGE command
SET-MARGINS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PANEL-FILLING-MODE Command

Host Syntax

$^{E}{}_{C}$**MS** int:*overstrike/replace*
int:*cover-boundary*
int:*pattern-keying-mode*

Setup Syntax

$^{E}{}_{C}$**MS** $^{S}{}_{P}$ *overstrike/replace*
cover-boundary
pattern-keying-mode

PARAMETERS

overstrike/replace (0 or 1)
Specifies Overstrike or Replace mode.

0	Replace mode
1	Overstrike mode

cover-boundary (0 or 1)
Specifies whether the boundary of the panel is covered when the panel is filled.

0	no
1	yes

pattern-keying-mode (0 to 3)
Specifies the position (direction) of the fill-pattern within the panel.

0	no change to keying
1	keyed to viewport position
2	keyed to lower left panel pixel
3	keyed to absolute screen position

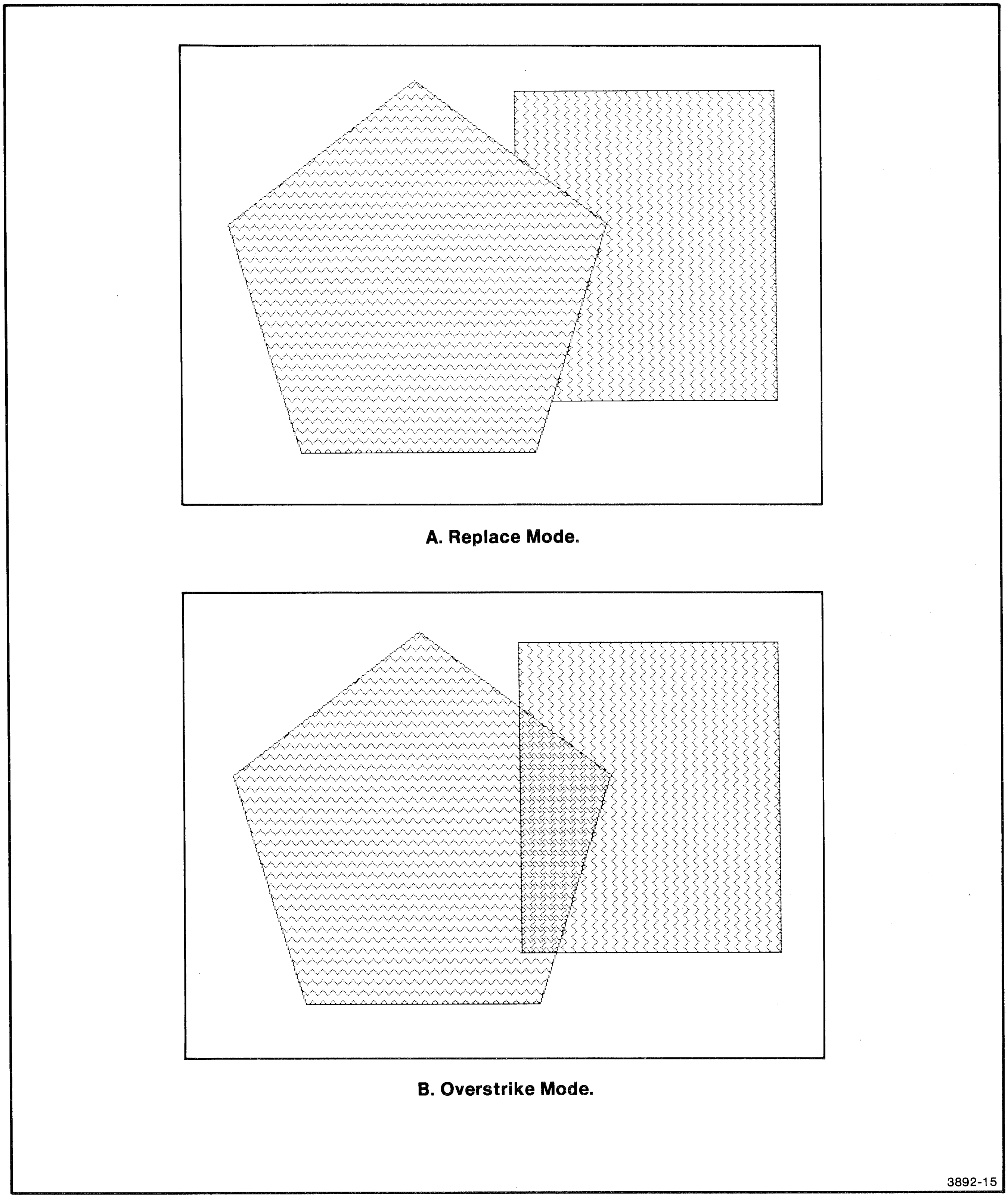
DESCRIPTION

This command determines how subsequent panels are to be filled.

Overstrike/Replace. The *overstrike/replace* parameter determines the effect of zeroes in the fill pattern. If this parameter is one ("overstrike"), then when subsequent panels are filled, pixels in the panel interior which correspond to zeroes in the fill pattern are left unchanged. If, however, the overstrike/replace parameter is zero ("replace"), then when subsequent panels are filled, zeroes in the fill pattern cause the corresponding pixels in the panel interior to be set to color index 0. Figure 7-19 shows the effect of the overstrike/replace parameter.

Cover-Boundary. The *cover-boundary* parameter determines whether, when panels are filled, their boundaries are filled as well as their interiors. If this parameter is zero, only panel interiors will be filled; if one, then the interiors are filled and the boundaries are covered.

If a BEGIN-PANEL-BOUNDARY command specifies that the panel boundary is to be drawn, but the SET-PANEL-FILLING-MODE command specifies that the filling operation is to include the boundary as well as the interior, then the "covering the boundary" supersedes "drawing the boundary." That is, the boundary is first drawn, and then later (when an END-PANEL command is received), it is covered over with the fill pattern.



A. Replace Mode.

B. Overstrike Mode.

Figure 7-19. Effect of Overstrike/Replace Parameter.

Pattern Keying. The *pattern-keying-mode* parameter determines how the fill pattern itself is positioned within a panel being filled.

If this parameter is 0, the present pattern keying mode is not changed.

If this parameter is 1, the fill pattern is "keyed to the current viewport." Figure 7-20 shows the concept. The fill pattern to be used is replicated over the entire viewport, starting at the lower left corner of the viewport. Then that part of the (replicated) fill pattern which falls within the panel to be filled is used to fill the panel.

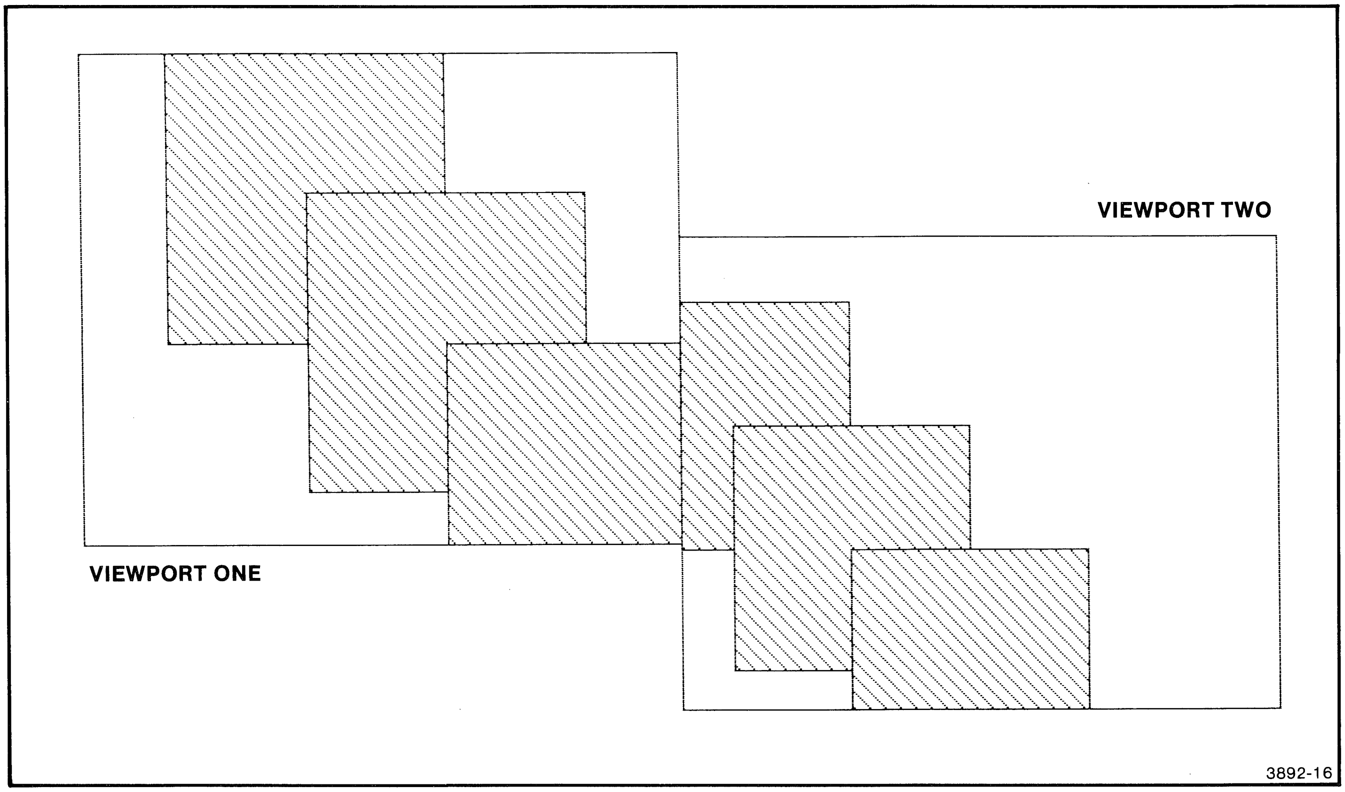


Figure 7-20. Keying a Fill Pattern to the Current Viewport.

If the *pattern-keying-mode* parameter is 2, then the fill pattern is keyed to the leftmost pixel in the bottom row of pixels in the panel being filled. The process is the same as described above, except that when replicating the fill pattern, the terminal starts at the lower left corner of the panel rather than the lower left corner of the current viewport. Figure 7-21 shows the result.

If the parameter is 3, then the fill pattern is keyed to the lower left corner of the screen. The process is as before, except that, when replicating the fill pattern, the terminal starts at the lower left corner of raster memory space (rather than the lower left corner of the viewport or the leftmost pixel in the bottom line of the panel being filled). Figure 7-22 shows the effect.

When keying to the viewport or to the screen, panels join each other smoothly, with no discernible interruption in the fill pattern. When keying to the panel being filled, however, each panel is filled starting at the leftmost pixel of that panel's bottom line, so there will usually be a "jump" or interruption in the fill pattern at the boundaries between adjacent panels.

DEFAULTS

overstrike/replace
as shipped — 0
on power-up — 0
if omitted — 0

cover-boundary
as shipped — 0
on power-up — 0
if omitted — 0

pattern-keying-mode
as shipped — 1
on power-up — 1
if omitted — 0

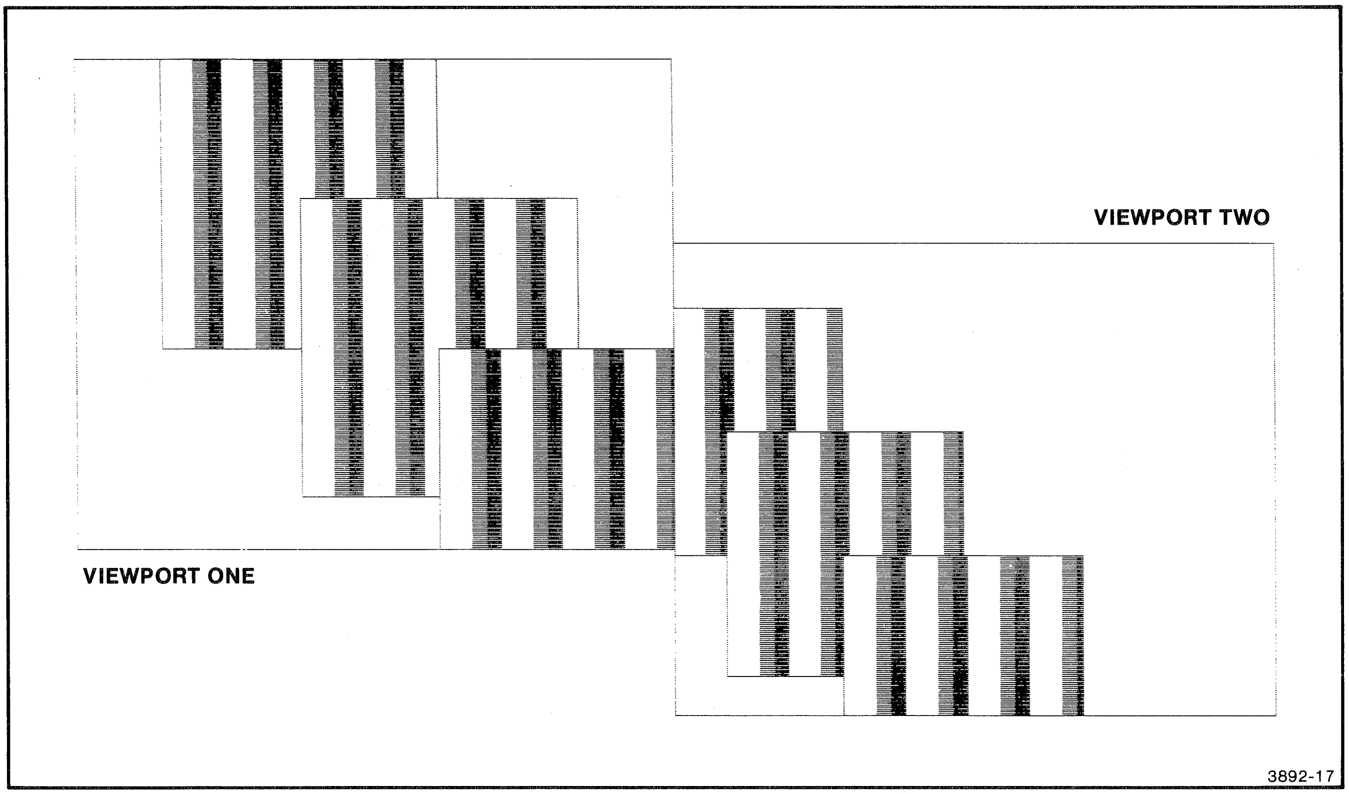


Figure 7-21. Keying a Pattern to the Panel Being Filled.

ERRORS

MS11 (Level 2): Invalid *overstrike/replace* parameter.

MS21 (Level 2): Invalid *cover-boundary* parameter.

MS31 (Level 2): Invalid *pattern-keying-mode.*

REFERENCES

Panel-definition syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

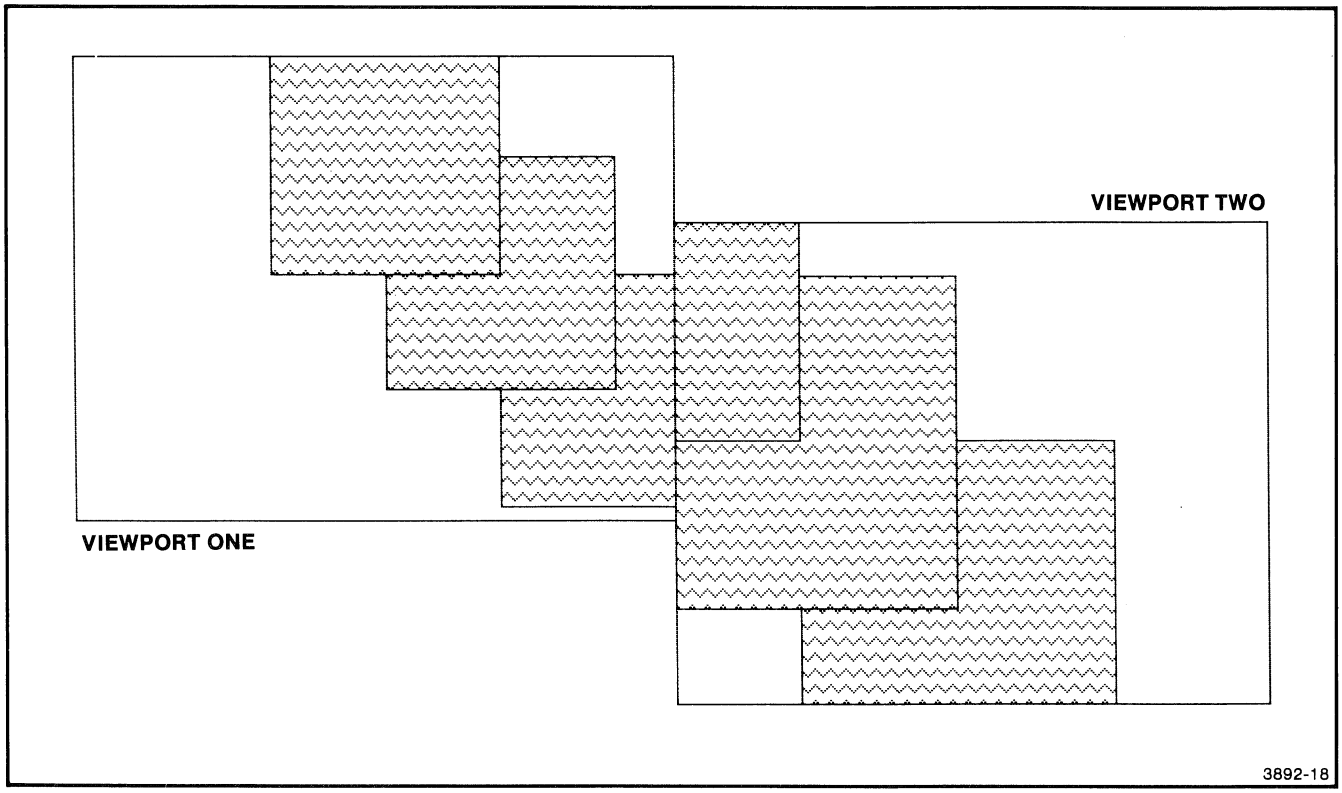


Figure 7-22. Keying a Fill Pattern to the Screen.

SET-PARITY Command

Host Syntax

$^{E}{}_{C}$**NP** int:*parity-mode*

Setup Syntax

PARITY $^{S}{}_{P}$ *parity-mode*

PARAMETERS

parity-mode (0 to 4)
Specifies the parity mode the terminal uses in sending data to the host. Setup mode parameters are **NONE**, **ODD**,**EVEN**, **HIGH**, and **DATA**.

0 **NONE**; the terminal ignores parity in character it receives from the host, and sets the parity bit to zero in characters it sends to the host.

1 **ODD**; the terminal ignores parity in characters it receives, but uses odd parity in characters it transmits to the host.

2 **EVEN**; the terminal ignores parity in characters it receives, but uses even parity in characters it transmits.

3 **HIGH**; the terminal ignores parity in characters it receives, and sets the parity bit to one in characters it transmits.

4 **DATA**; the terminal treats the eighth bit in each character as another data bit.

DESCRIPTION

Places the terminal in the specified parity mode.

This command determines how the parity bit is set in characters which the terminal sends to the host. (The terminal ignores the parity bit in characters it receives from the host except if the *parity-mode* is 4.)

Parity NONE. If mode 0 (NONE) is selected, the terminal sets the parity bit to zero in characters it sends to the host.

ODD Parity. If mode 1 (ODD) is selected, then when the terminal sends a character to the host, it sets the parity bit in that character to one or zero, whichever is required so that the character will have an odd number of bits set to one.

EVEN Parity. If mode 2 (EVEN) is selected, then the terminal sets the parity bit to one or zero, whichever is required so that the character will have even parity.

HIGH Parity. If mode 3 (HIGH) is selected, then the terminal, when sending a character to the host, sets the parity bit to one.

DATA Parity. If mode 4 (DATA) is selected, then the terminal uses the eighth bit in each character as another data bit. Using this *parity-mode* setting with the high bit (parity bit) set to zero is the same as 7-bit ASCII.

DEFAULTS

parity-mode
as shipped — 0
on power-up — remembered
if omitted — 0

ERRORS

NP11 (Level 2): Invalid *parity-mode.*

REFERENCES

SET-PORT-PARITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PICK-APERTURE Command

Host Syntax

$^{E}_{C}$**IA** int:*aperture-width*

Setup Syntax

GINPICKAPERTURE $^{S}_{P}$ *aperture-width*

PARAMETERS

aperture-width (0 to 4095)
Specifies the width of the pick aperture.

DESCRIPTION

This command sets the size of the pick aperture.

The pick aperture is a square in normalized screen space, centered on the current beam position. During a GIN pick operation part of the segment picked must fall within the pick aperture. (See ENABLE-GIN command for details.)

The *aperture-width* parameter sets the width of the pick aperture square, in normalized screen coordinate units. If the width is zero, then part of the segment being picked must fall exactly at the current cursor position. If the width is 4, then part of the segment being picked must have its x-coordinate in the range from X0 -2 to X0 +2, and its y-coordinate in the range from Y0 -2 to Y0 +2, where (X0,Y0) is the current beam position in normalize screen space.

The specified size of the pick aperture is an approximate number. Because of the rounding that occurs when transforming coordinates from terminal space to raster memory space (and vice-versa), it is possible to pick a line that is slightly outside the stated aperture.

DEFAULTS

aperture-width
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IA11 (Level 2): Invalid *aperture-width.*

REFERENCES

ENABLE-GIN command
GIN-report message type
INCLUDE-COPY-OF-SEGMENT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PICK-ID Command

Host Syntax

$^{E}{}_{C}$**MI** int:*pick-ID-number*

Setup Syntax

$^{E}{}_{C}$**MI** $^{S}{}_{P}$ *pick-ID-number*

PARAMETERS

pick-ID-number (0 to 32767)
Specifies the *pick-ID* for parts of the currently open segment.

DESCRIPTION

This command inserts a pickID into the the segment being defined, for use in GIN pick operations and segment editing. See ENABLE-GIN, DELETE-PART-OF-SEGMENT, INSERT-INTO-SEGMENT, and REPLACE-PART-OF-SEGMENT commands for details.

If you issue a SET-PICK-ID command while no segment is currently being defined, the terminal detects a type MI03 error.

DEFAULTS

pick-ID-number
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

MI03 (Level 2): Command is invalid at this time. (No segment is currently being defined.)

MI11 (Level 2): Invalid *pick-ID-number*.

REFERENCES

DELETE-PART-OF-SEGMENT command
ENABLE-GIN command
GIN-report message type
INSERT-INTO-SEGMENT command
REPLACE-PART-OF-SEGMENT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PIVOT-POINT Command

Host Syntax

EcSP *xy* + :*pivot-point*

Setup Syntax

EcSP Sp *pivot-point*

PARAMETERS

pivot-point
Specifies the pivot point of subsequent segment definitions and graphtext character definitions. The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$
coordinate mode 2: X,Y,Z = -2^{23} to $2^{23}-1$

DESCRIPTION

This command sets the pivot point for use in subsequent segment and graphtext character definitions, and also sets the position for future segments (segment −2) (as the SET-SEGMENT-POSITION, SET-SEGMENT-IMAGE-TRANSFORM, and SET-SEGMENT-MATRIX-TRANSFORM commands do).

The 4128 and 4129 terminals have two pivot points: one for both 2D segments and graphtext characters and one for 3D segments. The 2D pivot point is set and inquired about when the terminal is in coordinate mode 0 or 1. The 3D pivot point is set and inquired about when the terminal is in coordinate mode 2.

See the *Segment-definition* syntactic construct and the *Graphtext-font-definition* syntactic construct for details on the use of the pivot point.

DEFAULTS

pivot-point
as shipped — (0,0) 2D; (0,0,0) 3D
on power-up — (0,0) 2D; (0,0,0) 3D
if omitted — (0,0) 2D; (0,0,0) 3D

ERRORS

SP11 (Level 2): *Pivot-point* out of valid range.

REFERENCES

Graphtext-font-definition syntactic construct
Segment-definition syntactic construct
SET-SEGMENT-IMAGE-TRANSFORM command
SET-SEGMENT-MATRIX-TRANSFORM command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PIXEL-BEAM-POSITION Command

Host Syntax

$^{E}\mathrm{c}$**RH** xy:*beam-position*

Setup Syntax

$^{E}\mathrm{c}$**RH** xy:*beam-position*

PARAMETERS

beam-position (4111: X = 0 to 1023, Y = 0 to 767;
4115/4120 Series: X = 0 to 1279, Y = 0
to 1023)
Position, relative to the pixel viewport's lower left corner, where the next RASTER-WRITE or RUNLENGTH-WRITE command will take effect.

DESCRIPTION

This command sets the *pixel-beam-position* in the pixel viewport, for use by subsequent RASTER-WRITE, RUNLENGTH-WRITE, and SAVE commands. The *pixel-beam-position* coordinates are relative to the lower left corner of the pixel viewport in raster memory space.

The SET-PIXEL-VIEWPORT command initializes the *pixel-beam-position* to the pixel viewport home position, which is defined by the SET-PIXEL-WRITING-FACTORS command. On power-up, this is the upper-left corner of the pixel viewport. If the proposed *pixel-beam-position* is outside the current pixel viewport, then it is set to the nearest pixel within the pixel viewport that allows one pixel rectangle (as defined by the SET-PIXEL-WRITING-FACTORS command) to be written inside the viewport.

DEFAULTS

beam-position
as shipped — 0,1023 (4111: 0,767)
on power-up — 0,1023 (4111: 0,767)
if omitted — 0,0

ERRORS

RH11 (Level 2): Invalid *beam-position.*

REFERENCES

RASTER-WRITE command
RUNLENGTH-WRITE command
SAVE command
SET-PIXEL-VIEWPORT command
SET-PIXEL-WRITING-FACTORS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PIXEL-VIEWPORT Command

Host Syntax

$^{\mathrm{E}}_{\mathrm{C}}$**RS** xy:*first-corner* xy:*second-corner*

Setup Syntax

$^{\mathrm{E}}_{\mathrm{C}}$**RS** $^{\mathrm{S}}_{\mathrm{P}}$ *first-corner* *second-corner*

PARAMETERS

first-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series: X = 0 to 1279, Y = 0 to 1023)
Specifies, in raster memory space coordinates, one corner of the pixel viewport.

second-corner (4111: X = 0 to 1023, Y = 0 to 767; 4115/4120 Series:X = 0 to 1279, Y = 0 to 1023)
Specifies the opposite corner of the pixel viewport.

DESCRIPTION

This command sets the pixel viewport position on the pixel-operations writing surface. (The "pixel-operations writing surface" is the surface named in the most recent BEGIN-PIXEL-OPERATIONS command.) The pixel viewport is used by the Option 19 device SC:, and by the RASTER-WRITE, RUNLENGTH-WRITE, and SAVE commands. The SET-PIXEL-VIEWPORT command also updates the current pixel beam position to the upper left corner of the pixel the point specified by the last SET-PIXEL-WRITING-FACTORS command.

The *first-corner* and *second-corner* coordinates may be the coordinates of any two diagonally opposite corners of the pixel viewport. The terminal will set Xmin (the x-coordinate of the leftmost pixel in the pixel viewport) to the lesser of the two specified x-coordinates. It will set Xmax (the x-coordinate of the rightmost pixel) to the larger of the two x-coordinates. Likewise, it will sort the y-coordinates in the proper order to determine the y-coordinates of the bottom and top of the pixel viewport.

If the pixel viewport is smaller than the pixel rectangle defined by the most recent SET-PIXEL-WRITING-FACTORS command, an error is detected and the pixel viewport is not changed.

DEFAULTS

first-corner
as shipped — (0,0)
on power-up — remembered
if omitted — (0,0)

second-corner
as shipped — 1279,1023 (4111: 1023,767)
on power-up — remembered
if omitted — 0,0

ERRORS

RS11 (Level 2): Invalid *first-corner* coordinate.

RS21 (Level 2): Invalid *second-corner* coordinate.

REFERENCES

BEGIN-PIXEL-OPERATIONS command
Device parameter type
RASTER-WRITE command
RUNLENGTH-WRITE command
SAVE command
SET-PIXEL-BEAM-POSITION command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PIXEL-WRITING-FACTORS Command

Host Syntax

$^{E}_{C}$**RT** int:*pixel-width* int:*pixel-height*
int:*major-axis*

Setup Syntax

$^{E}_{C}$**RT** $^{S}_{P}$ *pixel-width* *pixel-height* *major-axis*

PARAMETERS

pixel-width (4111: –1023 to 1023;
4115/4120 Series: –1280 to 1280)
Specifies the width of the pixel and the horizontal direction of display. With positive widths, pixels are placed from left to right; with negative widths, pixels are placed from right to left. A value of 0 is interpreted as 1.

pixel-height (4111: –1023 to 1023;
4115/4120 Series: –1024 to 1024)
Specifies the height of the pixel and the vertical direction of display. With positive heights, pixels are placed from top to bottom; with negative heights, pixels are placed from bottom to top. A value of 0 is interpreted as 1.

major-axis (0 or 1)
Specifies the direction in which adjacent pixels are placed, horizontal or vertical.

0 adjacent pixels are placed horizontally

1 adjacent pixels are placed vertically

DESCRIPTION

This command allows you to control the size of the pixel rectangle that is produced for each pixel contained in RASTER-WRITE and RUNLENGTH-WRITE commands, and to control the direction in which pixels are written on the screen. The home position of the pixel viewport is also affected (see Table 7-24).

If the pixel rectangle's width or height are larger than the current pixel viewport, an error is detected and this command is ignored.

When a pixel is being displayed as the result of a RASTER-WRITE or RUNLENGTH-WRITE command, a pixel rectangle the size of the absolute values of the *pixel-width* and *pixel-height* and of all the same color index is put into raster memory, starting at the current pixel-beam-position and extending in the directions indicated by the signs of the *pixel-width* and *pixel-height* parameters. If the entire pixel rectangle does not fit into the pixel viewport, the pixel-beam-position is first moved to the beginning of the next row (or column, depending on the direction of display), before the rectangle is displayed.

The number of pixels in the display space which are actually filled for each pixel specified in a RASTER-WRITE or RUNLENGTH-WRITE command is specified by the *pixel-width* and *pixel-height* parameters. The writing direction is specified by the signs of those two parameters, and by the *major-axis* parameter, as shown in Table 7-24.

Table 7-24

COMBINATIONS OF DIRECTIONS

major-axis	pixel-width sign	pixel-height sign	Direction Pixels Are Placed	Pixel Viewport Home Position
0	+	+	L to R, top-down	Top-left corner
0	+	–	L to R, bottom-up	Bottom-left corner
0	–	+	R to L, top-down	Top-right corner
0	–	–	R to L, bottom-up	Bottom-right corner
1	+	+	Top-down, L to R	Top-left corner
1	+	–	Bottom-up, L to R	Bottom-left corner
1	–	+	Top-down, R to L	Top-right corner
1	–	–	Bottom-up, R to L	Bottom-right corner

If there have been RASTER-WRITE or RUNLENGTH-WRITE commands since the last SET-PIXEL-BEAM-POSITION or SET-PIXEL-VIEWPORT command, and you change the *pixel-width* parameter from positive to negative, or vice versa, then the pixel-beam-position moves one pixel in the new *pixel-width* direction. Likewise, if you change the *pixel-height* parameter from positive to negative, or vice versa, then the pixel-beam-position moves one pixel in the new *pixel-height* direction.

If either one of these adjustments results in the beam position moving outside the pixel viewport, then the beam is moved to the opposite edge of the viewport.

You can avoid this adjustment of the pixel-beam-position by setting the *pixel-width* or *pixel-height* parameter to 0 (which is neither positive nor negative) before assigning the parameter a value of the opposite sign.

DEFAULTS

pixel-width
as shipped — 0
on power-up — 0
if omitted — 0

pixel-height
as shipped — 0
on power-up — 0
if omitted — 0

major-axis
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

RT11	(Level 2):	Invalid *pixel-width* or width is larger than the pixel viewport width.
RT21	(Level 2):	Invalid *pixel-height* or height is larger than the pixel viewport height.
RT31	(Level 2):	Invalid *major-axis.*

REFERENCES

BEGIN-PIXEL-OPERATIONS command
RASTER-WRITE command
RUNLENGTH-WRITE command
SET-PIXEL-VIEWPORT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-PORT-BAUD-RATE Command

4111 or OPTION 10

Host Syntax

$^{E}{}_{C}$**PR** device:*port* int:*baud-rate*

Setup Syntax

PBAUD $^{S}{}_{P}$ *port* *baud-rate*

PARAMETERS

port

Names the RS-232 peripheral port whose baud rate is being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)
DMA (4111 only)

baud-rate

The data rate (bits/second) used at the specified peripheral port. Valid rates are: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, and 9600.

For the 4111, data rates used at the **DMA** device port are: 9600, 19200, 38400, 76800, 153600, 230400, 307200, 409600, 526628, 614400, and 737280. For **DMA**, if a value other than these is specified, the next lower value will be selected from a table. If a value less than 9600 is specified, 9600 will be selected. The range of valid values is 0 to $2^{31}-1$. The P0: settings can not be changed while DM: is active (while DMA is assigned). The DM: settings can be changed any time DM: is not busy.

DESCRIPTION

This command sets the baud rate used at one of the three RS-232 peripheral ports. The terminal "remembers" this setting even when powered off.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PR11

baud-rate
as shipped — 2400
on power-up — remembered
if omitted — error PR21

ERRORS

PR00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PR11	(Level 2):	Invalid *port.*
PR12	(Level 3):	Out of memory while parsing the parameter.
PR13	(Level 2):	Port is busy.
PR21	(Level 2):	Invalid *baud-rate.*

REFERENCES

SET-BAUD-RATES command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PORT-EOF-STRING Command

4111 or OPTION 10

Host Syntax

$^{E}{}_{C}$PE device:*port* int-array:*PEOF-string*

Setup Syntax

PEOF $^{S}{}_{P}$ *port* *PEOF-string*

PARAMETERS

port
Names the peripheral port for which an end-of-file string is being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)

PEOF-string
An array with length of up to ten *int*s. Each represents a single ASCII character, and so must be in the range from 0 to 127.

DESCRIPTION

This command sets the port-end-of-file string used when a peripheral device is connected to the specified RS-232 peripheral port. When, during a copy to that device, an end-of-file condition (EOF) is detected, the terminal sends the specified port-end-of-file string and terminates the copy operation.

Likewise, during a COPY to or from a peripheral port, the *PEOF-string* marks the end of the copy or pcopy operation. That is, the terminal terminates the COPY or PCOPY when it detects the *PEOF-string* in the characters coming from the peripheral device.

Port Name. The first parameter in the SET-PORT-EOF-STRING command is a string naming the peripheral port. This must be **P0:**, **P1:**, or **P2:**.

End-of-File String. The command's second parameter is an int-array holding the numeric equivalents of the characters which comprise the *PEOF-string*. Each int in this array is the numeric equivalent of an ASCII character, and so must be in the range from 0 to 127. There may be as many as ten ints in the array.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PE11

PEOF-string
as shipped — empty array
on power-up — remembered
if omitted — empty array

ERRORS

PE00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PE11	(Level 2):	Invalid *port*.
PE12	(Level 3):	Out of memory while parsing the parameter.
PE13	(Level 2):	Port is busy.
PE21	(Level 2):	Invalid *PEOF-string*.
PE22	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

SET-EOF-STRING command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PORT-EOL-STRING Command

4111 or OPTION 10

Host Syntax

${}^{E}{}_{C}$**PM** device:*port*
int-array:*EOL-string*

Setup Syntax

PEOL ${}^{S}{}_{P}$ *port* *EOL-string*

PARAMETERS

port
Names the RS-232 peripheral port for which an *EOL-string* is being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)

EOL-string
An array with length of up to two *int*s. Each *int* is the numeric equivalent of a single ASCII character, and so must be in the range from 0 to 127.

DESCRIPTION

This command sets the end-of-line string for the specified port. This string replaces every ${}^{C}{}_{R}$ in the data going to a port if (and only if) the **4643** protocol is set for that port. The most used *EOL-string* is ${}^{C}{}_{R}{}^{L}{}_{F}$.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PE11

EOL-string
as shipped — empty array
on power-up — remembered
if omitted — empty array

ERRORS

PM00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PM11	(Level 2):	Invalid *port*.
PM12	(Level 3):	Out of memory while parsing the parameter.
PM13	(Level 2):	Port is busy.
PM22	(Level 3):	Out of memory while parsing the parameter.

REFERENCES

PORT-ASSIGN command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PORT-FLAGGING-MODE Command

4111 or OPTION 10

Host Syntax

$^{E}{}_{C}$**PF** device:*port* int:*flagging-mode*
int:*GO-character* int:*STOP-character*

Setup Syntax

PFLAG $^{S}{}_{P}$ *port* *flagging-mode*
GO-character
STOP-character

PARAMETERS

port

Names the RS-232 peripheral port for which a flagging mode is being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)

flagging-mode (0, 1, or 2)

Specifies the type of flagging the terminal uses for the specified port. Setup mode parameters are **NONE**, **CHAR**, and **DTR/CTS**.

0	**NONE**; no flagging
1	**CHAR**; character flagging
2	**DTR/CTS**; flagging with DTR, CTS signal lines

GO-character (0 to 127)

The numeric equivalent of a single ASCII character. If *flagging-mode* is 1 (CHAR), this character means, "I am ready for more data; go ahead and send it," when the terminal sends it to the device attached to the specified port.

STOP-character (0 to 127; if not 0, must be different from GO-character)

The numeric equivalent of a single ASCII character. If *flagging-mode* is 1 (CHAR), this character means, "I am not ready for data; stop sending data to me," when the terminal sends it to the device attached to the specified port.

DESCRIPTION

Sets the flagging mode for the specified host port number to which the peripheral will be attached.

Character Flagging. When the terminal sends data to a peripheral device, that device can signal that it is not ready for more characters by sending a *STOP-character*. The terminal then stops sending characters to the peripheral device; it resumes transmission only on receipt of a *GO-character* from the peripheral device.

Likewise, when receiving data from a peripheral device, the terminal may send a *STOP-character* to the peripheral device when it is not yet ready to receive more characters. When the terminal is ready, it sends the *GO-character* to the peripheral device, causing the peripheral device to resume transmission.

This flagging mode is the default mode, with $^{D}{}_{1}$ — ASCII decimal equivalent 17 — as the *GO-character*, and $^{D}{}_{3}$ — ADE 19 — as the STOP-character. If the third parameter is zero (or omitted by terminating the command early), the terminal sets the GO-character to $^{D}{}_{1}$. If the fourth parameter is zero (or omitted), the terminal sets the *STOP-character* to $^{D}{}_{3}$.

DTR/CTS Flagging. In sending data to a peripheral device (typically, a printer for this kind of flagging), the terminal continues to send characters so long as the peripheral device places a positive voltage on the DTR (Data Terminal Ready) line at the RS-232 connector for that peripheral device. If the printer (or other peripheral device) places a negative voltage on DTR, the terminal stops sending characters until DTR goes positive again.

In receiving data from a peripheral device, the terminal places a positive voltage on the CTS (Clear To Send) line at the RS-232 connector for that peripheral device. Should the terminal momentarily be "too busy" to receive characters, it places a negative voltage on CTS. This should cause the peripheral device to stop transmitting characters. When the terminal is ready to receive more characters, it sends CTS-positive again.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PF11

flagging-mode
as shipped — 1
on power-up — remembered
if omitted — 0

GO-character
as shipped — 17
on power-up — remembered
if omitted — 17

STOP-character
as shipped — 19
on power-up — remembered
if omitted — 19

ERRORS

PF00	(Level 0):	Unrecognized command. (Not a 4111 or Option 10 is not installed.)
PF11	(Level 2):	Invalid *port.*
PF12	(Level 3):	Out of memory while parsing the parameter.
PF13	(Level 2):	Port is busy.
PF21	(Level 2):	Invalid *flagging-mode.*
PF31	(Level 2):	Invalid *GO-character.*
PF41	(Level 2):	Invalid *STOP-character.*

REFERENCES

SET-FLAGGING command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PORT-PARITY Command

4111 or OPTION 10

Host Syntax

$^{E}_{C}$**PP** device:*port* int:*parity-mode*

Setup Syntax

PPARITY $^{S}_{P}$ *port* *parity-mode*

PARAMETERS

port
Names the RS-232 peripheral port whose parity mode is being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)

parity-mode (0 to 4)
Specifies the parity mode for data sent through the specified port. Setup mode parameters are **LOW**, **ODD**, **EVEN**, **HIGH**, and **NONE**.

0	**LOW**; the parity bit is set to zero
1	**ODD**
2	**EVEN**
3	**HIGH**; the parity bit is set to one
4	**NONE**; the parity bit is omitted

DESCRIPTION

The SET-PORT-PARITY command determines whether characters at the peripheral ports have parity bits. If the characters do have parity bits, the SET-PORT-PARITY command also specifies how those parity bits are set.

NOTE

The meaning of the "parity type" parameter in this command is not the same as the meaning of the corresponding parameter in the SET-PARITY command.

Figure 7-23 shows the format of characters exchanged between the terminal and a device attached to an RS-232 peripheral port. Each character begins with a start bit. After the start bit come from 5 to 8 data bits, an optional parity bit, and one or two stop bits. The start bit is always zero ("pace" or "travail" condition on the data communications line). The stop bits are always one ("mark" or "repos" condition). The numbers of data bits and of stop bits are determined by the SET-PORT-STOP-BITS command. Whether or not there is a parity bit is determined by the SET-PORT- PARITY command.

Port Specifier String. The SET-PORT-PARITY command's first parameter is a string naming the peripheral port whose parity attribute is being set.

Parity Type. The command's second parameter specifies the parity type.

If the parity type parameter is 0, then low parity is used at the peripheral port. Each character includes a parity bit. The terminal transmits this bit as a "0," and ignores this bit in characters it receives from a peripheral device.

If this parameter is 1, then odd parity is used at the specified peripheral port. Each character (or other six-, seven-, or eight-bit data byte) includes a parity bit. In transmitting characters to a peripheral device, the terminal sets this bit to 1 or 0, whichever is needed to give an odd number of "1" bits (not counting the start and stop bits). Likewise, when receiving characters from a device at the peripheral port, the terminal checks for odd parity. That is, it checks that there are an odd number of "1" bits (not counting the start and stop bits).

If the parity type parameter is 2, then even parity is used at the peripheral port. Each transmitted character includes a parity bit. The parity bit is 1 or 0, whichever is necessary to make the character have an even number of "1" bits (not counting the start and stop bits).

If the parity type parameter is 3, then high parity is used at the peripheral port. Each character includes a parity bit. The terminal always transmits this bit as a "1". The parity bit is ignored in characters received from a peripheral device.

If this parameter is 4, the parity bit is omitted.

For normal operation with the ASCII character set, there should be seven data bits, one parity bit, and one or two stop bits. Therefore, the SET-PORT-PARITY command should have a parity mode parameter of 0, 1, 2, or 3. (Parity mode 4 should be avoided, as that causes the parity bit to be omitted.)

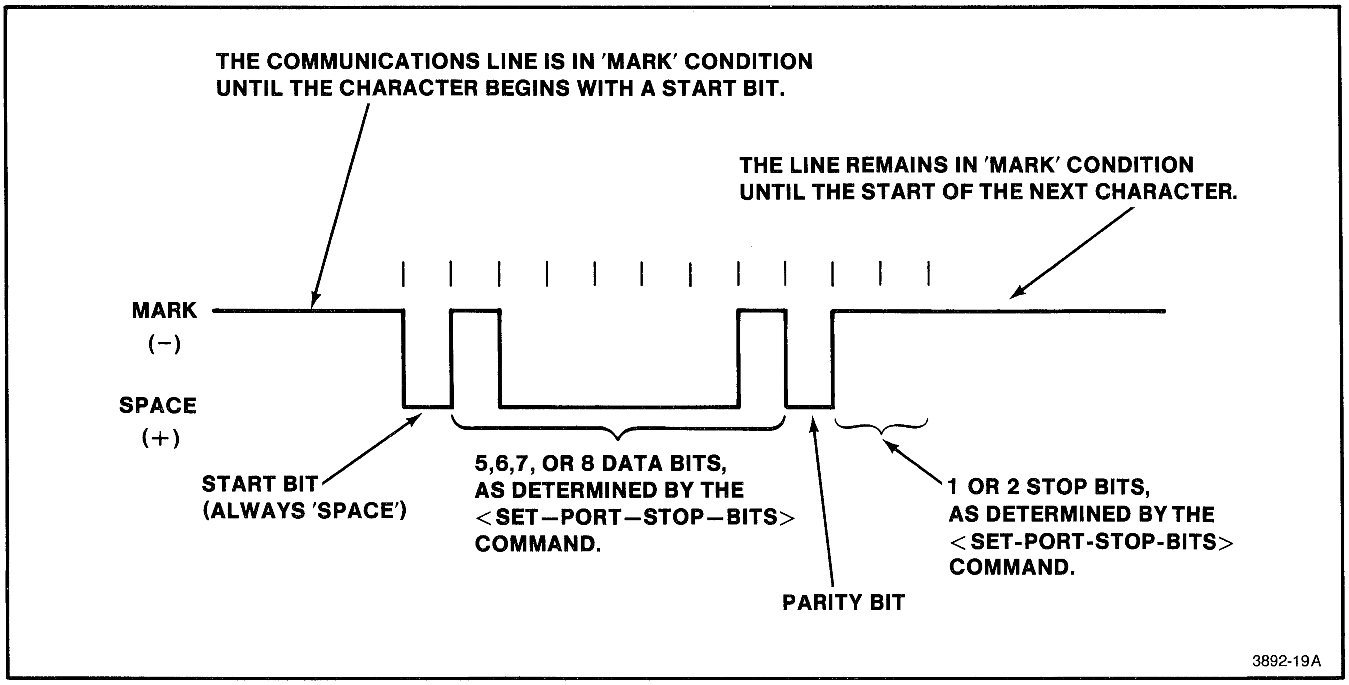

THE COMMUNICATIONS LINE IS IN 'MARK' CONDITION UNTIL THE CHARACTER BEGINS WITH A START BIT.
THE LINE REMAINS IN 'MARK' CONDITION UNTIL THE START OF THE NEXT CHARACTER.
MARK
(−)
SPACE
(+)
START BIT (ALWAYS 'SPACE')
5,6,7, OR 8 DATA BITS, AS DETERMINED BY THE <SET−PORT−STOP−BITS> COMMAND.
PARITY BIT
1 OR 2 STOP BITS, AS DETERMINED BY THE <SET-PORT-STOP-BITS> COMMAND.
3892-19A

Figure 7-23. Format of a Character at the RS-232 Peripheral Ports.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PP11

parity-mode
as shipped — 4
on power-up — remembered
if omitted — 0

ERRORS

PP00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PP11 (Level 2): Invalid *port.*

PP12 (Level 3): Out of memory while parsing the parameter.

PP13 (Level 2): Port is busy.

PP21 (Level 2): Invalid parity-mode.

REFERENCES

SET-PARITY command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PORT-STOP-BITS Command

4111 or OPTION 10

Host Syntax

```
EcPB   device:port   int:number-of-stop-bits
       int:number-of-data-bits
```

Setup Syntax

```
PBITS SP   port   number-of-stop-bits
           number-of-data-bits
```

PARAMETERS

port (P0:, P1:, or P2:)
Names the RS-232 peripheral port for which the numbers of stop bits and data bits are being specified. Valid ports are:

P0:
P1:
P2: (Not valid for 4111)

number-of-stop-bits (1 or 2)
Specifies the number of stop bits in each byte of data for the specified port.

number-of-data-bits (5, 6, 7, or 8)
Specifies the number of data bits in each byte of data for the specified port. This count does not include the parity bit, whose presence or absence is determined by the SET-PORT-PARITY command.

DESCRIPTION

The SET-PORT-STOP-BITS command sets the number of stop bits and data bits used in characters sent between the terminal and its RS-232 peripheral ports. This command requires that Option 10 be installed.

Figure 7-24 shows the format of a character (or data byte) transmitted between the terminal and one of its RS-232 peripheral ports. Each character has a start bit, which is always 0 (a positive voltage, corresponding to the "space" or "travail" condition at the RS-232 interface). After the start bit come from 5 to 8 data bits; the number of data bits is determined by the most recent SET-PORT-STOP-BITS command. After the data bits comes the parity bit; this bit may be omitted, depending on the most recent SET-PORT-PARITY command. After the parity bit comes one or two stop bits. The number of stop bits is determined by the most recent SET-PORT-STOP-BITS command. The stop bits are always 1 (a negative voltage, corresponding to a "mark" or "repos" condition at the RS-232 interface).

Port Name. The first parameter in the SET-PORT-STOP-BITS command is a string naming the RS-232 peripheral port to which the command applies.

Number of Stop Bits. The second parameter is an *int* naming the number of stop bits. This parameter must be either 1 or 2.

Number of Data Bits. The third parameter is an *int* naming the number of data bits. This parameter must be 5, 6, 7, or 8.

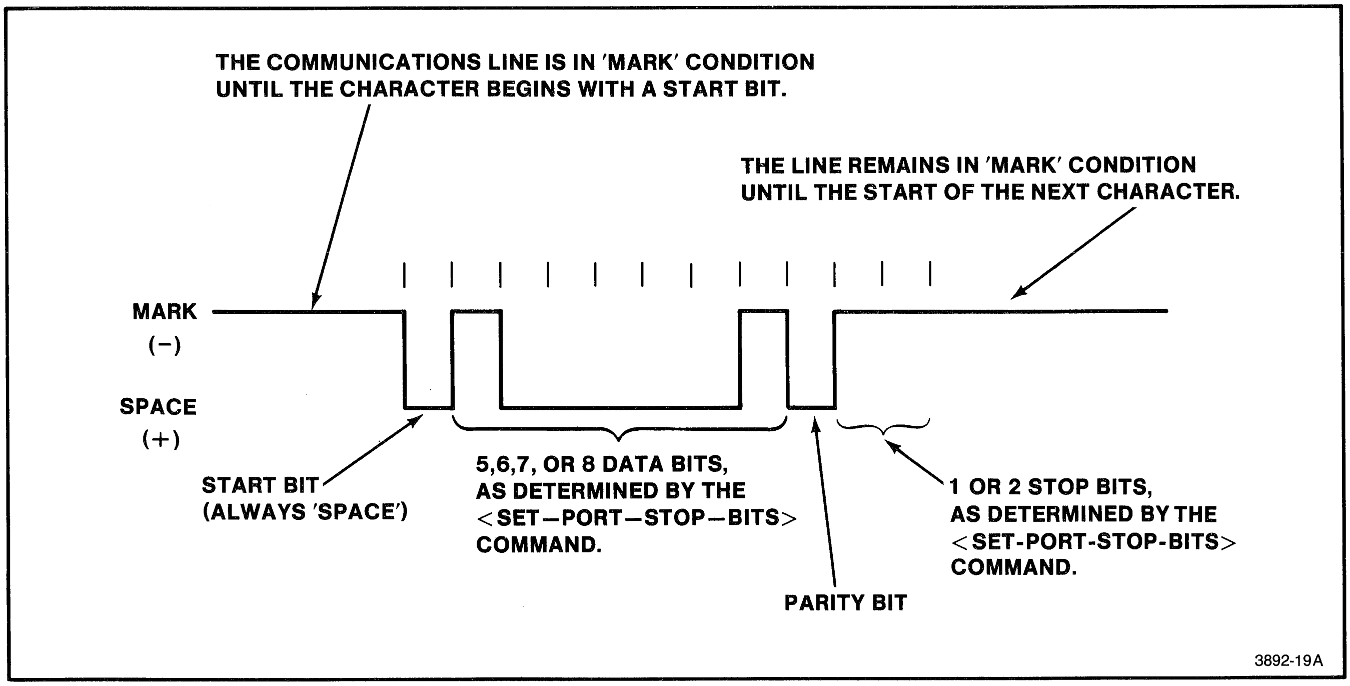

THE COMMUNICATIONS LINE IS IN 'MARK' CONDITION UNTIL THE CHARACTER BEGINS WITH A START BIT.
THE LINE REMAINS IN 'MARK' CONDITION UNTIL THE START OF THE NEXT CHARACTER.
MARK
(−)
SPACE
(+)
START BIT (ALWAYS 'SPACE')
5,6,7, OR 8 DATA BITS, AS DETERMINED BY THE <SET−PORT−STOP−BITS> COMMAND.
1 OR 2 STOP BITS, AS DETERMINED BY THE <SET-PORT-STOP-BITS> COMMAND.
PARITY BIT
3892-19A

Figure 7-24. Format of Characters at the RS-232 Peripheral Ports.

DEFAULTS

port
as shipped — none
on power-up — none
if omitted — error PB11

number-of-stop-bits
as shipped — 2
on power-up — remembered
if omitted — error PB21

number-of-data-bits
as shipped — 8
on power-up — remembered
if omitted — error PB21

ERRORS

PB00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PB11 (Level 2): Invalid *port.*

PB12 (Level 3): Out of memory while parsing the parameter.

PB13 (Level 2): Port is busy.

PB21 (Level 2): Invalid *number-of-stop-bits.*

PB31 (Level 2): Invalid *number-of-data-bits.*

REFERENCES

SET-PORT-PARITY command
SET-STOP-BITS command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-PROMPT-STRING Command

Host Syntax

$^E{}_C$**NS** int-array:*prompt-string*

Setup Syntax

PROMPTSTRING $^S{}_P$ *prompt-string*

PARAMETERS

prompt-string (length 0 to 10)
An array with length of up to ten *int*s. Each *int* is the numeric equivalent of a single ASCII character, and so must be in the range from 0 to 127.

DESCRIPTION

Specifies the character sequence that is accepted as a prompt sequence when received from the host. The string can be up to 10 characters. If the string is a null string (has zero characters), prompt mode is disabled, since a null string is always there when the terminal searches for it.

DEFAULTS

prompt-string
as shipped — 63,32
on power-up — remembered
if omitted — empty array

ERRORS

NS11 (Level 2): Invalid *prompt-string.*

NS12 (Level 3): Out of memory while parsing the parameter.

REFERENCES

PROMPT-MODE command

INQUIRY COMMAND

REPORT-PORT-STATUS command

SET-QUEUE-SIZE Command

Host Syntax

```
EcNQ   int:queue-size
```

Setup Syntax

```
QUEUESIZE Sp   queue-size
```

PARAMETERS

queue-size (1 to 65535)
Number of memory bytes reserved for the communications input queue.

DESCRIPTION

This command reserves part of the terminal's memory as a storage area for the terminal's communications input queue.

If the queue size requested is larger than the available memory, then all available memory is allocated to the communications queue and a type NQ02 error is detected.

Explanation. Sometimes characters come from the host computer at a rate faster than the terminal can process them. This can happen especially when those characters comprise commands to perform lengthy and complicated operations : INCLUDE-COPY-OF-SEGMENT, SAVE: segments-all, etc.

When characters arrive faster than the terminal can process them, the terminal stores them in its communications input queue until it has a chance to process them — or until the memory allocated for that queue is exhausted. (If the queue memory is exhausted, incoming characters are lost.)

Specifying a large maximum queue size permits the terminal to buffer more characters before data is lost. Specifying a small communications queue size leaves more of the terminal's memory available for other uses.

Issuing a SET-QUEUE-SIZE : N command causes the terminal to reserve at least N eight-bit bytes for its communications input queue. The actual queue size may be somewhat larger than N; however, it will be at least N bytes large.

The terminal "remembers" the queue size setting even when turned off. When the terminal is shipped from the factory, it maximum queue size is set to 300 bytes.

NOTE

The 4125, 4128, and 4129 can display simple alphanumerics and graphics up to a maximum continuous data rate of 19200 bits/second. (This does not include commands which require more than routine processing, such as the LOAD or INCLUDE-COPY-OF-SEGMENT commands.) At higher data rates, some "handshaking" protocol must be used to prevent the terminal's communications input queue from overflowing.

Moreover, even at slow data rates, it is prudent to use a handshaking protocol. The terminal can take an appreciable amount of time to execute some commands — such as LOAD or SAVE : segments-all — which can be issued using only a very few characters. If a handshaking protocol is not used, the terminal's input queue may overflow while executing such commands.

Such a handshaking protocol might be as simple as issuing a Report-4010-STATUS command from time to time, and waiting to receive the reply before issuing more commands to the terminal. Alternatively, any of several data communications protocols may be used: flagging mode, prompt mode, or block mode. Any of these communications modes will prevent the input queue from overflowing.

DEFAULTS

queue-size
as shipped — 300>
on power-up — remembered
if omitted — error NQ11

ERRORS

NQ01 (Level 3): Cannot free enough currently allocated queue memory.

NQ02 (Level 3): Out of memory while performing SET-QUEUE-SIZE command.

NQ11 (Level 2): Invalid *queue-size.*

REFERENCES

ARM-FOR-BLOCK-MODE command
SET-BAUD-RATES command
SET-FLAGGING-MODE command
SET-PROMPT-MODE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-REPORT-EOM-FREQUENCY Command

Host Syntax

$^{E}_{C}$**IM** int:*EOM-frequency*

Setup Syntax

REOM $^{S}_{P}$ *EOM-frequency*

PARAMETERS

EOM-frequency (0 or 1)
Specifies whether *EOM-indicators* should be sent more or less frequently in reports to the host. "More frequently" means at the end of each part of the message; "less frequently" means only when needed to prevent the maximum line length from being exceeded.

0	less frequently
1	more frequently

DESCRIPTION

The command controls how frequently the terminal intersperses *EOM-indicators* among the characters that comprise a "report message" that the terminal sends to the host computer: *GIN-report-sequences, device-status-reports, port-status-reports, segment-status-reports, terminal-settings-reports,* etc.

Generally speaking, setting the *EOM-frequency* to "1" (more frequent) causes a long message to be broken into separate lines of text for each part of the report. For instance, each *GIN-report* within a *GIN-report-sequence* would occupy a separate line of text, terminated with an *EOM-indicator.*

Setting the *EOM-frequency* to "0" (less frequent) permits several parts of a report to fit on the same line of text. For instance, several *GIN-reports* could fit on the same line. The line is terminated with an *EOM-indicator* only when the terminal's maximum line length is about to be exceeded.

EOM-Indicators. The *EOM-indicator* (end-of-message indicator) serves to mark the end of a "line of text" in data being sent to the host. If the terminal is not in block mode, the *EOM-indicator* is the current end-of-line string, as defined by the most recent SET-EOL-STRING command. In block mode, the terminal sends an *EOM-indicator* by terminating the block and setting the "end-of-message" bit in the *block-control-bytes.*

Report Messages. "Report messages" are messages which the terminal sends in response to such commands as ENABLE-GIN, REPORT-ERRORS, REPORT-DEVICE-STATUS, and so on. Exactly where *EOM-indicators* occur within a report messages depends on the syntax for the particular type of report message:

- For GIN reports, see the descriptions of *GIN-report* message type.
- For other reports ("inquiry" reports), see the descriptions of the REPORT-DEVICE-STATUS, REPORT-ERRORS, REPORT-PORT-STATUS, REPORT-SEGMENT-STATUS, and REPORT-TERMINAL-SETTINGS commands.

DEFAULTS

EOM-frequency
as shipped — 1
on power-up — remembered
if omitted — 1

ERRORS

IM11 (Level 2): Invalid *EOM-frequency* setting.

REFERENCES

EOM-indicator syntactic construct
GIN-report message type
REPORT-DEVICE-STATUS command
REPORT-ERRORS command
REPORT-PORT-STATUS command
REPORT-SEGMENT-STATUS command
REPORT-TERMINAL-SETTINGS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-REPORT-MAX-LINE-LENGTH Command

Host Syntax

$^{E}_{C}$**IL** int:*max-line-length*

Setup Syntax

RLINELENGTH$^{S}_{P}$ *max-line-length*

PARAMETERS

max-line-length (0 to 65535)
The maximum number of characters per line in reports which the terminal sends to the host. Setting this parameter to zero disables the maximum-line-length feature.

DESCRIPTION

The command sets the maximum line length for report messages which the terminal sends to the host computer. It also determines how frequently the terminal intersperses *EOM-indicators* among files being transferred to the host in a COPY, SPOOL, SAVE, PLOT, or DIRECTORY command. Specifying a maximum line length of zero disables this feature; in effect, the maximum line length is set to infinity.

Report Messages. If the terminal has a report to send to the host, and that report would cause the maximum line length to be exceeded, then the terminal inserts an *EOM-indicator* into the report. The *EOM-indicator* serves to terminate the current line of the report.

Files Sent To The Host. When the terminal sends a file to the HO: device, it inserts and *EOM-indicator* after every *max-line-length* character (including $^{C}_{R}$'s and other control characters), unless it is 0, when none are inserted.

DEFAULTS

max-line-length
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IL11 (Level 2): Invalid *max-line-length.*

REFERENCES

EOM-indicator syntactic construct

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-REPORT-SIG-CHARS Command

Host Syntax

```
EcIS   int:report-type-code   int:sig-char
       int:term-sig-char
```

Setup Syntax

```
RSIGCHARS Sp   report-type-code   sig-char
               term-sig-char
```

PARAMETERS

report-type-code (–3 to –1, or a valid GIN *device-function*)
If this is a GIN *device-function* code (see the ENABLE-GIN command), then the signature characters defined in this command will be used in GIN-report-sequences for that *device-function* code. If this parameter is –3, then these signature characters are used in non-GIN report messages. If the parameter is –2, these signature characters are used only when responding to REPORT GIN-POINT: –2 commands. If the parameter is minus one, then these signature characters are used in all report messages (both GIN reports and non-GIN reports).

DESCRIPTION

This command assigns the "signature characters" to be used within report messages which the terminal sends to the host computer.

Report Type Code. The first parameter specifies in which type of report these signature characters will be used.

0 or a positive number represents a GIN *device-function* code; see the description of the ENABLE-GIN command for details. Future GIN-report-sequences for that *device-function* combination will use the signature characters specified in this SET-REPORT-SIG-CHARS command.

–3 specifies "inquiry reports" — report messages sent in response to non-GIN commands, such as REPORT-TERMINAL-SETTINGS, REPORT-ERRORS, REPORT-PORT-STATUS, etc.

–2 specifies responses to REPORT-GIN-POINT: –2 commands only.

–1 specifies "all reports" — inquiry reports and reports for all GIN *device-function* combinations.

Sig-Char. The second parameter is the numeric equivalent of the ASCII character which is to be used as the *sig-char* in reports of the specified type. If the *sig-char* is set to $^{\mathrm{N}}\mathrm{L}$ — numeric equivalent of zero — then that *sig-char* is omitted from reports sent to the host. Valid range is 0 to 127.

Term-Sig-Char. The third parameter specifies the character to be used as the *term-sig-char* in reports of the specified type. If the *term-sig-char* is $^{\mathrm{N}}\mathrm{L}$, then it is omitted in reports sent to the host. Valid range is 0 to 127.

When the *report-type-code* is set to -3, the *term-sig-char* is not included in some non-GIN report messages. The *term-sig-char* is not included, for example, in response to the REPORT-DEVICE-STATUS, REPORT-PORT-STATUS, or REPORT-TERMINAL-SETTINGS commands.

NOTE

> *If GIN is enabled for more than one device at a time, then, for best results, set different* sig-chars *and* term-sig-chars *for each enabled GIN device. This is necessary in order that the host computer can parse the interleaved* GIN-report-sequences *which can be sent to the host if more than one GIN device is active.*
>
> *Also, if non-GIN reports are requested while GIN is enabled, then different* sig-chars *and* term-sig-chars *are needed to distinguish non-GIN reports from GIN reports.*

On power up, all signature characters are initialized to $^{\mathrm{N}}\mathrm{L}$.

DEFAULTS

report-type-code
as shipped — none
on power-up — none
if omitted — 0

sig-char
as shipped — 0
on power-up — 0
if omitted — 0

term-sig-char
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

I011	(Level 2):	Invalid *report-type-code.*
IS21	(Level 2):	Invalid *sig-char.*
IS31	(Level 2):	Invalid *term-sig-char.*

REFERENCES

Device-status-report message type
ENABLE-GIN command
Error-report message type
GIN-report message type
Port-status-report message type
Segment-status-report message type
Terminal-settings-report message type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SECTIONING-PLANES Command

4128, 4129

Host Syntax

${}^{E}{}_{C}$WS int-array:*sectioning*
xy + -array:*plane-points*

Setup Syntax

${}^{E}{}_{C}$WS ${}^{S}{}_{P}$ *sectioning* *plane-points*

PARAMETERS

sectioning (length 0 or 1)
Specifies how *plane-points* determines the sectioning volume.

first element (–2 to 2)

not present	no change
–2	OR halfspaces
–1	AND halfspaces
0	no sectioning
1	first halfspace only
2	second halfspace only

plane-points (length 0, 3, or 6)
Specifies triples of points which determine sectioning plane halfspaces.

first three elements: (xyz = -2^{23} to $2^{23}-1$)
Specify one sectioning plane.

not present no change

second three elements: (xyz = -2^{23} to $2^{23}-1$)
Specify a second sectioning plane.

not present no change

DESCRIPTION

This command enables and disables sectioning and specifies the sectioning-volume for the current view only (but not for other views in the current view's cluster; see the SET-VIEW-DISPLAY-CLUSTER command).

Sectioning only effects "shaded-surface" and "hidden-line" images. Since sectioning does not effect "wire-frame" images, this command has no effect on a 4128 terminal.

The *sectioning* parameter indicates whether sectioning is enabled for the view, and specifies how the halfspaces are combined to form the sectioning volume. The sectioning volume is defined as either the logical AND (intersection) of the two halfspaces, the OR (intersection) of the two halfspaces, or only the first or second halfspace. The two halfspaces are determined by the points in the *plane-points* array. The first three points, P1,P2, and P3 define the first halfspace as the set of points P satisfying

(P-P1) • ((P2-P1) X (P3-P1)) > 0

where "•" represents the vector dot product operator and "X" the vector cross product operator. If these three points are colinear, then the first halfspace is the set of points with W coordinates greater than the plane parallel to the UV-plane (defined in the SET-VIEWING-TRANSFORM command) containing P1, so that the halfspace is oriented in UVW space.

The next three points, P4,P5, and P6, define the second halfspace as the set of points P satisfying

(P-P4) • ((P5-P4)) X (P6-P4)) > 0

If these three points are colinear, then the second halfspace is the set of points with W coordinates less than the plane parallel to the UV-plane containing P,4 so the halfspace is oriented in UVW space.

The sectioning volume determines which areas of the shaded image will be solid, translucent, or transparent when the view is renewed (see the SELECT-TRANSLUCENCY-PATTERN command). The unsectioned images will be solid or translucent, while the sectioned images will be translucent or transparent. An image is sectioned when it appears within the sectioning volume.

The default condition of *plane-points* produces a first halfspace of all points with positive W coordinates and a second halfspace of all points with negative coordinates.

DEFAULTS

sectioning
as shipped — 0
on power-up — 0
if omitted — 0

plane-points
as shipped — <0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0>
on power-up — <0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0>
if omitted — no change

ERRORS

WS00	(Level 0):	Unrecognized command (terminal is not a 4128 or 4129).
WS03	(Level 2):	Invalid coordinate mode (must not be 0 or 1).
WS11	(Level 2):	Invalid *sectioning.*
WS12	(Level 3):	Out of memory while parsing the *sectioning* parameter.
WS21	(Level 2):	Invalid *plane-points.*
WS22	(Level 3):	Out of memory while parsing the *plane-points* parameter.

REFERENCES

SELECT-TRANSLUCENCY-PATTERN command
SET-VIEW-DISPLAY-CLUSTER command
SET-VIEWING-TRANSFORM command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SEGMENT-CLASS Command

Host Syntax

$^{E}{}_{C}$**SA** int:*segment-number*
int-array:*removal-array*
int-array:*addition-array*

Setup Syntax

$^{E}{}_{C}$**SA** $^{S}{}_{P}$ *segment-number* *removal-array*
addition-array

PARAMETERS

segment-number (–3 to –1, or 1 to 32767)
Names the segment whose classes are being altered.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
1 to **32767**	a specific segment

removal-array (length 0 to 32768; –1, 1 to 64)
Lists the segment classes that are being removed from the specified segment.

–1	all classes
1 to **64**	a specific class

addition-array (length 0 to 32768; –1, 1 to 64)
Lists the segment classes that are being added to the specified segment.

–1	all classes
1 to **64**	a specific class

DESCRIPTION

The SET-SEGMENT-CLASS command alters the classes that are assigned to a segment for use in segment matching operations.

The segment number may be –3, –2, –1, or any positive segment number for an existing segment. Segment –3 means all segments that match the current matching class. Segment –2 means future segments. Segment –1 means all currently existing segments.

Class numbers may be –1 (meaning "all classes"), or any number in the range from 1 to 64.

Each segment has a set of classes which belong to it. This command alters this set by first removing the classes specified in the *removal-array* and then adding the classes specified in the *addition-array*. Removing a class that is not in a segment's set, or adding a class that is already there, does not cause an error.

The *4110/4120 Series Host Programmers Manual* contains further discussion of segment classes and how to set a segment's class-set.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SA11

removal-array
as shipped — empty array for segment -2
on power-up — empty array for segment -2
if omitted — empty array

addition-array
as shipped — empty array for segment -2
on power-up — empty array for segment -2
if omitted — empty array

ERRORS

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*.
SA21	(Level 2):	Invalid *removal-array*.
SA22	(Level 3):	Out of memory while parsing parameter.
SA31	(Level 2):	Invalid *addition-array*.
SA32	(Level 3):	Out of memory while parsing parameter.

REFERENCES

SET-CURRENT-MATCHING-CLASS command
4110/4120 Host Programmers Manual

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-DETECTABILITY Command

Host Syntax

EcSD int:*segment-number* int:*detectability*

Setup Syntax

EcSD **SP** *segment-number* *detectability*

PARAMETERS

segment-number (–3 to –1, 1 to 32767)
Names the segment whose detectability mode is being set.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
0	the crosshair cursor
1 to **32767**	a specific segment

detectability (0 or 1)
Specifies whether a segment can be picked in a GIN "pick" operation or not.

0	can't be picked
1	can be picked

DESCRIPTION

This command sets the detectability of a segment. If a *detectability* of 1 is specified, and the segment is visible in the current view, it may be picked when the pick function is enabled. If 0 is specified, the segment may not be picked.

Segment-Number Parameter. The segment number may be a positive number, or any of the special negative numbers 0, −1, −2, and −3. A positive number refers to a specific single segment, previously defined with BEGIN-SEGMENT and END-SEGMENT commands. Detectability can be specified for segment 0, but it is not set (that is, the crosshair cursor is not detectable). Segment −1 means all segments. Segment −2 means future segments. Segment −3 means all segments that match the current segment matching class.

The default segment detectability is 1 (detectable), unless the detectability for segment −2 has been set to 0 (nondetectable), in which case, the default detectability is 0.

All vectors, text, and markers are "pickable." Only the center point of a marker is pickable. The filled portion of a panel is also pickable on these terminals.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SD11

detectability
as shipped — 1 for segment −2
on power-up — 1 for segment −2
if omitted — 0

ERRORS

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*.
SD21	(Level 2):	Invalid *detectability*.

REFERENCES

BEGIN-SEGMENT command
ENABLE-GIN command
END-SEGMENT command
SET-CURRENT-MATCHING-CLASS command
SET-SEGMENT-CLASS command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-DISPLAY-PRIORITY Command

Host Syntax

```
EcSS   int:segment-number   int:priority-number
```

Setup Syntax

```
EcSS Sp   segment-number   priority-number
```

PARAMETERS

segment-number (−3 to −1, or 1 to 32767)
Specifies the segment for which a display priority is being set.

−3	all segments that match the current matching class
−2	the future segment
−1	all segments
1 to **32767**	a specific segment

priority-number (−32768 to 32767)
Specifies the display priority of the specified segment.

DESCRIPTION

This command sets the display priority of the specified segment to a number in the range from −32768 to 32767. A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist. Segment −1 refers to all currently defined segments (with numbers from 1 to 32767). Segment −3 refers to all segments that match the current segment matching class. Segment −2 refers to future segments at the time they are opened.

Segment display priority determines the order in which segments are displayed when a view is renewed or segment −1 or −3 is made visible, and which segment is picked in a GIN pick operation. See the RENEW-VIEW, SET-SEGMENT-VISIBILITY, and ENABLE-GIN commands for details.

On power-up, the display priority for future segments is 0.

For segments which have the same display priority, the order of display and picking is not defined.

DEFAULTS

segment-number
as shipped — none
n power-up — none
if omitted — error SS11

priority-number
as shipped — 0 for segment -2
on power-up — 0 for segment -2
if omitted — 0

ERRORS

SS02 (Level 3): Out of memory while processing the command.

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.*

SS21 (Level 2): Invalid *priority-number.*

REFERENCES

ENABLE-GIN command
RENEW-VIEW command
SET-SEGMENT-VISIBILITY command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-EDIT-MODE Command

4111, 4115P51, or 4120 SERIES

Host Syntax

$^{E}_{C}$**UH** int:*edit-mode*

Setup Syntax

$^{E}_{C}$**UH** $^{S}_{P}$ *edit-mode*

PARAMETERS

edit-mode (0 to 3)
The two low bits of this int parameter are two separate flags that control the appearance of the trailing part of an edited segment. Bit 0 controls xy translation (for 2D segments) or xyz translation (for 3D segments) and bit 1 controls primitive attributes.

0 Do not translate; restore primitive attributes.

1 Translate; restore primitive attributes.

2 Do not translate; do not restore primitive attributes.

3 Translate; do not restore primitive attributes.

DESCRIPTION

This command determines the position and status of primitive attributes of the trailing portion of a segment when it is closed and redisplayed, after it was reopened by a DELETE-PART-OF-SEGMENT, INSERT-INTO-SEGMENT, or REPLACE-PART-OF-SEGMENT command.

At the time a segment is reopened, the trailing portion of the segment that follows the insertion point and/or deleted pickID groups is in effect hidden (it is not redisplayed if the segment is redrawn). The primitive attributes at that point are remembered if they are not the terminal default; default values are not remembered. The terminal beam position (2D or 3D) is also remembered. When the segment is closed, the remembered primitive attributes and beam position may or may not be restored before the trailing portion, depending on *edit-mode.*

If *edit-mode* is 0 or 2, the primitives in the trailing part are not translated; they stay in here as they were before the segment was edited. If *edit-mode* is 1 or 3, the initial position of the trailing part is moved to the last position in the leading part (if no primitives were inserted) or the position of the last inserted primitive. All other primitives in the trailing part are translated by the same amount. Since dot-matrix characters (alphatext or string precision graphtext) update the beam position in terminal space only when the characters are displayed, inserting after them will occur at the last position in front of them.

If *edit-mode* is 0 or 1, the primitive attributes in the trailing part will be restored. If *edit-mode* is 2 or 3, the primitive attributes of the trailing part are not restored, so that changes due to editing will take effect until they are explicitly set in the trailing part.

DEFAULTS

edit-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

UH11 (Level 2): Invalid *edit-mode*.

REFERENCES

DELETE-PART-OF-SEGMENT command
INSERT-INTO-SEGMENT command
REPLACE-PART-OF-SEGMENT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SEGMENT-HIGHLIGHTING Command

Host Syntax

ECSH int:*segment-number* int:*highlighting*

Setup Syntax

ECSH SP *segment-number* *highlighting*

PARAMETERS

segment-number (–3 to –1, or 1 to 32767)
Names the segment for which highlighting is being specified.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
1 to **32767**	a specific segment

highlighting (4125: 0 or 1; 4128, 4129: 0 or 1 for 2D segments, –3 to 3 for 3D segments)
Specifies how a segment is highlighted.

DESCRIPTION

This command sets a segment's highlight attribute. *Highlighting* controls whether a segment blinks and whether facets are outlined on a 4129 when the view's object-surface-display attribute (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is "shaded-surface" or "hidden-line."

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist or if it is a 2D segment and *highlighting* is not 0 or 1. Segment –1 refers to all currently defined segments (with numbers from 1 to 32767). Segment –3 refers to all segments that match the current segment matching class. Segment –2 refers to future segments at the time they are opened.

When segment −1 or −3 is specified, negative *highlighting* values and *highlighting* values greater than 1 affect only 3D segments.

The low-order two bits of *highlighting* control blinking and outlining. The 4125 and 4128 support only blinking. The 4129 supports blinking and outlining.

The Table 7-25 gives the meaning of the *highlighting* values.

Table 7-25

HIGHLIGHTING VALUES

Highlight Value	Terminal	Blinking	Outlining	Segments Affected
-3	4128/29	off	off	3D only
-2	4128/29	no change	off	3D only
-1	4128/29	off	no change	3D only
0	all	off	off	all
1	all	on	no change	all
2	4128/29	no change	on	3D only
3	4128/29	on	on	3D only

Blinking. The terminal blinks a visible segment by making it visible and invisible. An invisible segment does not blink. This does not affect the segment's visibility attribute. Whether the segment is actually blinked depends on the current fixup level of the terminal. (See the SET-FIXUP-LEVEL command.)

A segment blinks only in the current view. If a segment is blinking when views are changed, the segment will stop blinking and be visible or invisible depending on where the segment was in the blinking cycle. Blinking will carry over into the new view if the segment is visible in that new view.

If more than one segment is being blinked in the current view, all segments are made visible and then all are made invisible alternately. On a 4129, if the view's object-surface-display attribute is "shaded-surface" or "hidden-line", then a "wireframe" version of the segment is blinked in XOR writing mode. See the SET-SEGMENT-WRITING-MODE command for details on how segments with a *writing-mode* of XOR are displayed.

An open segment does not blink (if blinking is on for segment -2), but will start blinking when it is closed.

XOR Facet Display. On the 4128 and 4129, this mode determines how lines in 3D segments are displayed when the *writing-mode* is XOR. See the SET-SEGMENT-WRITING-MODE command for details.

Outlining. On the 4129, when outlining is "on" and a 3D segment is displayed in a view whose object-surface-display attribute is "shaded-surface", the edges of 3D facets which would be drawn in wireframe or hidden-line mode will be "outlined" by drawing them in the line-index in effect for the outlined primitive. When outlining is "off", these edges are not drawn. Only "shaded-surface" images are affected.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SH11

highlighting
as shipped — 0 for segment -2
on power-up — 0 for segment -2
if omitted — 0

ERRORS

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.*
SH13	(Level 2):	A 3D *highlighting* value is used for a 2D segment.
SH21	(Level 2):	Invalid *highlighting.*

REFERENCES

DRAW-FACETS command
SET-FIXUP-LEVEL command
SET-LINE-INDEX command
SET-SEGMENT-WRITING-MODE command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SEGMENT-IMAGE-TRANSFORM Command

Host Syntax

```
EcSI  int:segment-number
      real:x-scale-factor
      real:y-scale-factor
      real:rotation-angle
      xy:position
```

Setup Syntax

```
EcSI Sp  segment-number
         x-scale-factor
         y-scale-factor
         rotation-angle
         position
```

PARAMETERS

segment-number (–3 to –1, 0, or 1 to 32767)
Names the segment for which an image transform is being specified.

–3	all segments that match the current matching class
–2	the future segment
–1	all segments
0	the crosshair cursor (2D only)
1 to **32767**	a specific segment

x-scale-factor (–32767.0 to +32767.0)
The factor by which the segment is scaled in the x-direction. 0 is not valid.

y-scale-factor (–32767.0 to +32767.0)
The factor by which the segment is scaled in the y-direction. 0 is not valid.

rotation-angle (–32767.0 to +32767.0)
The counterclockwise rotation angle, in degrees. (A negative number specifies a clockwise rotation.)

position
Specifies where in terminal space the segment's pivot point is located. The valid ranges for the different coordinate modes are:

coordinate mode 0:	X,Y = 0 to 4095;
coordinate mode 1:	X,Y = -2^{31} to $2^{31}-1$

DESCRIPTION

This command transforms a 2D segment as follows:

1. First, the segment is scaled in the x- and y-directions away from the segment's pivot point by the amounts specified by the x and y scale factors. A negative scale factor indicates an inversion about the appropriate axis.

2. Next, the segment is rotated counterclockwise about its pivot point by the number of degrees specified in the rotation parameter. If this parameter is negative, then the segment is rotated clockwise.

3. Finally, the segment is moved (translated) so that its pivot point is at the position specified by the position xy parameter.

Image transform operations are not cumulative. They always start at the size and position of the segment as originally defined.

When the segment is next displayed, the xyz coordinates of points in primitives such as vectors and facets within segments are transformed about the segment's pivot point. This transformation can be described as follows: Let the matrix

$$\begin{bmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{bmatrix}$$

represent the segment's current matrix. Then the point (X,Y,Z) is transformed about the segment's pivot point (P_x,P_y,P_z) into the point (X′,Y′,Z′) as follows:

$$(X' - P_x) = (X - P_x) \bullet S_{11} + (Y - P_y) \bullet S_{21} + (Z - P_z) \bullet S_{31}$$

$$(Y' - P_y) = (X - P_x) \bullet S_{12} + (Y - P_y) \bullet S_{22} + (Z - P_z) \bullet S_{32}$$

$$(Z' - P_z) = (X - P_x) \bullet S_{13} + (Y - P_y) \bullet S_{23} + (Z - P_z) \bullet S_{33}$$

The pixel representations of alphatext, string-precision graphtext, and markers within segments are not scaled or rotated although their positions are. The current fixup level determines when the screen is updated to remove the old image and display the new image of the transformed segment. (See the SET-FIXUP-LEVEL command for details.) If the transformed image is displayed immediately, it is displayed in wireframe mode, even if the view's object-surface-display attribute is "shaded-surface" or"hidden-line."

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist. Segment –1 refers to all currently defined 2D segments (with numbers from 1 to 32767). Segment –3 refers to all 2D segments that match the current segment matching class. Segment –2 refers to future 2D segments at the time they are opened.

The position of 2D segments –3, –2, –1, 0, and 1 to 32767 may also be changed by the SET-SEGMENT-POSITION command. The position of segments 1 to 32767 are changed when used as the graphic input cursor. (See the SET-GIN-CURSOR command.) The position of segment –2 is also changed by the SET-PIVOT-POINT command.

When a segment is being used as the GIN cursor in a GIN operation, then the segment's position may be changed by the SET-SEGMENT-IMAGE-TRANSFORM command:

- For an Absolute mode GIN device (that is, Device Codes 1, 3, 4, and 5; see ENABLE-GIN command), the current graphic input location *is not changed*. The graphic input location will be updated and the segment serving as the GIN cursor will be moved back to it the next time the operator moves the GIN device.
- For Relative mode GIN devices (Device Codes 0, 6, 8, and 9), the current graphic input location *is changed*, and the cursor moves from the new position the next time the operator moves the GIN device.

The scale and rotation of a 2D segment can also be set by the SET-SEGMENT-SCALE-ROTATION command and includes future called 2D segments.

If the internal stack is nearly full when you attempt to use this command (due to nested CALL-SEGMENT commands or the INCLUDE-COPY-OF-SEGMENT commands), the internal stack may overflow causing an error to be detected.

If a segment is scaled too large relative to the current window/viewport transformation, internal calculations will result in singularities and an error will be detected.

When either of the above errors is detected the segment that caused the error is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when a segment is deleted.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SI11

x-scale-factor
as shipped — 1.0 segment -2
on power-up — 1.0 segment -2
if omitted — 0.0

y-scale-factor
as shipped — 1.0 segment -2
on power-up — 1.0 segment -2
if omitted — 0.0

rotation-angle
as shipped — 0.0 segment -2
on power-up — 0.0 segment -2
if omitted — 0.0

position
as shipped — (0,0) segment -2
on power-up — (0,0) segment -2
if omitted — (0,0)

ERRORS

SI02	(Level 3):	Out of memory while transforming segment.
SI03	(Level 2):	Command is invalid at this time. (The specified segment is currently being defined.)
SI03	(Level 3):	Internal transformation calculation can not be performed.
SI09	(Level 3):	Internal stack overflow.
SI10	(Level 2):	Segment does not exist.
SI11	(Level 2):	Invalid *segment-number.*
SI13	(Level 2):	Segment is not a 2D segment.
SI21	(Level 2):	Invalid *x-scaling-factor.*
SI31	(Level 2):	Invalid *y-scaling-factor.*
SI41	(Level 2):	Invalid *rotation-angle.*
SI51	(Level 2):	*Position* out of valid range.

REFERENCES

SET-FIXUP-LEVEL command
SET-PIVOT-POINT command
SET-SEGMENT-POSITION command
SET-SEGMENT-SCALE-ROTATION command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-MATRIX-TRANSFORM Command

4128, 4129

Host Syntax

```
EcSW   int:segment
       int:xform-mode
       real-array:xform-matrix
       xy + -array:points
```

Setup Syntax

```
EcSW Sp   segment
          xform-mode
          xform-matrix
          points
```

PARAMETERS

segment

The segment for which a matrix-transform is being specified. Valid segment numbers are:

–5	called 3D segments
–3	all 3D segments that match the current matching class
–2	future 3D segments
–1	all 3D segments
1 to **32767**	a specific 3D segment

xform-mode

Specifies how the transform is to be applied.

0 = set transform mode

1 = concatenate transform mode

xform-matrix (length 0, 1, or 9; elements –32767.0 to 32767.0)

Specifies the 3x3 transform matrix.

points (length 0 or 1; XYZ = -2^{23} to $2^{23}-1$)

Specifies the segment position.

DESCRIPTION

This command sets a 3D segment's matrix-transform and position as follows:

1. If *xform-mode* is 0, the segment's matrix is set to the values in *xform-matrix*. If the length of *xform-matrix* is 0, no change occurs. If the length of *points* is 1, the segment's position is set to the values in *points*. If the length of *points* is 0, the segment's position is not changed.
2. If *xform-mode* is 1, the segment's current matrix is post-multiplied by *xform-matrix*. If length of *xform-matrix* is 0, no change occurs. If the length of *points* is 1, the xyz values in *points* are added to the segment's current position. If length of *points* is 0, the segment's position is not changed.

If *xform-matrix* contains 1 element, the matrix used in 1 and 2 above is the 3x3 identity matrix multiplied by this element.

When the segment is next displayed, the xyz coordinates of points in primitives such as vectors and facets within segments are transformed about the segment's pivot point. The pixel representations of alphatext, string-precision graphtext, and markers within segments are not scaled or rotated although their positions are. The current fixup level determines when the screen is updated to remove the old image and display the new image of the transformed segment. (See the SET-FIXUP-LEVEL command for details.) If the transformed image is displayed immediately, it is displayed in wireframe mode, even if the view's object-surface-display attribute is "shaded-surface" or "hidden-line."

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist. Segment –1 refers to all currently defined 3D segments (with numbers from 1 to 32767). Segment –3 refers to all 3D segments that match the current segment matching class. Segment –5 refers to called 3D segments at the time they are called and is used only to set up transforms. Segment –2 refers to future 3D segments at the time they are opened.

The position of 3D segments –3, –2, –1, and 1 to 32767 may also be changed by the SET-SEGMENT-POSITION command when the terminal is in coordinate mode 2. The position of a user-defined segment is changed when it is used as the GIN cursor. (See the SET-GIN-CURSOR command.) The position of segment –2 is also changed by the SET-PIVOT-POINT command.

When a segment is being used as the GIN cursor in a GIN operation, the segment's position may be changed by the SET-SEGMENT-MATRIX-TRANSFORM command:

- For an Absolute mode GIN device (that is, Device Codes 1, 3, 4, and 5; see ENABLE-GIN command), the current graphic input location *is not changed.* The graphic input location will be updated and the segment serving as the GIN cursor will be moved back to it the next time the operator moves the GIN device.
- For Relative mode GIN devices (Device Codes 0, 6, 8, and 9), the current graphic input location *is changed,* and the cursor moves from the new position the next time the operator moves the GIN device.

If the segment specified is currently being defined or is a 2D segment, or the terminal's coordinate mode is not 2, an error is generated and no further processing takes place.

If the internal stack is nearly full when you attempt to use this command (due to nested CALL-SEGMENT commands or the INCLUDE-COPY-OF-SEGMENT commands), the internal stack may overflow causing an error to be detected.

If a segment is scaled too large relative to the current window/viewport transformation, internal calculations will result in singularities and an error will be detected.

When either of the above errors is detected the segment that caused the error is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when a segment is deleted.

DEFAULTS

segment
as shipped — 0
on power-up — 0
if omitted — error SW11 level 2

xform-mode
as shipped — 0
on power-up — 0
if omitted — 0

xform-matrix
as shipped — 3x3 identity matrix for segment -2
on power-up — 3x3 identity matrix for segment -2
if omitted — no change

points
as shipped — (0,0,0)
on power-up — (0,0,0)
if omitted — no change

ERRORS

SW00	(Level 0):	Unrecognized command (the terminal is not a 4128 or 4129).
SW02	(Level 3):	Out of memory while transforming segment.
SW03	(Level 2):	Command is invalid at this time (coordinate mode 2 is not in effect or specified segment is currently being defined).
SW03	(Level 3):	Internal transformation calculation can not be performed.
SW09	(Level 3):	Internal stack overflow.
SW10	(Level 2):	Segment does not exist.
SW11	(Level 2):	Invalid *segment* number.
SW13	(Level 2):	Segment is not a 3D segment.
SW21	(Level 2):	Invalid *xform-mode.*
SW31	(Level 2):	Invalid *xform-matrix.*
SW41	(Level 2):	Invalid *points* array.

REFERENCES

CALL-SEGMENT command
INCLUDE-COPY-OF-SEGMENT command
SET-FIXUP-LEVEL command
SET-GIN-CURSOR command
SET-PIVOT-POINT command
SET-SEGMENT-POSITION command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-POSITION Command

Host Syntax

EcSX int:*segment-number* xy+:*position*

Setup Syntax

EcSX Sp *segment-number* *position*

PARAMETERS

segment-number (–3 to 32767)
Names the segment whose position is being specified.

–3	all segments that match the current matching class and whose dimensionality matches the terminal's coordinate mode
–2	future segments
–1	all defined segments whose dimensionality matches the terminal's coordinate mode
0	the crosshair GIN cursor
1 to **32767**	a specific segment

position
Specifies where in terminal space the segment is to be positioned. The valid ranges for the different coordinate modes are:

coordinate mode 0:	X,Y	= 0 to 4095
coordinate mode 1:	X,Y	= -2^{31} to $2^{31}-1$
coordinate mode 2:	X,Y, Z	= -2^{23} to $2^{23}-1$

DESCRIPTION

This command moves the specified segment so that its pivot point is at the position specified by *position.* A 2D (including segment 0) segment can only be positioned when the terminal is in coordinate mode 0 or 1. A 3D segment can only be positioned when the terminal is in coordinate mode 2. The image of the segment will be repositioned immediately or the next time the view is renewed, depending on the fixup level of the terminal. (See the SET-FIXUP-LEVEL command.) If a 3D transformed image is displayed immediately, it is displayed in wireframe mode even if the view's object-surface-display attribute is "shaded-surface" or "hidden-line."

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist or if the dimensionality of the specified segment does not match the coordinate mode of the terminal.

Segment 0 refers to the crosshair GIN cursor. Segment –1 refers to all currently defined segments (with numbers from 1 to 32767) whose dimensionality matches the current coordinate mode. Segment –3 refers to all segments that match the current segment matching class and whose dimensionality matches the current coordinate mode of the terminal. Segment –2 refers to future segments at the time they are opened. The 4128 and 4129 have two future segment *positions,* one for 2D and one for 3D.

The position of segments –3, –2, –1, and 1 to 32767 may also be changed by the SET-SEGMENT-IMAGE-TRANSFORM command (2D segments) or by the SET-SEGMENT-MATRIX-TRANSFORM command (3D segments). The position of segments 1 to 32767 are changed when used as the graphic input cursor. (See the SET-GIN-CURSOR command.) The position of segment –2 is also changed by the SET-PIVOT-POINT command.

When a segment is being used as the GIN cursor in a GIN operation, the segment's position may be changed by the SET-SEGMENT-POSITION command:

- For an Absolute mode GIN device (that is, Device Codes 1, 3, 4, and 5; see ENABLE-GIN command), the current graphic input location *is not changed.* The graphic input location will be updated and the segment serving as the GIN cursor will be moved back to it the next time the operator moves the GIN device.
- For Relative mode GIN devices (Device Codes 0, 6, 8, and 9), the current graphic input location *is changed,* and the cursor moves from the new position the next time the operator moves the GIN device.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — 0

position
as shipped — (0,0) (2D) segment –2;
(0,0,0) (3D) segment –2
on power-up — (2D) segment –2;
(0,0,0) (3D) segment –2
if omitted — (0,0) (2D); (0,0,0) (3D)

ERRORS

SX02	(Level 3):	Out of memory.
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*.
SX13	(Level 2):	The specified segment's dimensionality does not match the current coordinate mode of the terminal.
SX21	(Level 2):	*Position* out of valid range.

REFERENCES

SET-COORDINATE-MODE command
SET-FIXUP-LEVEL command
SET-GIN-CURSOR command
SET-PIVOT-POINT command
SET-SEGMENT-IMAGE-TRANSFORM command
SET-SEGMENT-MATRIX-TRANSFORM command

INQUIRY COMMAND

Report-SEGMENT-STATUS command

SET-SEGMENT-SCALE-ROTATION Command

4111, 4115P51, or 4120 SERIES

Host Syntax

```
EcSJ  int:segment-number
      real:x-scale-factor
      real:y-scale-factor
      real:rotation-angle
```

Setup Syntax

```
EcSJ Sp  segment-number   x-scale-factor
         y-scale-factor   rotation-angle
```

PARAMETERS

segment-number (–5, –3, –2, –1, 1 to 32767)
Names the segment to be scaled and/or rotated.

–5	future called 2D segment
–3	all 2D segments that match the current matching class
–2	future 2D segments
–1	all 2D segments
1 to **32767**	a specific 2D segment

x-scale-factor (–32767.0 to 32767.0)
The factor to scale the segment in the x-direction.

y-scale-factor (–32767.0 to 32767.0)
The factor to scale the segment in the y-direction.

rotation-angle (–32767.0 to 32767.0)
The counterclockwise rotation angle in degrees. A negative number specifies clockwise rotation.

DESCRIPTION

This command sets the scale and rotation parts of a 2D segment's image transformation. See the SET-SEGMENT-IMAGE-TRANSFORM command for details.

Segment –5 defines a scale factor and rotation angle for future segment subroutine calls. The CALL-SEGMENT command specifies the use of segment –5 scale factors and rotation angle.

If the coordinate mode of the terminal does not match the dimensionality of the segment, an error is detected.

If the internal stack is nearly full when you attempt to use this command (due to nested CALL-SEGMENT commands or the INCLUDE-COPY-OF-SEGMENT commands), the internal stack may overflow causing an error to be detected.

If a segment is scaled too large relative to the current window/viewport transformation, internal calculation will result in singularities and an error will be detected.

When either of the above errors is detected the segment that caused the error is not displayed. This condition occurs each time the segment is made visible or invisible, such as during a PAGE operation or when a segment is deleted.

The error reported is SJ03 (Level 3) only when the transform is applied to Segment –5 during a call operation. For all other conditions the error detected is SI03 (Level 3).

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — error SJ11

x-scale-factor
as shipped — 1.0
on power-up — 1.0
if omitted — 0.0

y-scale-factor
as shipped — 1.0
on power-up — 1.0
if omitted — 0.0

rotation-angle
as shipped — 0.0
on power-up — 0.0
if omitted — 0.0

ERRORS

SJ02	(Level 3):	Out of memory while transforming segment.
SJ03	(Level 2):	The specified segment is open or coordinate mode is 2.
SJ03	(Level 3)	Internal transformation calculation cannot be performed.
SJ09	(Level 3):	Internal stack overflow.
SJ10	(Level 2):	Segment does not exist.
SJ11	(Level 2):	Invalid *segment number.*
SJ13	(Level 2):	The segment is not a 2D segment.
SJ21	(Level 2):	Invalid *x-scaling-factor.*
SJ31	(Level 2):	Invalid *y-scaling-factor.*
SJ41	(Level 2):	Invalid *rotation-angle.*

REFERENCES

CALL-SEGMENT command
SET-SEGMENT-IMAGE-TRANSFORM command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-VISIBILITY Command

Host Syntax

$^{E}_{C}$**SV** int:*segment-number* int:*visibility*

Setup Syntax

$^{E}_{C}$**SV** $^{S}_{P}$ *segment-number* *visibility*

This is the correction for Section 7-287

PARAMETERS

segment-number (–3 to 32767)
Names the segment whose visibility is being specified.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
0	the crosshair cursor
1 to **32767**	a specific segment

visibility (0 or 1)
Specifies whether a segment is visible in the current view or not.

0	makes segment invisible
1	makes segment visible

DESCRIPTION

This command sets the visibility attribute of the specified segment (or segments) in the current view to "visible" or "invisible".

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist.

Segment 0 refers to the crosshair GIN cursor. Segment –1 refers to all currently defined segments (with numbers from 1 to 32767). Segment –3 refers to all segments that match the current segment matching class. Segment –2 refers to future segments at the time they are opened.

When segments –1 or –3 are made visible, they are made visible in display priority order, so that segments with higher priority appear in front of segments with lower priority. All 3D segments are made visible before any 2D segments, regardless of display priority.

The segment(s) will be made visible or invisible either immediately or the next time the view is renewed, depending on the fixup level of the terminal.

When the current view's object-display-attribute (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is "shaded-surface" or "hidden-line", the wireframe image of a 3D segment is made visible or invisible immediately; the "shaded-surface" or "hidden-line" image is changed only when the view is renewed.

Each segment contains a list of the views in which the segment is visible. If segment –2 is visible when a segment is opened, then when the segment is closed, the current view is added to this list. A view is also added to the list by selecting the view and making the segment visible. A view is removed from the list by selecting the view and making the segment invisible.

When a segment becomes an active GIN cursor, its visibility attribute is saved and the segment is then made visible. The SET-SEGMENT-VISIBILITY command can then change the visibility of an active GIN cursor if it is in the current view, but the new visibility attribute applies only while the segment continues to act as the GIN cursor. When GIN is disabled or a SET-GIN-CURSOR command selects a different segment as the GIN cursor, the visibility attribute that the segment had before becoming the GIN cursor is restored. See SET-GIN-CURSOR for more information.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — 0

visibility
as shipped — 1 for segment –2
on power-up — 1 for segment –2
if omitted — 0

ERRORS

SV02	(Level 3):	Out of memory.
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*.
SV21	(Level 2):	Invalid *visibility*.

REFERENCES

PAGE command
RENEW-VIEW command
SET-FIXUP-LEVEL command
SET-GIN-CURSOR command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SEGMENT-WRITING-MODE Command

Host Syntax

$^{E}_{C}$**SM** int:*segment-number* int:*writing-mode*

Setup Syntax

$^{E}_{C}$**SM** $^{S}_{P}$ *segment-number* *writing-mode*

PARAMETERS

segment-number (–3 to 32767)
Names the segment for which a writing mode is being specified.

–3	all segments that match the current matching class
–2	future segments
–1	all segments
0	the crosshair cursor
1 to **32767**	a specific segment

writing-mode (1 to 4)

1	SET mode
2	XOR mode
3	AND mode (2D only)
4	OR mode (2D only)

DESCRIPTION

This command specifies the writing mode for the specified segment. If the segment is visible, this command may make the segment invisible with its current writing mode, and visible again with its new writing mode, depending on the fixup level of the terminal. (See the SET-FIXUP-LEVEL command for details.)

NOTE

When the fixup level is set to 4, changing a 2D or 3D wireframe segment's writing mode from SET to XOR causes the segment's image to disappear. Renewing the view or making the segment invisible and visible again causes the segment to reappear.

A segment number in the range from 1 to 32767 specifies a single user-defined segment. An error is detected if the specified segment does not exist or if it is a 3D segment and *writing-mode* is 3 or 4. Segment 0 refers to the crosshair GIN cursor. If segment 0 is specified, no action occurs and no error is reported. Segment –1 refers to all currently defined segments (with numbers from 1 to 32767). Segment –3 refers to all segments that match the current segment matching class. Segment –2 refers to future segments at the time they are opened. If *writing-mode* is 3 or 4 and segment –1 or –3 is specified, only 2D segments are affected.

If the *segment-number* is –2 (future segments), this command also sets the writing mode for all graphic primitives, whether a segment is open or not.

The writing mode of a segment is analogous to the ALU mode of pixel operations. (See the BEGIN-PIXEL-OPERATIONS command.)

Mode 1 (SET Mode). The index of each pixel in the image of the graphic primitive being displayed is set to the color index of the graphic primitive. Mode 1 is equivalent to pixel ALU mode 11. A SET mode segment is "undrawn" (made invisible) by redrawing it with all primitive indices set to the view's wipe index.

Mode 2 (XOR (EXCLUSIVE OR) Mode). The index of each pixel in the image of the graphic primitive being displayed is set to the XOR of its current index and the index of the graphic primitive. Mode 2 is equivalent to pixel ALU mode 7. An XOR mode segment is "undrawn" (made invisible) by redrawing it in XOR mode again.

In the 4129, a 3D segment displayed in a view with an object-surface-display attribute of "shaded-surface" or "hidden-line" will always be drawn in SET mode when a PAGE or RENEW-VIEW command is executed (even if its *writing-mode* is XOR). Also, changes made to the segment's visibility or position when the fixup level is set to 4 or 6 *will not* cause the segment's "shaded-surface" or "hidden-line" image to disappear. These changes will cause the "wireframe" image of the segment to be drawn in XOR mode, just as it would be if the view's object-surface-display attribute were "wireframe."

Mode 3 (AND Mode; 2D segments only). The index of each pixel in the image of the graphic primitive being displayed is set to the AND of its current index and the index of the graphic primitive. Mode 3 is equivalent to pixel ALU mode 12. A segment is "undrawn" (made invisible) by redrawing it with all primitive indices set to the view's wipe index.

Mode 4 (OR Mode; 2D segments only). The index of each pixel in the image of the graphic primitive being displayed is set to the OR of its current index and the index of the graphic primitive. Mode 3 is equivalent to pixel ALU mode 15. A segment is "undrawn" (made invisible) by redrawing it with all primitive indices set to the view's wipe index.

DEFAULTS

segment-number
as shipped — none
on power-up — none
if omitted — 0

writing-mode
as shipped — 1 for segment -2
on power-up — 1 for segment -2
if omitted — error SM21

ERRORS

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*.
SM13	(Level 2):	*Writing-mode* cannot be 3 or 4 for 3D segments.
SM21	(Level 2):	Invalid *writing-mode*.

REFERENCES

BEGIN-PIXEL-OPERATIONS command
PAGE command
RENEW-VIEW command
SET-FIXUP-LEVEL command
SET-VIEW-DISPLAY-ATTRIBUTES command

INQUIRY COMMAND

REPORT-SEGMENT-STATUS command

SET-SNOOPY-MODE Command

Host Syntax

EcKS int:*snoopy-mode*

Setup Syntax

SNOOPY Sp *snoopy-mode*

PARAMETERS

snoopy-mode (0 or 1)
Specifies whether the terminal is to be in or out of Snoopy mode. Setup mode parameters are **YES** and **NO**.

0 **NO**; in Setup mode, a **SNOOPY NO** command removes the terminal from Snoopy mode. (The terminal cannot be removed from Snoopy mode with the escape-sequence version of this command, because escape-sequence commands are ignored in Snoopy mode.)

1 **YES**; puts the terminal in Snoopy mode.

DESCRIPTION

A SET-SNOOPY-MODE: 1 command puts the terminal in Snoopy mode. This lets the operator view control characters received from the host (or typed on the keyboard in LOCAL mode).

Control characters such as Ec, Gs, Us, etc., are processed like any other printing ASCII character. (Each control character has a corresponding Snoopy mode mnemonic character which is displayed on the screen.) The CR character, however, is not only displayed, but also advances the cursor to the start of the next line.

It is impossible to execute commands from the host or the keyboard (except in SETUP mode) while Snoopy mode is in effect.

Only the operator can remove the terminal from Snoopy mode. The operator can do this in two ways:

- The operator can press the CANCEL key; among other effects, this key removes the terminal from Snoopy mode. (See the descriptions of the CANCEL key and the CANCEL command for details.)
- The operator can press the SETUP key to put the terminal in Setup mode. Then, while in Setup mode, the operator would type the **SNOOPY NO** command.

Typing **SNOOPY NO** in Setup mode not only removes the terminal from Snoopy mode, but also puts the terminal in alpha mode.

DEFAULTS

snoopy-mode
as shipped — 0
on power-up — 0
if omitted — 1

ERRORS

KS11 (Level 2): Invalid *snoopy-mode* (must be 0 or 1).

REFERENCES

CANCEL command
CANCEL key
ENTER-ALPHA-MODE command
SETUP key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-STOP-BITS Command

Host Syntax

$^{E}{}_{C}$**NB** int:*number-of-stop-bits*

Setup Syntax

STOPBITS $^{S}{}_{P}$ *number-of-stop-bits*

PARAMETERS

number-of-stop-bits (1 or 2)
Specifies the number of stop bits appended to each character transmitted from the terminal.

DESCRIPTION

This command specifies the number of stop bits appended to each character transmitted from the terminal to the host.

DEFAULTS

number-of-stop-bits
as shipped — 1
on power-up — remembered
if omitted — error NB11

ERRORS

NB11 (Level 2): Invalid *number-of-stop-bits.*

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SURFACE-COLOR-MAP Command

Host Syntax

$^{E}{}_{C}$**TG** int:*surface-number*
int-array:*color-mixtures*

int-array:*color-mixtures* = int:*count*
[*quadruple* . . .]

quadruple = int:*color-index*
int:*first-color-coordinate*
int:*second-color-coordinate*
int:*third-color-coordinate*

Setup Syntax

CMAP $^{S}{}_{P}$ *surface-number color-mixtures*

color-mixtures = {*quadruple* . . .

quadruple = *color-index*
first-color-coordinate
second-color-coordinate
third-color-coordinate

PARAMETERS

surface-number (–1, or 1 to 8)
Names the surface for which color mixtures are being defined. -1 means a "super surface" consisting of all bit planes of all surfaces presently defined.

color-mixtures array (length 0 to 32768)
An array of int parameters, which are considered in groups of four (quadruples). The first int in each quadruple names a color index, while the following three ints specify the color mixture for that color index. The color mixture is specified in the HLS, RGB, CMY, or Machine RGB color coordinate system, according to the most recent SET-COLOR-MODE command. A blinking color can be specified by adding 1000 to the third coordinate of an index.

If the first int in a quadruple is 0, then the following three ints specify the color of the background which is behind all the writing surfaces.

The valid ranges for the first, second, and third coordinates in each system are, respectively:

HLS:
–2767 to 32767; 0 to 100; 0 to 100 or 1000 to 1100

RGB and CMY:
0 to 100; 0 to 100; 0 to 100 or 1000 to 1100

Machine RGB:
0 to 255; 0 to 255; 0 to 255 or 1000 to 1255.

DESCRIPTION

The SET-SURFACE-COLOR-MAP command sets the "color map" for a particular surface. That is, it determines the mapping from that surface's color indices to particular color mixtures.

This command also sets the surface gray levels to NTSC gray levels equivalent to the colors set using the following function:

Gray level = 30% (Red level) + 59% (Green level) + 11% (Blue level)

The resulting gray level value is always rounded to the nearest integer.

The effect of the SET-SURFACE-COLOR-MAP command continues until superseded by another SET-SURFACE-COLOR-MAP or SET-SURFACE-GRAY-LEVELS command, until surfaces are redefined with a SET-SURFACE-DEFINITIONS command, or until the terminal is reset (by the reset command or by pressing the RESET button) or turned off.

In this command, surface –1 means a "super-surface" consisting of all bit planes in all surfaces presently defined. This is for use in advanced applications, such as controlling the exact color displayed where images on one surface overlap images on another surface. If you plan to use this "super-surface" feature, be sure to read the *4110/4120 Series Host Programmers Manual*, which explains the super-surface and its side effects.

Adding 1000 to the third color coordinate of the index causes the color to blink by alternately becoming visible and invisible. For example, in the HLS color mode a normal red is indicated by (120,50,100), and a blinking red is given by (120,50,1100).

Appendix E lists the color coordinates in RGB mode of the 256 default colors.

ERRORS

TG10	(Level 2):	Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).
TG11	(Level 2):	Invalid *surface-number*.
TG21	(Level 2):	Invalid *color-mixtures* array.
TG22	(Level 3):	Out of memory while parsing parameter.

REFERENCES

Appendix E, "Color Coordinates"
SET-BACKGROUND-COLOR command
SET-BACKGROUND-GRAY-LEVEL command
SET-COLOR-MODE command
SET-SURFACE-DEFINITIONS command
SET-SURFACE-GRAY-LEVELS command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SURFACE-DEFINITIONS Command

Host Syntax

^{E}c**RD** int-array:*surface-defs*

int-array:*surface-defs* = int:*number-of-surfaces*
[*surface-specifier . . .*]

surface-specifier = int:*number-of-bit-planes*

Setup Syntax

^{E}c**RD** ^{S}P *surface-defs*

surface-defs = [*surface-specifier . . .*]

surface-specifier = *number-of-bit-planes*

PARAMETERS

surface-defs (length 1 to 8; ints 0 to 8)
An array of ints specifying the number of bit planes for each surface. The first int in the array (the array count) tells how many writing surfaces the terminal is to have. Subsequent ints specify the number of bit planes for their respective surfaces.

DESCRIPTION

This command erases the screen and sets the number of bit planes in each surface. It also initializes the surface color map, the surface gray levels, surface priorities, and surface visibilities.

Here, the "number of bit planes" in each surface determines the maximum color index which may be written into pixels on that surface. If there are N bit planes in a surface, the maximum color index is $2^{N}-1$.

If a surface is currently defined as the current view surface, dialog area surface, or pixel operation surface and is not specified in the *surface-defs* array, an error is detected. If more planes are specified than can be installed in the terminal, an error is detected.

Example. To assign one bit plane to surface one, and two bit planes to surface two, issue the following command:

SET-SURFACE-DEFINITIONS:(1, 2)
= ^{E}c**RD** int-array : (1,2)
= ^{E}c**RD** int:2 int:1 int:2
= ^{E}c**RD212**

In this example, surface 3 does not exist; an error will occur if you attempt to reference surface 3 in another command.

Another Example. To assign one bit plane to surface 1, zero bit planes to surface 2, and two bit planes to surface 3, issue the following command:

SET-SURFACE-DEFINITIONS: (1, 0, 2)
= ^{E}c**RD** int-array : (1,0,2)
= ^{E}c**RD int:3 int:1 int:0 int:2**
= ^{E}c**RD3102**

In this example, surface 2 has zero bit planes. It is, however, possible to create a view whose viewport is on surface 2. In that case, nothing in that view would be visible on the screen. This may be useful for double buffering. Segments can be built in a view whose viewport is on a surface to which zero bit planes are assigned. Later, another SET-SURFACE-DEFINITIONS command can assign bit planes to that surface, and the segments become visible (when the view is renewed).

Appendix E lists the color coordinates in RGB mode of the 256 default colors.

Initializing Visibility. The SET-SURFACE-DEFINITIONS command causes each surface it defines to be visible, as if a SET-SURFACE-VISIBILITY command were issued for that surface.

Initializing Surface Priorities. The SET-SURFACE-DEFINITIONS command assigns the following default priorities to the surfaces it defines. Surface one is in front; surface two, if it exists, is behind surface one; surface three, if it exists is behind surface two; and so on. You can change this ordering with the SET-SURFACE-PRIORITIES command.

DEFAULTS

surface-defs
as shipped — 2, 4, 6, or 8, depending on the number of bit planes installed
on power-up — 2, 4, 6, or 8, depending on the number of bit planes installed
if omitted — error RD11

ERRORS

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 *surface-defs* array.

RD12 (Level 3): Out of memory while trying to parse parameter.

REFERENCE

Appendix E, *Color Coordinates*
SET-SURFACE-COLOR-MAP command
SET-SURFACE-GRAY-LEVELS command
SET-SURFACE-PRIORITIES command
SET-SURFACE-VISIBILITY command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SURFACE-GRAY-LEVELS Command

Host Syntax

```
EcRG   int:surface-number
       int-array:indices-and-gray-levels
```

Setup Syntax

```
GRAY Sp   surface-number
          indices-and-gray-levels
```

PARAMETERS

surface-number (−1, 1 to 8)
Names the surface for which gray-levels are being specified.

indices-and-gray-levels (length 0 to 32768)
This array holds an even number of *int* parameters. The *int*s are considered in pairs, with the first *int* in each pair being a color index, and the second *int* in the pair specifying the gray-level (range 0 to 100 and 1000 to 1100 for RGB, CMY, and HLS, 0 to 255 and 1000 to 1255 for Machine RGB) for that color index.

DESCRIPTION

This command defines the color indices for a particular surface, assigning to each color index a corresponding shade of gray (gray-level).

Color-Indices and Gray-Levels. Think of the color indices as ink bottles which can hold different shades of ink. Each writing surface has its own set of ink bottles, into which you can dip your pen when drawing lines on that surface. The number of ink bottles for a given surface is two raised to the power N, where N is the number of bit planes assigned to that surface by the SET-SURFACE-DEFINITIONS command.

Ink bottle number 0 (color index 0) always holds "ink eradicator." That is, whenever you draw a line in color index 0, that line is transparent, wiping out any previous line drawn on that surface.

The SET-SURFACE-GRAY-LEVELS command loads different inks, in shades of gray (gray-levels) into a surface's ink bottles. Another command, SET-SURFACE-COLOR-MAP, lets you load colored inks as well as gray inks into the ink bottles.

If you want all areas displayed in a given gray-level to blink at 1½ cycles per second (alternate between visible and invisible), add 1000 to the gray-level number in the *indices-and-gray-levels* array. That is, if you want an index of gray-level 56 to blink, use 1056 in the array instead of 56.

Surface-Number Parameter. The first parameter is an *int* naming the writing surface for which gray-levels are being specified. That is, it names the surface whose ink bottles (color indices) are to be filled with various inks (shades of gray).

In this command, surface −1 means a super surface of all bit planes in all defined surfaces.

Indices-and-Gray-Levels Parameter. The second of the command's parameters is an *int-array* holding pairs of numbers. The first number in each pair is a color index, while the second number in the pair is a gray-level. The gray levels are in the range 0 to 100 for the RGB, HLS, and CMY color-specifying system and 0 to 255 for the Machine RGB color-specifying system.

The SET-SURFACE-GRAY-LEVELS command sets the surface color map as follows:

RGB System:
R = gray level
G = gray level
B = gray level

CMY System:
C = 100 — gray level
M = 100 — gray level
Y = 100 — gray level

HLS System:
Hue = 0
Lightness = gray level
Saturation = 0

Machine RGB System:
MR = gray level
MG = gray level
MB = gray level

See the description of SET-SURFACE-COLOR-MAP for more information.

ERRORS

RG10	(Level 2):	Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).
RG11	(Level 2):	Invalid *surface-number.*
RG21	(Level 2):	Invalid *indices-and-gray-levels* array. (The array count must be even; the first *int* in each pair must be a color index in the range from 1 to 32767; the second *int* in each pair must be a valid gray-level: a number from 0 to 100 and 1000 to 1100).
RG22	(Level 3):	Out of memory while parsing parameter.

REFERENCES

SET-BACKGROUND-COLOR command
SET-BACKGROUND-GRAY-LEVEL command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-DEFINITIONS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SURFACE-PRIORITIES Command

Host Syntax

```
EcRN    int-array:priorities
```

Setup Syntax

```
EcRN Sp    priorities
```

PARAMETERS

priorities (length 2 to 32768 [even])
The *int*s in this array are grouped in pairs. The first *int* in each pair is a surface number, while the second *int* of the pair is a priority number for that surface.

DESCRIPTION

This command determines which of the transparent writing surfaces are "in front" of others, and which are "behind" other surfaces.

Objects (especially filled areas) drawn on one surface will obscure objects drawn on a surface which is "behind" that surface if the SET-COLOR-MODE command's *overlay-mode* setting is OPAQUE rather than SUBTRACTIVE or ADDITIVE.

Surface Priorities Array. The *int-array* parameter holds an even number of integers, which are regarded in pairs. The first integer in each pair is a surface number; the second integer in the pair is a priority number (0 to 9) for that surface. Each surface is "in front of" all surfaces which have a larger priority number than itself, and "behind" all surfaces which have lesser priority numbers.

The resulting priorities, as are visible on the screen and reported in a *terminal-settings-report*, are determined by a combination of current priorities and the priorities set with this command. Surface priorities are set as follows:

1. The surfaces in the command are given their assigned priorities.
2. All surfaces are assigned new priorities so that each surface has a unique priority, with the "front" surface having priority 1. If two surfaces were assigned the same priority, the lower numbered surface is given the higher priority.

For example, if the surfaces are defined with default priorities, and surface one is assigned priority 3, the result is that surface two has priority 1, surface one has priority 2, and surface three has priority 3.

Any surface can be put "in front" of all other surfaces by assigning it priority 0, or "behind" all other surfaces by assigning it priority one greater than the maximum number of possible surfaces for that terminal.

Example. Suppose we want surface 3 to be in front, and surface 2 in back, with surface 1 in between. Then surface 3 should be assigned priority 1, surface 1 should be assigned priority 2, and surface 2 should be assigned priority 3. We would issue the following command:

SET-SURFACE-PRIORITIES : surface 3, priority 1; surface 1, priority 2; surface 2, priority 3
= ^E_C**RN** int-array:(3,1,1,2,2,3)
= ^E_C**RN** int:6 int:3 int:1 int:1 int:2 int:2 int:3
= ^E_C**RN6311223**

Special Cases. If two surfaces are assigned same priority, then the lower-numbered surface is deemed to be in front of the higher-numbered surface. If more than one priority is assigned to the same surface, then the last priority assigned that surface is the one which has effect.

Default. When multiple surfaces are defined (with the SET-SURFACE-DEFINITIONS command), the surface priorities are: Surface 1 is the front surface; surface 2, if it exists, is behind surface 1; surface 3, if it exists, is behind surface 2; and so on.

On power-up only surface 1 exists.

DEFAULTS

priorities
as shipped — 1,1
on power-up — 1,1
if omitted — error RN11

ERRORS

RN10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RN11 (Level 2): Invalid *priorities* array.

RN12 (Level 3): Out of memory while parsing parameter.

REFERENCES

SET-SURFACE-DEFINITIONS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-SURFACE-VISIBILITY Command

Host Syntax

^{E}c**RI** int-array:*surface-numbers-and-visibilities*

Setup Syntax

^{E}c**RI** ^{S}P *surface-numbers-and-visibilities*

PARAMETERS

surface-numbers-and-visibilities (length 2 to 32768 [even])
This *int-array* holds an even number of *int* parameters, which are regarded in pairs.

The first *int* in each pair is a surface number. It must be in the range 1 to 8.

The second *int* in each pair specifies the visibility mode, and must be in the range from 0 to 2:

0 invisible

1 visible

2 blinking (alternating) between visible and invisible

DESCRIPTION

This command changes the visibility of one or more surfaces without affecting surface priorities or the visibility attributes of any segments in the surfaces. Normally, a surface is visible (visibility mode 1). If a surface is invisible (visibility mode 0), then no objects on that surface are displayed. Visibility mode 2 causes a surface to "blink", or alternate between being visible and being invisible at 1 1/2 cycles per second. Blinking indices and blinking surfaces blink in unison; highlighted segments do not necessarily blink in phase with each other or with other blinking graphic objects.

This command is primarily for use in double buffering. It may also be used to provide an alternate, high-speed way to highlight segments. (Rather than issue a SET-SEGMENT-HIGHLIGHTING command in order to cause a segment to blink, instead display the segment in a view whose viewport is on a blinking surface.)

DEFAULTS

surface-numbers-and-visibilities
as shipped — 1,1
on power-up — 1,1
if omitted — error RI11

ERRORS

RI10 (Level 2): A surface in *surface-numbers-and-visibilities* does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RI11 (Level 2): Invalid *surface-numbers-and-visibilities* array.

RI12 (Level 3): Out of memory while parsing parameter.

REFERENCES

SET-SURFACE-DEFINITIONS command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TABLET-HEADER-CHARACTERS Command

4111 or OPTION 10

Host Syntax

$^{E}_{C}$IH int:*character-set-selector*

Setup Syntax

TBHEADERCHARS $^{S}_{P}$ *character-set-selector*

PARAMETERS

character-set-selector (0 or 1)
Specifies whether letters or control characters are used as the header characters used in *4953-tablet-GIN-reports* and the key characters used in *GIN-stroke-reports*. Setup mode parameters are **LETTER** and **CONTROL**.

0 **LETTERS**; **M**, **J**, and **O**

1 **CONTROL**; $^{G}_{S}$, $^{S}_{B}$, and $^{U}_{S}$

DESCRIPTION

This command selects the "header" characters used in *4953-tablet-GIN-reports* and the "key" characters used in *GIN-stroke-reports*.

Header Characters. If the parameter in the SET-TABLET-HEADER-CHARACTERS command is 0 (the default), then the header characters in *4953-tablet-GIN-reports* are **M**,**J**, and **O**.

If the parameter is 1, then the header characters are $^{G}_{S}$, $^{S}_{B}$, and $^{U}_{S}$.

Key Characters. If the parameter in the SET-TABLET-HEADER-CHARACTERS command is 0, then the key characters in *GIN-stroke-reports* are:

- **0**, **Z**, **1**, **2**, or **3** for the first point in a stroke. The character **M** is used with the tablet stylus, and the characters **Z**, **1**, **2**, and **3** are used with the optional tablet puck.
- **J** for subsequent points in a stroke.
- **O** for the last point in a stroke.

A SET-TABLET-HEADER-CHARACTERS: 1 command, however, can change these as follows:

- **0**, **Z**, **1**, **2**, or **3** for the first point in a stroke. (These characters do not change.)
- $^{S}_{B}$ for subsequent points in a stroke.
- $^{U}_{S}$ for the last point in a stroke.

For more details, see the *GIN-stroke-report* (in the *GIN-report* message type) and the *4953-tablet-GIN-report* message type descriptions.

DEFAULTS

character-set-selector
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IH00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

IH11 (Level 2): Invalid *character-set-selector*.

REFERENCES

ENABLE-GIN command
ENABLE-4953-TABLET-GIN command
GIN-report message type
4953-tablet-GIN-report message type

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TABLET-SIZE Command

4111 or OPTION 10

Host Syntax

$^{E}_{C}$IN int:*tablet-size*

Setup Syntax

TBSIZE $^{S}_{P}$ *tablet-size*

PARAMETERS

tablet-size (0, 1, 2, or 3)
Specifies the size of the active tablet area. Setup mode parameters are **AUTOMATIC**, **SMALL**, **LARGE**, and **XLARGE**.

0 **AUTOMATIC**; when the tablet is enabled with the ENABLE-GIN command, the LARGE active area is used. When the tablet is enabled with the ENABLE-4953-TABLET-GIN command, the SMALL active area is used.

1 **SMALL**; the 4957 tablet's active area is 10.24 × 10.24 in. or 260 × 260 mm;

2 **LARGE**; the 4957 tablet's active area is 11 × 11 in. or 279 × 279 mm;

3 **XLARGE**; the 4957 tablet's active area is 11.7 × 11.7 in. or 297 × 297 mm.

DESCRIPTION

This command sets the tablet size used during tablet GIN operations.

The 4957 Graphics Tablet has a slightly larger active area than the TEKTRONIX 4953 and 4954 tablets used with earlier Tektronix terminals. This command is included so the larger tablets can emulate the smaller tablets.

The XLARGE size (*tablet-size* = 3) of the active area of the 4957 tablet is 11.7 × 11.7 in. or 297 × 297 mm.

The LARGE size (*tablet-size* = 2) of the active area of the 4957 tablet is 11 × 11 in. or 279 × 279 mm.

The SMALL size (*tablet-size* = 1) of the active area of the 4957 tablet is 10.24 × 10.24 in. or 260 × 260 mm (the same as the 4953 tablet).

In AUTOMATIC size (*tablet-size* = 0), LARGE size is automatically selected when the tablet is enabled with the ENABLE-GIN command and SMALL is automatically selected when the tablet is enabled with the ENABLE-4953-TABLET-GIN command.

DEFAULTS

tablet-size-mode
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

IN00 (Level 0): Unrecognized command. (Must be a 4111 or Option 10 must be installed.)

IN11 (Level 2): Invalid *tablet-size*.

REFERENCES

ENABLE-GIN command
ENABLE-4953-TABLET-GIN command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TABLET-STATUS-STRAP Command

4111 or OPTION 10

Host Syntax

$^{E}_{C}$**IT** int:*strap-setting*

Setup Syntax

TBSTATUS $^{S}_{P}$ *strap-setting*

PARAMETERS

strap-setting (0 or 1)
This parameter emulates the STATUS strap on a TEKTRONIX 4953 or 4954 tablet controller board. Setup mode parameters are **STATUS OUT** and **STATUS IN**.

0 **STATUS OUT**

1 **STATUS IN**

DESCRIPTION

This command helps a 4110/4120-series terminal emulate TEKTRONIX 4010-series terminals which have accessory 4953 or 4954 graphic tablets. The command emulates the STATUS strap on the 4953/4954 tablet controller board. (This strap controls whether not a terminal status byte is sent when the tablet stylus is lifted away from the tablet.)

NOTE

This command only affects the 4953-tablet-status-reports *sent to the host when the terminal has been enabled for graphic input with an ENABLE-4953-TABLET-GIN command. This command does not affect normal GIN operation with the ENABLE-GIN command.*

If the parameter is 0, the terminal emulates the STATUS OUT setting of the 4953 tablet's STATUS strap. In this case, a status byte is not sent when the tablet stylus leaves presence.

If the parameter is 1, the terminal emulates the STATUS IN strap setting. In this case, if the tablet has been enabled for "disable on leaving presence" mode, then a status byte is sent when the tablet stylus leaves presence.

For more information, see the description of the *4953-tablet-status-report* message type. (See also the TEKTRONIX *4953/4954 Graphics Tablet Instruction Manual.*)

DEFAULTS

strap-setting
as shipped — 1
on power-up — 1
if omitted — 1

ERRORS

IT00 (Level 0): Unrecognized command. (Not a 4111 or Option 10, Version 6 or later is not installed.)

IT11 (Level 2): Invalid *strap-setting*.

REFERENCES

ENABLE-4953-TABLET-GIN command
4953-tablet-GIN-report message type
4953/4954 Graphics Tablet Instruction Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TEXT-INDEX Command

Host Syntax

${}^{E}{}_{C}$MT int:*text-index*

Setup Syntax

${}^{E}{}_{C}$MT ${}^{S}{}_{P}$ *text-index*

PARAMETERS

text-index (0 to 32767)
Specifies the pen-index or color index used for subsequent alphatext and graphtext.

DESCRIPTION

This command specifies the color index to be used for subsequent graphtext and alphatext in the graphics area. (See the DISPLAY-ALPHA-TEXT command for details.)

There is a maximum color index for any particular writing surface: one less than 2^M, where M is the number of bit planes assigned to that surface. (See the SET-SURFACE-DEFINITIONS command.) If a text-index greater than the maximum is used, the terminal displays it as if it were the maximum color index for that surface.

DEFAULTS

text-index
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

MT11 (Level 2): Invalid *text-index*.

REFERENCES

DISPLAY-ALPHA-TEXT command
MAP-INDEX-TO-PEN command
SET-DIALOG-AREA-INDEX command
SET-SURFACE-DEFINITIONS command
SET-VIEW-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TRANSMIT-DELAY Command

Host Syntax

$^{E}_{C}$**ND** int:*transmit-delay*

Setup Syntax

XMTDELAY $^{S}_{P}$ *transmit-delay*

PARAMETERS

transmit-delay (0 to 65535)
The number of milliseconds which the terminal is to wait after sending each line of text before it starts transmitting the next line of text.

DESCRIPTION

This command sets the transmit delay. The transmit delay is the delay between when the terminal receives a prompt string or transmits an *EOM-character* and when the terminal sends data to the host. This command is used in conjunction with Prompt mode, *EOM-chars*, and *EOL-strings* to delay transmission from the terminal until the host is ready to accept.

After sending one of the *EOM-characters* (set by the SET-EOM-CHARS command), the terminal pauses a short time before resuming transmission. The SET-TRANSMIT-DELAY time determines the duration of that "short time."

The actual delay time may be up to 25 milliseconds greater than that specified, because of the resolution of the internal timer.

If the terminal receives any characters after a prompt string, but before the transmit delay has elapsed, the prompt string is not recognized as such.

DEFAULTS

transmit-delay
as shipped — 100
on power-up — remembered
if omitted — 0

ERRORS

ND11 (Level 2): Invalid *transmit-delay.*

REFERENCES

ENTER-BYPASS-MODE command
SET-EOM-CHARS command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-TRANSMIT-RATE-LIMIT Command

Host Syntax

```
EcNL   int:rate-limit
```

Setup Syntax

```
XMTLIMIT Sp   rate-limit
```

PARAMETERS

rate-limit (110 to 65535)
Transmit rate limit, in bits per second.

DESCRIPTION

The SET-TRANSMIT-RATE-LIMIT command imposes an upper bound on how fast the terminal may send characters to the host computer. This may be useful in circumstances where the host cannot process characters as fast as the terminal can send them over the communications line.

Example. Suppose the transmit baud rate (set by the SET-BAUD-RATES command) is 1200 bits/sec — about 120 characters per second. Suppose further that a SET-TRANSMIT-RATE-LIMIT: 300 command is sent to the terminal. Then, when transmitting characters to the host, the terminal will send each character at the full data rate (1200 bits/sec), but will space the characters apart so that the average bit rate is no more than 300 bits per second (about 30 characters per second).

DEFAULTS

rate-limit
as shipped — 19200
on power-up — remembered
if omitted — error NL11

ERRORS

NL11 (Level 2): Invalid *rate-limit.*

REFERENCES

SET-BAUD-RATES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-USER-NUMBER Command

Host Syntax

```
EcJU   int:user-number
```

Setup Syntax

```
USERNUMBER Sp   user-number
```

PARAMETERS

user-number (0 to 15)
The user number that the terminal should use when accessing disk files.

DESCRIPTION

This command tells the terminal to access only those files marked with the user number specified by the *user-number* parameter. It also marks all files subsequently created with the same user number.

If you request a directory list with the DIRECTORY command, only those files with the current *user-number* are reported.

If you request a copy or spool operation of the entire contents of a disk, only those files with the current *user-number* are transferred.

DEFAULTS

user-number
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

JU00 (Level 0): Unrecognized command.

JU11 (Level 2): Invalid *user-number*.

REFERENCES

COPY command
DIRECTORY command
SPOOL command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-VIEW-ATTRIBUTES Command

Host Syntax

```
EcRA  int:surface-number   int:wipe-index
      int:border-index
```

Setup Syntax

```
EcRA Sp  surface-number   wipe-index
         border-index
```

PARAMETERS

surface-number (–1 to 8)
Specifies the surface on which the view's viewport is located. –1 specifies the "super-surface."

wipe-index (–1 to 32767)
The color index used for wiping (erasing) the viewport.

–1	viewport not erased
0 to **32767**	a specific index

border-index (0 to 32767)
The color index used for displaying a border around the viewport.

DESCRIPTION

This command sets the surface, wipe index, and border index for the current view. Here, "current view" means the view most recently selected with the SELECT-VIEW command or the NEXTVIEW key.

Surface Number. The first *int* parameter determines on which surface the view is to be displayed: that is, on which surface the viewport for that view exists. The default is surface 1.

If surface 0 is specified, the current surface for the view is left unchanged.

If surface −1 is specified, then a super surface is used. This super surface consists of all bit planes on all of the presently defined surfaces. The super surface is for use in advanced applications. If you plan to use the super surface, be sure to read the *4110/4120 Series Host Programmers Manual,* which explains the super surface and its side effects.

If a surface is specified which does not exist, then an error occurs. (A surface does not exist if it was not defined in the most recent SET-SURFACE-DEFINITIONS command.)

Wipe Index. The second parameter specifies the *wipe-index* for the viewport of the specified view. This is the color index to which all pixels in the viewport are set during an erase operation. The default is color index 0, "transparent."

If *wipe-index* is −1, the viewport is not erased during a page or renew operation.

There is a maximum color index for any particular surface: one less than 2^M, where M is the number of bit planes assigned to that surface. If a wipe index greater than this maximum is specified, the terminal uses the maximum index as the wipe index.

Border Index. The third parameter specifies the color index in which the viewport border is to be drawn. (The border is drawn only if the BORDER key or SET-BORDER-VISIBILITY command has made the border visible.)

If a border index greater than the maximum index for the surface is specified, the terminal uses the maximum index as the border index.

The border segment also controls:

- segment 0 (the crosshair cursor). In the 4111, the border index doesn't affect segment 0. The 4111 crosshair cursor is always white.
- the Local Viewing function box (for ZOOM and PAN)
- the blinking alpha cursor in the graphics area.

DEFAULTS

surface-number
as shipped — 1
on power-up — 1
if omitted — 0

wipe-index
as shipped — 0
on power-up — 0
if omitted — 0

border-index
as shipped — 1
on power-up — 1
if omitted — 0

ERRORS

RA10	(Level 2):	Surface does not exist (has not been defined with SET-SURFACE-DEFINITIONS command).
RA11	(Level 2):	Invalid *surface-number.*
RA21	(Level 2):	Invalid *wipe-index.*
RA31	(Level 2):	Invalid *border-index.*

REFERENCES

BORDER key
NEXTVIEW key
SELECT-VIEW command
SET-BORDER-VISIBILITY command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-DEFINITIONS command
SET-SURFACE-GRAY-LEVELS command
4110/4120 Series Host Programmers Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-VIEW-DISPLAY-ATTRIBUTES Command

4128, 4129

Host Syntax

$^{E}_{C}$**WA** int-array:*attributes*

Setup Syntax

$^{E}_{C}$**WA** *attributes*

PARAMETERS

attributes (length 0 to 3)
Specifies 3D view display attributes.

first element: *pan-dimensionality* (0, 1, or 2)

0 or not included	no change
1	pan-dimensionality is same as dimensionality of terminal's coordinate mode
2	2D
3	3D

second element: *object-surface-display* (0, 1, 2, or 3)

0 or not included	no change
1	wireframe
2	hidden line
3	shaded surface

third element: *color-smoothing* (0, 1, 2, 3, 4, or 5)

0 or not included	no change
1	none
2	2x2 dithering
3	halftoning with 2x2 dithering
4	4x4 dithering
5	halftoning with 4x4 dithering

DESCRIPTION

This command sets some of the current view's 3D display attributes.

The pan-dimensionality attribute specifies whether Zoom/ Pan actions change the 2D or 3D viewing transformations. (See the ZOOM and PAN keys and the *4128/4129 Operators Manual* for details.) An error is generated if pan-dimensionality is changed while a Local Viewing function is enabled.

The object-surface-display attribute determines how facets in segments are displayed on the 4129 when the view is renewed. The possible values are: (1) wireframe, (2) wireframe with hidden lines removed, and (3) shaded-surface. See the RENEW-VIEW command for details. This attribute has no effect on a 4128 where images are always displayed as wireframe.

The color-smoothing attribute determines whether dithering, halftoning, a combination of the two, or neither are to be applied to the 3D "shaded-surface" or "hidden line" image for the view. (See Appendix G for details on shaded surfaces.) The color-smoothing parameter has no effect on a 4128.

DEFAULTS

attributes
as shipped — < 1,1,1 >
on power-up — < 1,1,1 >
if omitted — < > (no change)

ERRORS

WA00	(Level 0):	Unrecognized command. (The terminal is not a 4128 or 4129.)
WA11	(Level 2):	Invalid *attributes* array.
WA12	(Level 3):	Out of memory while parsing *attributes* array.

REFERENCES

Appendix G, "Shaded Surfaces"
PAN Key
RENEW-VIEW command
ZOOM Key
4128/4129 Operators Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-VIEW-DISPLAY-CLUSTER Command

Host Syntax

$^{E}{}_{C}$**RQ** int-array:*view-numbers*

Setup Syntax

$^{E}{}_{C}$**RQ** $^{S}{}_{P}$ *view-numbers*

PARAMETERS

view-numbers (length 0 to 32768)
Specifies the view-number.

Length 0 deletes all view-display-clusters.
Length 1 to 32768 specifies a view-display-cluster.

Valid element values are in the range from –2 to 64, as follows:

–2	deletes the cluster containing the current view
–1	all views 1 to 64
0	the current view
1 to **64**	a specific view

DESCRIPTION

This command defines a new view-display-cluster. All views in the new view-display-cluster are removed from any other cluster, so that a view can belong to only one cluster at a time. A view does not need to exist to be in a view-display-cluster.

If *view-numbers* has a length of 0, all view-display-clusters are deleted. If *view-numbers* has a length of 1 or more, then element 1 to 64 specifies a particular view. Element –1 specifies all views. Element 0 specifies the currently selected view. Element –2 does not specify a view; it has the special effect of deleting the cluster containing the currently selected view, and of preventing all views in the deleted cluster from being added to the cluster being defined.

A view-display-cluster is a list of views that have certain operations applied to all views when these operations are performed on any one view in the cluster. These operations are: changing the 2D window with the SET-WINDOW command or with the VIEW, CTRL-VIEW, RESTORE, CTRL-RESTORE, OVERVIEW, or CTRL-OVERVIEW keys; renewing the view with the RENEW-VIEW or PAGE commands, or with the PAGE, VIEW, CTRL-VIEW, RESTORE, CTRL-RESTORE, OVERVIEW, or CTRL-OVERVIEW keys. On the 4128 and 4129, these operations also include changing the 3D viewing transform with the SET-VIEWING-TRANSFORM command or with the VIEW, RESTORE, and CTRL-RESTORE keys.

DEFAULTS

view-numbers
as shipped — no clusters
on power-up — no clusters
if omitted — no clusters

ERRORS

RQ11 (Level 2): Invalid *view-numbers* array.

RQ12 (Level 3): Out of memory while parsing parameter.

REFERENCES

OVERVIEW key
PAGE command
PAGE key
RENEW-VIEW command
RESTORE key
SET-VIEWING-TRANSFORM command
SET-WINDOW command
VIEW key

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-VIEWING-TRANSFORM Command

4128, 4129

Host Syntax

$^{E}_{C}$UV xy + -array: *uvw-space*
int-array: *uv-window-size*
xy + : *eye-position*
int: *projection-type*
int-array: *distances*

Setup Syntax

$^{E}_{C}$UV *uvw-space*
uv-window-size
eye-position
projection-type
distances

PARAMETERS

uvw-space (length 0, 1, 2, or 3)
Specifies the location of the UVW origin and the orientation of the UV plane in XYZ terminal space.

first element: *view-reference-point* (xyz = -2^{30} to $2^{30}-1$)

not present no change

second element: *view-plane-normal* (xyz = -2^{30} to $2^{30}-1$, except cannot be <0,0,0>)

not present no change

third element: *view-up* (xyz = -2^{30} to $2^{30}-1$, except cannot be <0,0,0>)

not present no change

uv-window-size (length 0 or 2)
Specifies the width and height of the 3D viewing window in *uvw-space.*

first element: *width* (0 to $2^{30}-2$)

not present no change

second element: *height* (0 to $2^{30}-2$)

not present no change

eye-position (xy = -2^{30} to $2^{30}-1$; z = -2^{30} to 0)
Specifies the eye position.

not present no change

projection-type (0, 1, or 2)
Specifies whether the 3D to 2D projection is parallel or perspective.

0 no change

1 parallel projection

2 perspective projection

distances (length 0 to 3)
Specifies the ROTATE-radius, the front plane distance, and the back plane distance of the viewing volume.

first element: *ROTATE-radius* (-2^{30} to $2^{30}-1$)

not present no change

second element: *front-distance* (-2^{30} to $2^{30}-1$; *front-distance* must be greater than the Z of *eye-position*)

not present no change

third element: *back-distance* (-2^{30} to $2^{30}-1$)

not present no change

DESCRIPTION

This command specifies the viewing volume and ROTATE-radius for the current view and all the views in the view-display cluster containing the current view. The viewing volume depends on the orientation of UVW-space, UV-window, eye-position, front and back planes, and the projection type as described in the following paragraphs.

UVW-space is defined by the *uvw-space* parameter containing *view-reference-point*, *view-plane-normal*, and the *view-up* vector. *View-reference-point* specifies the origin of UVW viewing space in XYZ terminal space coordinates. (The view-reference-point has UVW coordinates of 0,0,0.) The UV-plane (where W = 0 for all points) is perpendicular to the *view-plane-normal* vector. The V axis is the projection of the *view-up* vector onto the UV-plane.

The UV-window is the rectangle on the UV-plane which is centered on the UVW origin (the view-reference-point) and has width and height specified by *UV-window-size.* If the specified width or height is an odd number, it is incremented by one to make it an even number.

The eye-position is specified by *eye-position.* If the Z coordinate of *eye-position* is 0, then none of the coordinates of the eye-position are changed.

The front-plane is parallel to the UV-plane with a W coordinate of *front-distance*. This W coordinate must be greater than the W coordinate of the eye-position. The back-plane is parallel to the UV-plane with a W coordinate of *back-distance*. If *front-distance* is greater than *back-distance*, then the location of the front plane is not changed.

The center of the ROTATE-sphere used in the 3D ROTATE Local Viewing function is on the W axis (U = 0 and V = 0) with the W coordinate equal to *ROTATE-radius*. (See the *4128/4129 Operators Manual* for further details on the ROTATE-sphere.)

The viewing volume consists of the volume between two rectangles defined on the front and back planes. The center of the front and back rectangles are on the line from the eye-position through the UVW origin. The size of the front and back rectangles depend on projection type. See Figure 7-25.

If the projection type is parallel, the front and back rectangles are the same size as the UV-window.

If the projection type is perspective, the corners of the front and back rectangles are on the lines from the eye-position through the corners of the UV-window.

Graphic primitives inside the viewing volume are displayed, while those outside the viewing volume are clipped.

The *terminal-settings-report* for this command uses the *Intc-array-report* message type to report the second and fifth parameters (*UV-window-size* and *distances*) of this command.

DEFAULTS

uvw-space
as shipped — <(0,0,4095),(0,0,-1),(0,1,0)>
on power-up — <(0,0,4095),(0,0,-1),(0,1,0)>
if omitted — no change

uv-window-size
as shipped — <8190,6552>
on power-up — <8190,6552>
if omitted — no change

eye-position
as shipped — (0,0,-4095)
on power-up — (0,0,-4095)
if omitted — no change

projection-type
as shipped — 1
on power-up — 1
if omitted — (no change)

distances
as shipped — <4095,0,8190>
on power-up — <4095,0,8190>
if omitted — (no change)

ERRORS

UV03	(Level 2):	Invalid coordinate mode (must not be 0 or 1).
UV11	(Level 2):	Invalid *uvw-space.*
UV12	(Level 3):	Out of memory parsing parameter.
UV21	(Level 2):	Invalid *UV-window.*
UV22	(Level 3):	Out of memory parsing parameter.
UV31	(Level 2):	Invalid *eye-position.*
UV41	(Level 2):	Invalid *projection-type.*
UV51	(Level 2):	Invalid *distances.*
UV52	(Level 3):	Out of memory parsing parameter.

REFERENCES

Intc-array-report message type
4128/4129 Operators Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

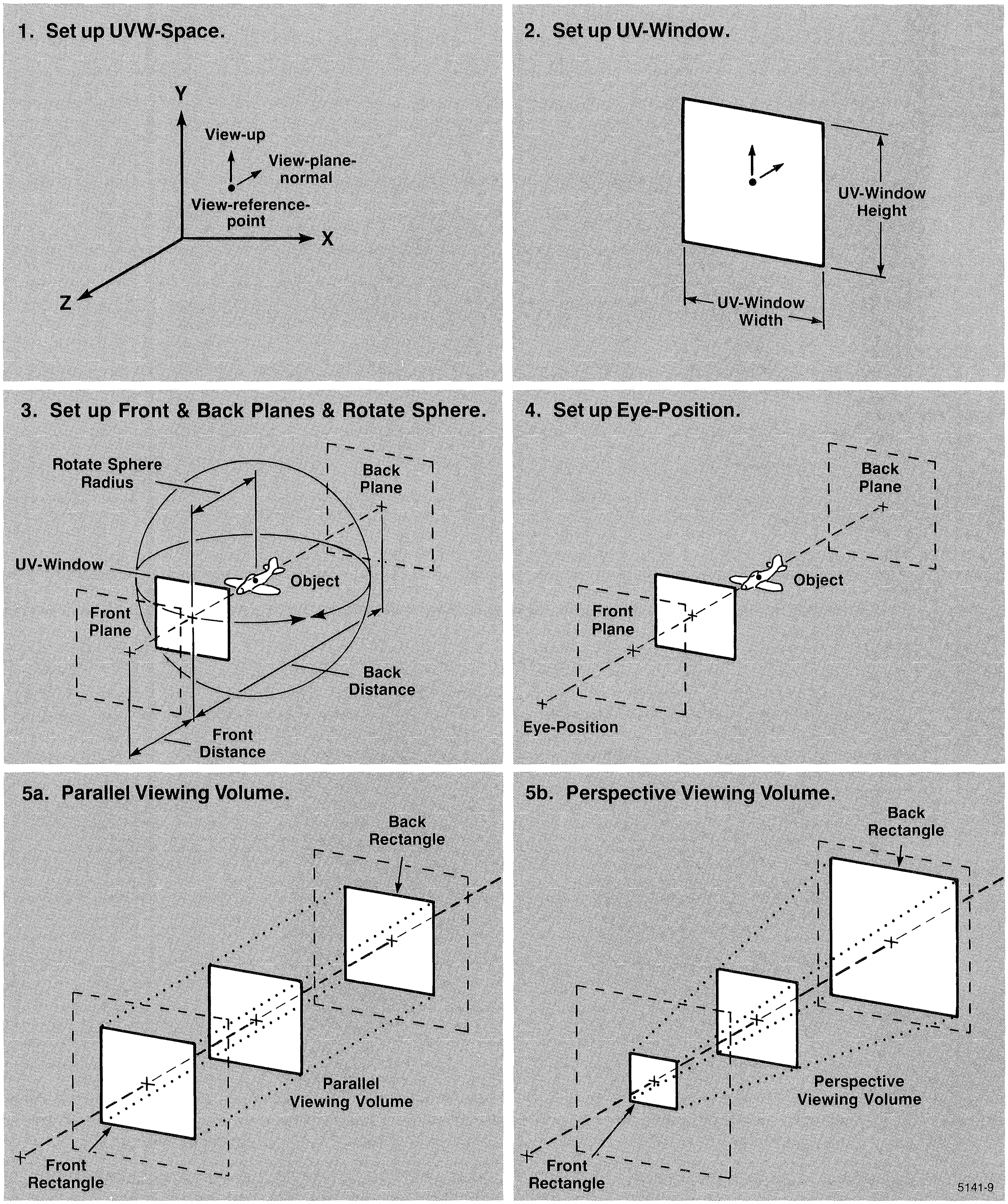


Figure 7-25. How the Viewing Volume is Defined

SET-VIEWPORT Command

Host Syntax

$^{E}_{C}$**RV** xy:*first-corner* xy:*second-corner*

Setup Syntax

$^{E}_{C}$**RV** $^{S}_{P}$ *first-corner* *second-corner*

PARAMETERS

first-corner (X = 0 to 4095, Y = 0 to 3276)
Location of one corner of the current view's viewport, in normalized screen coordinates.

second-corner (X = 0 to 4095, Y = 0 to 3276)
Location of the opposite corner of the viewport, in normalized screen coordinates.

DESCRIPTION

This command sets the position of the current view's viewport in normalized screen coordinate space.

The two xy parameters specify any two opposite corners of the viewport.

Even though the viewport exists in raster memory space, the viewport corners are specified in "normalized screen coordinate space." The normalized coordinate space is 4096-by-3277; the terminal converts normalized coordinates by multiplying by 5 and dividing by 16.

NOTE

Changing the viewport changes the location on the screen where existing segments are displayed. However, segments that are visible when the viewport change occurs do not automatically move to their new screen locations. To make the terminal redraw segments at their new screen locations, you should issue a RENEW-VIEW or PAGE command immediately after changing the viewport.

If you don't do this, and the fixup level (as set by the SET-FIXUP-LEVEL command) is greater than or equal to 4, then moving a segment may not cause that segment's old image to be properly removed from the screen. Multiple images of the segment will appear. The remedy is to issue RENEW-VIEW or PAGE either immediately after the SET-VIEWPORT command or immediately after moving a segment.

DEFAULTS

first-corner
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

second-corner
as shipped — (4095,3276)
on power-up — (4095,3276)
if omitted — (0,0)

ERRORS

RV01	(Level 2):	Invalid viewport size.
RV11	(Level 2):	Invalid *first-corner.*
RV21	(Level 2):	Invalid *second-corner.*

REFERENCES

DELETE-VIEW command
PAGE command
RENEW-VIEW command
SELECT-VIEW command
SET-FIXUP-LEVEL command
SET-VIEW-ATTRIBUTES command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-WINDOW Command

Host Syntax

$^{E}_{C}$**RW** xy:*first-corner* xy:*second-corner*

Setup Syntax

$^{E}_{C}$**RW** $^{S}_{P}$ *first-corner* *second-corner*

PARAMETERS

first-corner
Specifies one corner of the window in 2D terminal space for the current view.

second-corner
Specifies the opposite corner of the window.

The valid ranges for the different coordinate modes are:

coordinate mode 0: X,Y = 0 to 4095
coordinate mode 1: X,Y = -2^{31} to $2^{31}-1$

DESCRIPTION

This command sets the 2D window of the current view and all other views in the same view display cluster. (See the SET-VIEW-DISPLAY-CLUSTER command.) The window is the rectangular region in 2D terminal space whose contents are displayed in a viewport on the screen. If the 4128 or 4129 is in 3D mode (coordinate mode 2), an error is detected.

The parameters specify two opposite corners of the window. These may be any two opposite corners; the terminal sorts the two x-coordinates and the two y-coordinates in the proper order.

If the two x-coordinates are the same, this x-coordinate is used for the lower left corner and the x-coordinate of the upper right corner is calculated by the terminal so that the window has the same aspect ratio as the viewport. This can result in a coordinate greater than $2^{31}-1$. If this occurs, error RW21 is detected and the window is unchanged.

If the two y-coordinates are the same, this y-coordinate is used for the lower left corner and the y-coordinate of the upper right corner is calculated by the terminal so that the window has the same aspect ratio as the viewport. This can result in a coordinate greater than $2^{31}-1$. If this occurs, error RW21 is detected and the window is unchanged.

If *first-corner* is the same as *second-corner*, then the window is set to the partial overview window as defined by the SET-OVERVIEW-WINDOW command.

A view's window determines part of the view's window-viewport transform; the view's viewport determines the other part. This transform maps the lower left corner of the window to the lower left corner of the viewport, and the upper right corner to upper right corner. When any graphic primitive is displayed, its terminal space coordinates are transformed into pixel space coordinates by this transform. See the DRAW, DRAW-MARKER, and DISPLAY-ALPHATEXT commands for details on how primitives are displayed.

NOTE

Changing the window changes the location on the screen where existing segments are displayed. However, segments that are visible when the window change occurs do not automatically move to their new screen locations. To make the terminal redraw segments at their new screen locations, you should issue a RENEW-VIEW or PAGE command immediately after changing the window.

If you don't do this, and the fixup level is greater than or equal to 4, then moving a segment may not cause that segment's old image to be properly removed from the screen. Multiple images of the segment will appear. The remedy is to issue RENEW-VIEW or PAGE either immediately after the SET-WINDOW command or immediately after moving a segment.

DEFAULTS

first-corner
as shipped — (0,0)
on power-up — (0,0)
if omitted — (0,0)

second-corner
as shipped — depends on SET-OVERVIEW-WINDOW
on power-up — depends on SET-OVERVIEW-WINDOW
if omitted — depends on SET-OVERVIEW-WINDOW

ERRORS

RW03 (Level 2): Context error (the 4128 or 4129 must be in coordinate mode 0 or 1).

RW11 (Level 2): *First-corner* out of valid range.

RW21 (Level 2): *Second-corner* out of valid range.

REFERENCES

DISPLAY-ALPHA-TEXT command
DRAW command
DRAW-MARKER command
PAGE command
RENEW-VIEW command
SET-OVERVIEW-WINDOW command
SET-VIEW-DISPLAY-CLUSTER command
SET-VIEWPORT command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SET-4014-ALPHATEXT-SIZE Command

Host Syntax

^{E}c**8**
^{E}c**9**
^{E}c**:**
^{E}c**;**

DESCRIPTION

This command selects between the two character sizes available in graphics and dialog areas:

^{E}c**8** = 16 by 30 pixels, 80 characters per line and 34 lines

^{E}c**9** = same as ^{E}c**8**

^{E}c**:** = 8 by 16 pixels, 160 characters per line and 64 lines

^{E}c**;** = same as ^{E}c**:**

This command is not valid for the 4111.

DEFAULT

as shipped — ^{E}c**8**
on power-up — ^{E}c**8**
if omitted — N/A

INQUIRY COMMAND

none

SET-4014-LINE-STYLE Command

Host Syntax

Ec char:*line-style-code*

PARAMETERS

line-style-code (` to **o**)
An ASCII character with decimal equivalent in the range from 96 to 111.

DESCRIPTION

The SET-4014-LINE-STYLE command is an alternate to the SET-LINE-STYLE command. It is compatible with the command used to set dash patterns on earlier TEKTRONIX terminals. The line-styles that are set by this command are:

SET-4014-LINE-STYLE	**line-style**
Ec `	0
Ec**a**	1
Ec**b**	2
Ec**c**	3
Ec**d**	4
Ec**e**	5
Ec**f**	6
Ec**g**	7
Ec**h**	0
Ec**i**	1
Ec**j**	2
Ec**k**	3
Ec**l**	4
Ec**m**	5
Ec**n**	6
Ec**o**	7

This command sets the dashed-line generator to the beginning of the new line-style.

REFERENCES

SET-LINE-STYLE command

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

SETUP Key

DESCRIPTION

Pressing the SETUP key puts the terminal in Setup mode and turns on the light on the key. Pressing the key again removes the terminal from Setup mode and extinguishes the light.

Pressing SHIFT-SETUP has the same effect as pressing SETUP.

The SETUP key does not auto-repeat.

If you enter Setup mode while GIN is enabled, the graphic cursor disappears until you exit Setup mode. While the terminal is in Setup mode, the enabled GIN device is not used for input to the suspended GIN operation.

When the dialog area is not enabled, alphatext typed in Setup mode is displayed together with graphics, and updates the graphic beam position. When the dialog area is enabled, alphatext typed in Setup mode is displayed in the dialog area, and has no effect on the graphic beam position.

REFERENCES

Description of Setup mode in the Operator's Manual for the particular terminal.

$\mathrm{S_I}$ Character

DESCRIPTION

NOTE

This description is for the 4111 in TEK mode and for the 4115/4120 Series in either TEK or ANSI mode. For 4111 operation in ANSI mode, see ANSI Commands *(Section 8).*

This character selects the JIS Roman font when the terminal has the Katakana keyboard (Option 4K) installed. The JIS Roman font remains in effect until an $\mathrm{S_O}$ character is received.

If the terminal is in eight-bit block mode, or is using DATA parity and it receives alphatext, the font is selected by the eighth bit, and not by the $\mathrm{S_I}$ character. Characters with ASCII decimal equivalents (ADEs) from 0 to 127 are displayed in JIS Roman font; characters with ADEs from 128 to 255 are displayed in JIS Katakana font.

REFERENCES

SET-PARITY command
$\mathrm{S_O}$ character

INQUIRY COMMAND

none

SO Character

DESCRIPTION

NOTE

This description is for the 4111 in TEK mode and for the 4115/4120 Series in either TEK or ANSI mode. For 4111 operation in ANSI mode, see ANSI Commands *(Section 8).*

This character selects the JIS Katakana font when the terminal has the keyboard (Option 4K) installed. The JIS Katakana font remains in effect until an SI character is received.

If the terminal is in eight-bit block mode, or is using DATA parity and it receives alphatext, the font is selected by the eighth bit, and not by the SO character. Characters with ASCII decimal equivalents (ADEs) from 0 to 127 are displayed in JIS Roman font; characters with ADEs from 128 to 255 are displayed in JIS Katakana font.

REFERENCES

SET-PARITY command
SI character

INQUIRY COMMAND

none

SPOOL Command

Host Syntax

```
EcJS   device:source   string:separator
       device:destination
```

Setup Syntax

```
SPOOL SP  source   separator   destination
```

PARAMETERS

source
The source of data for the spool operation. See *Device* parameter type for valid specifiers.

separator
The empty string or the two-character string, **TO**.

destination
The destination for the spool operation. See *Device* parameter type for valid specifiers.

DESCRIPTION

This command spools files from the device specified by the *source* parameter to the device specified by the *destination* parameter. This command is similar to the COPY command, except that once initiated, the operation proceeds as a background task. Although the SPOOL operation may take more time than COPY, the terminal can be used for other tasks during a SPOOL operation. All terminal activity can proceed while a spool operation is currently in progress; although, if HO: is one of the devices, no other communication with the host can be done.

The SPOOL command cannot be used to make a copy of the entire disk. That is, a specific disk file must be named in the input specifier: "F0:FILENAME" is allowed, but "F0:" is not.

If a SPOOL command is given while a spool operation is currently in progress, an error occurs, but the current spooling operation is not disturbed.

A spooling operation cannot be aborted by pressing the CANCEL key on the keyboard. The STOP-SPOOLING command must be used.

DEFAULTS

source
as shipped — none
on power-up — none
if omitted — error JS11

separator
as shipped — none
on power-up — none
if omitted — error JS21

destination
as shipped — none
on power-up — none
if omitted — error JS31

ERRORS

JS02 (Level 3): Out of memory while attempting DMA transfer (4111 or Option 3A only).

JS03 (Level 2): Command context error: a spooling operation is already in progress or the image will not fit (4111 or Option 19, version 5 or later).

JS10 (Level 2): Specified *source* does not exist.

JS11 (Level 2): Invalid *source* specifier.

JS12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JS13 (Level 2): Parameter 1 context error. (Not a valid source device, or device is busy.)

JS19 (Level 2): Device hardware error (disk hardware error, drive not ready, or DMA block transfer error).

JS21 (Level 2): Invalid *separator* (must be empty string or **TO**. Setup mode must be **TO**).

JS22 (Level 3): Out of memory while parsing parameter.

JS30 (Level 2): Specified *destination* does not exist.

JS31 (Level 2): Invalid *destination* device specifier.

JS32 (Level 3): Out of memory while parsing parameter, or while executing the command.

JS33 (Level 2): Parameter 3 context error. (Not a valid destination, device is busy, or existing file is write protected or open.)

JS39 (Level 2): Device hardware error (disk hardware error, drive not ready, or DMA block transfer error).

REFERENCES

COPY command
Device parameter type
STOP-SPOOLING command

INQUIRY COMMAND

none

STATUS Command

There is no escape sequence for the STATUS command. This command can only be typed by the operator while the terminal is in Setup mode.

DESCRIPTION

This command causes the terminal to display the current values of many of its settings. For details, see the Operator's Manual for the particular terminal.

INQUIRY COMMAND

none

STOP-SPOOLING Command

Host Syntax

${}^{E}_{C}$JE

Setup Syntax

STOP

DESCRIPTION

This command aborts the spooling operation currently in progress. The output file is closed.

REFERENCES

SPOOL command

INQUIRY COMMAND

none

String Parameter Type

Syntax

```
string = char-array
```

DESCRIPTION

A string is an array of chars; see the description of the *Char-array* parameter type for details.

REFERENCES

Char-array parameter type

String-Report Message Type

Syntax

```
string-report = char-array-report
```

DESCRIPTION

A *string-report* is a *char-array-report*; see the *Char-array-report* message type description for details.

REFERENCES

Char-array-report message type

Terminal-Settings-Report Message Type

Syntax

```
terminal-settings-report  =  [EOM-indicator]
                             [sig-char]
                             op-code-report
                             [parameter-report . . . EOM-indicator]

op-code-report  =  char-report char-report

                      ( int-report          )
                      ( intc-report         )
                      ( real-report         )
parameter-report  =   { char-report         }
                      { xy-report           }
                      ( xy + -report        )
                      ( int-array-report    )
                      ( xy-array-report     )
                      ( xy + -array-report  )
```

DESCRIPTION

The *terminal-settings-report* message is sent to the host computer in response to a REPORT-TERMINAL-SETTINGS command.

When the terminal sends a report to the host, Bypass mode is entered. (See ENTER-BYPASS-MODE.)

Optional EOM-Indicator. An optional *EOM-indicator* is provided at the start of the terminal-settings-report because of the terminal's "maximum line length" feature. (See the description of the SET-REPORT-MAX-LINE-LENGTH command for details.) This *EOM-indicator* is only sent if not sending it would cause the terminal's maximum line length to be exceeded.

Signature Character. The *sig-char* (signature character) is sent only if it is not NL. This character is the current *sig-char* for non-GIN reports, as set by the most recent SET-REPORT-SIG-CHARS command.

Op-Code-Report. Next comes the *op-code-report*, consisting of two *char-reports*. The two chars being reported are the same two characters which were used in the REPORT-TERMINAL-SETTINGS command; they comprise either an op code for one of the terminal's commands, or else a special inquiry code.

However, if the REPORT-TERMINAL-SETTINGS command specified an op code for a command which does not exist, or which is not installed in the terminal, then the *op-code-report* is SPSP.

The special inquiry codes are listed in Table 7-26. For each inquiry code, the table also shows the *parameter-reports* which are included in that inquiry code's *terminal-settings-report*.

Table 7-26
SPECIAL INQUIRY CODES

Code	Associated Parameter Reports
?M	int-report : available-memory int-report : largest-contiguous-block (The available memory, and the size of the largest contiguous block, are reported as a number of 16-byte units of memory.)
?T	int-report : mode-number-code (For the 4128, the model number code is 4128; for the 4129, the code is 4129, etc.
00	int-report : standard-firmware-version-number
01 to 98	int-report : optional-firmware-version-number (If a firmware option is not installed, the *op-code-report* is SPSP and this int-report is not sent. See Table 7-27 below for Options 22/23 reports.
99	int-report : low-part-number. The low part number on the FE ROM. For internal use by service personnel.

Table 7-27 shows what's reported with Option 22/23.

Table 7-27
OPTION 22/23 REPORTS

Terminal Type	Number of Bit Planes	What's Reported	
		Option 22	Option 23
4111	4	1	0
4125	2	0	0
4125	4	1	0
4125	6	0	1
4125	8	1	1
4128/4129	4	0	0
4128/4129	8	0	1

Parameter-Report. If the command being reported is "inquireable", the *terminal-settings-report* contains a *parameter-report* for each parameter in the command's syntax. A command is inquireable if the INQUIRY COMMAND section of its command description in this manual includes the REPORT-TERMINAL-SETTINGS command. If the command is not "inquireable", no *parameter-reports* are sent.

See Table 7-26 for the type of *parameter-report* for the special inquiry codes.

Most *parameter-reports* are "mirror images" of the parameters of the commands, containing identical values. For a few commands the meanings of the *parameter-reports* differ from the meanings of the parameters in the commands. These commands are:

PROMPT-MODE (op code NM)
LOCK-VIEWING-KeyS (op code RJ)
SET-SURFACE-GRAY-LEVELS (op code RG)
SET-SURFACE-COLOR-MAP (op code TG)
SET-VIEW-DISPLAY-CLUSTER (op code RQ)

PROMPT-MODE. Prompt mode can be turned on with a parameter of 1 or 2. However, the terminal only reports whether the Prompt mode is on (1) or off (0).

LOCK-VIEWING-KEYS. This command takes the form^{E}C**RJ** *int*. Therefore, a Report-TERMINAL-SETTINGS:**RJ** command causes the terminal to send a *terminal-settings-report* which has one *parameter-report*, of the *int-report* type:

terminal-settings-report
= {*EOM-indicator*}
{*sig-char*}
RJ
int-report :*viewing-key-status*
EOM-indicator.

In this report, however, the "viewing key status" integer can assume more values than just zero and one. It can assume values from zero to three. The meanings of these are as follows:

0 The terminal is in not in Local Viewing function. (That is, it is neither in ZOOM function nor in PAN function.) Moreover, the viewing keys are not locked. (This does not preclude the entire keyboard's being locked as a result of a lock-keyboard command.)

1 The terminal is in ZOOM function.

2 The terminal is in PAN function.

3 The viewing keys are locked. (Therefore, the terminal is neither in a ZOOM function nor in PAN function.)

SET-SURFACE-GRAY-LEVELS Command. This command has the syntax $^{E}_{C}$**RG** *int int-array*. Therefore, a REPORT-TERMINAL-SETTINGS: **RG** causes the terminal to send a *terminal-settings-report* message, which has two *parameter-reports*, of the *int-report* and *int-array-report* types. However, the meanings of these *parameter-reports* differ from those of the corresponding parameters in the SET-SURFACE-GRAY-LEVELS command:

terminal-settings-report
= {*EOM-indicator*}
{*sig-char*}
RG
int-report:*number-of-surfaces*
int-array-report:*gray-levels-for-surfaces*
EOM-indicator.

In this *terminal-settings-report*, the *int-report* tells how many surfaces are defined.

The *int-array-report* contains surface numbers (expressed as negative integers), followed by the gray-levels for each surface (expressed as positive integers). Consider, for instance, the following *int-array-report*:

int-array-report: (–1, 90, –2, 30, 60, 90)

This is interpreted as meaning that surface one has color index one set to 90% lightness, while surface two has color index one set to 30%, color index two set to 60%, and color index three set to 90%. (Since color index zero always means "transparent," no gray-level is reported for color index zero.)

The number of gray-levels reported for each surface is one less than two to the power N, where N is the number of bit planes assigned to that surface. Thus, a surface with one bit plane has one gray-level reported, while a surface with two bit planes has three gray-levels reported. If a surface has zero bit planes, then no gray-levels are reported for it. Consider, for instance, the following *int-array-report*:

int-array-report: (–1,–2,10,15,30,45,55,70,90)

This would be interpreted to mean that surface one has no bit planes (and therefore no gray-levels to report), while surface two has three bit planes. The gray-level for the color index one on surface two is 10%, that for color index two on surface two is 15%, and so on.

SET-SURFACE-COLOR-MAP Command. This command has the syntax $^{E}_{C}$**TG** *int int-array*. Therefore, a REPORT-TERMINAL-SETTINGS: **TG** command causes the terminal to send to the host a *terminal-settings-report* with the following syntax:

terminal-settings-report
= {*EOM-indicator*}
{*sig-char*}
TG
int-report:*number-of-surfaces*
int-array-report:*color-info*
EOM-indicator

Here, the *int-report* tells the number of surfaces currently defined, while the *int-array-report* contains information about the background color mixture, and about the color mixtures for each of the color indices on each of the surfaces. (This is different from the meaning these parameters have when they are in SET-SURFACE-COLOR-MAP command sent from the host to the terminal.

int-array-report:*color-info*
= int-report:*number-of-int-reports-to-follow*
triple-report:*background-color*
{*colors-for-one-surface . . .* }

colors-for-one-surface
= int-report:*negative-surface-number*
{triple:*color-coordinates-for-one-color-index . . .* }

triple-report = *RGB-triple-report*
CMY-triple-report
HLS-triple-report

RGB-triple-report = int-report:*red-percentage*
int-report:*green-percentage*
int-report:*blue-percentage*

CMY-triple-report = int-report:*cyan-percentage*
int-report:*magenta-percentage*
int-report:*blue-percentage*

HLS-triple-report = int-report:*hue-angle-in-degrees*
int-report:*lightness-percentage*
int-report:*saturation-percentage*

Machine-RGB-report = int-report:*machine-red-value*
int-report:*machine-green-value*
int-report:*machine-blue-value*

The RGB, CMY, HLS, or Machine RGB color coordinate system is used in the report, depending on which color specifying mode was selected by the most recent SET-COLOR-MODE command.

Suppose, for instance, that HLS color coordinates are being used. (This is the default when the terminal is turned on.) Consider the following *int-array-report*:

int-array-report:*color-info* = int-report: 17
int-report: 0
int-report: 0
int-report: 0
int-report: -1
int-report: 120
int-report: 50
int-report: 100
int-report: -2
int-report: 180
int-report: 50
int-report: 100
int-report: 240
int-report: 50
int-report: 100
int-report: 0
int-report: 100
int-report: 0

Here, the first *int-report* says that there are 17 *int-reports* to follow in the *int-array-report.*

The next three *int-report*s say that the background color is black (hue zero degrees, lightness zero, saturation zero).

The following four *int-report*s carry the numbers –1, 120, 50, 100. Thus, on surface one, color index one is displayed as a red color (hue 120 degrees, lightness 50, saturation 100). Since only one set of three coordinates follows the –1, there must be only one no n-zero color index for that surface. In other words, surface one has only one bit plane assigned to it.

Likewise, the next ten *int-report*s carry the numbers –2, 180, 50, 100, 240, 50, 100, 0, 100, 0. These give three sets of color coordinates for surface number two. On that surface, color index one is displayed as yellow (hue 180, lightness 50, saturation 100), color index two as green (hue 240, lightness 50, saturation 100), and color index three as white (hue 0, lightness 100, saturation 0). Since three sets of color coordinates are reported for surface two, that surface must have two bit planes assigned to it.

SET-VIEW-DISPLAY-CLUSTER Command. This command has the syntax $^{E}_{C}$**RQ** *int-array*. Therefore, the command,

REPORT-TERMINAL-SETTINGS : **RQ**
= $^{E}_{C}$**IQRQ**

causes the terminal to send to the host a *terminal-settings-report* with the following syntax:

terminal-settings-report
= {*EOM-indicator*}
{*sig-char*}
RQ int-array-report
EOM-indicator

Here, the *int-array-report* tells how views are grouped into view display clusters. For the purposes of this report, the clusters are assigned numbers. The first number in the *int-array-report* tells to which cluster, if any, view number one is assigned. Likewise, the second number in the array tells to which cluster view two is assigned, — and so on. If a view has not been assigned to any display cluster, then "cluster number zero" is reported for that view.

Suppose, for instance, that the signature character for non-GIN reports is the tilde (~), and that the host has issued the character sequence, $^{E}_{C}$**IQRQ**. (This is a REPORT-TERMINAL-SETTINGS command which inquires about the view display cluster settings.) One possible response from the terminal would be:

~ **RQ**$^{S}_{P}$$^{S}_{P}$**6**$^{S}_{P}$$^{S}_{P}$**1**$^{S}_{P}$$^{S}_{P}$**1**$^{S}_{P}$$^{S}_{P}$**1**$^{S}_{P}$$^{S}_{P}$**0**$^{S}_{P}$$^{S}_{P}$**2**$^{S}_{P}$$^{S}_{P}$**2**$^{C}_{R}$

Here, the tilde (~) is the *sig-char*, the characters **RQ** signify that the report is for the RQ op code, and the final $^{C}_{R}$ is the *EOM-indicator*. The other characters comprise an *int-array-report*, as follows:

$^{S}_{P}$$^{S}_{P}$6 = int-report: 6 The array has six items.
$^{S}_{P}$$^{S}_{P}$1 = int-report: 1 View 1 is in display cluster 1.
$^{S}_{P}$$^{S}_{P}$1 = int-report: 1 View 2 is in display cluster 1.
$^{S}_{P}$$^{S}_{P}$1 = int-report: 1 View 3 is in display cluster 1.
$^{S}_{P}$$^{S}_{P}$0 = int-report: 0 View 4 is not in any display cluster.
$^{S}_{P}$$^{S}_{P}$2 = int-report: 2 View 5 is in display cluster 2.
$^{S}_{P}$$^{S}_{P}$2 = int-report: 2 View 6 is in display cluster 2.

Thus, the *terminal-settings-report* tells the host these things:

- The highest-numbered view which is in a view display cluster is view number six.
- Views one, two, and three are in the same display cluster.
- Views five and six are in the same display cluster.
- View four is not in any view display cluster.

REFERENCES

Char-report message type
Int-report message type
Int-array-report message type
Intc-array-report message type
Intc-report message type
Real-report message type
XY-report message type
XY-array-report message type
XY+-report message type
XY+-array-report message type
LOCK-VIEWING-KeyS command
REPORT-TERMINAL-SETTINGS command
SET-PROMPT-MODE command
SET-SURFACE-COLOR-MAP command
SET-SURFACE-GRAY-LEVELS command
SET-VIEW-DISPLAY-CLUSTER command

^{U}S Character

See the description in the ENTER-ALPHA-MODE command.

INQUIRY COMMAND

REPORT-4010-STATUS command

VIEW Key

Pressing this key causes the current viewing parameters (in 2D, set by the SET-WINDOW command; in 3D, set by the SET-VIEWING-TRANSFORM command) to be updated according to the previous Local Viewing function actions and renews all views in the current view's cluster (see the SET-VIEW-DISPLAY-CLUSTER command). The updated viewing parameters are pushed into the terminal's circular restore-queue for access by the RESTORE key. If there were four entries in this queue, the "back" one is deleted.

On the 4128 and 4129, 3D segments are drawn only in wireframe when the VIEW key is pressed, even if the view's object-surface-display attribute is "shaded-surface" or "hidden-line".

Pressing CTRL-VIEW (pressing the VIEW key while holding down the CTRL key) causes the following to occur:

1. The next higher-numbered view is selected.
2. The VIEW function is performed on that view.
3. The original view is selected again.

In other words, pressing CTRL-VIEW is equivalent to pressing the following keys: NEXTVIEW, VIEW, CTRL-NEXT-VIEW. If the same segments are visible in both views, the CTRL-VIEW feature makes it easy to use one viewport to show an enlargement of part of the picture shown in the other viewport.

This key does not auto-repeat.

See the *4128/4129 Operators Manual* for further details.

REFERENCES

NEXTVIEW key
PAN key
RESTORE key
SET-VIEW-DISPLAY-CLUSTER command
SET-VIEWING-TRANSFORM command
SET-WINDOW command
ZOOM key
4128/4129 Operators Manual

VT Character

DESCRIPTION

The effect of a VT character depends on whether a dialog area is enabled and on the current coordinate mode of the terminal.

The VT character causes different actions depending on whether the terminal is in TEK mode or ANSI mode (as set by the SELECT-CODE command). See the ANSI Commands section in this manual for details on what happens when the terminal is in ANSI mode.

Alpha Mode; Dialog Area Disabled. When the terminal is in alpha mode with the dialog area disabled, the ASCII VT character ("vertical tab" character) causes the alpha cursor to move up one line. If the terminal is in 2D mode (coordinate mode 0 or 1), the 2D graphic beam position is also moved up one line. If the new 2D position is above the top of the alphatext window (see the SET-OVERVIEW-WINDOW command), no action occurs. If the 4128 or 4129 terminal is in 3D mode (coordinate mode 2), the 3D graphic beam position is not affected and the "carriage return position" is moved up one line in pixel (frame buffer) space.

Alpha Mode; Dialog Area Enabled. If the dialog area is enabled, no action occurs.

Vector and marker modes. When the terminal is in vector mode or marker mode, the VT character has no effect.

REFERENCES

ANSI Commands section
ENTER-ALPHA-MODE command
ENTER-MARKER-MODE command
ENTER-VECTOR-MODE command
SELECT-CODE command
SET-OVERVIEW-WINDOW command
VT Character ANSI command

INQUIRY COMMAND

none

XY-Array Parameter Type

Syntax

```
xy-array    = int[xy . . . ]
```

DESCRIPTION

The *xy-array* parameter type provides the capability to send a variable number of *xy* parameters to various terminal commands. An *xy-array* starts with an *int* that specifies the number of *xys* that follow. If the array is terminated with a command terminator (EC, FS, GS, or US) before the specified number of *xys* have been received by the terminal, the number is reduced to the number actually received. In coordinate modes 1 and 2, the first *xy* in the array is absolute and the rest are relative, specifying the displacement from the position specified by the preceding *xy* in the array.

REFERENCES

Int parameter type
XY parameter type

XY-Array-Report Message Type

Syntax

xy-array-report: *int-report*[*xy-report* . . .]

DESCRIPTION

The *xy-array-report* is sent from the terminal to the host as part of various status reports. All *xy-reports* are absolute, unlike the *xy-array* parameter type, where some *xys* are relative.

REFERENCES

Int-report message type
XY-array parameter type
XY-report message type

XY Parameter Type

Syntax

$$\text{xy} = \left\{\begin{matrix}\textit{12-bit-xy}\\ \textit{32-bit-xy}\end{matrix}\right\}$$

12-bit-xy = {*HiY*} { {*Extra*} *LoY* {*HiX*} } *LoX*
HiY = ASCII character with bits 01yyyyy
Extra = ASCII character with bits 110yyxx or 111yyxx
LoY = ASCII character with bits 11yyyyy
HiX = ASCII character with bits 01xxxxx
LoX = ASCII character with bits 10xxxxx
32-bit-xy = int:*x-coord* int:*y-coord*

DESCRIPTION

There are two syntaxes for *xy* parameters. The one to use depends on the current coordinate mode of the terminal. In coordinate mode 0, use the *12-bit-xy* syntax in Figure 7-26. In coordinate modes 1 and 2, use the *32-bit-xy* syntax, also described below.

Format of 12-Bit-XY Coordinate Bytes

Each *12-bit-xy* parameter consists of from one to five ASCII characters (seven-bit bytes). The bytes are sent in this order: *HiY Extra LoY HiX LoX.* Figure 7-27 shows the formats of the five bytes.

Order and Meaning of the XY Characters

1. The *HiY* (high-order y) character comes first. This character contains the most-significant five bits of the y-coordinate. You can omit the *HiY* byte if the high-order five bits of the y-coordinate have not changed since the last *xy* coordinate sent to the terminal.

2. Next comes the *Extra* character. This character contains the least-significant two bits of the x-coordinate, and the least-significant two bits of the y-coordinate. You can omit the *Extra* character if the least-significant bits of the x- and y-coordinates have not changed since the last *xy* coordinate sent to the terminal. If you do send the *Extra* character, you must follow it with the *LoY* character.

3. Next comes the *LoY* (low-order y) character. Despite its name, this character contains the intermediate five bits of the 12-bit y-coordinate. You can omit the *LoY* character provided: (a) you are sending neither the *Extra* character nor the *HiX* character in this *xy* coordinate, and (b) the intermediate five bits of the y-coordinate have not changed since the last *xy* coordinate sent to the terminal.

4. Next comes the *HiX* (high-order x) character. This character contains the high-order (most-significant) five bits of the x-coordinate. You can omit the *HiX* character if the x-coordinate's most-significant bits have not changed since the last *xy* parameter sent to the terminal. If you do send the *HiX* character, then you must precede it with the *LoY* byte.

5. Finally comes the *LoX* (low-order x) character. Again, despite its name, this character contains the intermediate five bits of the x-coordinate. This character is always required, because it serves to terminate the *xy* parameter sequence.

NOTE

The LoY *and* Extra *bytes each have high-order bits of 11. Thus the* DT *character (binary 1111111) is a possible* LoY *or* Extra *byte. Since some host computers use* DT *as a filler character, this could be a problem in some installations.*

The 4110/4120 terminals include two features which together provide a way to overcome this difficulty. First, the terminal treats **Ec?** *as a synonym for the* DT *character. Secondly, it can be set to ignore* DT *characters. (Use the* IGNORE-DELETES *command.)*

If your host computer uses DT *as a filler character, then you should use these features. Have the host send* **Ec?** *in place of* DT *in* xy *parameters, and send an* IGNORE-DELETES *command to the terminal.*

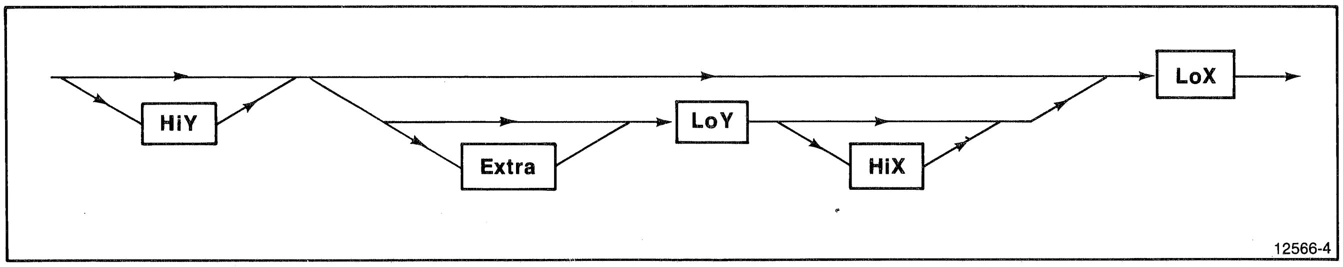


Figure 7-26. *12-Bit-XY* Parameter Syntax.

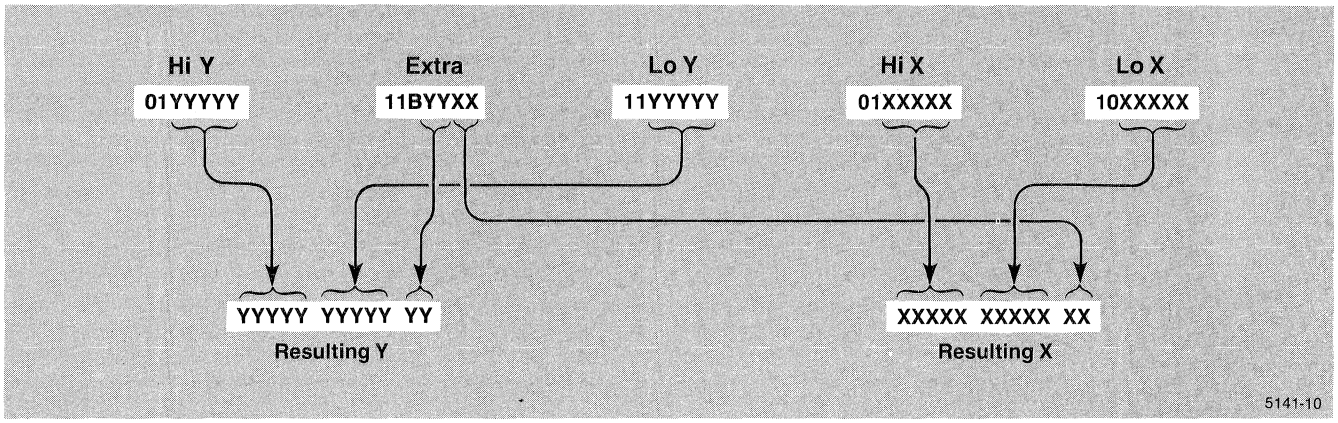


Figure 7-27. Format of *12-Bit-XY* Bytes.

Format of 32-Bit-XY Coordinates

Each *32-bit-xy* parameter consists of two *int* parameters, the first for the x value, the second for the y value. The x and y values either are absolute or are relative. In all explicit commands, the *32-bit-xy* values are always absolute (unless part of an *xy-array*; see below). In the implicit forms of MOVE, DRAW, and DRAW-MARKER, the first *32-bit-xy* after the ^{G}s or ^{F}s is absolute, and all subsequent *32-bit-xys* are relative, specifying the displacement from the position specified by the preceding implicit *32-bit-xy*. In *xy-arrays*, the first *32-bit-xy* is absolute and the rest are relative, specifying the displacement from the position specified by the preceding *32-bit-xy* in the array.

REFERENCES

DRAW command
DRAW-MARKER command
IGNORE-DELETES command
Int parameter type
MOVE command
SET-COORDINATE-MODE command

XY-Report Message Type

Syntax

xy-report = { *12-bit-xy-report*
32-bit-xy-report }

12-bit-xy-report = [*EOM-indicator*]
char:*HiY-report*
char:*Extra-report*
char:*LoY-report*
char:*HiX-report*
char:*LoX-report*

32-bit-xy-report = intc-report:*x-coord*
intc-report:*y-coord*

DESCRIPTION

The terminal sends *xy-report* messages to the host as part of larger report sequences in response to report commands requesting the status of *xy* parameters. In coordinate mode 0, the terminal sends *xy-reports* in the *12-bit-xy-report* format. In coordinate modes 1 and 2, *xy-reports* are sent in *32-bit-xy-report* format.

12-Bit-XY-Reports

A *12-bit-xy-report* includes five characters and may include an *EOM-indicator*. The five report characters are all in the ADE (ASCII Decimal Equivalent) range 32 to 63.

EOM-indicator. The *12-bit-xy-report* includes an*EOM-indicator* only when there are less than five characters left before the report-max-line-length is reached.

Parsing a 12-Bit-XY-Report. To parse the five data bytes for the *12-bit-xy-reports*, use this algorithm:

x = ((*HiX-report*))–32 ∗ 128 + ((*LoX-report*)–32) ∗ 4
+ (*Extra-report*) MOD 4

y = ((*HiY-report*)–32) ∗ 128 + ((*LoY-report*)–32) ∗ 4 +
((*Extra-report*)/4) MOD 4

32-Bit-XY-Reports

A *32-bit-xy-report* consists of two *intc-reports*, the first for the x and the second for the y.

See the description of the *Intc-report* message type for details.

REFERENCES

EOM-indicator syntactic construct
Intc-report message type
SET-REPORT-MAX-LINE-LENGTH command

XY + -Array Parameter Type

4128, 4129

Syntax

```
xy + -array    = int[xy + . . . ]
```

DESCRIPTION

The *xy + -array* parameter type provides the capability to send a variable number of *xy +* parameters to various terminal commands. An *xy + -array* starts with an *int* that specifies the number of *xy + s* that follow. If the array is terminated with a command terminator (^{E}C, ^{F}S, ^{G}S, or ^{U}S) before the specified number of *xy + s* have been received by the terminal, the number is reduced to the number actually received. In coordinate modes 1 and 2, the first *xy +* in the array is absolute and the rest are relative, specifying the displacement from the position specified by the preceding *xy +* in the array.

REFERENCES

Int parameter type
XY + parameter type

XY + -Array-Report Message Type

4128, 4129

Syntax

xy + -array-report: *int-report*[*xy + -report* . . .]

DESCRIPTION

The *xy + -array-report* is sent from the terminal to the host as part of various status reports.

REFERENCES

Int-report message type
XY + -report message type

XY + Parameter Type

4128, 4129

Syntax

$$xy+ \;=\; \left\{ \begin{matrix} xy \\ xyz \end{matrix} \right\}$$

xyz = int:*x-coord* int:*y-coord* int:*z-coord*

DESCRIPTION

XY+ parameters represent points in 2D coordinate space when the terminal is in coordinate modes 0 and 1 and in 3D coordinate space when the terminal is in coordinate mode 2.

In 2D mode (coordinate modes 0 and 1), *xy+* parameters have the same syntax as *xy* parameters.

In 3D mode (coordinate mode 2), *xy+* parameters consist of three *int* parameters:

xy+ = int:*x-coord* int:*y-coord* int:*z-coord*

The values of the x,y, and z values are either absolute or relative, following the same rules as *32-bit-xy* parameters.

See the description of the *XY* and *Int* parameter types for details.

REFERENCES

Int parameter type
SET-COORDINATE-MODE command
XY parameter type

XY + -Report Message Type

4128, 4129

Syntax

$$\textit{xy + -report} = \left\{ \begin{matrix} \textit{xy-report} \\ \textit{xyz-report} \end{matrix} \right\}$$

$$\textit{xyz-report} = \left\{ \begin{matrix} \textit{intc-report}\text{:x-coord} \\ \textit{intc-report}\text{:y-coord} \\ \textit{intc-report}\text{:z-coord} \end{matrix} \right\}$$

DESCRIPTION

The terminals send *xy + -report* messages as part of larger report sequences in response to report commands requesting the status of *xy* + - parameters. In coordinate modes 0 and 1, the terminal sends *xy + -reports* in the *xy-report* format. In coordinate mode 2, *xy + -reports* consist of three *intc-reports*, the first for the x, the second for the y, and the third for the z.

See the descriptions of the *Intc-report* and *XY-report* message types for details.

REFERENCES

Intc-report message type
XY-report message type

ZOOM Key

DESCRIPTION

Pressing this key when its light is off enables the 2D or 3D (4128 and 4129 only) ZOOM Local Viewing function and turns the key's light on. Pressing the key when the terminal is in the ZOOM Local Viewing function disables the function and turns the key's light off. Pressing this key when its light is on, but the terminal is in a different 3D Local Viewing function, enables the 3D ZOOM Local Viewing function and leaves the key's light on. The key does not auto-repeat.

When the ZOOM Local Viewing function is enabled, the ZOOM symbol is displayed on the screen. The 2D ZOOM symbol includes a rectangle with two corners of a smaller rectangle inside it. This smaller rectangle is one half as large as the symbol, and is centered on the symbol. On the 4128 and 4129, the 3D ZOOM symbol is similar to the 2D ZOOM symbol, but the vertical sides of the corners of the smaller rectangle are longer. If the PAN or DEPTH Local Viewing function was enabled when the ZOOM key was pressed, then the ZOOM symbol is the same size as the previous symbol.

On the 4128 and 4129, if the view's pan-dimensionality (as set by the SET-VIEW-DISPLAY-ATTRIBUTES command) is 2D, the 2D ZOOM Local Viewing function is enabled. If it is 3D, then the 3D ZOOM Local Viewing function is enabled.

The size of the ZOOM symbol is changed without changing its aspect-ratio when the thumbwheels are moved. Pressing the SHIFT key while moving either thumbwheel causes the size to change more slowly. This allows for finer adjustments in the size of the symbol. In 3D, the limits on the size of the ZOOM symbol are the same as in 2D.

With a pan-dimensionality of 2D, pressing the CTRL key while moving the horizontal thumbwheel changes only the horizontal size; pressing the CTRL key while moving the vertical thumbwheel changes only the vertical size. This will change the aspect-ratio of the symbol. The "normal shape" of the symbol (where the potential 2D window has the same aspect-ratio as the viewport) can be restored by pressing the NORMAL key.

The boundary of the 2D ZOOM symbol indicates the 2D window which will take effect if the VIEW key is pressed. The size and center of the 3D ZOOM symbol indicates the size of the 3D UV-window and the location of the 3D view-reference-point which will take effect if the VIEW key is pressed.

The operator cannot, by moving the thumbwheels while in the ZOOM function, make the symbol less than one sixteenth the size of the current 2D window or 3D UV-window, or less than 8 terminal space units wide or high.

If the terminal is in either a 2D ZOOM or PAN function, then pressing CTRL-ZOOM shrinks the symbol about its center by approximately 5/8 and performs a view function as if the VIEW key was pressed.

REFERENCES

NORMAL key
PAN key
SET-VIEW-DISPLAY-ATTRIBUTES command
SET-VIEWING-TRANSFORM command
SET-VIEWPORT command
SET-WINDOW command
VIEW key
4128/4129 Operators Manual

INQUIRY COMMAND

REPORT-TERMINAL-SETTINGS command

4010-GIN-Report Message Type

Syntax

```
4010-GIN-report  =  char:key-pressed
                         point-report
                         EOM-indicator

point-report  =  { xy + -report    }
                 { 4010-xy-report  }

4010-xy-report  =  HiX-report
                   LoX-report
                   HiY-report
                   LoY-report
```

PARTS OF THE Report

key-pressed
A single ASCII character, corresponding to the key which the operator pressed to initiate the 4010-GIN-report.

point-report
Tells the location of the GIN cursor when the operator pressed a key to initiate a GIN report. The location is reported in either a *xy + -report* or *4010-xý-report* format.

HiX-report, LoX-report, HiY-report, LoY-report
These are all seven-bit ASCII characters with high-order bits "01" characters in the "figures" column of the ASCII chart.

DESCRIPTION

When the terminal responds to an ENABLE-4010-GIN command — the sequence of characters EcSB — it is emulating a TEKTRONIX 4010 Series computer display terminal. Therefore, it sends coordinate information to the host computer using the *4010-GIN-report* syntax rather than the *GIN-report-sequence* syntax that it uses when responding to the ENABLE-GIN command.

When the terminal sends a report to the host, Bypass mode is entered. (See ENTER-BYPASS-MODE.)

Char Parameter. The first part of the *4010-GIN-report* is a single ASCII character. This is the character for the key which the operator pressed to initiate the GIN report.

Point-Report Parameter. The second part of the *4010-GIN-report* is the *point-report.*

If the terminal's coordinate mode is 1 or 2, then the *point-report* is formatted as an *xy+-report*. See the *XY+-report* message type for details.

If the terminal's coordinate mode is 0, then the *point-report* is in the *4010-xy-report* format. This is the format used by Tektronix 4010 Series terminals. Only the ten most significant bits are sent to the host; the least significant two bits are omitted. Consequently, the position is reported only to a precision of ten (rather than 12) binary bits.

A *4010-xy-report* consists of four ASCII characters, corresponding to the *HiX*, *LoX*, *HiY*, and *LoY* characters in the *12-bit-xy* parameters that the host may send when displaying graphics on the screen. However, these characters differ from those in *xy* syntax in that their high-order bits ("tag bits") are always "01". To parse the 10-bit *4010-xy-report*, use this algorithm:

$$x = ((\mathit{HiX\text{-}report}) - 32) * 128 + ((\mathit{LoX\text{-}report}) - 32) * 4$$
$$y = ((\mathit{HiY\text{-}report}) - 32) * 128 + ((\mathit{LoY\text{-}report}) - 32) * 4$$

Also, the characters in the 4010-xy-report are sent in the order *HiX LoX HiY LoY*. This is different from the order used in *xy* parameters. It is also different from the order used in *xy-report*s sent as part of *GIN-report-sequences* in response to the ENABLE-GIN command.

EOM-indicator. The *EOM-indicator* ("end-of-message" indicator) marks the end of the report. See the description of the *EOM-indicator* syntactic construct.

To most closely emulate the 4010 Series terminals, the terminal should not be in block mode. The *EOL-string* should be set (by a SET-EOL-STRING command) as in Table 7-28.

Table 7-28

EOL-STRING SETTINGS TO EMULATE 4010 SERIES TERMINALS

GIN Strap (in 4010 Terminal)	EOL-string (in 4110/4120 Series Terminal)
CR and EOT	CRET
CR only	CR
no CR, no EOT	empty string

REFERENCES

EOM-indicator syntactic construct
SET-COORDINATE-MODE command
SET-EOL-STRING command
XY parameter type
XY+-report message type

4010-HARDCOPY Command

Host Syntax

$^{E}{}_{C}{}^{E}{}_{B}$

Setup Syntax

$^{E}{}_{C}{}^{E}{}_{B}$

DESCRIPTION

The 4010-HARDCOPY command is provided for compatibility with host software written for use with earlier TEKTRONIX terminals. This command has the same effect as pressing the HARDCOPY key. A hard copy of the display is made on the hard copy unit (provided one is attached).

Issuing the 4010-HARDCOPY command is equivalent to issuing the HARDCOPY command with a parameter of zero:

HARDCOPY: 0 = $^{E}{}_{C}$**KH0**

If Option 19 is installed, this command causes a hard copy of the display to be sent to the interface selected with the SELECT-HARDCOPY-INTERFACE command.

REFERENCES

HARDCOPY command
HARD COPY key
SELECT-HARDCOPY-INTERFACE command

INQUIRY COMMAND

none

4010-Status-Report Message Type

Syntax

```
4010-status-report  =  { 4010-GIN-status-report     }
                       { 4010-non-GIN-status-report }

4010-GIN-status-report  =  point-report
                           EOM-indicator

4010-non-GIN-status-report  =  4010-status-byte
                               point-report
                               EOM-indicator

4010-status-byte  =  a seven-bit ASCII character whose high-order two bits are "01" and whose least-significant
                     five bits hold status information, described below.

point-report  =  { xy+-report      }
                 { 4010-xy-report  }

4010-xy-report  =  HiX-report
                   LoX-report
                   HiY-report
                   LoY-report
```

PARTS OF THE REPORT

4010-GIN-status-report
If the terminal is enabled for graphic input, the report message does not include a *4010-status-byte.*

4010-non-GIN-status-report
If the terminal is not enabled for graphic input, the report message begins with a *4010-status-byte.*

4010-status-byte
A seven-bit ASCII character, in the range from SP to **?** — a character whose two high-order bits are "01". The least-significant five bits hold status information.

point-report
The location of the graphic cursor (if GIN is enabled) or the current beam position (if GIN is not enabled). The point is reported in either *xy+-report* or *4010-xy-report* format.

DESCRIPTION

The *4010-status-report* is sent by the terminal in response to a REPORT-4010-STATUS command (the EcEQ sequence). It may also be sent if the terminal has received an ENABLE-4953-TABLET-GIN command and the operator moves the stylus (or four-button puck) away from the tablet surface. When the terminal sends a report to the host, it enters Bypass mode. (See the ENTER-BYPASS-MODE command.)

Responding to a Report-4010-STATUS Command. If the terminal is responding to an EcEQ sequence, it sends to the host either a *4010-GIN-status-report* or a *4010-non-GIN-status-report*, depending on whether or not it has been enabled for graphic input.

If enabled for 4010-GIN (with an ENABLE-4010-GIN command), the terminal responds to the EcEQ sequence by sending a *4010-GIN-status-report.* This consists of a *point-report* for the current location of the GIN cursor, followed by an *EOM-indicator.*

If the terminal is not enabled for 4010-GIN, then it responds to the EcEQ sequence with a *4010-non-GIN-status-report.* This consists of a *4010-status-byte* that holds status information about the terminal, a *point-report* for the current graphic beam position, and an *EOM-indicator.*

4010-Status-Byte. The *4010-status-byte* is a seven-bit ASCII character, as follows:

b7	b6	b5	b4	b3	b2	b1
0	1	HCU	NOLI	GRAPH	0	1

HCU — The HCU (Hard Copy Unit) bit, bit b5, is zero if a hard copy unit is attached to the terminal and is ready to accept a copy request. Otherwise, this bit is one. This bit reflects the status of the standard hard copy interface only, even if Option 19 (the color hard copy board and interface) is installed.

NOLI,GRAPH — In the 4110/4120, the NOLI (No Linear Interpolation) and GRAPH (Graph Mode) bits have the following meanings:

NOLI	GRAPH 0	
0	0	The terminal is in marker mode.
0	1	The terminal is in alpha mode.
1	0	The terminal is in vector mode.
1	1	(This combination does not occur.)

Point-Report. If the terminal's coordinate mode is 1 or 2, then the *point-report* is formatted as an *xy + -report*. See the *XY + -report* message type for details.

If the terminal's coordinate mode is 0, then the *point-report* is in the *4010-xy-report* format. This is the format used by Tektronix 4010 Series terminals. Only the ten most significant bits are sent to the host; the least significant two bits are omitted. Consequently, the position of the graphic cursor or the current beam is reported only to a precision of ten (rather than 12) binary bits.

To parse a 10-bit *4010-xy-report*, use this algorithm:

$$x = ((HiX\text{-}report) - 32) * 128 + ((LoX\text{-}report) - 32) * 4$$
$$y = ((HiY\text{-}report) - 32) * 128 + ((LoY\text{-}report) - 32) * 4$$

If full twelve-bit precision is required for the *4010-xy-report*, use the ENABLE-GIN or Report-GIN-POINT command rather than the ENABLE-4010-GIN or Report-4010-STATUS command.

EOM-indicator. The *EOM-indicator* marks the end of the report.

REFERENCES

EOM-indicator syntactic construct
REPORT-4010-STATUS command
SET-COORDINATE-MODE command
XY + -report message type

4953-Tablet-GIN-Report Message Type

Syntax

```
4953-tablet-GIN-report  =  [EOM-indicator]
                           header-char
                           point-report
                           [EOM-indicator]

header-char  =  ⎧ SB ⎫
                ⎪ GS ⎪
                ⎨ US ⎬
                ⎪ M  ⎪
                ⎪ J  ⎪
                ⎩ O  ⎭

point-report  =  ⎰ xy + -report           ⎱
                 ⎱ 4953-tablet-xy-report  ⎰

4953-tablet-xy-report  =  ⎰ xy-report         ⎱
                          ⎱ 10-bit-xy-report  ⎰

10-bit-xy-report  =  HiY-report
                     LoY-report
                     HiX-report
                     LoX-report
```

PARTS OF THE REPORT

EOM-indicator

An *EOM-indicator* is only sent at the start of the report message if both the following conditions are met: (a) At least one character has already been sent on this line (that is, since the last *EOM-indicator*). (b) If the current line were not terminated (by sending this *EOM-indicator*), then the rest of this *4953-tablet-GIN-report* would cause the current maximum line length to be exceeded.

An *EOM-indicator* is only sent at the end of the report message if the SET-REPORT-EOM-FREQUENCY command specified "more frequent" rather than "less frequent."

header-char

A single ASCII character. The possible header characters are determined by the most recent SET-TABLET-HEADER-CHARACTERS command.

point-report

The current cursor position. The position is reported as either an *xy + -report* or a *4953-tablet-xy-report*, depending on the type of terminal and the terminal's coordinate mode. If the *point-report* is formatted as a *4953-tablet-xy-report*, the cursor position may be reported to 12 bits of precision (in the standard *xy-report* format) or to ten bits of precision (in the *10-bit-xy-report* format), depending on the ENABLE-4953-TABLET-GIN command.

10-bit-xy-report

Gives the most-significant 10 bits of the cursor's x- and y-coordinates. The format is similar to that of the standard *xy-report*, except that the *Extra-Report* byte is omitted.

DESCRIPTION

By issuing the ENABLE-4953-TABLET-GIN command, you can cause the terminal to emulate a TEKTRONIX 4010 Series terminal with accessory 4953 or 4954 graphics tablet. (See the description of the ENABLE-4953-TABLET-GIN command for details.)

In response to an ENABLE-4953-TABLET-GIN command, the terminal sends graphic input information to the host computer in a format compatible with the format used by TEKTRONIX 4953 or 4954 graphic tablets; this format comprises the *4953-tablet-GIN-report.*

When the terminal sends a report to the host, Bypass mode is entered. (See ENTER-BYPASS-MODE.)

First EOM-Indicator. The optional *EOM-indicator* at the start of the report occurs in the syntax because of the terminal's "maximum report line length" feature. (See the description of the SET-REPORT-MAX-LINE-LENGTH command for details.) This *EOM-indicator* is prefixed to the report if it is needed to prevent the terminal's maximum line length from being exceeded.

Header-Char. Which characters are used for the *header-char* depends on the most recent SET-TABLET-HEADER-CHARACTERS command. If the *int* parameter in that command was 0, then the *header-char* is one of the ASCII control characters GS, SB, and US. If the *int* parameter in that command was 1 (the default), then the *header-char* is one of the uppercase letters **M**, **J**, and **O**.

For the first point in a stroke, the *header-char* is GS or **M**. For subsequent points in a stroke, the header-char is SB or **J**. For the last point in a stroke (when the operator ceases to press the stylus against the tablet), the *header-char* is US or **O**.

Point-Report. The *point-report* reports the cursor position in either the *xy + -report* or the*4953-tablet-xy-report* format, depending on the type of terminal and the terminal's coordinate mode.

If the terminal's coordinate mode is 1 or 2, then the *point-report* is formatted as an *xy + -report*. See the *XY + -report* message type for details. If the terminal's coordinate mode is 0, then the *point-report* is formatted as a*4953-tablet-xy-report*.

The *4953-tablet-xy-report* is either an *xy-report* or a *10-bit-xy-report*, depending on bit 5 of the ENABLE-4953-TABLET-GIN command. To parse a *10-bit-xy-report*, use this algorithm (note that the *HiY-report* and *LoY-report* precedes the *HiX-report* and *LoX-report*):

$$y = ((\mathit{HiY\text{-}report}) - 32) * 128 + ((\mathit{LoY\text{-}report}) - 32) * 4$$
$$x = ((\mathit{HiX\text{-}report}) - 32) * 128 + ((\mathit{LoX\text{-}report}) - 32) * 4$$

If the ENABLE-4953-TABLET-GIN command specified 12-bit data, then the *point-report* is formatted as an *xy-report*. (See the*XY-Report* message type for details.)

Final EOM-Indicator. The *EOM-indicator* at the end of the report is sent if the most recent SET-REPORT-EOM-FREQUENCY command specified that reports are to be sent "more frequently" rather than "less frequently."

REFERENCES

ENABLE-4953-TABLET-GIN command
ENTER-BYPASS-MODE command
EOM-indicator syntactic construct
SET-COORDINATE-MODE command
SET-REPORT-EOM-FREQUENCY command
SET-REPORT-MAX-LINE-LENGTH command
SET-REPORT-SIG-CHARS command
SET-TABLET-HEADERS command
XY-report message type
XY + -report message type

Section 8

ANSI X3.64 COMMANDS

INTRODUCTION

The terminal is equipped with two command sets. This section discusses the ANSI X3.64 commands (ANSI mode and other command modes are selected by the SELECT-CODE command).

TERMINAL MODES

Terminal modes (e.g., Overstrike/Replace, Echo, etc) remain set in ANSI modes even though they were set while the terminal was in TEK mode. Terminal modes set while the terminal is in ANSI mode remain set after the terminal returns to TEK mode.

Before changes can be made to the mode (TEK or ANSI) environment, the appropriate mode must be selected.

CONTROL CHARACTERS

The terminal reacts to a different set of control characters while processing a command in ANSI mode. All control characters that do not prompt a terminal or cursor action (including the TEK mode command terminators) are ignored, and the control characters that do prompt an action cause that action without terminating the command. (These control characters are ^{E}Q, ^{B}L, ^{B}S, ^{H}T, ^{L}F, ^{V}T, ^{F}F, and ^{C}R.) For example, the ^{B}L character causes the terminal bell to ring, the ^{C}R character causes the action as set by CRLF mode, etc.

Some control characters affect the dialog area the same, whether the terminal is in TEK, ANSI, or VT52 mode.

ANSI Syntax

The ANSI X3.64 commands that are in Section 8 use a different syntax than the TEK commands in Section 7. ANSI commands (including alphatext) affect only the dialog area, whether the dialog area is enabled or not, and are not valid in the graphics area.

There are two syntax forms for the ANSI commands. Commands of one of these syntax forms consist of an ^{E}C character and a unique final character. The final character is what identifies each command. The commands with this syntax form do not have parameters.

Commands of the other syntax form consist of a *control-sequence-introducer* (*CSI*), zero or more parameters of either *Pn* or *Ps* types, and a unique final character. The *CSI* and the terminator together identify individual commands.

The *CSI* syntax is:

^{E}C[

Substitute these two characters whenever the *CSI* is called for.

The symbols that are used to represent syntax elements and conventions for ANSI commands are the same as those used for TEK commands (see Table 2-1).

ANSI PARAMETER TYPES

There are two parameter types for ANSI X3.64 commands, *Pn* and *Ps*.

Pn

Pn is a numeric parameter ranging from 0 to 32767. Send *Pn*'s as a sequence of digits. For example, send the number 75 as the two characters **7** and **5**. There is no special packing scheme for this parameter type. If the *Pn* is 0 or missing it is interpreted as a default value (usually 1).

Ps

Ps is a parameter selected from a given list. When the parameter type is *Ps*, the command description gives you a choice of parameters. There is no special packing scheme for this parameter type. They all have the format of a *Pn* or (char) *Pn* where (char) is **<**, **:**, **>**, or **?**.

The semicolon (**;**) separates parameters in a command string. Enter a semicolon between parameters when you are entering more than one parameter for a command. You can enter up to 46 characters of *Ps* parameters for a command that expects *Ps* parameters.

Command TERMINATORS

ANSI mode uses the $^{\mathrm{C}}\mathrm{N}$ and $^{\mathrm{S}}\mathrm{B}$ characters as command terminators. When these characters are received in the middle of a command, the command is terminated and any characters belonging to that command that have already been received are discarded. When this occurs, a snoopy character $^{\mathrm{C}}\mathrm{N}$ or $^{\mathrm{S}}\mathrm{B}$ will appear on the terminal screen.

ANSI/VT52-MODE Command

Syntax

$^{E}{}_{C}$<

DESCRIPTION

This command is treated as a no-op by the terminal, but is included for compatibility with certain editors.

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

The $^{B}{}_{L}$ character sounds the terminal's bell.

$^{\mathrm{B}}\mathrm{S}$ Character (Backspace)

The $^{\mathrm{B}}\mathrm{S}$ character moves the cursor one character position to the left. If the cursor is at column 1, $^{\mathrm{B}}\mathrm{S}$ has no effect.

$^{\mathrm{C}}\mathrm{N}$ Character (Cancel)

DESCRIPTION

This character is a command terminator. When it is received in the middle of a command, the command is terminated and any characters belonging to that command that have already been received are discarded. When this occurs, a snoopy character $^{\mathrm{C}}\mathrm{N}$ will appear on the terminal screen.

The $^{\mathrm{S}}\mathrm{B}$ character has a similar effect.

REFERENCES

$^{\mathrm{S}}\mathrm{B}$ character

CR Character (Carriage Return)

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

REFERENCES

CRLF Command

CONTROL-SEQUENCE-INTRODUCER

Acronym: CSI

Syntax

```
EC[
```

DESCRIPTION

The *control-sequence-introducer*, *CSI*, serves to introduce control sequence type commands. (Many ANSI X3.64 commands are of the control sequence type.)

CURSOR-BACKWARD Command

Acronym: CUB

Syntax

```
CSI  [Pn:number-of-columns]  D
```

PARAMETERS

number-of-columns (0 to 32767)
The number of columns to move the cursor to the left.

DESCRIPTION

This command causes the cursor to move to the left the specified number of columns in the active line.

If the *number-of-columns* parameter is greater than the number of columns to the left of the cursor position, the cursor moves to column 1 (the left-most column) in the current line.

DEFAULT

number-of-columns
if 0 or omitted — 1

ERRORS

[D11 (Level 2): Invalid *number-of-columns* parameter (range is 0 to 32767).

CURSOR-BACKWARD-TAB Command

Acronym: CBT

Syntax

```
CSI  [Pn:number-of-tabs]  Z
```

PARAMETERS

number-of-tabs (0 to 32767)
The number of tab stops to move the cursor backward.

DESCRIPTION

This command causes the cursor to move backward (to the left) the specified *number-of-tabs* tab stops on the current line.

If a parameter value of n is received, the cursor moves back to the nth tab stop preceding or column 1 of the current line, whichever comes first. No wrap-around occurs.

DEFAULTS

number-of-tabs
if 0 or omitted — 1

ERRORS

[Z11 (Level 2): Invalid *number-of-tabs* parameter (range is 0 to 32767).

REFERENCES

CURSOR-HORIZONTAL-TAB command
$^{H}{}_{T}$ character

CURSOR-DOWN Command

Acronym: CUD

Syntax

```
CSI  [Pn:number-of-lines]  B
```

PARAMETERS

number-of-lines (0 to 32767)
The number of lines to move the cursor down.

DESCRIPTION

This command causes the cursor to move down the number of lines (rows) specified in the *number-of-lines* parameter.

If the CUD command moves the cursor from inside the viewport (the visible area of the dialog area) to below the viewport, or from above the viewport to below the viewport, then the text scrolls so as to keep the cursor in view.

If the cursor was not inside the viewport when the command is received it becomes visible only if it is moved through the viewport.

If the number of lines specified is greater than the number of lines of text remaining in the scroll buffer (as set by the SET-DIALOG-AREA-BUFFER-SIZE command), the cursor stops at the bottom of the scroll buffer.

In the 4111, if Edit margins have been set, the CUD command will not move the cursor from the scrolling region down to the lower fixed region.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[B11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

REFERENCES

SET-DIALOG-AREA-BUFFER-SIZE command (TEK command set)
SET-TOP-AND-BOTTOM-MARGINS command
SET-MODE command
RESET-MODE command

CURSOR-FORWARD Command

Acronym: CUF

Syntax

```
CSI  [Pn:number-of-columns]  C
```

PARAMETERS

number-of-columns (0 to 32767)
The number of columns to move the cursor forward.

DESCRIPTION

This command causes the cursor to move to the right the number of columns specified by the *number-of-columns* parameter.

If the number received is greater than the number of columns to the right of the cursor position in the current line, the cursor moves to the right-most column.

DEFAULT

number-of-columns
if 0 or omitted — 1

ERRORS

[C11 (Level 2): Invalid *number-of-columns* parameter (range is 0 to 32767).

CURSOR-HORIZONTAL-TAB Command

Acronym: CHT

Syntax

```
CSI  [Pn:number-of-tabs]  I
```

PARAMETERS

number-of-tabs (0 to 32767)
The number of tab stops to move the cursor forward.

DESCRIPTION

This command moves the cursor to the right *number-of-tabs* tab stops on the current line.

If the parameter value is 1 (or 0 or omitted), the cursor moves forward to the tab stop immediately following the current cursor position. If a parameter value of n is received, the cursor moves forward to the nth tab stop following, or to the right-hand margin of the current line, whichever comes first. No wrap-around occurs.

DEFAULTS

number-of-tabs
if 0 or omitted — 1

ERRORS

[I11 (Level 2): Invalid *number-of-tabs* (range is 0 to 32767).

REFERENCES

CURSOR-BACKWARD-TAB command
$^{H}{}_{T}$ character

CURSOR-POSITION Command

Acronym: CUP

Syntax

CSI [Pn:*line-number*]
[; [Pn:*column-number*]] **H**

PARAMETERS

line-number (0 to 32767)
The number of the line to which to move the cursor.

column-number (0 to 32767)
The number of the column to which to move the cursor.

DESCRIPTION

This command moves the cursor to the line and column within the scroll buffer specified in the *line-number* and *column-number* parameters.

If the specified coordinates move the cursor from inside the dialog area viewport to coordinates that are currently outside the viewport, the text scrolls so that the cursor remains in view. This scrolling does not occur if the cursor moves from outside the viewport to another position outside the viewport.

In the 4111, the cursor address is relative to the first line of the buffer, except when Origin mode is "relative" and edit margins are set, in which case the cursor address is relative to the first line of the scrolling region.

If *line-number* is greater than the number of lines in the scroll buffer (as set by the SET-DIALOG-AREA-BUFFER-SIZE command), the cursor is positioned at the bottom of the scroll buffer. If *column-number* is greater than the number of columns in the scroll buffer (as set by the SET-DIALOG-AREA-CHARS command or the column-mode parameter of the SET-MODE or RESET-MODE command), the cursor is positioned at the right-most column.

DEFAULT

line-number
if 0 or omitted — 1

column-number
if 0 or omitted — 1

ERRORS

[H11 (Level 2): Invalid *line-number* parameter (range is 0 to 32767).

[H21 (Level 2): Invalid *column-number* parameter (range is 0 to 32767)

REFERENCES

RESET-MODE command
SET-DIALOG-AREA-BUFFER-SIZE command (TEK command set)
SET-DIALOG-AREA-CHARS command (TEK command set)
SET-MODE command
SET-TOP-AND-BOTTOM-MARGINS command

CURSOR-POSITION-REPORT Message Type

Acronym: CPR

Syntax

```
CSI  Pn:line-number ;
     Pn:column-number   R
```

PARAMETERS

line-number
The number of the line the cursor is in.

column-number
The number of the column the cursor is in.

DESCRIPTION

The terminal sends this response to the host when it receives the DEVICE-STATUS-Report command with a *status* parameter of 6.

The message contains the line (row) and column coordinates of the cursor position in the scroll buffer. "Line 1, column 1" indicates the upper left-hand corner of the scroll buffer.

In the 4111, these coordinates are relative to the first line of the buffer, except when Origin mode is relative and edit margins are set, in which case the position is relative to the first line of the scrolling region.

When the terminal sends this message, it does not append an *EOM-character* or enter Bypass mode.

REFERENCES

DEVICE-STATUS-REPORT command (ANSI command set)

CURSOR-UP Command

Acronym: CUU

Syntax

```
CSI  [Pn:number-of-lines]  A
```

PARAMETERS

number-of-lines (0 to 32767)
The number of lines to move the cursor up.

DESCRIPTION

This command moves the cursor up the number of lines specified in the *number-of-lines* parameter.

If the cursor moves from inside the viewport to a position currently above the viewport, or from below the viewport to above the viewport, then the text scrolls so as to keep the cursor in view.

If the cursor was not inside the viewport when the command was received, it only becomes visible if it is moved through the viewport.

If the *number-of-lines* parameter specified is greater than the number of lines of text remaining in the scroll buffer, the cursor stops at the first line in the scroll buffer.

In the 4111, if edit margins are set, the CUP command will not move the cursor upward from the scrolling region to the top fixed region.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[A11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

DELETE-CHARACTER Command

Acronym: DCH

Syntax

```
CSI  [Pn:number-of-characters]  P
```

PARAMETERS

number-of-characters (0 to 32767)
The number of characters to be deleted.

DESCRIPTION

This command deletes *number-of-characters* characters from the current line, starting at the current cursor position. Only characters on the current line of text are deleted, even if *number-ofcharacters* is larger than the number of of characters remaining on the line.

Characters to the right of the deleted characters in the current line are moved to the left so that there is no gap left in the line. The characters from the next line are not moved up to fill the empty spaces left by this command.

The cursor position does not change.

DEFAULT

number-of-characters
if 0 or omitted — 1

ERRORS

[P11 (Level 2): Invalid *number-of-characters* parameter.

DELETE-LINE Command

Acronym: DL

Syntax

```
CSI  [Pn:number-of-lines]  M
```

PARAMETERS

number-of-lines (0 to 32767)
The number of lines to be deleted.

DESCRIPTION

This command deletes *number-of-lines* lines from the scroll buffer, starting at the current line.

If the *number-of-lines* parameter less one is equal to or greater than the remaining number of lines in the scroll buffer, the remainder of the scroll buffer, including the line holding the cursor, is deleted.

All lines in the scroll buffer following the deleted lines are shifted up, so that the line immediately following the deleted lines becomes the current line (contains the cursor).

All lines following the shifted portion are erased (i.e., the text in the shifted lines is not repeated).

The cursor position does not change.

In the 4111, if edit margins are set, the DL command's effect will not extend beyond the fixed or scrolling region in which the cursor is located.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[M11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

DEVICE-ATTRIBUTES Command

4111

Acronym: DA

Syntax

CSI [Pn:*device-status-request*] **c**

PARAMETERS

device-status-request (0 to 32767)
The type of status requested.

0 terminal type

DESCRIPTION

This command requests information about terminal characteristics. The terminal responds with the report ^{E}c[?1;2c which means that the terminal is similar to a VT100 with Advance Video Option, including 132 column mode, bold, blink, underline, and reverse image character attributes.

The terminal ignores values between 1 and 32767.

DEFAULT

device-status-request
if omitted — 0

ERRORS

[c11 (Level 2): Invalid *device-status-request* (value greater than 32767)

DEVICE-STATUS-REPORT Command

Acronym: DSR

Syntax

CSI Ps:*status* **n**

PARAMETERS

status
Indicates whether the DSR is a command from the host or a report from the terminal.

0 Report from terminal: ready — no malfunctions detected

5 Command from host: report status using a DSR

6 Command from host: report cursor position using a CPR (see CURSOR-POSITION-Report command)

DESCRIPTION

The DEVICE-STATUS-REPORT either causes the terminal to send a CURSOR-POSITION-REPORT (CPR) or a DEVICE-STATUS-REPORT (DSR) to the host or is a report to the host.

A status value of 5, causes the terminal to send a DEVICE-STATUS-REPORT to the host with a *status* value of 0.

A status value of 6, causes the terminal to send a CURSOR-POSITION-REPORT to the host.

DEFAULT

status
  as shipped — none
  on power-up — none
  if omitted — error [n11

ERRORS

[n11 (Level 2): Invalid *status* parameter; the only valid status specifiers are 0, 5, and 6.

REFERENCES

Cursor-position-report message type

DISABLE-MANUAL-INPUT Command

Acronym: DMI

Syntax

```
Ec `   (Hex 1B 60)
```

DESCRIPTION

This command locks the keyboard. When the keyboard is locked, pressing any key except CANCEL and BREAK rings the terminal bell.

This command has the same effect as the LOCK-KEYBOARD: 1 command.

REFERENCES

ENABLE-MANUAL-INPUT command
LOCK-KEYBOARD command (TEK command set)

ENABLE-MANUAL-INPUT Command

Acronym: EMI

Syntax

```
Ecb
```

DESCRIPTION

This command unlocks the terminal keyboard. It has the same effect as the LOCK-KEYBOARD: 0 command.

REFERENCES

DISABLE-MANUAL-INPUT command
LOCK-KEYBOARD command (TEK command set)

$^{E}_{Q}$ Character (Enquiry)

DESCRIPTION

Receipt of this character from the host causes the answerback message to be sent to the host. If the terminal is in local mode, the $^{E}_{Q}$ character is treated as a no-op and the answerback message is not transmitted.

The terminal does not enter bypass mode when transmitting the answerback message or append optional EOM indicators to the message. If you want to suppress the echo of the message from the host, you must do so at the host.

REFERENCES

SET-EOM-CHARS command
SET-ANSWERBACK-MESSAGE command
SET-REPORT-MAX-LINE-LENGTH command

ERASE-CHARACTER Command

Acronym: ECH

Syntax

```
CSI  [Pn:number-of-characters]  X
```

PARAMETERS

number-of-characters (0 to 32767)
The number of characters to be erased.

DESCRIPTION

This command erases *number-of-characters* characters starting at the current cursor position.

As many characters are erased as are specified in the *number-of-characters* parameter, including characters in the lines that follow the line that holds the cursor (unlike the characters deleted by the DELETE-CHARACTER command). The erasure of characters in a new line begins in column 1 of that line.

Characters that follow the erased characters are not shifted forward to fill the emptied spaces; that is, characters are removed from the scroll buffer, but the character cell is not deleted.

The cursor position does not change.

DEFAULT

number-of-characters
if 0 or omitted — 1

ERRORS

[X11 (Level 2): Invalid *number-of-characters* parameter (range is 0 to 32767).

REFERENCES

DELETE-CHARACTER command

ERASE-IN-DISPLAY Command

Acronym: ED

Syntax

```
CSI  [Ps:erase-extent]  J
```

PARAMETERS

erase-extent
The part of the scroll to be erased. Valid selectors are:

0 from the cursor through the end of the scroll buffer, including the cursor position.

1 from the beginning of the scroll buffer to and including the cursor position.

2 the entire scroll buffer

DESCRIPTION

This command causes all or part of the scroll to be erased, with the erased portion determined by the *erase-extent* parameter and the cursor position.

Characters that follow the erased portion of the scroll are not shifted forward to fill the emptied spaces; that is, characters are removed from the scroll buffer, but the character cell is not deleted.

The cursor position does not change.

DEFAULT

erase-extent
as shipped — none
on power-up — none
if omitted — 0

ERRORS

[J11 (Level 2): Invalid *erase-extent* parameter (must be 0, 1, or 2).

ERASE-IN-LINE Command

Acronym: EL

Syntax

```
CSI  [Ps:erase-extent]  K
```

PARAMETERS

erase-extent
The part of the line to be erased. Valid selectors are:

0 from the cursor position to the right-most occupied column, including the cursor position.

1 from column 1 to and including the cursor position.

2 the entire line.

DESCRIPTION

This command causes all or part of a line to be erased, with the erased part determined by the *erase-extent* parameter and the cursor position.

Characters that follow the erased portion of the scroll are not shifted forward to fill the emptied spaces; that is, characters are removed from the scroll buffer, but the character cell is not deleted.

The cursor position does not change.

DEFAULT

erase-extent
as shipped — none
on power-up — none
if omitted — 0

ERRORS

[K11 (Level 2): Invalid *erase-extent* parameter (must be 0, 1, or 2).

FF Character (Form Feed)

DESCRIPTION

When the terminal receives this character in ANSI mode, it acts as if it had received the LF character.

REFERENCES

LF character

HORIZONTAL-AND-VERTICAL-POSITION Command

Acronym: HVP

Syntax

CSI [Pn:*line-number*]
[; [Pn:*column-number*]] **f**

PARAMETERS

line-number (0 to 32767)
The number of the row to which to move the cursor.

column-number (0 to 32767)
The number of the column to which to move the cursor.

DESCRIPTION

This command moves the cursor to the line and column coordinates specified in the *line-number* and *column-number* parameters.

If the cursor moves to a position outside the current scrolling region, the region is scrolled up or down such that the cursor remains or becomes visible. The new cursor position is either at the bottom or top edge of the scrolling region, depending on the direction of the scroll action and on which edge was closer to the old cursor position.

In the 4111, the cursor address is relative to the first line of the dialog buffer, except when Origin mode is "relative" and edit margins are set, in which case the cursor address is relative to the first line of the scrolling region.

DEFAULT

line-number
if 0 or omitted — 1

column-number
if 0 or omitted — 1

ERRORS

[f11 (Level 2): Invalid *line-number* parameter (range is 0 to 32767).

[f21 (Level 2): Invalid *column-number* parameter (range is 0 to 32767).

HORIZONTAL-TAB-SET Command

Acronym: HTS

Syntax

$^{\mathrm{E}}{}_{\mathrm{C}}\mathrm{H}$

DESCRIPTION

This command sets a tab stop at the current cursor position. The tab position is set in the current column, and is set for all lines (not just the current line).

REFERENCES

$^{\mathrm{H}}{}_{\mathrm{T}}$ character
TABULATION-CLEAR command

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

DESCRIPTION

If the terminal receives the $^{\mathrm{H}}{}_{\mathrm{T}}$ character while in ANSI mode, the cursor moves to the right to the next tab stop set by the HORIZONTAL-TAB-SET command, or to the right-most column in the current line, whichever comes first.

REFERENCES

HORIZONTAL-TAB-SET command

INDEX Command

Acronym: IND

Syntax

```
EcD
```

DESCRIPTION

This command causes the cursor to move down one line without changing column position.

If the cursor is on the bottom line of the viewport, but is not on the bottom line of the scroll buffer, the text is scrolled up. If the cursor is at the bottom line of the scroll buffer, a blank line is added to the bottom of the scroll, the top line of the scroll buffer is removed, and the text is then scrolled up.

If a line is removed at the top of the scroll buffer, the remaining lines in the scroll buffer are renumbered.

In the 4111, if edit margins are set, the IND command will not cause the cursor to move downward from the scrolling region to the bottom fixed region. Instead, the scrolling region will scroll upward.

INSERT-CHARACTER Command

Acronym: ICH

Syntax

```
CSI   [Pn:number-of-characters]   @
```

PARAMETERS

number-of-characters (0 to 32767)
The number of empty spaces to be inserted.

DESCRIPTION

This command shifts the character currently at the cursor position and the characters to its right *number-of-characters* positions to the right. Characters shifted off the end of the line are lost.

The cursor position does not change.

DEFAULTS

number-of-characters
if 0 or omitted — 1

ERRORS

[@11 (Level 2): Invalid *number-of-characters* parameter (range is 0 to 32767).

INSERT-LINE Command

Acronym: IL

Syntax

CSI [Pn:*number-of-lines*] **L**

PARAMETERS

number-of-lines (0 to 32767)
The number of blank lines to be inserted.

DESCRIPTION

This command inserts *number-of-lines* empty lines at the cursor position. The line that contains the cursor and all succeeding lines in the scroll buffer are shifted downward. The last lines in the scroll buffer are lost if the scroll buffer is full.

The cursor position does not change.

In the 4111, if edit margins are set, the effect of the IL command will not extend outside the fixed region or scrolling region in which the cursor is located.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[L11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

LF Character (Line Feed)

Moves the cursor position down one line. Resultant column position is dependent on the setting of Linefeed/Newline Mode (LNM) with the SET-MODE command (SM) or RESET-MODE command (RM).

REFERENCES

RESET-MODE command
SET-MODE command

KEYPAD-APPLICATION-MODE Command

4111

Acronym: TEKKPAM

Host Syntax

```
EC =
```

Setup Syntax

```
KEYPADMODE APPLICATION
```

DESCRIPTION

This command sets the numeric keypad to send characters distinct from the numeric keys on the main keyboard. Table 8-1 lists the characters sent in numeric and application modes. These characters are independent of the state of the Caps Lock, Shift, and Control keys. If a key is redefined with the DEFINE-MACRO command, the new definition is sent if KEYPADMODE is NUMERIC and key expansions are enabled (KEYEXPAND YES). If KEYPADMODE is APPLICATION, however, the application mode character codes prevail. The default setting on power-up is KEYPADMODE NUMERIC.

Table 8-1

KEYPADMODE CHARACTER CODES

Numeric Keypad Key	Numeric Mode	Application Mode
0	0	ECOp
1	1	ECOq
2	2	ECOr
3	3	ECOs
4	4	ECOt
5	5	ECOu
6	6	ECOv
7	7	ECOw
8	8	ECOx
9	9	ECOy
-	-	ECOm
,	,	ECOl
.	.	ECOn
Enter	CR	ECOM
F5	ECOP	ECOP
F6	ECOQ	ECOQ
F7	ECOR	ECOR
F8	ECOS	ECOS

REFERENCES

DEFINE-MACRO command (TEK command set)
ENABLE-KEY-EXPANSION command (TEK command set)
KEYPAD-NUMERIC-MODE command

KEYPAD-NUMERIC-MODE Command

4111

Host Syntax

```
EC>
```

Setup Syntax

```
KEYPADMODE NUMERIC
```

DESCRIPTION

This command sets the keys on the numeric keypad to their default meanings as shown in Table 8-1 (KEYPAD-APPLICATION-MODE).

When KEYPADMODE is NUMERIC and key expansions are enabled (KEYEXPAND YES), any meanings programmed into the keys with the DEFINE-MACRO command will supersede the character codes in Table 8-1.

REFERENCES

DEFINE-MACRO command (TEK command set)
KEYPAD-APPLICATION-MODE command

NEXT-LINE Command

Acronym: NEL

Syntax

```
ECE
```

DESCRIPTION

This command causes a carriage return/line feed action, moving the cursor to the beginning of the next line.

If the cursor is on the bottom line of the viewport, but is not on the bottom line of the scroll buffer, the text is scrolled up. If the cursor is at the bottom line of the scroll buffer, a blank line is added to the bottom of the scroll, the top line of the scroll buffer is removed, and the text is then scrolled up.

In the 4111, if edit margins are set, the NEL command will not move the cursor downward from the scrolling region into the bottom fixed region. Instead, the scrolling region will scroll upward.

REPORT-SYNTAX-MODE Command

4111

Syntax

^{E}c#! *report-type*

PARAMETERS

report-type

Type of report requested.

0	Command code

DESCRIPTION

This command causes the terminal to send a terminal-settings-report to the host. The report contains the terminal's command code status.

This command has the same effect as a REPORT-TERMINAL-SETTINGS command issued to the SELECT-CODE command (as if ^{E}cIQ%! was sent from the host in TEK mode).

This command is recognized in all modes: ANSI, Edit, TEK, and VT52.

ERRORS

#!11 (Level 2): Invalid *report-type* parameter

REFERENCES

REPORT-TERMINAL-SETTINGS command

RESET-MODE Command

Acronym: RM

Host Syntax

CSI [Ps:*mode* [; [Ps:*mode*]...]] **l**

Setup Syntax

```
LOCKKEYBOARD NO
INSERTREPLACE REPLACE
ECHO YES
LFCR YES
DAMODE OVERSTRIKE
CURSORKEYMODE NO
CODE VT52
DACHARS 80
ORIGINMODE ABSOLUTE
AUTOWRAP NO
AUTOREPEAT NO
```

PARAMETERS

mode

The terminal mode you want to reset. Valid selectors are:

Selector	Mode	Action
2	Keyboard Action	Unlocks the keyboard; same as ENABLE-MANUAL-INPUT and LOCK-KEYBOARD: 0
4	Insertion/ Replacement	Existing characters are replaced as new characters are entered.
12	Send/Receive	Sets the terminal to local echo (Echo mode is turned on.)
20	Linefeed/ Newline	^{L}F characters received move the cursor down only. (LFCR mode turned off.)
<1	Overstrike/ Replace	Sets dialog-area-writing-mode to Replace mode.

?1	Cursor Keys	(4111 only.) Sets function keys F1 to F4 to transmit ANSI cursor control commands (unless they have been programmed with the DEFINE-MACRO command). See Table 8-2.
?2	ANSI/VT52	(4111 only.) Sets the terminal to VT52 syntax mode.
?3	Column	(4111 only.) Sets dialog-area-characters to 80 and toggles DA visibility so the new setting has immediate effect. (In Setup, DACHARS can accept values from 5 to 80.) Setting DACHARS erases the dialog area.
?6	Origin	(4111 only.) Sets the origin mode to Absolute (line numbering begins with the first line of the dialog buffer), reduces the dialog area buffer size if it is greater than 48 lines, and moves the cursor to line 1, column 1.
?7	Auto-wrap	Disables the auto-wrap feature — the cursor does not automatically wrap to the next line at the end of a line of data.
?8	Auto-repeat	Keys do not automatically repeat when held down.

Table 8-2

CURSOR KEY MODE CODES (4111)

Function Key	Code Sent When Set	Code Sent When Reset
F1	EcOA	Ec[A
F2	EcOB	Ec[B
F3	EcOC	Ec[C
F4	EcOD	Ec[D

NOTE

When cursor key mode is reset, and key expansions are enabled, any meanings programmed into function keys F1 through F4 (with the DEFINE-MACRO command) will supersede the codes in Table 8-2. However, when cursor key mode is set, the codes in Table 8-2 take precedence.

DESCRIPTION

This command resets the action of various terminal modes.

Reset is the default for the following modes:

- Keyboard Action mode
- Insertion/Replacement mode
- Overstrike/Replace mode
- Linefeed/Newline mode
- Cursor Keys mode
- Column mode
- Origin mode

Once a mode is reset, it remains reset until set by a SET-MODE command, until a command affecting that particular mode is received, or until a RESET-TO-INITIAL-STATE command is received.

You can reset several terminal modes with one RESET-MODE command. However, private parameters (those that begin with < or **?**) and standard parameters (those consisting only of the digits from **0** to **9**) should be sent in separate RM commands. Do not mix standard and private parameters in the same command.

ERRORS

[l*1 (Level 0): Invalid or missing *mode* selector value. The character position shown as an asterisk is the selector value in error. Values 10 and above are reported as 9.

REFERENCES

DEFINE-MACRO command (TEK command set)
ECHO command (TEK command set)
ENABLE-KEY-EXPANSION command (TEK command set)
INDEX command
LFCR command (TEK command set)
LOCK-KEYBOARD command (TEK command set)
RESET-TO-INITIAL-STATE command
SET-DIALOG-AREA-CHARS command (TEK command set)
SET-DIALOG-AREA-WRITING-MODE command
(TEK command set)
SET-MODE command

RESET-TO-INITIAL-STATE Command

Acronym: RIS

Syntax

^{E}cc

DESCRIPTION

This command causes the terminal to reinitialize the dialog area. Dialog area initialization erases the dialog area if it is enabled, moves the cursor to Row 1, Column 1, resets the tab stops to default positions, clears the scroll buffers, moves the screen viewport to the top of the scroll buffer, and sets the graphic rendition to default.

This command has the same effect as pressing the MASTER RESET button or entering the RESET command.

REFERENCES

RESET command (TEK command set)

RESTORE-CURSOR Command

Acronym: TEKRC

Syntax

^{E}c8

DESCRIPTION

This command restores the cursor position and graphic rendition previously saved with the SAVE-CURSOR command. If the SAVE-CURSOR command was not used to save the cursor position, the cursor is moved to the upper left corner of the scroll buffer (line 1, column 1), and the graphic rendition is set to the default (plain text).

REFERENCES

SAVE-CURSOR command
SET-GRAPHIC-RENDITION command

REVERSE-INDEX Command

Acronym: RI

Syntax

$^{E}_{C}$M

DESCRIPTION

This command moves the cursor up one line up without changing the column position.

If the cursor is at the top of the viewport, but is not at the top of the scroll buffer, the text is scrolled down and the cursor remains in the viewport.

If the cursor is at the top of the viewport and is also at the top of the scroll buffer, a blank line is added to the top of the scroll and the buffer is scrolled down. If the scroll buffer is full the last line of data is lost.

In the 4111, if edit margins are set, the RI command will not move the cursor upward from the scrolling region to the top fixed region.

SAVE-CURSOR Command

Acronym: TEKSC

Syntax

$^{E}_{C}$7

DESCRIPTION

This command causes the terminal to save the present cursor position and graphic rendition.

You can restore the saved information with the RESTORE-CURSOR command.

REFERENCES

RESTORE-CURSOR command
SET-GRAPHIC-RENDITION command

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

DESCRIPTION

This character is a command terminator. When it is received in the middle of a command, the command is terminated and any characters belonging to that command that have already been received are discarded. When this occurs, a snoopy character $^{S}{}_{B}$ appears on the terminal screen.

The $^{C}{}_{N}$ character has a similar effect.

REFERENCES

$^{C}{}_{N}$ character

SCROLL-DOWN Command

Acronym: SD

Syntax

CSI [Pn:*number-of-lines*] **T**

PARAMETERS

number-of-lines (0 to 32767)
The number of lines to be scrolled.

DESCRIPTION

This command shifts the scroll buffer down within the dialog area scrolling region the specified number of lines. Lines at the bottom of the region are scrolled out of the view as lines appear at the top.

Scrolling stops when *number-of-lines* lines have been scrolled or when the top line of the scroll buffer is at the top of the scrolling region.

The cursor position within the scroll buffer does not change, so this command may move the cursor out of view.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[T11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

REFERENCES

SCROLL-UP command

SCROLL-UP Command

Acronym: SU

Syntax

CSI [Pn:*number-of-lines*] **S**

PARAMETERS

number-of-lines (0 to 32767)
The number of lines to be scrolled.

DESCRIPTION

This command shifts the scroll buffer up within the dialog area scrolling region by the specified number of lines. Lines at the top of the region are scrolled out of view as lines appear at the bottom.

Scrolling stops when *number-of-lines* lines have been scrolled or when the bottom line of the scroll buffer is in view.

The cursor position within the scroll buffer does not change, so this command may move the cursor out of view.

DEFAULT

number-of-lines
if 0 or omitted — 1

ERRORS

[S11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

REFERENCES

SCROLL-DOWN command

SELECT-CHARACTER-SET Command

Acronym: SCS

Host Syntax (to select G0)

^{E}**c(** *character-set*

Host Syntax (to select G1)

^{E}**c)** *character-set*

Setup Syntax (to select G0)

SELECTCHARSET G0 *character-set*

Setup Syntax (to select G1)

SELECTCHARSET G1 *character-set*

PARAMETERS

character-set
The code that represents the character set to be assigned.

ASCII Code	Character Set
A	United Kingdom
B	US (ASCII)
G	Swedish
K	German
f (or R)	French (see note)
\	Danish/Norwegian (ADE 96)
0	Ruling character set
3	Supplementary
I	Japanese Katakana (Option 4K only)
J	Japanese Roman (Option 4K only)

NOTE

To designate the French character set, the code **f** *is the current standard escape sequence parameter. For compatibility with an earlier version of the French character set, the terminal will accept* **R** *as a synonym for* **f**.

Valid codes for Option 4K are A, B, I, J, and 0.

DESCRIPTION

On the 4115/4120 Series, this command is recognized, but ignored. On the 4111, it is used to assign special character sets to the G0 and G1 character sets.

You may access two different character fonts by using the SI (Shift In) and SO (Shift Out) commands to select either of the currently defined G0 or G1 character sets.

On power-up, the character set associated with the currently attached keyboard is assigned to both G0 and G1. This command allows you to assign different fonts to G0 and G1. If the keyboard is detached and reattached, G0 and G1 will reset to the power-up default.

REFERENCES

SI Character
SO Character

SELECT-CODE Command

Host Syntax

```
EC%!  int:command-set
```

Setup Syntax

```
CODE SP  command-set
```

PARAMETERS

command-set (0 to 3)
Selects the command set to be recognized by the terminal.

0	TEK command set
1	ANSI X3.64 command set
2	EDIT environment, ANSI X3.64 command set
3	VT52 command set

Setup mode parameters are TEK, ANSI, EDIT, and VT52.

DESCRIPTION

This command sets the terminal to recognize the commands of either the TEKTRONIX 4100 System, ANSI X3.64, or the VT52 system. Since there are some syntax commonalities between the different command sets, the terminal can recognize only one command set at a time.

If *command-set* is **0**, the terminal is set so that it understands the TEK 4100 commands, which are described in Section 7 of this manual. The terminal is also put into Alpha mode.

If *command-set* is **1**, the terminal is set so that it understands the ANSI X3.64 and ISO 6429 commands, which are described in this section.

If *command-set* is **2**, the terminal is set up for running VT100 applications programs. This option has the following effects:

- Sets the terminal to recognize ANSI commands.
- Sets Origin mode to absolute.
- Sets dialog area and dialog buffer to 24 lines.
- If no dialog area is enabled, enables dialog area 1 and makes it visible.
- Defines a scrolling region of 24 lines.
- Disables all expansions of programmed keys. All keys assume their default meanings (the programmed meanings can be enabled from Setup by typing KEYEXPAND YES).
- Sets Insert/Replace mode to Replace.

If *command-set* is **3**, the terminal is set so that it understands the VT52 commands described in Section 9. Entering VT52 mode causes the default character set (determined by the keyboard attached) to be selected as the G0 and G1 sets.

In the 4111, if no dialog area is currently enabled when the terminal exits TEK mode to enter ANSI, EDIT, or VT52 mode, then dialog area 1 is enabled. In that case, when the terminal returns to TEK mode all dialog areas will be disabled. If the terminal receives a Setup DAENABLE NO command while in ANSI, EDIT, or VT52 mode, it defers executing that command until it next enters TEK mode. Commands to enable a specific dialog area, however, have immediate effect, and that effect persists even when the terminal returns to TEK mode.

DEFAULTS

command-set
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

%!00	(Level 0):	Unrecognized command; terminal firmware is Version 3.
%!11	(Level 2):	Invalid command-set (must be 0, 1, 2 or 3).

REFERENCES

ENTER-ALPHA-MODE command (TEK command set)
RESET-MODE command
SET-MODE command
SET-TOP-AND-BOTTOM-MARGINS command
(ANSI/Edit command set)

SELECT-GRAPHIC-RENDITION Command

Acronym: SGR

Syntax

```
CSI  [Ps:rendition  [;[Ps:rendition]...]]  m
```

PARAMETERS

rendition
The style(s) in which you want text characters to appear. Valid specifiers are:

0 primary rendition (default): no blink, no underscore, positive image, current dialog area foreground index

1 alternate foreground index (bold)

4 underscore

5 slow blink

7 reverse image: foreground and background indices are interchanged

24 (4111 only) no underscore. Cancels *rendition* 4

25 (4111 only) no blink. Cancels *rendition* 5

27 (4111 only) positive video text. Cancels *rendition* 7

30 (4111 only) selects character color pair 8 (default: black on white)

31 (4111 only) selects character color pair 2 (default: red on transparent)

32 (4111 only) selects character color pair 3 (default: green on transparent)

33 (4111 only) selects character color pair 4 (default: yellow on transparent)

34 (4111 only) selects character color pair 5 (default: blue on transparent)

35 (4111 only) selects character color pair 6 (default: magenta on transparent)

36 (4111 only) selects character color pair 7 (default: cyan on transparent)

37 (4111 only) selects character color pair 1 (default: white on transparent)

39 (4111 only) selects default character color pair. (On the 4111 this is color pair 1, which has a default of white on transparent)

DESCRIPTION

This command invokes the graphic rendition that is specified by the *rendition* parameters. All following characters in the data stream are displayed according to the specified parameters until the next occurrence of a SET-GRAPHIC-RENDITION command, a RESTORE-CURSOR command, or a RESET-TO-INITIAL-STATE command.

On the 4111, color pair indices are set with the SET-DIALOG-AREA-INDEX command.

The alternate foreground index (bold) is that set by the SET-DIALOG-AREA-ALTERNATE-INDEX command.

This command controls the graphic rendition of dialog area characters. The 4111 can have up to 8 pairs of foreground/background color characters defined for the dialog area. The 4115/4120 Series have two character colors at one time. Color pairs can be redefined with the SET-DIALOG-AREA-INDEX command. The colors of an index can be redefined with the SET-SURFACE-COLOR-MAP command.

If you include *rendition* **0** in the list of *rendition* specifiers, all renditions previously set in the same list are cancelled, and only the specifiers that occur following the occurrence of *rendition* **0** are executed.

DEFAULT

rendition
as shipped — none
on power-up — 0
if omitted — 0

ERRORS

[m11 (Level 2): Invalid *rendition* parameter.

REFERENCES

RESET-TO-INITIAL-STATE command
RESTORE-CURSOR command
SAVE-CURSOR command
SET-DIALOG-AREA-ALTERNATE-INDEX command
(TEK command set)
SET-DIALOG-AREA-INDEX command

SET-ANSWERBACK-MESSAGE Command

4111

Setup Syntax

```
ANSWERBACK   answerback-message
```

PARAMETERS

answerback-message
A character array of length 0 to 20. The characters must be ASCII characters in the range 0 to 127 (decimal). The characters must be delimited, at each end of the string, with identical ASCII characters not included in the message string.

DESCRIPTION

This command defines the answerback message that will be sent from the terminal to the host in response to receipt of an EQ (Enquiry) character from the host or when the CTRL-BREAK character is pressed on the keyboard. Except for Local mode, the terminal will respond with the answerback message while in any of its operating modes (TEK, ANSI, Edit, or VT52), but this command is recognized only in Setup mode.

The answerback message is saved in non-volatile memory when the SAVE-NONVOLATILE-PARAMETERS command is issued and remembered when the terminal is powered-down.

DEFAULTS

answerback-message
as shipped — empty array
on power-up — remembered
if omitted — empty array

ERRORS

N011 (Level 2): Invalid answerback-message

REFERENCES

EQ Character
BREAK key

SET-MODE Command

Acronym: SM

Host Syntax

```
CSI  [Ps:mode  [;[Ps:mode]...]]  h
```

Setup Syntax

```
LOCKKEYBOARD YES
INSERTREPLACE INSERT
ECHO NO
LFCR NO
DAMODE REPLACE
CURSORKEYMODE YES
DACHARS 132
ORIGINMODE RELATIVE
AUTOWRAP YES
AUTOREPEAT YES
```

PARAMETERS

mode
The terminal mode you want to reset. Valid selectors are:

Selector	Mode	Action
2	Keyboard Action	Locks the keyboard. Same as DISABLE-MANUAL-INPUT and LOCK-KEYBOARD: 1
4	Insertion/ Replacement	As new characters are entered, existing characters are moved to the right, being lost off the right edge.
12	Send/Receive	Sets the terminal to remote echo (Echo mode off)
20	Linefeed/ Newline	LF characters received move the cursor to column one of the next line (LFCR mode on)
<1	Overstrike/ Replace	Dialog-area-writing- mode set to Overstrike.
?1	Cursor Keys	(4111 only.) Sets Function Keys F1 to F4 to transmit application program codes. See RESET-MODE, Table 8-1, for a description of these codes. If these keys are defined with the DEFINE-MACRO command, they will still transmit their application values.
?3	Column	(4111 only.) Sets dialog-area-characters to 132 and toggles DA visibility so the new setting has immediate effect. (In Setup, DACHARS can accept values from 5 to 132.) Setting DACHARS erases the dialog area.
?6	Origin	Sets origin-mode to relative (line numbering begins with the first line in the scrolling region), and moves the cursor to line 1, column 1.
?7	Auto-wrap	Enables the auto-wrap feature — the cursor automatically wraps to the next line at the end of a line of data.
?8	Auto-repeat	Most keys automatically repeat when held down.

DESCRIPTION

This command sets the action of various terminal modes.

SET is the default for the following modes:

- Auto-wrap mode (AUTOWRAP YES)
- Auto-repeat mode (AUTOREPEAT YES)
- Send/Receive mode (ECHO NO)

Once a mode is set, it remains set until reset by a RESET-MODE command, a command that affects that particular mode, or a RESET-TO-INITIAL-STATE command.

You can set several terminal modes with one SET-MODE command. However, private parameters (those that begin with < or **?**) and standard parameters (those consisting only of the digits from **0** to **9**) should be sent in separate SM commands. Do not mix standard and private parameters in the same command.

ERRORS

[h11 (Level 2): Invalid or missing *mode* value.

REFERENCES

DISABLE-MANUAL-INPUT command
ECHO command (TEK command set)
LFCR command (TEK command set)
LOCK-KEYBOARD command (TEK command set)
RESET-MODE command
RESET-TO-INITIAL-STATE command
SET-DIALOG-AREA-CHARS command
SET-DIALOG-AREA-WRITING-MODE command
(TEK command set)
SET-SURFACE-COLOR-MAP command

SET-TOP-AND-BOTTOM-MARGINS Command

4111

Acronym: TEKSTBM

Host Syntax

CSI [Pn:*top-margin*] [; [Pn:*bottom-margin*]] **r**

Setup Syntax

EDITMARGIN *top-margin* *bottom-margin*

PARAMETERS

top-margin (0 to 32767)
The row number of the top-margin.

bottom-margin (0 to 32767)
The row number of the bottom margin.

DESCRIPTION

This command sets the dialog buffer's edit margins. The *top-margin* value specifies the dialog buffer row to be the top line of the scrolling region. Similarly, the *bottom-margin* value specifies the dialog buffer row to be the scrolling region bottom line.

This command also forces the cursor to the home position (row 1, column 1) and sets the dialog area buffers to the maximum displayable size (48), unless the dialog area buffer size is already set to less than 48.

Rows in the dialog buffer above the top margin and the rows below the bottom margin become fixed regions. No scrolling action can occur in these fixed regions.

If you or the host program reset the size of the dialog buffer, the scrolling region is automatically reestablished as the entire buffer.

If *top-margin* is zero or omitted, it defaults to one. If *bottom-margin* is zero or omitted, it defaults to the maximum number of lines in the dialog area (for the 4111, this is 48).

ERRORS

[r11	(Level 2):	Invalid *top-margin* value.
[r21	(Level 2):	Invalid *bottom-margin* value.

REFERENCES

SET-DIALOG-AREA-BUFFER-SIZE command
(TEK command set)

S_I Character (Shift In)

DESCRIPTION

The terminal allows you to access two different character sets by using the S_I (Shift In) and S_O (Shift Out) commands to switch between the currently defined G0 or G1 character sets.

The S_I command invokes the currently defined G0 character set. This may be the 94 graphic characters from the ASCII character set, or the corresponding 94 characters from the United Kingdom, French, Swedish, Danish/Norwegion, German, supplementary, or special rulings character sets. Changing the keyboard automatically selects the corresponding keyboard as the default, but all character sets are available. You may use the SCS (SELECT-CHARACTER-SET) command to assign to G0 the character set you want no matter which keyboard is connected.

To select the G1 character set, use the S_O (Shift Out) command.

REFERENCES

SELECT-CHARACTER-SET command
S_O Character

S_O Character (Shift Out)

DESCRIPTION

The terminal allows you to access two different character sets by using the S_I (Shift In) and S_O (Shift Out) commands to switch between the currently defined G0 or G1 character sets.

The S_O command invokes the currently defined G1 character set. When a keyboard is plugged into the terminal, the character set associated with that keyboard is designated as both the G0 and G1 set. You may use the SCS (SELECT-CHARACTER-SET) command to assign to G1 a different character set than the one associated with the current keyboard.

To select the G0 character set, use the S_I (Shift In) command.

REFERENCES

SELECT-CHARACTER-SET command
S_I Character

TABULATION-CLEAR Command

Acronym: TBC

Syntax

CSI [Ps:*tab-clear-extent*] **g**

PARAMETERS

tab-clear-extent
Defines the tab stops you want cleared. Valid selectors are:

0 Clear the tab stop at the cursor position

2 Clear all tab stops in the active line (same as **3**).

3 Clear all tab stops (same as **2**).

DESCRIPTION

This command clears one or all tab stops, according to the selected *tab-clear-extent* parameter.

If you select **0**, clear the tab at the cursor position, and there is no tab at the cursor position, the command is ignored.

DEFAULT

tab-clear-extent
as shipped — none
on power-up — none
if omitted — 0

ERRORS

[g11 (Level 2): Invalid *tab-clear-extent* parameter.

REFERENCES

HORIZONTAL-TAB-STOPS command

VT Character (Vertical Tab)

DESCRIPTION

When the terminal receives this character, it acts as if it had received an INDEX command: it moves down one line.

In the 4111, if edit margins are set, VT will not move the cursor downward from the scrolling region into the bottom fixed region. Instead, the scrolling region will scroll upward.

REFERENCES

INDEX command

Section 9

VT52 MODE (4111 ONLY)

INTRODUCTION

VT52 mode configures the 4111 terminal to run most VT52 application programs. You may select VT52 mode with one of the following methods:

- Enter CODE VT52 while in Setup mode
- Send an RM command (^{E}c[?2l) from the host while in ANSI/Edit command set
- Send a SELECT-CODE command (^{E}c%!3) from the host while in TEK, ANSI, or Edit mode

While in VT52 mode, the terminal only recognizes the following commands:

- The VT52 commands described in this section.
- Control characters (bytes in the range from ADE 0 to ADE 31). In VT52 mode, these characters perform the same functions as in ANSI mode, which is described in in Section 8.
- The SELECT-CODE and Report-Syntax-MODE commands, both of which work in all syntax modes.

While in VT52 mode, only two character sets are available: the default character set for the keyboard attached to the terminal (typically ASCII) and the rulings character set. The ENTER-GRAPHICS-MODE and EXIT-GRAPHICS-MODE commands are used to switch between these two character sets. When entering VT52 mode, the keyboard default character set is selected.

When the terminal exits VT52 mode, it also exits the graphics submode (if it was in that submode) and selects the default character set.

CURSOR-DOWN Command

Syntax

$^{E}_{C}$B

DESCRIPTION

The CURSOR-DOWN command moves the cursor down one line without moving it horizontally. The cursor moves with respect to the dialog buffer and stops at the last row of the dialog buffer. However, if margins are set and the cursor is within the scrolling region, the cursor stops at the bottom margin of the scrolling region.

REFERENCES

SET-TOP-AND-BOTTOM-MARGINS Command (ANSI/Edit command set)

CURSOR-LEFT Command

Syntax

$^{E}_{C}$D

DESCRIPTION

The CURSOR-LEFT command moves the cursor one column to the left. This command works as does the ANSI mode command CUB (CURSOR-BACKWARD) with a parameter of 1.

REFERENCES

CURSOR-BACKWARD command (ANSI/Edit command set)

CURSOR-RIGHT Command

Syntax

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

DESCRIPTION

The CURSOR-RIGHT command moves the cursor one column to the right. The cursor does not move beyond the right-most column. This command works as does the ANSI mode command CUF (CURSOR-FORWARD) with a parameter of 1.

REFERENCES

CURSOR-FORWARD command (ANSI/Edit command set)

CURSOR-TO-HOME Command

Syntax

${}^{E}{}_{C}$H

DESCRIPTION

The CURSOR-TO-HOME command moves the cursor to the "home" position. The home position depends upon the state of the ANSI mode Origin. If Origin mode is Absolute (TEKOM reset), then the "home" position is line 1, column 1 of the dialog buffer. If Origin mode is Relative (TEKOM set), then the "home" position is the first line of the scrolling region, column 1.

The command works as does the ANSI mode command CUP (CURSOR-POSITION) with parameters of 1 and 1.

REFERENCES

CURSOR-POSITION command (ANSI/Edit command set)
RESET-MODE command (ANSI/Edit command set)
SET-MODE command (ANSI/Edit command set)

CURSOR-UP Command

Syntax

$^{E}_{C}$**A**

DESCRIPTION

The CURSOR-UP command moves the cursor up one line without moving it horizontally. The cursor moves with respect to the dialog buffer and stops at the first row of the dialog buffer. However, if margins are set and the cursor is within the scrolling region, the cursor stops at the top margin of the scrolling region.

DIRECT-CURSOR-ADDRESS Command

Syntax

$^{E}_{C}$**Y** *line* *column*

PARAMETERS

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

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

DESCRIPTION

The DIRECT-CURSOR-ADDRESS command moves the cursor to the specified line and column. For example, to move the cursor to line 3, column 1, give the command $^{E}_{C}$Y"$^{S}_{P}$ since the decimal equivalent of " is 34 (3 + 31) and the decimal equivalent of $^{S}_{P}$ is 32 (1 + 31).

Line numbering depends upon the current state of the Origin (Absolute or Relative).

If a parameter is out of range, the cursor will not change position for that parameter. However, the cursor will move to the other parameter position if it is within range.

REFERENCES

RESET-MODE command (ANSI/Edit command set)
SET-MODE command (ANSI/Edit command set)

ENTER-ALTERNATE-KEYPAD-MODE Command

Syntax

```
Ec =
```

DESCRIPTION

The ENTER-ALTERNATE-KEYPAD-MODE command sets the numeric keypad keys and function keys F5 through F8 to assume their Alternate Keypad mode code values. Table 9-1 lists the codes sent for each of the keys in both modes.

Table 9-1

NUMERIC KEYPAD CODES

Numeric Keypad Key	Characters Sent as Factory Default	Characters Sent in Alternate Keypad Mode
0	0	Ec?p
1	1	Ec?q
2	2	Ec?r
3	3	Ec?s
4	4	Ec?t
5	5	Ec?u
6	6	Ec?v
7	7	Ec?w
8	8	Ec?x
9	9	Ec?y
-	-	Ec?m
,	,	Ec?l
.	.	Ec?n
Enter	CR>	Ec?M
F5	EcP	EcP
F6	EcQ	EcQ
F7	EcR	EcR
F8	EcS	EcS

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

If you alternate between ANSI (or Edit) mode and VT52 mode, the numeric keypad retains its setting even though the selected command set changes. That is, if the keypad is set to send the default numeric character codes while in ANSI mode, then it sends the same default codes while in VT52 mode. If the keypad is set to send application codes while in ANSI mode, then it sends those alternate keypad codes in VT52 mode.

This command works the same as the ANSI mode command TEKKPAM (KEYPAD-APPLICATION-MODE) except that the alternate keypad key definitions are different.

REFERENCES

DEFINE-MACRO command (TEK command set)
KEYPAD-APPLICATION-MODE (ANSI/Edit command set)

ENTER-ANSI-MODE Command

Syntax

```
EC<
```

DESCRIPTION

This command places the terminal in ANSI mode. The terminal will interpret all subsequent commands according to ANSI standard X3.64. This command has the same effect as the SELECT-CODE command with a parameter of **1**, or the Setup mode CODE ANSI command.

ENTER-GRAPHICS-MODE Command

Syntax

```
ECF
```

DESCRIPTION

The ENTER-GRAPHICS-MODE command selects the rulings character set as the G0 character set. The terminal will remain in Graphics mode until you issue an EXIT-GRAPHICS-MODE command except if you issue the ENTER-ANSI-MODE command while the terminal is still in Graphics mode, the terminal will exit Graphics mode before it exits VT52 mode.

REFERENCES

ENTER-ANSI-MODE command
EXIT-GRAPHICS-MODE command

$^{E}\mathrm{Q}$ Character (Enquiry)

DESCRIPTION

Receipt of this character by the terminal causes the answerback message to be sent to the host. If the terminal is in local mode, the $^{E}\mathrm{Q}$ character is treated as a no-op and the answerback message is not transmitted.

The terminal does not enter bypass mode when transmitting the answerback message or append optional EOM indicators to the message. If you want to suppress the echo of the message from the host, you must do so at the host.

REFERENCES

SET-EOM-CHARS command
SET-ANSWERBACK-MESSAGE command
SET-Report-MAX-LINE-LENGTH command

ERASE-TO-END-OF-LINE Command

Syntax

$^{E}\mathrm{C}$K

DESCRIPTION

The ERASE-TO-END-OF-LINE command erases all characters from the cursor to the end of the current line. The cursor position does not change. This command works as does the ANSI mode command EL (ERASE-IN-LINE) with a parameter of 0.

REFERENCES

ERASE-IN-LINE command (ANSI/Edit command set)

ERASE-TO-END-OF-SCREEN Command

Syntax

```
ᴱcJ
```

DESCRIPTION

The ERASE-TO-END-OF-SCREEN command erases all characters from the cursor to the end of the dialog buffer. The cursor position does not change. This command works as does the ANSI mode command ED (ERASE-IN-DISPLAY) with a parameter of 0.

REFERENCES

ERASE-IN-DISPLAY command (ANSI/Edit command set)

EXIT-ALTERNATE-KEYPAD-MODE Command

Syntax

```
ᴱc>
```

DESCRIPTION

The EXIT-ALTERNATE-KEYPAD-MODE command resets the numeric keypad keys and function keys F5 through F8 to their factory default values, or their programmed values if they have been programmed. The factory default character codes are shown in Table 9-1 (under the ENTER-ALTERNATE-KEYPAD-MODE command).

This command works the same as the ANSI mode command TEKKPNM (KEYPAD-NUMERIC-MODE) except that the default key definitions for the function keys F5 through F8 are different.

REFERENCES

ENTER-ALTERNATE-KEYPAD-MODE command
KEYPAD-NUMERIC-MODE command
(ANSI/Edit command set)

EXIT-GRAPHICS-MODE Command

Syntax

$^{E}{}_{C}$G

DESCRIPTION

The EXIT-GRAPHICS-MODE command restores the default character set that was in effect before the current ENTER-GRAPHICS-MODE command was issued.

REFERENCES

ENTER-GRAPHICS-MODE command

IDENTIFY Command

Syntax

$^{E}{}_{C}$Z

DESCRIPTION

This command identifies the terminal to the host. When the host issues this command, the terminal sends its identifier escape sequence $^{E}{}_{C}$/Z to the host.

REPORT-SYNTAX-MODE Command

Syntax

^{E}c#!0

DESCRIPTION

The REPORT-SYNTAX-MODE command sends a Terminal Settings Report that contains the syntax mode status to the host. This command has the same effect as a REPORT-TERMINAL-SETTINGS command issued for the SELECT-CODE command (as if ^{E}cIQ%! were sent from the host).

This command is recognized in all modes: ANSI, Edit, TEK, and VT52.

ERRORS

#!11 (Level 2): Invalid report-type parameter (must be 0).

REFERENCES

Report-TERMINAL-SETTINGS command
(TEK command set)

REVERSE-LINEFEED Command

Syntax

^{E}cI

DESCRIPTION

The REVERSE-LINEFEED command moves the cursor up one line without affecting the cursor position within the line. This command works the same as the ANSI mode command RI (REVERSE-INDEX).

REFERENCES

REVERSE-INDEX command (ANSI/Edit command set)

SELECT CODE Command

Host Syntax

```
EC%!   int:command-set
```

Setup Syntax

```
CODESP   command-set
```

PARAMETERS

command-set (0 to 3)
Selects the command set to be recognized by the terminal.

0 TEK command set
1 ANSI X3.64 command set
2 EDIT environment, ANSI X3.64 command set
3 VT52 command set

Setup mode parameters are TEK, ANSI, EDIT, and VT52.

DESCRIPTION

This command sets the terminal to recognize the commands of either the TEKTRONIX 4100 System, ANSI X3.64, or the VT52 system. Since there are some syntax commonalities between the different command sets, the terminal can recognize only one command set at a time.

If *command-set* is **0**, the terminal is set so that it understands the TEK 4100 commands, which are described in Section 7 of this manual. The terminal is also put into Alpha mode.

If *command-set* is **1**, the terminal is set so that it understands the ANSI X3.64 and ISO 6429 commands, which are described in Section 8.

If *command-set* is **2**, the terminal is set up for running VT100 applications programs. This option has the following effects:

- Sets the terminal to recognize ANSI commands.
- Sets Origin mode to absolute.
- Sets dialog area and dialog buffer to 24 lines.
- If no dialog area is enabled, enables dialog area 1 and makes it visible.
- Defines a scrolling region of 24 lines.
- Disables all expansions of programmed keys. All keys assume their default meanings (the programmed meanings can be enabled from Setup by typing KEY-EXPAND YES).
- Sets Insert/Replace mode to Replace.

If *command-set* is **3**, the terminal is set so that it understands the VT52 commands described in this section. Entering VT52 mode causes the default character set (determined by the keyboard attached) to be selected as the G0 and G1 sets.

In the 4111, if no dialog area is currently enabled when the terminal exits TEK mode to enter ANSI, EDIT, or VT52 mode, then dialog area 1 is enabled. In that case, when the terminal returns to TEK mode all dialog areas will be disabled. If the terminal receives a setup DAENABLE NO command while in ANSI, EDIT, or VT52 mode, it defers executing that command until it next enters TEK mode. Commands to enable a specific dialog area, however, have immediate effect, and that effect persists even when the terminal returns to TEK mode.

DEFAULTS

command-set
as shipped — 0
on power-up — 0
if omitted — 0

ERRORS

%!00 (Level 0): Unrecognized command; terminal firmware is Version 3.

%!11 (Level 2): Invalid command-set (must be 0, 1, 2 or 3).

REFERENCES

ENTER-ALPHA-MODE command (TEK command set)
RESET-MODE command
SET-MODE command
SET-TOP-AND-BOTTOM-MARGINS command
(ANSI/Edit command set)

Appendix A

ASCII CODE 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
A-1	ASCII Code Chart
A-2	Characters Used in *Char* Parameters
A-3	Characters Used in *Int* Parameters
A-4	Characters Used in *Int-Report* and *Intc-Report* Parameters
A-5	Characters Used in *Xy* Parameters
A-6	Characters Used in *Xy-Report* Parameters

Table A-1

ASCII (IS0-7-US) CODE CHART

B7 B6 B5 BITS				Ø Ø Ø	Ø Ø 1	Ø 1 Ø	Ø 1 1	1 Ø Ø	1 Ø 1	1 1 Ø	1 1 1
B4	B3	B2	B1	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
Ø	Ø	Ø	Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
Ø	Ø	Ø	1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1	Ø	Ø	1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1	Ø	1	Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1	1	Ø	1	CR 13	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1	1	1	Ø	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1	1	1	1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

4526-21

Table A-2

CHARACTERS USED IN CHAR PARAMETERS

B7 B6 B5 BITS				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	
B4	B3	B2	B1								
Ø	Ø	Ø	Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
Ø	Ø	Ø	1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1	Ø	Ø	1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1	Ø	1	Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1	1	Ø	1	CR 13	GS 29	- 45	= 61	M 77	] 93	m 109	} 125
1	1	1	Ø	SO 14	RS 30	. 46	> 62	N 78	^ 94	n 110	~ 126
1	1	1	1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

(4526) 3892-52

Table A-3

CHARACTERS USED IN INT PARAMETERS

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		FIGURES		UPPERCASE		LOWERCASE	
Ø Ø Ø Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	SX 2	D2 18	″ 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BL 7	EB 23	′ 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 Ø Ø 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 Ø 1 Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 Ø Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 Ø 1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
1 1 1 Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

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		FIGURES		UPPERCASE		LOWERCASE	
Ø Ø Ø Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	\ 96	p 112
Ø Ø Ø 1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø Ø 1 Ø	SX 2	D2 18	″ 34	2 50	B 66	R 82	b 98	r 114
Ø Ø 1 1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø 1 Ø Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø 1 Ø 1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø 1 1 Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø 1 1 1	BL 7	EB 23	′ 39	7 55	G 71	W 87	g 103	w 119
1 Ø Ø Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1 Ø Ø 1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1 Ø 1 Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1 Ø 1 1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1 1 Ø Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1 1 Ø 1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
1 1 1 Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1 1 1 1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

(4526) 3892-53

Table A-4

CHARACTERS USED IN INT-REPORT AND INTC-REPORT PARAMETERS

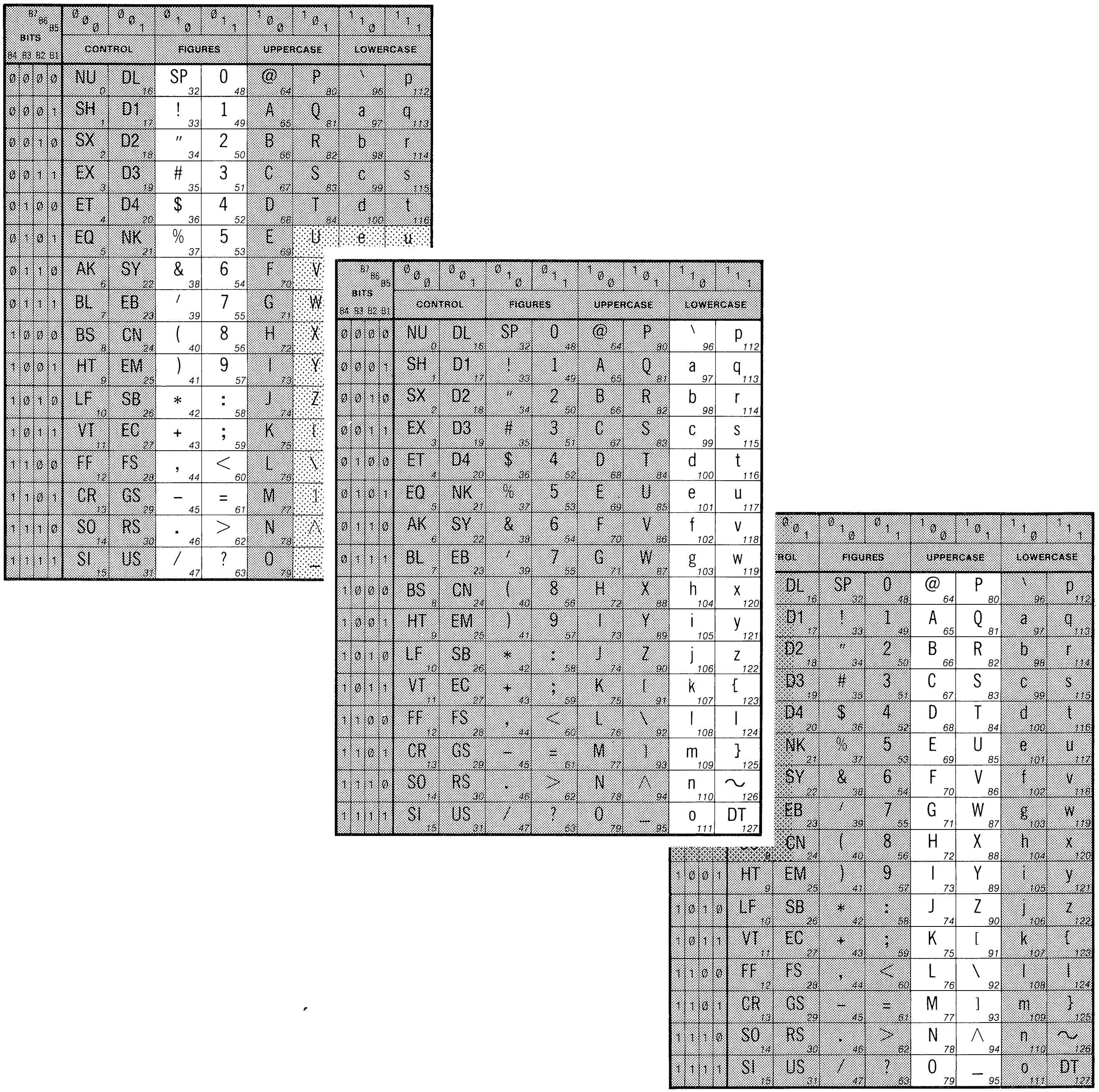
B7 B6 B5				Ø Ø Ø	Ø Ø 1	Ø 1 Ø	Ø 1 1	1 Ø Ø	1 Ø 1	1 1 Ø	1 1 1
BITS				CONTROL		FIGURES		UPPERCASE		LOWERCASE	
B4	B3	B2	B1								
Ø	Ø	Ø	Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
Ø	Ø	Ø	1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	SX 2	D2 18	″ 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BL 7	EB 23	′ 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1	Ø	Ø	1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1	Ø	1	Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1	1	Ø	1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
1	1	1	Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1	1	1	1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

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
BITS				CONTROL		FIGURES		UPPERCASE		LOWERCASE	
B4	B3	B2	B1								
Ø	Ø	Ø	Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
Ø	Ø	Ø	1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	SX 2	D2 18	″ 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BL 7	EB 23	′ 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1	Ø	Ø	1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1	Ø	1	Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1	1	Ø	1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
1	1	1	Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1	1	1	1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

(4526) 3892-54

Table A-5

CHARACTERS USED IN XY PARAMETERS



BITS B7 B6 B5 / B4 B3 B2 B1	ØØØ	ØØ1	Ø1Ø	Ø11	1ØØ	1Ø1	11Ø	111
	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
ØØØØ	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
ØØØ1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
ØØ1Ø	SX 2	D2 18	" 34	2 50	B 66	R 82	b 98	r 114
ØØ11	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø1ØØ	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø1Ø1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø11Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø111	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1ØØØ	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1ØØ1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1Ø1Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1Ø11	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
11ØØ	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
11Ø1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
111Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1111	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

(4526) 3892-55

Table A-6

CHARACTERS USED IN XY-REPORT PARAMETERS

B7 B6 B5 BITS				Ø Ø Ø	Ø Ø 1	Ø 1 Ø	Ø 1 1	1 Ø Ø	1 Ø 1	1 1 Ø	1 1 1
B4	B3	B2	B1	CONTROL		FIGURES		UPPERCASE		LOWERCASE	
Ø	Ø	Ø	Ø	NU 0	DL 16	SP 32	0 48	@ 64	P 80	` 96	p 112
Ø	Ø	Ø	1	SH 1	D1 17	! 33	1 49	A 65	Q 81	a 97	q 113
Ø	Ø	1	Ø	SX 2	D2 18	″ 34	2 50	B 66	R 82	b 98	r 114
Ø	Ø	1	1	EX 3	D3 19	# 35	3 51	C 67	S 83	c 99	s 115
Ø	1	Ø	Ø	ET 4	D4 20	$ 36	4 52	D 68	T 84	d 100	t 116
Ø	1	Ø	1	EQ 5	NK 21	% 37	5 53	E 69	U 85	e 101	u 117
Ø	1	1	Ø	AK 6	SY 22	& 38	6 54	F 70	V 86	f 102	v 118
Ø	1	1	1	BL 7	EB 23	' 39	7 55	G 71	W 87	g 103	w 119
1	Ø	Ø	Ø	BS 8	CN 24	(40	8 56	H 72	X 88	h 104	x 120
1	Ø	Ø	1	HT 9	EM 25	) 41	9 57	I 73	Y 89	i 105	y 121
1	Ø	1	Ø	LF 10	SB 26	* 42	: 58	J 74	Z 90	j 106	z 122
1	Ø	1	1	VT 11	EC 27	+ 43	; 59	K 75	[91	k 107	{ 123
1	1	Ø	Ø	FF 12	FS 28	, 44	< 60	L 76	\ 92	l 108	| 124
1	1	Ø	1	CR 13	GS 29	– 45	= 61	M 77	] 93	m 109	} 125
1	1	1	Ø	SO 14	RS 30	. 46	> 62	N 78	∧ 94	n 110	~ 126
1	1	1	1	SI 15	US 31	/ 47	? 63	O 79	_ 95	o 111	DT 127

(4526) 3892-56

Appendix B

INT PARAMETERS

This appendix describes how to manually construct *int* parameters. Note that the sign information for an *int* representation is contained in the final character, and therfore, the final character is treated differently.

The *4110/4120 Series Host Programmers Manual*, Appendix B, contains a list of already packed *int* parameters from –1049 to 1049 in Appendix B. If you want an *int* parameter in that range quickly, see that manual and appendix.

The algorithm is:

1. Check if the integer value is present in the Tables B-1 to B-6. If it is, the *int* characters are next to it, and you are done. If not, go to Step 2.
2. Let NUMBER be the absolute value of your number.
3. Until NUMBER is less than 16, repeat these steps:
   a. Find the largest integer that is less than NUMBER in the Tables B-1 to B-6.
   b. List the *int* representation of this "largest integer" (directly below any previous representation if this is not the first iteration).
   c. Subtract the integer in the table from NUMBER and put the result in NUMBER.
4. If the original number is positive, take the last character for your *int* from the left half of the Table B-1. If the original number is negative, use the right half of Table B-1. Put this *int* into the list of *int* representations.
5. Add (symbolically) the list of *int* representations using the two following definitions:
   - @ is equivalent to 0 in non-right-most positions.
   - **0** is equivalent to 0 in the right-most position.

See the following Examples to clarify the use of the algorithm.

EXAMPLES

Positive Number. Suppose you want to convert the number 20491 to *int* format.

1. Does 20491 appear in the the tables? No
2. NUMBER = 20491
3. NUMBER – Number In Table *INT* Representation

```
   20491  –  20480          T@0
   =  11                      ;
                          ----------
                            T@;
```

Negative Integer. Suppose you want to convert the number –133120 to *int* format.

1. Does –133120 appear in the the tables? No
2. NUMBER = 133120
3. NUMBER – Number In Table *INT* Representation

```
   133120  –  131072        B@@0
   = 2048  –  2048            B@0
      = 0                       SP
                          ----------
                            BB@SP
```

Table B-1

ONE CHARACTER

Integer Value	*Int* Representation	Integer Value	*Int* Representation
0	**0**	-0	SP
1	**1**	–1	**!**
2	**2**	–2	”
3	**3**	–3	#
4	**4**	–4	**$**
5	**5**	–5	**%**
6	**6**	–6	**&**
7	**7**	–7	’
8	**8**	–8	**(**
9	**9**	–9	**)**
10	**:**	–10	*
11	**;**	–11	+
12	<	–12	**,**
13	=	–13	**–**
14	>	–14	**.**
15	**?**	–15	/

Table B-2

TWO CHARACTER

Integer Value	*Int* Representation	Integer Value	*Int* Representation
16	**A0**	528	**a0**
32	**B0**	544	**b0**
48	**C0**	560	**c0**
64	**D0**	576	**d0**
80	**E0**	592	**e0**
96	**F0**	608	**f0**
112	**G0**	624	**g0**
128	**H0**	640	**h0**
144	**I0**	656	**i0**
160	**J0**	672	**j0**
176	**K0**	688	**k0**
192	**L0**	704	**l0**
208	**M0**	720	**m0**
224	**N0**	736	**n0**
240	**O0**	752	**o0**
256	**P0**	768	**p0**
272	**Q0**	784	**q0**
288	**R0**	800	**r0**
304	**S0**	816	**s0**
320	**T0**	832	**t0**
336	**U0**	848	**u0**
352	**V0**	864	**v0**
368	**W0**	880	**w0**
384	**X0**	896	**x0**
400	**Y0**	912	**y0**
416	**Z0**	928	**z0**
432	**[0**	944	**{0**
448	**\0**	960	**|0**
464	**]0**	976	**}0**
480	**∧0**	992	**~0**
496	**_0**	1008	**DL0**
512	**‘0**		

Table B-3

THREE CHARACTER

Integer Value	*Int* Representation	Integer Value	*Int* Representation
1024	A@0	33792	a@0
2048	B@0	34816	b@0
3072	C@0	35840	c@0
4096	D@0	36864	d@0
5120	E@0	37888	e@0
6144	F@0	38912	f@0
7168	G@0	39936	g@0
8192	H@0	40960	h@0
9216	I@0	41984	i@0
10240	J@0	43008	j@0
11264	K@0	44032	k@0
12288	L@0	45056	l@0
13312	M@0	46080	m@0
14336	N@0	47104	n@0
15360	O@0	48128	o@0
16384	P@0	49152	p@0
17408	Q@0	50176	q@0
18432	R@0	51200	r@0
19456	S@0	52224	s@0
20480	T@0	53248	t@0
21504	U@0	54272	u@0
22528	V@0	55296	v@0
23552	W@0	56320	w@0
24576	X@0	57344	x@0
25600	Y@0	58368	y@0
26624	Z@0	59392	z@0
27648	[@0	60416	{@0
28672	\@0	61440	|0
29696	]@0	62464	}@0
30720	∧@0	63488	~@0
31744	_@0	64512	DL@0
32768	‘@0		

Table B-4

FOUR CHARACTER

Integer Value	*Int* Representation	Integer Value	*Int* Representation
65536	A@@0	2162688	a@@0
131072	B@@0	2228224	b@@0
196608	C@@0	2293760	c@@0
262144	D@@0	2359296	d@@0
327680	E@@0	2424832	e@@0
393216	F@@0	2490368	f@@0
458752	G@@0	2555904	g@@0
524288	H@@0	2621440	h@@0
589824	I@@0	2686976	i@@0
655360	J@@0	2752512	j@@0
720896	K@@0	2818048	k@@0
786432	L@@0	2883584	l@@0
851968	M@@0	2949120	m@@0
917504	N@@0	3014656	n@@0
983040	O@@0	3080192	o@@0
1048576	P@@0	3145728	p@@0
1114112	Q@@0	3211264	q@@0
1179648	R@@0	3276800	r@@0
1245184	S@@0	3342336	s@@0
1310720	T@@0	3407872	t@@0
1376256	U@@0	3473408	u@@0
1441792	V@@0	3538944	v@@0
1507328	W@@0	3604480	w@@0
1572864	X@@0	3670016	x@@0
1638400	Y@@0	3735552	y@@0
1703936	Z@@0	3801088	z@@0
1769472	[@@0	3866624	{@@0
1835008	\@@0	3932160	|@@0
1900544	]@@0	3997696	}@@0
1966080	∧@@0	4063232	~@@0
2031616	_@@0	4128768	DL@@0
2097152	‘@@0		

Table B-5

FIVE CHARACTER

Integer Value	*Int* Representation	Integer Value	*Int* Representation
4194304	**A**@@@**0**	138412032	**a**@@@**0**
8388608	**B**@@@**0**	142606336	**b**@@@**0**
12582912	**C**@@@**0**	146800640	**c**@@@**0**
16777216	**D**@@@**0**	150994944	**d**@@@**0**
20971520	**E**@@@**0**	155189248	**e**@@@**0**
25165824	**F**@@@**0**	159383552	**f**@@@**0**
29360128	**G**@@@**0**	163577856	**g**@@@**0**
33554432	**H**@@@**0**	167772160	**h**@@@**0**
37748736	**I**@@@**0**	171966464	**i**@@@**0**
41943040	**J**@@@**0**	176160768	**j**@@@**0**
46137344	**K**@@@**0**	180355072	**k**@@@**0**
50331648	**L**@@@**0**	184549376	**l**@@@**0**
54525952	**M**@@@**0**	188743680	**m**@@@**0**
58720256	**N**@@@**0**	192937984	**n**@@@**0**
62914560	**O**@@@**0**	197132288	**o**@@@**0**
67108864	**P**@@@**0**	201326592	**p**@@@**0**
71303168	**Q**@@@**0**	205520896	**q**@@@**0**
75497472	**R**@@@**0**	209715200	**r**@@@**0**
79691776	**S**@@@**0**	213909504	**s**@@@**0**
83886080	**T**@@@**0**	218103808	**t**@@@**0**
88080384	**U**@@@**0**	222298112	**u**@@@**0**
92274688	**V**@@@**0**	226492416	**v**@@@**0**
96468992	**W**@@@**0**	230686720	**w**@@@**0**
100663296	**X**@@@**0**	234881024	**x**@@@**0**
104857600	**Y**@@@**0**	239075328	**y**@@@**0**
109051904	**Z**@@@**0**	243269632	**z**@@@**0**
113246208	**[**@@@**0**	247463936	{@@@**0**
117440512	\\@@@**0**	251658240	|@@@**0**
121634816	**]**@@@**0**	255852544	}@@@**0**
125829120	∧@@@**0**	260046848	~@@@**0**
130023424	_@@@**0**	264241152	ᴰL@@@**0**
134217728	‘@@@**0**		

Table B-6

SIX CHARACTER

Integer Value	*Int* Representation
268435456	**A**@@@@**0**
536870912	**B**@@@@**0**
805306368	**C**@@@@**0**
1073741824	**D**@@@@**0**
1342177280	**E**@@@@**0**
1610612736	**F**@@@@**0**
1879048192	**G**@@@@**0**
2147483648	**H**@@@@**0**
2415919104	**I**@@@@**0**
2684354560	**J**@@@@**0**
2952790016	**K**@@@@**0**
3221225472	**L**@@@@**0**
3489660928	**M**@@@@**0**
3758096384	**N**@@@@**0**
4026531840	**O**@@@@**0**

Appendix C

ERROR CODES

INTRODUCTION

Each error condition which a 4110/4120 Series terminal can detect has an error code and a severity level.

When the terminal detects an error condition, it stores the error code and severity level in a limited-size queue for later retrieval by a REPORT-ERRORS commmand from the host.

If the error's severity level is greater than or equal to the current error threshold, then the terminal displays a message for the operator. When the terminal is powered up or reset, its error threshold is set to 2, so that the only errors displayed are those with a severity level of 2 or more. The error threshold can be changed with the SET-ERROR-THRESHOLD command from the host or the ERRORLEVEL setup command from the keyboard.

NOTE

If memory is insufficient to issue an error message, the terminal rings its bell. Press the cancel key.

Since the terminal has two command sets, TEK commands and ANSI commands, the error codes for the two sets are listed separately. TEK error codes begin with two alphanumeric (usually alpha) characters, while ANSI error codes begin with either a bracket ([) and a character, or a character and a space. Other than this difference, errors from the two command sets are treated the same.

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 4128/4129 commands are sent to a 4125, 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 word, "Terminal system error . . . ".
- Level 4. Level four errors are the internal errors. For these errors, the message starts with the words, "Terminal Internal, DS:, or SG: Error ??09 . . . ".

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* (operation code) for the command which causes the error. For example, error IA11 is associated with the SET-PICK-APERTURE 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 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." An error in the second parameter, for example, is indicated by the digit **2**. Errors in parameters 10 and above are indicated by the digit **9**.

- The fourth character in an error code is also a digit. The most frequently used digits here are 0, 1, 2, 3 and 9:

  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.)

  9 : A hardware error prevents or halts execution. (For example, the door is open on a disk drive when you try to copy from it.)

For example, consider the "SO10" error code. Here, "SO" means the BEGIN-SEGMENT command, which has the syntax ^{E}c**SO** *int*. The "1" refers to the first (and only) parameter of that command, which is the segment number. The "0" indicates an "existence problem;" the segment referred to already exists.

Commands Not Installed in the Terminal

Op codes beginning with letters from I to Z may occur in present or future 4110/4120 Series terminals. When the terminal receives one of these escape-sequence commands and does not recognize that command, it detects an error 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 01 installed, and the host sends it the following character sequence:

^{E}cOD0^{E}cKA1

Since Option 01 is not installed, the terminal does not recognize the SET-DUPLEX-MODE: 0 command, ^{E}c**OD0**. On receiving the ^{E}c**OD** op code, it 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, it resumes processing of the characters received, so that it correctly interprets and executes the ENABLE-DIALOG-AREA: 1 command, ^{E}c**KA1**.

If your host program sends commands that may not be installed in all 4110/4120 Series terminals, then these commands should be followed by other commands which are recognized by all terminals in the series.

HARDWARE ERRORS

Disk Hardware Errors

If the terminal has any of the disk options (Options 42, 43, 45, 46, or 47), it may issue disk hardware errors. The errors have this format:

```
>>Terminal Detects Error xxn9
>> Device yy: Hardware Error zzz
```

In the first line, *xx* refers to the command (with the two-letter error code) and *n* refers to the number of the parameter that caused the error. In the second line of the hardware error, *yy:* is the device name and *zzz* refers to the error number listed in Table C-1.

Table C-1

DISK HARDWARE ERRORS

Device Description	Error No.	Explanation
8" Floppy Drive (Devices F0: or F1:)	2	No seek complete.
	3	Write fault.
	4	Drive not ready.
	6	No track 0.
	7	Multiple drives selected.
	10	Insufficient capacity.
	13	Controller chip error.
	16	ID CRC error.
	17	Data CRC error.
	18	ID address mark not found.
	19	Data address mark not found.
	20	Record not found.
	21	Seek error.
	22	DMA timeout error.
	23	Write-protected.
	25	Bad block found.
	26	Interleave error.
	28	Unable to determine a disk format.
	32	Invalid command from disk driver.
	34	Insufficient internal data buffer.
	35	Controller chip system error.

Table C-1 (cont)

DISK HARDWARE ERRORS

Device Description	Error No.	Explanation
5-1/4" Floppy (4925 or 4926 Option 25)	2	No seek complete.
	3	Write fault.
	4	Drive not ready.
	5	Drive not selected.
	6	No track 0.
	7	Illegal or multiple drives selected.
	10	Insufficient capacity.
	16	ID CRC error.
	17	Data CRC error.
	18	ID address mark not found.
	19	Data address mark not found.
	20	Record not found.
	21	Seek error.
	22	DMA timeout error.
	23	Write-protected.
	25	Bad block found.
	32	Invalid command from disk driver.
	34	System error.
	35	Power fail.
	48	RAM diagnostic failure.
	49	Program memory checksum error.

Table C-1 (cont)

DISK HARDWARE ERRORS

Device Description	Error No.	Explanation
10-Megabyte Hard Disk (4926 or Option 46/47)	1 or 129	No index detected from disk drive.
	2 or 130	No seek complete from disk drive.
	3 or 131	Write fault from disk drive.
	4 or 132	Drive not ready after it was selected.
	6 or 133	Track zero not found.
	16 or 144	ID field read error.
	17 or 145	Uncorrectable data error.
	18 or 146	Address mark not found.
	20 or 148	Target sector not found.
	21 or 149	Seek error.
	24 or 152	Correctable data error.
	25 or 153	Bad track flag detected.
	26 or 154	Format error.
	28 or 156	Illegal (direct) access to an alternate track.
	29 or 157	Alternate track to be formatted is either a bad track or already an alternate.
	30 or 158	Target alternate track is not formatted as an alternate.
	31 or 159	Alternate track to be formatted is the same track as the bad track itself.
	32 or 160	Invalid command from disk driver.
	33 or 161	Illegal disk address.
	48 or 176	RAM diagnostic failure.
	49 or 177	Program memory checksum error.
	50 or 178	ECC diagnostic failure.

Disk System Context Errors

For some file system errors, a supplemental error message may be displayed. This supplemental message describes the type of error which has occurred. Table C-2 explains these context error messages:

Table C-2

DISK SYSTEM CONTEXT ERROR

Error	Explanation
Disk Full	There is no more room to write data on the specified disk.
Drive Not Ready	There is no disk in the specified drive, or the door is open.
Write Protected	The disk to which you are trying to write can only be read at this time.
Invalid Device Specifier	The device you specified is not valid for this command.
File Not Found	The terminal cannot find the specified file on the specified device.
Directory Full	The disk already contains the maximum number of files permitted.
File Full	There is no more room to write data in the specified file.
File Currently Being Written	Trying to access a file that is currently being written to.
Invalid Media Format	The disk in the specified device is formatted with an unsupported format.
Device Busy	The specified device is currently active and therefore cannot be accessed.
Invalid File Specifier	You specified a device and filename when only a filename is valid.
File Busy	Trying to protect or rename a file that is currently being written to or read.
File Already Exists	You are trying to rename a file to a name that already exists.

3PPI Hardware Errors

For the Three Port Peripheral Interface option (Option 10), the terminal reports hardware errors in this format:

```
>>Terminal Detects Error xxn9
>> Port Hardware Error Type yy, Parameter n.
```

In the first line, *xx* refers to the command (with the two-letter error code). The *n* in both lines refers to the number of the parameter that caused the error. *Yy* refers to the error number listed in Table C-3.

Table C-3

3PPI HARDWARE ERRORS

Error No.	Explanation
33	3PPI: circular buffer overrun.
34	3PPI:8250 data overrun.
35	3PPI: Errors 33 and 34.
36	3PPI: 8250 parity error.
37	3PPI: Errors 33 and 36.
38	3PPI: Errors 34 and 36.
39	3PPI: Errors 33, 34, and 36.
40	3PPI: 8250 framing error.
41	3PPI: Errors 33 and 40.
42	3PPI: Errors 34 and 40.
43	3PPI: Errors 33, 34, and 40.
44	3PPI: Errors 36 and 40.
45	3PPI: Errors 33, 36, and 40.
46	3PPI: Errors 34, 36, and 40.
47	3PPI: Errors 33, 34, 36, and 40.

DMA Transfer Errors

DM: Device Errors. When DM: is specified as a device, there are error messages that may be reported if certain errors are generated (instead of the standard "Terminal detects . . . "). These error messages are:

"DMA Option 3A failed to power up"	There is not enough memory available to satisfy the requirements for DM: use.
"Invalid DMA Parameter"	The parameter portion of the DM:parameter device is invalid.

Also, for some file system errors ("type 9" errors such as JC09, JC39, JD39, etc.), a supplemental error message is displayed when an error is generated by a transfer involving the DMA. This supplemental message describes the type of error which has occurred. Table C-4 lists these errors:

Table C-4

DMA TRANSFER ERRORS

Error	Explanation
Host not ready for DMA transfer	The host computer is not ready to start a DMA transfer (as indicated by the READY line of the DR11B board).
Host aborted DMA transfer	The host computer has sent an ABORT code to the terminal (via the FNCT lines).
DMA transfer failed (Option 3A)	The host computer tried to transfer more data in a block than was specified in the last SET-DMA-BLOCK-SIZE command OR the host tried to initiate a block transfer before the end of the last terminal-to-host DMA block transfer.
DMA transfer failed (Option 3C)	Illegal length acknowledged, or frame too big, or retry count exceeded, or block size mismatch.
Host termination signal invalid	The signals sent by the host at the end of a DMA block transfer are invalid.

Pseudo Devices. When the DMA pseudo devices are specified as devices, there are error messages that may be reported if certain errors are generated (instead of the standard "Terminal detects . . . "). These error messages are:

"DMA Option 3A Pseudo Devices failed to power up"	There is not enough memory available to satisfy the requirements for DMA pseudo device use.
"Invalid Pseudo Device Parameter"	The parameter portion of the pseudo device specifier is invalid.
"Data Format Error"	There is an error in the data received from the host computer by a DS: or SG: pseudo device.

Internal Errors

For internal errors, the terminal reports in this format:

```
>>Terminal internal, DS:, or SG: Error ??09
>>Diagnostic Code AAAA,BBBB, . . .,GHII,
    JJKK,LLLL,MMMM
```

The Diagnositc Code is a series of ten 4-bit hexadecimal numbers and are level 4 errors. Each of the ten numbers represent:

AAAA	— Error Type (see Table C-5)
BBBB and CCCC	— Micro-processor memory address of the op-code that caused the error
DDDD and EEEE	— Micro-processor memory address of the previous op-code that executed without error
FFFF	— Internal word used for panel filling

GHII and JJKK	— Time and date microcode was created
	G = month
	H = year
	I = hour
	J = unused
	K = day of month

LLLL	— The word at the address specified by BBBB and CCCC
MMMM	— The word at the address specified by DDDD and EEEE

Table C-5

INTERNAL ERROR TYPES

Error Number	Explanation
0001	The microcode call stack (5 deep) has overflowed. Only Diagnostic Codes GHII and JJKK are valid with this error number.
0002	Illegal op-code
0003	Illegal operand in op-code
0004	Stack underflow
0005	Internal stack overflow
0006	Interal stack imbalance
0007	PUSHBC used illegally with BPANEL and EPANEL (i.e. BPANEL, PUSHBC, BPANEL, PUSHBC causes error; BPANEL, PUSHBC, BPANEL, . . ., EPANEL,POPBC, . . . ,EPANEL is OK).
0008	Not enough memory available to fill panel
000A	Transform overflow
000B	Transform cannot be inverted (the concatenated transform is too close to singular to be inverted).
000C	Picture processor RAM exhausted
0042	Divide fault in 3D perspective transform
0043	Too many lightsources defined
0044	Error with microcode overlaying scheme
0045	Tiling processor is busy
0046	Tried to open a facet with a previous facet already open

TEK ERROR CODES

%! SELECT-CODE = ^{E}c%! int

%!11 (Level 2): Invalid *command-set.*

I0 (For several GIN commands.)

I002 (Level 2): Insufficient memory availabe for GIN functions. (Only detected at power-up or during a RESET.)

I003 (level 2): A 3D Segment is being used as the cursor table GIN device.

I011 (Level 2): Invalid *device-function.* (See the ENABLE-GIN command for a table of *device-function* codes.)

I1 ENABLE-4953-TABLET-GIN = ^{E}c! char

I100 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

IA SET-PICK-APERTURE = ^{E}cIA int

IA11 (Level 2): Invalid *aperture-width.*

IC SET-GIN-CURSOR = ^{E}cIC int int

IC13 (Level 2): GIN 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.*

IC23 (Level 2): Context error. A 3D segment cannot be used as a GIN cursor for a pick or stroke function.

ID DISABLE-GIN = ^{E}cID int

I011 (Level 2): Invalid *device-function.* (See the description for a table of *device-function* codes.)

IE ENABLE-GIN = ^{E}cIE int int

I011 (Level 2): Invalid *GIN-code.* (See the description for a table of *device-function* codes.)

IE00 (Level 2): The cursor segment for the specified *device-function* does not exist. (It has been deleted since the SET-GIN-CURSOR command which assigned it to that *device-function.*)

IE03 (Level 2): Command is invalid at this time. (The segment being used as the cursor for the specified *device-function* is a segment which is currently being defined; or, in the 4128 and 4129, a 3D cursor is specified for a pick or stroke function.)

IE10 (Level 2): The specified GIN device is not installed in the terminal.

IE13 (Level 2): The specified device is already enabled, or the protocol assigned to the peripheral port does not support GIN, or, on the 4128 and 4129, a locator function is enabled for GIN with a 3D cursor and either inking or rubberbanding is specified.

IE21 (Level 2): Invalid *GIN-count.*

IF SET-GIN-STROKE-FILTERING = ^{E}cIF int int int

IF10 (Level 2): Stroke filtering not valid for specified *device-function.*

IF21 (Level 2): Invalid *distance-filter.*

IF31 (Level 2): Invalid *time-filter.*

IG SET-GIN-GRIDDING = ^{E}cIG 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-grid-spacing*

IG31 (Level 2): Invalid *y-grid-spacing*

IH SET-TABLET-HEADER-CHARACTERS = ^{E}cIH int

IH00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

IH11 (Level 2): Invalid *character-set-selector.*

II SET-GIN-INKING = $^{E}{}_{C}$II int int

II11 (Level 2): Inking does not apply to the specified *device-function* code. (Inking is not allowed for the pick function.)

II13 (Level 2): Context error. Inking is not allowed with locator functions which have a 3D segment as the GIN cursor for currently enabled GIN.

II21 (Level 2): Invalid *inking-mode* (must be 0, 1, or 2; 0 or 1 if firmware is Version 3 or earlier).

IK SET-GIN-REPORT-FORMAT = $^{E}{}_{C}$IK int

IK11 (Level 2): Invalid *GIN-report-format.*

IL SET-REPORT-MAX-LINE-LENGTH = $^{E}{}_{C}$IL int

IL11 (Level 2): Invalid *max-line-length.*

IM SET-REPORT-EOM-FREQUENCY = $^{E}{}_{C}$IM int

IM11 (Level 2): Invalid *EOM-frequency* setting.

IN SET-TABLET-SIZE = $^{E}{}_{C}$IN int

IN00 (Level 0): Unrecognized command. (Must be a 4111 or Option 10 must be installed.)

IN11 (Level 2): Invalid *tablet-size.*

IO SET-GIN-LOGICAL-FUNCTION = $^{E}{}_{C}$IO int-array real-array xy + -array char-array

IO00 (Level 0): Unrecognized command (the terminal is not a 4128 or 4129).

I011 (Level 2): Invalid first element in the *functions* array.

IO11 (Level 2): Invalid *functions* array.

IO12 (Level 3): Out of memory while parsing parameter.

IO21 (Level 2): Invalid *reals* array.

IO22 (Level 3): Out of memory while parsing parameter.

IO31 (Level 2): Invalid *points* array.

IO32 (Level 3): Out of memory while parsing parameter.

IO41 (Level 2): Invalid *chars* array.

IO42 (Level 3): Out of memory while parsing parameter.

IP REPORT-GIN-POINT = $^{E}{}_{C}$IP int

I011 (Level 2): Invalid *code.*

IE03 (Level 2): Context error. Command is invalid at this time. (The segment being used as the cursor for the specified *device-function* is a segment which is currently being defined; or, in the 4128 and 4129, a 3D segment is specified as a GIN cursor for a pick or stroke function.)

IE10 (Level 2): The specified GIN device is not installed in the terminal.

IE13 (Level 2): The protocol assigned to the peripheral port does not support GIN.

IP13 (Level 2): The *code* names a GIN device which has already been enabled for a different GIN function.

IR SET-GIN-RUBBERBANDING = $^{E}{}_{C}$IR 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.)

IR13 (Level 2): Context error. Rubberbanding is not allowed with locator functions which have a 3D segment as the GIN cursor for currently enabled GIN.

IR21 (Level 2): Invalid *rubberbanding-mode* (must be 0, 1, or 2; 0 or 1 if firmware is Version 3 or earlier).

IS SET-REPORT-SIG-CHARS = $^{E}{}_{C}$IS int int int

IS21 (Level 2): Invalid *sig-char.*

IS31 (Level 2): Invalid *term-sig-char.*

IT SET-TABLET-STATUS-STRAP = $^{E}{}_{C}$IT int

IT00 (Level 0): Unrecognized command. (Not a 4111 or Option 10, Version 6 or later is not installed.)

IT11 (Level 2): Invalid *strap-setting.*

IU SET-GIN-RATES = $^{E}{}_{C}$IUnt

IU10 (Level 2): Device code or sub-device code not present (0,0 for thumbwheels, 8,0 for mouse, 9,0 for joystick).

IU11 (Level 2): Invalid *rate-table* entries (array length must be 0 to 14 for thumbwheels, 1 to 10 for mouse, and 1 to 4 for joystick. Array elements 1 and 2 must be –32768 to 32767, array elements 3 and after must be 0 to 4095).

IU12 (Level 3): Out of memory while parsing array.

IV SET-GIN-AREA = $^{E}{}_{C}$IV int int xy xy

IV03 (Level 3): Out of memory while processing command.

IV11 (Level 2): Invalid *device-function* (see ENABLE-GIN).

IV21 (Level 2): Invalid *window-specifier.*

IV31 (Level 2): Invalid *first-corner.*

IV41 (Level 2): Invalid *second-corner* or zero width or height GIN area.

IW SET-GIN-WINDOW = $^{E}{}_{C}$IW xy xy

IW00 (Level 0): Unrecognized command; firmware is Version 3 or earlier.

IW11 (Level 2): Invalid *first-corner.*

IW21 (Level 2): Invalid *second-corner.*

IX SET-GIN-DISPLAY-START-POINT = $^{E}{}_{C}$IX int xy

IX03 (Level 0): Context error. The terminal must be in coordinate mode 0 or 1.

IX11 (Level 2): Invalid *device-function* parameter.

IX21 (Level 2): Invalid *start-point.*

J0 & J1 Disk System Errors on Power-Up

J002 (Level 3): Memory error detected by standard firmware.

J008 (Level 3): Bus timeset error.

J102 (Level 3): Memory error detected by optional peripheral firmware.

J109 (Level 3): Hardware initialization error in controller board.

JB ACTIVATE-LPOS = $^{E}{}_{C}$JB

JB03 (Level 2): A filename was specified when LPOS was resident in terminal memory, or a filename was not specified, but LPOS was active.

JB10 (Level 2): The *boot-file* was not found.

JB11 (Level 2): Illegal filename.

JB12 (Level 3): Out of memory while performing command.

JB13 (Level 2): Invalid device specifier, invalid medium format or the file is currently being written.

JB19 (Level 2): The drive is not ready or a hardware error occurred while reading from the drive.

JC COPY = $^{E}{}_{C}$JC device string device

JC00 (Level 2): Existence problem in op-code parameter (devices DS: and SG: only).

JC01 (Level 2): Data format error (devices DS:, SG:, and HC: only).

JC02 (Level 3): Out of memory while attempting DMA transfer (device DM: only.)

JC03 (Level 2): Attempt to copy an entire disk volume onto itself. Or HC: Image will not fit.

JC10 (Level 2): Specified *source* does not exist, or cannot be found.

JC11 (Level 2): Invalid *source* specifier.

JC12 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JC13 (Level 2): Context error (not an input device, or device is busy).

JC19 (Level 2): Disk hardware error or drive not ready on the *source* device, or error in DMA block transfer.

JC21 (Level 2): Invalid *separator* (must be empty string or **TO**, in Setup mode, must be **TO**).

JC22 (Level 3): Out of memory while parsing the parameter.

JC30 (Level 2): Specified *destination* does not exist.

JC31 (Level 2): Invalid *destination* specifier.

JC32 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JC33 (Level 2): Context error. (Invalid destination device, device is busy, or existing disk file is protected.)

JC39 (Level 2): Disk hardware error on the *destination* device, or error in DMA block transfer.

JD DIRECTORY = $^{E}_{C}$JD device string device

JD00 (Level 0): Unrecognized command. (Disk drive option is not installed.)

JD10 (Level 2): The specified *source* device does not exist or cannot be found.

JD11 (Level 2): Invalid *source* specifier.

JD12 (Level 3): Out of memory while parsing the parameter, 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* (must be empty string or **TO**).

JD22 (Level 3): Out of memory while parsing the parameter.

JD30 (Level 2): The specified *destination* does not exist.

JD31 (Level 2): Invalid *destination* specifier.

JD32 (Level 3): Out of memory while parsing the parameter, or while executing the command.

JD33 (Level 2): Context error. (The device specified is not a valid destination device, or is write-protected.)

JD39 (Level 2): Hardware error for the destination device.

JF FORMAT-VOLUME = $^{E}_{C}$JF device

JF00 (Level 0): Unrecognized command. (Disk drive option not installed.)

JF10 (Level 2): *Device* is not installed.

JF11 (Level 2): Invalid *device* specifier.

JF12 (Level 3): Out of memory while parsing the parameter.

JF13 (Level 2): The device specified is not a disk drive, is write-protected, is busy, detects a verify error, detects a bit map error, or is not mounted.

JF19 (Level 2): Hardware error at the specified disk drive. (Format error, drive not ready, or write-protect switch or notch error.)

JH SET-DMA-BLOCK-SIZE = $^{E}_{C}$JH int

JH00 (Level 0): Unrecognized command; Not a 4111 or Option 3A or 3C is not installed.

JH03 (Level 2): SET-DMA-BLOCK-SIZE command received after DMA failed to power up.

JH11 (Level 2): Invalid *DMA-block-size*.

JJ DISMOUNT = $^{E}_{C}$JJ device

JJ00 (Level 0): Unrecognized command (there are no disk options installed).

JJ10 (Level 2): *Device* is not installed.

JJ11 (Level 2): Invalid *device* parameter.

JJ13 (Level 2): Context error (device is busy).

JK DELETE-FILE = ${}^{E}c$JK device

JK00 (Level 0): Unrecognized command. (Disk drive option is not installed.)

JK10 (Level 2): The specified file or disk drive does not exist.

JK11 (Level 2): Invalid *file-name.*

JK12 (Level 3): Out of memory while parsing parameter.

JK13 (Level 2): The specified device is not a disk drive, is write-protected, is busy, or detects a bit map error.

JK19 (Level 2): Disk hardware error.

JL LOAD = ${}^{E}c$JL device

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 *source* specifier.

JL12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JL13 (Level 2): Context error. (Not a valid source device, device is busy, or command detects a disk format error.)

JL19 (Level 2): Device hardware error.

JP PROTECT-FILE = ${}^{E}c$JP device int

JP00 (Level 0): 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): Out of memory while parsing parameter.

JP13 (Level 2): The specified device is not a disk drive.

JP19 (Level 2): Disk hardware error. (I/O error, drive not ready, or hardware write-protect error.)

JP21 (Level 2): Invalid *write-protect-mode.*

JQ EPORT-DEVICE-STATUS = ${}^{E}c$JQ device

JQ10 (Level 2): Device is not installed.

JQ11 (Level 2): Invalid device specifier.

JQ12 (Level 3): Out of memory while parsing parameter.

JR RENAME-FILE = ${}^{E}c$JR device string device

JR00 (Level 0): Unrecognized command. (Disk drive option is not installed.)

JR10 (Level 2): The specified file does not exist or the device is not installed.

JR11 (Level 2): Invalid *old-filename* specifier.

JR12 (Level 3): Out of memory while parsing parameter.

JR13 (Level 2): The specified file is write-protected or *old-filename* is not a disk device.

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): Out of memory while parsing parameter.

JR30 (Level 2): The device specified is not installed or is different from the device specified in parameter 1, or the *new-filename* is already present.

JR31 (Level 2): Invalid *new-filename* specifier.

JR32 (Level 3): Out of memory while parsing parameter.

JR33 (Level 2): *New-filename* is not a disk device.

JR39 (Level 2): Disk hardware error. (I/O or drive not ready.)

JS SPOOL = $^{E}_{C}$JS device string device

JS02 (Level 3): Out of memory while attempting DMA transfer (4111 or Option 3A only).

JS03 (Level 2): Command context error: a spooling operation is already in progress or the image will not fit (4111 or Option 19, version 5 or later).

JS10 (Level 2): Specified *source* does not exist.

JS11 (Level 2): Invalid *source* specifier.

JS12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JS13 (Level 2): Parameter 1 context error. (Not a valid source device, or device is busy.)

JS19 (Level 2): Device hardware error (disk hardware error, drive not ready, or DMA block transfer error).

JS21 (Level 2): Invalid *separator* (must be empty string or **TO**. Setup mode must be **TO**).

JS22 (Level 3): Out of memory while parsing parameter.

JS30 (Level 2): Specified *destination* does not exist.

JS31 (Level 2): Invalid *destination* device specifier.

JS32 (Level 3): Out of memory while parsing parameter, or while executing the command.

JS33 (Level 2): Parameter 3 context error. (Not a valid destination, device is busy, or existing file is write protected or open.)

JS39 (Level 2): Device hardware error (disk hardware error, drive not ready, or DMA block transfer error).

JU SET-USER-NUMBER = $^{E}_{C}$JU int

JU00 (Level 0): Unrecognized command.

JU11 (Level 2): Invalid *user-number.*

JV SAVE = $^{E}_{C}$JV string int string device

JV02 (Level 3): Out of memory while attempting DMA transfer (4111 or Option 3A or 3C only).

JV10 (Level 2): Invalid *item-type.*

JV12 (Level 3): Out of memory while parsing parameter, or while executing the command.

JV20 (Level 2): The specified macro or segment does not exist, or segment is being defined.

JV21 (Level 2): Invalid *item-number.*

JV31 (Level 2): Invalid *separator* (must be empty string or **TO**).

JV32 (Level 3): Out of memory while parsing the parameter.

JV40 (Level 2): The specified *destination* is not installed.

JV41 (Level 2): Invalid *destination* specifier.

JV42 (Level 2): Out of memory while parsing the parameter, or while executing the command.

JV43 (Level 2): Not a valid *destination* device, device is busy, or existing disk file is write protected or open.

JV49 (Level 2): Device hardware error. (I/O error, drive not ready, hardware write-protect error, or DMA block transfer error.)

K0 Keyboard System Errors

K002 (Level 3): Out of memory while initializing the keyboard system.

KA ENABLE-DIALOG-AREA = $^{E}_{C}$KA int

KA02 (Level 3): Insufficient memory to enable a new dialog area.

KA11 (Level 2): Invalid *dialog-area.*

KD DEFINE-MACRO = $^{E}_{C}$KD int int-array

KD11 (Level 2): Invalid *macro-number.*

KD21 (Level 2): Invalid *macro-contents.*

KD22 (Level 3): Insufficient memory to define macro.

KE SET-ECHO = $^{E}{}_{c}$KE int

KE11 (Level 2): Invalid *echo-mode.*

KF LFCR = $^{E}{}_{c}$KF int

KF11 (Level 2): Invalid *LFCR-mode.*

KH HARDCOPY = $^{E}{}_{c}$KH int

KH01 (Level 2): Copier fault condition; operator assistance required (4111 or Option 19 only).

KH03 (Level 2): Image too large for copier (4111 or Option 19 only).

KH11 (Level 2): Invalid *hard-copy-code.*

KH13 (Level 2): Wrong copier connected (4111 or Option 19, version 5 or later).

KH19 (Level 2): Color hardcopy device not ready (4111 or Option 19 only).

KI IGNORE-DELETES = $^{E}{}_{c}$KI int

KI11 (Level 2): Invalid *ignore-deletes-mode.*

KK DELETE-DIALOG-AREA = $^{E}{}_{c}$KK int

KK00 (Level 0): Command not recognized by terminal.

KK10 (Level 1): Specified *dialog-number* does not exist.

KK11 (Level 2): Invalid *dialog-number.*

KL LOCK-KEYBOARD = $^{E}{}_{c}$KL int

KL11 (Level 2): Invalid *locking-mode.*

KN RENEW-VIEW = $^{E}{}_{c}$KN int

KN02 (Level 3): Out of memory while attempting to renew a view. (This error can also occur as a result of pressing the PAGE key.)

KN10 (Level 2): The view specified does not exist.

KN11 (Level 2): Invalid *view-number.*

KP SET-PAGE-FULL-ACTION = $^{E}{}_{c}$KP int

KP11 (Level 2): Invalid *page-full-action.*

KR CRLF = $^{E}{}_{c}$KR int

KR11 (Level 2): Invalid *CRLF-mode.*

KS SET-SNOOPY-MODE = $^{E}{}_{c}$KS int

KS11 (Level 2): Invalid *snoopy-mode.*

KT SET-ERROR-THRESHOLD = $^{E}{}_{c}$KT int

KT11 (Level 2): Invalid *error-threshold-level.*

KX EXPAND-MACRO = $^{E}{}_{c}$KX int

KX11 (Level 2): Invalid *macro-number.*

KY SET-KEY-EXECUTE-CHAR = $^{E}{}_{c}$KY int

KY11 (Level 2): Invalid *key-execute-char.*

KZ SET-EDIT-CHARS = $^{E}{}_{c}$KZ int int int

KZ11 (Level 2): Invalid *char-delete* character.

KZ21 (Level 2): Invalid *line-delete* character.

KZ31 (Level 2): Invalid *take-literally* character.

LB SET-DIALOG-AREA-BUFFER-SIZE = $^{E}{}_{c}$LB int

LB03 (Level 2): Context error. Buffer size cannot be set when *dialog-area* –2 is enabled.

LB11 (Level 2): Invalid *number-of-lines.*

LC SET-DIALOG-AREA-CHARS = $^{E}{}_{c}$LC int

LC03 (Level 2): Context error. *Number-of-chars* cannot be set when *dialog-area* –2 is enabled.

LC11 (Level 2): Invalid *number-of-chars.*

LE END-PANEL = $^{E}{}_{c}$LE

LE02 (Level 3): Out of memory while performing END-PANEL command.

LE03 (Level 1): No panel is currently being defined.

LE09 (Level 3): Internal stack overflow.

LF MOVE = $^{E}{}_{c}$LF xy

LF11 (Level 2): *Position* out of valid range.

LG DRAW = $^{E}{}_{c}$LG xy

LG11 (Level 2): *Position* out of valid range.

LH DRAW-MARKER = $^{E}_{C}$LH xy

LH11 (Level 2): *Marker-position* out of valid range.

LI SET-DIALOG-AREA-INDEX = $^{E}_{C}$LI int int int

LI11 (Level 2): Invalid *character-index.*

LI21 (Level 2): Invalid *character-background-index.*

LI31 (Level 2): Invalid *color-pair.*

LJ SET-DIALOG-AREA-ALTERNATE-INDEX = $^{E}_{C}$LJ int

LJ11 (Level 2): Invalid *color-index.*

LK INCLUDE-COPY-OF-SEGMENT = $^{E}_{C}$LK int

LK02 (Level 3): Out of memory while performing INCLUDE-COPY-OF-SEGMENT.

LK09 (Level 3): Internal stack overflow.

LK10 (Level 2): Segment does not exist.

LK11 (Level 2): Invalid *segment-number.*

LK13 (Level 2): The segment specified is currently being defined, or, on a 4128 and 4129 terminal, the dimensionality of the segment does not match the current coordinate mode of the terminal.

SO00 (Level 4): INCLUDE-COPY-OF-SEGMENT commands are nested too deep.

LL SET-DIALOG-AREA-LINES = $^{E}_{C}$LL int

LL03 (Level 2): Context error. *Number-of-lines* cannot be set when *dialog-area* –2 is enabled.

LL11 (Level 2): Invalid *number-of-lines.*

LM SET-DIALOG-AREA-WRITING-MODE = $^{E}_{C}$LM int

LM11 (Level 2): Invalid *writing-mode.*

LP BEGIN-PANEL-BOUNDARY = $^{E}_{C}$LP xy int

LP02 (Level 3): Out of memory while defining panel.

LP03 (Level 2): Alphatext and graphtext are not allowed within a *panel-definition* or a subpanel cannot be added to and existing panel being edited, or coordinate mode is not 0 or 1.

LP11 (Level 2): *First-point* out of valid range.

LP21 (Level 2): *Draw-boundary* out of valid range.

LS SET-DIALOG-AREA-SURFACE = $^{E}_{C}$LS int

LS11 (Level 2): Invalid *surface-number.*

LT GRAPHIC-TEXT = $^{E}_{C}$LT string

LT03 (Level 2): Command is invalid at this time. (Graphtext is not allowed within a panel-definition or stroke precision graphtext is not allowed in coordinate mode 2.)

LT09 (Level 3): Internal stack overflow.

LT11 (Level 2): Invalid array count (must be in range from 0 to 65535).

LT12 (Level 3): Out of memory while parsing the parameter.

LV SET-DIALOG-AREA-VISIBILITY = $^{E}_{C}$LV 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 *visibility.*

LX SET-DIALOG-AREA-POSITION = $^{E}_{C}$LX xy

LX11 (Level 2): Invalid *lower-left-corner.*

LZ CLEAR-DIALOG-SCROLL = $^{E}_{C}$LZ

LZ03 (Level 2): Context error. The dialog area scroll buffer cannot be cleared when *dialog-area* –2 is enabled.

MA SET-GRAPHTEXT-SLANT = $^{E}_{C}$MA real

MA11 (Level 2): Invalid *slant-angle.*

MB SET-BACKGROUND-INDICES = $^{E}_{C}$MB int int

MB11 (Level 2): Invalid *text-background-index.*

MB21 (Level 2): Invalid *dash-gap-index.*

MC SET-GRAPHTEXT-SIZE = $^{E}_{C}$MC int int int

MC11 (Level 2): Invalid *cell-width.*

MC21 (Level 2): Invalid *cell-height.*

MC31 (Level 2): Invalid *character-spacing.*

MD BEGIN-FILL-PATTERN = ^{E}cMD int int int int

MD02 (Level 3): Not enough memory available for fill pattern.

MD03 (Level 2): Another fill pattern is currently being defined.

MD11 (Level 2): Invalid *fill-pattern-number.*

MD21 (Level 2): Invalid *pattern-width.*

MD31 (Level 2): Invalid *pattern-height.*

MD41 (Level 2): Invalid *bits-per-pixel.*

MF SET-GRAPHTEXT-FONT = ^{E}cMF int

MF10 (Level 2): Font does not exist.

MF11 (Level 2): Invalid *font-number.*

MG SET-GRAPHICS-AREA-WRITING-MODE = ^{E}cMG int

MG11 (Level 2): Invalid *writing-mode.* (Must be 0 or 1; in Setup mode, must be OVERSTRIKE or REPLACE.)

MH SELECT-TRAVERSAL-DIRECTION = ^{E}cMH int

MH00 (Level 2): Unrecognized command. (The terminal is not a 4128 or 4129.)

MH11 (Level 2): Invalid *traversal-direction.*

MI SET-PICK-ID = ^{E}cMI int

MI03 (Level 2): Command is invalid at this time. (No segment is currently being defined.)

MI11 (Level 2): Invalid *pick-ID-number.*

ML SET-LINE-INDEX = ^{E}cML int

ML11 (Level 2): Invalid *line-index.*

MM SET-MARKER-TYPE = ^{E}cMM int

MM11 (Level 2): Invalid *marker number.*

MO SELECT-TRANSLUCENCY-PATTERN = ^{E}cMO int

MO00 (Level 2): Unrecognized command. (The terminal is not a 4128 or 4129).

MO11 (Level 2): Invalid *translucency-pattern.*

MP SELECT-FILL-PATTERN = ^{E}cMP int

MP10 (Level 2): Specified fill pattern does not exist (has not been defined).

MP11 (Level 2): Invalid *fill-pattern-number.*

MQ SET-GRAPHTEXT-PRECISION = ^{E}cMQ int

MQ11 (Level 2): Invalid *precision.*

MR SET-GRAPHTEXT-ROTATION = ^{E}cMR real

MR11 (Level 2): Invalid *angle-in-degrees.*

MS SET-PANEL-FILLING-MODE = ^{E}cMS int int int

MS11 (Level 2): Invalid *overstrike/replace* parameter.

MS21 (Level 2): Invalid *cover-boundary* parameter.

MS31 (Level 2): Invalid *pattern-keying-mode.*

MT SET-TEXT-INDEX = ^{E}cMT int

MT11 (Level 2): Invalid *text-index.*

MV SET-LINE-STYLE = ^{E}cMV int

MV11 (Level 2): Invalid *line-style.*

NB SET-STOP-BITS = ^{E}cNB int

NB11 (Level 2): Invalid *number-of-stop-bits.*

NC SET-EOM-CHARS = ^{E}cNC int int

NC11 (Level 2): Invalid *EOM-char-1.*

NC21 (Level 2): Invalid *EOM-char-2.*

ND SET-TRANSMIT-DELAY = ^{E}cND int

ND11 (Level 2): Invalid *transmit-delay.*

NE SET-EOF-STRING = ^{E}cNE 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): Out of memory while parsing the parameter.

NF SET-FLAGGING-MODE = ^{E}cNF int

NF11 (Level 2): Invalid *flagging-mode.*

NK SET-BREAK-TIME = ${}^{E}{}_{c}$NK int

NK11 (Level 2): Invalid *break-time-in-milliseconds.*

NL SET-TRANSMIT-RATE-LIMIT = ${}^{E}{}_{c}$NL int

NL11 (Level 2): Invalid *rate-limit.*

NM PROMPT-MODE = ${}^{E}{}_{c}$NM int

NM11 (Level 2): Invalid *prompt-mode.*

NP SET-PARITY = ${}^{E}{}_{c}$NP int

NP11 (Level 2): Invalid *parity-mode.*

NQ SET-QUEUE-SIZE = ${}^{E}{}_{c}$NQ int

NQ01 (Level 3): Cannot free enough currently allocated queue memory.

NQ02 (Level 3): Out of memory while performing SET-QUEUE-SIZE command.

NQ11 (Level 2): Invalid *queue-size.*

NR SET-BAUD-RATES = ${}^{E}{}_{c}$NR int int

NR11 (Level 2): Invalid transmit (terminal-to-host) data rate.

NR21 (Level 2): Invalid receive (host-to-terminal) data rate.

NS SET-PROMPT-STRING = ${}^{E}{}_{c}$NS int-array

NS11 (Level 2): Invalid *prompt-string.*

NS12 (Level 3): Out of memory while parsing the parameter.

NT SET-EOL-STRING = ${}^{E}{}_{c}$NT int-array

NT11 (Level 2): Invalid *EOL-string* (contents: from 0 to 2 *int*s; each in the range from 0 to 127).

NT12 (Level 3): Out of memory while parsing the parameter.

NU SET-BYPASS-CANCEL-CHAR = ${}^{E}{}_{c}$NU int

NU11 (Level 2): Invalid numeric equivalent of bypass-cancel character.

OB ARM-FOR-BLOCK-MODE = ${}^{E}{}_{c}$OB int

OB00 (Level 0): Unrecognized command (Not a 4111 or Option 1 is not installed).

OB03 (Level 2): The communications queue size is smaller than the specified input block size.

OB11 (Level 2): Invalid *block-mode-arming* parameter.

OC SET-BLOCK-CONTINUE-CHARS = ${}^{E}{}_{c}$OC int int

OC00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OC03 (Level 2): Command is invalid at this time. (Terminal must not be armed for block mode.)

OC11 (Level 2): Invalid *transmit-continue-char.*

OC13 (Level 2): *Transmit-continue-char* must be different from *block-master-char* and *block-end-char.*

OC21 (Level 2): Invalid *receive-continue-char.*

OC23 (Level 2): *Receive-continue-char* must be different from *block-master-char* and *block-end-char.*

OD SET-DUPLEX-MODE = ${}^{E}{}_{c}$OD int

OD00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OD01 (Level 2): Invalid *duplex-mode.*

OE SET-BLOCK-END-CHARS = ${}^{E}{}_{c}$OE int int

OE00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OE03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OE11 (Level 2): Invalid *transmit-end-char.*

OE13 (Level 2): *Transmit-end-char* must be different from *block-master-char* and *block-continue-char.*

OE21 (Level 2): Invalid *receive-end-char.*

OE23 (Level 2): *Receive-end-char* must be different from *block-master-char* and *block-continue-char.*

OH SET-BLOCK-HEADERS = $^{E}{}_{c}$OH int-array int-array

OH00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OH02 (Level 3): Out of memory while performing command.

OH03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OH11 (Level 2): Invalid *char* (must be 0 to 127) or array count (must be in range 1 to 10) in *transmit-header*.

OH12 (Level 3): Out of memory while parsing the parameter.

OH21 (Level 2): Invalid *char* (must be 0 to 127) or array count (must be in range 1 to 10) in *receive-header*.

OH22 (Level 3): Out of memory while parsing the parameter.

OL SET-BLOCK-LINE-LENGTH = $^{E}{}_{c}$OL int

OL00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 not installed.)

OL03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OL11 (Level 2): Invalid *maximum-line-length*.

OM SET-BLOCK-MASTER-CHARS = $^{E}{}_{c}$OM int int

OM00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OM03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OM11 (Level 2): Invalid *transmit-master-char*.

OM13 (Level 2): *Transmit-master-char* must be different from *block-end-char* and *block-continue-char*.

OM21 (Level 2): Invalid *receive-master-char*.

OM23 (Level 2): *Receive-master-char* must be different from *block-end-char* and *block-continue-char*.

ON SET-BLOCK-NON-XMT-CHARS = $^{E}{}_{c}$ON int-array int-array

ON00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

ON03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

ON11 (Level 2): Invalid character code or array count in *transmit-chars* array. (The array count must range from 0 to 20, and the character codes must range from 0 to 127.)

ON12 (Level 3): Out of memory while parsing the parameter.

ON21 (Level 2): Invalid character code or array count in *receive-chars* array. (The array count must range from 0 to 20, and the character code must range from 0 to 127.)

ON22 (Level 3): Out of memory while parsing the parameter.

OP SET-BLOCK-PACKING = $^{E}{}_{c}$OP int int int int

OP00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OP03 (Level 2): Command invalid at this time. (Terminal must not be armed for block mode.)

OP11 (Level 2): Invalid *transmit-unpacked-bits*.

OP21 (Level 2): Invalid *transmit-packed-bits*.

OP31 (Level 2): Invalid *receive-unpacked-bits*.

OP41 (Level 2): Invalid *receive-packed-bits*.

OS SET-BLOCK-LENGTH = $^{E}{}_{c}$OS int int

OS00 (Level 2): Unrecognized command. (Not a 4111 or 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 *transmit-block-length*.

OS21 (Level 2): Invalid *receive-block-length*.

OT SET-BLOCK-TIMEOUT = $^{E}{}_{c}$OT int

OT00 (Level 2): Unrecognized command. (Not a 4111 or Option 1 is not installed.)

OT11 (Level 2): Invalid *number-of-seconds*.

PA PORT-ASSIGN = $^{E}{}_{C}$PA device string int

PA00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PA11 (Level 2): Invalid *port.*

PA12 (Level 3): Out of memory while parsing the parameter.

PA13 (Level 2): Port is in use.

PA21 (Level 2): Invalid *protocol-identifier.* (Must be **PPORT**, **4643**, **4662**, **4662/MP**,**4662/NT**, **4663**, **4663/NB**,**4663/NT**, **4957**, **4957/S**, **4958**,**4958/S**, **KANA**, or **DMA**).

PA22 (Level 3): Out of memory while parsing the parameter.

PB SET-PORT-STOP-BITS = $^{E}{}_{C}$PB device int int

PB00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PB11 (Level 2): Invalid *port.*

PB12 (Level 3): Out of memory while parsing the parameter.

PB13 (Level 2): Port is busy.

PB21 (Level 2): Invalid *number-of-stop-bits.*

PB31 (Level 2): Invalid *number-of-data-bits.*

PC PORT-COPY = $^{E}{}_{C}$PC device string device

PC00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PC02 (Level 3): Out of memory while processing command.

PC10 (Level 2): *Source* device does not exist.

PC11 (Level 2): Invalid *source.*

PC12 (Level 3): Out of memory while parsing the parameter.

PC13 (Level 2): *Source* is busy, or is a peripheral port that is not assigned the **PPORT** protocol.

PC21 (Level 2): Invalid *separator.*

PC22 (Level 3): Out of memory while parsing the parameter.

PC30 (Level 2): *Destination* device is not installed.

PC31 (Level 2): Invalid *destination.*

PC32 (Level 3): Out of memory while parsing the parameter.

PC33 (Level 2): *Destination* is busy is a peripheral port that is not assigned the **PPORT** protocol.

PE SET-PORT-EOF-STRING = $^{E}{}_{C}$PE device int-array

PE00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PE11 (Level 2): Invalid *port.*

PE12 (Level 3): Out of memory while parsing the parameter.

PE13 (Level 2): Port is busy.

PE21 (Level 2): Invalid *PEOF-string.*

PE22 (Level 3): Out of memory while parsing the parameter.

PF SET-PORT-FLAGGING-MODE = $^{E}{}_{C}$PF device int int int

PF00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PF11 (Level 2): Invalid *port.*

PF12 (Level 3): Out of memory while parsing the parameter.

PF13 (Level 2): Port is busy.

PF21 (Level 2): Invalid *flagging-mode.*

PF31 (Level 2): Invalid *GO-character.*

PF41 (Level 2): Invalid *STOP-character.*

PI MAP-INDEX-TO-PEN = ^{E}cPI device int int

PI00 (Level 0): Unrecognized command. (Not a 4111 or 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 MAP-INDEX-TO-PEN command should be issued immediately after power-up, or immediately after a RESET command.)

PI11 (Level 2): Invalid *port* identifier.

PI12 (Level 3): Out of memory while parsing the parameter.

PI13 (Level 2): Port busy.

PI21 (Level 2): Invalid *index-to-be-mapped.*

PI31 (Level 2): Invalid *pen-ID-number.*

PL PLOT = ^{E}cPL string device

PL00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PL02 (Level 3): Out of memory while attempting DMA transfer (4111 or Option 3A or 3C only).

PL11 (Level 2): Invalid first parameter. (Must be the empty string or **TO**.)

PL12 (Level 3): Out of memory while parsing parameter.

PL20 (Level 2): Destination device not installed.

PL21 (Level 2): Invalid destination.

PL22 (Level 3): Out of memory while parsing parameter.

PL23 (Level 2): Destination device is busy or is a write-protected file.

PL29 (Level 2): Hardware error on destination device.

PM SET-PORT-EOL-STRING = ^{E}cPM device int-array

PM00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PM11 (Level 2): Invalid *port.*

PM12 (Level 3): Out of memory while parsing the parameter.

PM13 (Level 2): Port is busy.

PM22 (Level 3): Out of memory while parsing the parameter.

PP SET-PORT-PARITY = ^{E}cPP device int

PP00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PP11 (Level 2): Invalid *port.*

PP12 (Level 3): Out of memory while parsing the parameter.

PP13 (Level 2): Port is busy.

PP21 (Level 2): Invalid parity-mode.

PQ REPORT-PORT-STATUS = ^{E}cPQ device

PQ00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PQ11 (Level 2): Invalid port identifier.

PQ12 (Level 3): Out of memory while parsing the parameter.

PR SET-PORT-BAUD-RATE = ^{E}cPR device int

PR00 (Level 0): Unrecognized command. (Not a 4111 or Option 10 is not installed.)

PR11 (Level 2): Invalid *port.*

PR12 (Level 3): Out of memory while parsing the parameter.

PR13 (Level 2): Port is busy.

PR21 (Level 2): Invalid *baud-rate.*

QB SET-COLOR-COPIER-DATA-RESOLUTION = ^{E}cQB int

QB00 (Level 0): Unrecognized command (Not a 4111 or Option 19 is not installed).

QB11 (Level 2): Invalid *number-of-bytes (must be 1 or 2).*

QD SELECT-HARDCOPY-INTERFACE = ^{E}cQD int

QD00 (Level 0): Unrecognized command (Not a 4111 or Option 19 is not installed).

QD11 (Level 2): Invalid *interface* parameter.

QF SET-COLORHARDCOPY-FORM-WIDTH = ^{E}cQF int

QF00 (Level 0): Unrecognized command. (Must be 4111 or Option 19 must be installed.)

QF11 (Level 2): Invalid *form-width.*

QL SET-DIALOG-HARDCOPY-ATTRIBUTES = ^{E}cQL int int int

QL00 (Level 0): Unrecognized command (Must be 4111 or Option 19 must be installed).

QL11 (Level 2): Invalid *number-of-pages.*

QL21 (Level 2): Invalid *page-origin.*

QL31 (Level 2): Invalid *page-definition.*

QM SET-COLORHARDCOY-DITHERING = ^{E}cQM int

QM00 (Level 0): Unrecognized command (Must be 4111 or Option 19, version 5 or later, must be installed).

QM11 (Level 2): Invalid *dither-code.*

QN SET-NUMBER-OF-COPIES = ^{E}cQN int

QN00 (Level 0): Unrecognized command (Not a 4111 or Option 19 is not installed).

QN11 (Level 2): Invalid *number-of-copies* parameter.

QO SET-IMAGE-ORIENTATION = ^{E}cQO int

QO00 (Level 0): Unrecognized command (Not a 4111 or Option 19 is not installed).

QO11 (Level 2): Invalid *orientation.*

QQ REPORT-COLORHARDCOPY-STATUS = ^{E}cQQ

QQ00 (Level 0): Unrecognized command (Must be 4111 or Option 19 must be installed).

QR RESERVE-COLOR-COPIER = ^{E}cQR int

QR00 (Level 0): Unrecognized command (Must be 4111 or Option 19, version 5 or later, must be installed).

QR03 (Level 2): The color hardcopy interface is busy.

QR11 (Level 2): Invalid *reserve-code.*

QR19 (Level 2): No color copier is connected.

QU SELECT-COLORHARDCOPY-IMAGE-DENSITY = ^{E}cQU int

QU00 (Level 0): Unrecognized command (Must be 4111 or Option 19, version 5 or later, must be installed).

QU11 (Level 2): Invalid *density-code.*

RA SET-VIEW-ATTRIBUTES = ^{E}cRA int int int

RA10 (Level 2): Surface does not exist (has not been defined with SET-SURFACE-DEFINITIONS command).

RA11 (Level 2): Invalid *surface-number.*

RA21 (Level 2): Invalid *wipe-index.*

RA31 (Level 2): Invalid *border-index.*

RB SET-BACKGROUND-GRAY-LEVEL = ^{E}cRB int

RB11 (Level 2): Invalid *gray-level.*

RC SELECT-VIEW = ^{E}cRC int

RC02 (Level 3): Insufficient memory to create a view.

RC11 (Level 2): Invalid *view-number.*

RD SET-SURFACE-DEFINITIONS = ^{E}cRD int-array

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 *surface-defs* array.

RD12 (Level 3): Out of memory while trying to parse parameter.

RE SET-BORDER-VISIBILITY = ^{E}cRE int

RE11 (Level 2): Invalid *border-visibility-mode* parameter.

RF SET-FIXUP-LEVEL = ^{E}cRF int

RF11 (Level 1): Invalid *fixup-level.*

RG SET-SURFACE-GRAY-LEVELS = ^{E}cRG int int-array

RG10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RG11 (Level 2): Invalid *surface-number.*

RG21 (Level 2): Invalid *indices-and-gray-levels* array. (The array count must be even; the first *int* in each pair must be a color index in the range from 1 to 32767; the second *int* in each pair must be a valid gray-level: a number from 0 to 100 and 1000 to 1100).

RG22 (Level 3): Out of memory while parsing parameter.

RH SET-PIXEL-BEAM-POSITION = ^{E}cRH xy

RH11 (Level 2): Invalid *beam-position.*

RI SET-SURFACE-VISIBILITY = ^{E}cRI int-array

RI10 (Level 2): A surface in*surface-numbers-and-visibilities* does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RI11 (Level 2): Invalid *surface-numbers-and-visibilities* array.

RI12 (Level 3): Out of memory while parsing parameter.

RJ LOCK-VIEWING-KEYS = ^{E}cRJ int

RJ11 (Level 2): Invalid *locking-mode.*

RK DELETE-VIEW = ^{E}cRK int

RK10 (Level 2): The designated view does not exist (has not been defined with the SELECT-VIEW command).

RK11 (Level 2): Invalid *view-number.*

RL RUNLENGTH-WRITE = ^{E}cRL int-array

RL11 (Level 2): Invalid *indices-array.*

RL12 (Level 3): Out of memory while parsing the parameter.

RN SET-SURFACE-PRIORITIES = ^{E}cRN int-array

RN10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RN11 (Level 2): Invalid *priorities* array.

RN12 (Level 3): Out of memory while parsing parameter.

RP RASTER-WRITE = ^{E}cRP int char-array

RP11 (Level 2): Invalid *number-of-pixels.*

RP21 (Level 2): There are too many or too few pixels in the *indices* array, or invalid code(s) are present (range is from ADE 32 to 96).

RP22 (Level 3): Out of memory while parsing the parameter.

RQ SET-VIEW-DISPLAY-CLUSTER = ^{E}cRQ int-array

RQ11 (Level 2): Invalid *view-numbers* array.

RQ12 (Level 3): Out of memory while parsing parameter.

RR RECTANGLE-FILL = ^{E}cRR xy xy int

RR11 (Level 2): Invalid *first-corner* coordinates.

RR21 (Level 2): Invalid *second-corner* coordinates.

RR31 (Level 2): Invalid *fill-index.*

RS SET-PIXEL-VIEWPORT = ^{E}cRS xy xy

RS11 (Level 2): Invalid *first-corner* coordinate.

RS21 (Level 2): Invalid *second-corner* coordinate.

RT SET-PIXEL-WRITING-FACTORS = ^{E}cRT int int int

RT11 (Level 2): Invalid *pixel-width* or width is larger than the pixel viewport width.

RT21 (Level 2): Invalid *pixel-height* or height is larger than the pixel viewport height.

RT31 (Level 2): Invalid *major-axis.*

RU BEGIN-PIXEL-OPERATIONS = ^{E}cRU int int int

RU10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

RU11 (Level 2): Invalid *surface-number.*

RU21 (Level 2): Invalid *ALU-mode.*

RU31 (Level 2): Invalid *bits-per-pixel.*

RV SET-VIEWPORT = ^{E}cRV xy xy

RV01 (Level 2): Invalid viewport size.

RV11 (Level 2): Invalid *first-corner.*

RV21 (Level 2): Invalid *second-corner.*

RW SET-WINDOW = $^{E}{}_{C}$RW xy xy

RW03 (Level 2): Context error (the 4128 or 4129 must be in coordinate mode 0 or 1).

RW11 (Level 2): *First-corner* out of valid range.

RW21 (Level 2): *Second-corner* out of valid range.

RX PIXEL-COPY = $^{E}{}_{C}$RX int xy xy xy

RX10 (Level 2): The specified *destination-surface* does not exist.

RX11 (Level 2): Invalid *destination-surface*.

RX21 (Level 2): Invalid *destination-lower-left-corner*.

RX31 (Level 2): Invalid *first-source-corner*.

RX41 (Level 2): Invalid *second-source-corner*.

SA SET-SEGMENT-CLASS = $^{E}{}_{C}$SA 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*.

SA21 (Level 2): Invalid *removal-array*.

SA22 (Level 3): Out of memory while parsing parameter.

SA31 (Level 2): Invalid *addition-array*.

SA32 (Level 3): Out of memory while parsing parameter.

SB BEGIN-LOWER-SEGMENT = $^{E}{}_{C}$SB

SB00 (Level 2): The indicated segment already exists.

SB01 (Level 2): Invalid value for next lower segment number (current segment ID is 1).

SB02 (Level 3): Out of memory while ending or beginning segment definition.

SB03 (Level 2): Context error; command is invalid at this time. No segment is currently being defined, or a graphtext character is currently being defined.

SC END-SEGMENT = $^{E}{}_{C}$SC

SC02 (Level 3): Out of memory while performing END-SEGMENT command.

SC03 (Level 1): Context error (no segment is currently being defined).

SD SET-SEGMENT-DETECTABILITY = $^{E}{}_{C}$SD 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*.

SD21 (Level 2): Invalid *detectability*.

SE BEGIN-NEW-SEGMENT = $^{E}{}_{C}$SE int

SE02 (Level 3): Not enough memory to begin segment, or out of memory while defining segment.

SE03 (Level 2): Command is invalid at this time (a graphtext character or a panel is currently being defined).

SE10 (Level 2): Segment already exists.

SE11 (Level 2): Invalid *segment-number*.

SF CALL-SEGMENT = $^{E}{}_{C}$SF int xy int

SF02 (Level 3): Not enough memory to make segment reference.

SF03 (Level 2): The specified segment is open or another segment is open for insertion inside a panel.

SF09 (Level 3): Internal stack overflow.

SF11 (Level 2): Invalid *segment-number*.

SF13 (Level 2): The specified segment's dimensionality does not match the current coordinate mode of the terminal.

SF31 (Level 2): Invalid *attribute-flag*.

SO00 (Level 4): Picture processor stack has overflowed (nesting depth up to 24 is valid). Picture processor error 0005 will also be part of the error message.

SG SET-GRAPHTEXT-FONT-GRID = $^{E}_{C}$SG 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*.

SG21 (Level 2): Invalid *grid-width*.

SG31 (Level 2): Invalid *grid-height*.

SH SET-SEGMENT-HIGHLIGHTING = $^{E}_{C}$SH 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*.

SH13 (Level 2): A 3D *highlighting* value is used for a 2D segment.

SH21 (Level 2): Invalid *highlighting*.

SI SET-SEGMENT-IMAGE-TRANSFORM = $^{E}_{C}$SI int real real real xy

SI02 (Level 3): Out of memory while transforming segment.

SI03 (Level 2): Command is invalid at this time. (The specified segment is currently being defined.)

SI03 (Level 3): Internal transformation calculation cannot be performed.

SI09 (Level 3): Internal stack overflow.

SI10 (Level 2): Segment does not exist.

SI11 (Level 2): Invalid *segment-number*.

SI13 (Level 2): Segment is not a 2D segment.

SI21 (Level 2): Invalid *x-scaling-factor*.

SI31 (Level 2): Invalid *y-scaling-factor*.

SI41 (Level 2): Invalid *rotation-angle*.

SI51 (Level 2): *Position* out of valid range.

SJ SET-SEGMENT-SCALE-ROTATION = $^{E}_{C}$SJ int real real real

SJ02 (Level 3): Out of memory while transforming segment.

SJ03 (Level 2): The specified segment is open or coordinate mode is 2.

SJ03 (Level 3): Internal transformation calculation cannot be performed.

SJ09 (Level 3): Internal stack overflow.

SJ10 (Level 2): Segment does not exist.

SJ11 (Level 2): Invalid *segment number*.

SJ13 (Level 2): The segment is not a 2D segment.

SJ21 (Level 2): Invalid *x-scaling-factor*.

SJ31 (Level 2): Invalid *y-scaling-factor*.

SJ41 (Level 2): Invalid *rotation-angle*.

SK DELETE-SEGMENT = $^{E}_{C}$SK int

SK02 (Level 3): Out of memory while attempting to delete a segment.

SK10 (Level 1): Segment does not exist.

SK11 (Level 2): Invalid segment number.

SK13 (Level 2): Segment specified is an active GIN-cursor.

SL SET-CURRENT-MATCHING-CLASS = $^{E}_{C}$SL int-array int-array

SL11 (Level 2): Invalid *inclusion-set* array.

SL12 (Level 3): Out of memory while parsing the parameter.

SL21 (Level 2): Invalid *exclusion-set*.

SL22 (Level 3): Out of memory while parsing the parameter.

SM SET-SEGMENT-WRITING-MODE = $^{E}_{C}$SM 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*.

SM13 (Level 2): *Writing-mode* cannot be 3 or 4 for 3D segments.

SM21 (Level 2): Invalid *writing-mode*.

SN BEGIN-HIGHER-SEGMENT = $^{E}_{C}$SN

SN00 (Level 2): The indicated segment already exists.

SN01 (Level 2): Invalid value for next higher segment number (current segment ID is 32767).

SN02 (Level 3): Out of memory while ending or beginning segment definition.

SN03 (Level 2): Context error; command is invalid at this time. No segment is currently being defined, or a graphtext character is currently being defined.

SO BEGIN-SEGMENT = $^{E}_{C}$SO int

SO00[1] (Level 4): Picture Processor error. Segment transform error; segment scaled out of window, or window decreased beyond scaled segment.

SO02 (Level 3): Not enough memory to begin segment, or 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*.

SP SET-PIVOT-POINT = $^{E}_{C}$SP xy

SP11 (Level 2): *Pivot-point* out of valid range.

SQ REPORT-SEGMENT-STATUS = $^{E}_{C}$SQ int char-array

SQ10 (Level 2): Segment does not exist.

SQ11 (Level 2): Invalid *segment-number*.

SQ21 (Level 2): Invalid *status-codes*.

SQ22 (Level 3): Out of memory while parsing the parameter.

SR RENAME-SEGMENT = $^{E}_{C}$SR int int

SR02 (Level 3): Out of memory while renaming a segment.

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 *old-segment-number*.

SR20 (Level 2): A segment with that segment number already exists.

SR21 (Level 2): Invalid *new-segment-number*.

[1] **Although error SO00 starts with the op code SO, it is a picture processor error related to segment. It is *not* caused by the BEGIN-SEGMENT ($^{E}_{C}$SO) command. The error has this form:**

```
>>Terminal Issues Message SO00
>>PP Error 000A 0002 2682 0000 9310 0001 0107 0608
```

The first hexadecimal number in the series indicates the type of picture processor error.

The most common numbers in this first position are 0004, 000A, or 000B, which indicate that the error is caused by the interaction of the SET-WINDOW ($^{E}_{C}$RW) and SET-SEGMENT-IMAGE-TRANSFORM ($^{E}_{C}$SI) commands. These two commands can produce the SO00 error, for example, in these situations:

- **The user specifies a scale factor of 0 to the SET-SEGMENT-IMAGE-TRANSFORM command ($^{E}_{C}$SI).**
- **When a window is decreased after a segment has been transformed, error SO00 may apear when the view is renewed. (For example, suppose the window is defined to be plus or minus 2 billion, and a transform is applied to scale all segments by 10000 in the X and Y directions. If the window is then decreased to the default window — 0,0 by 4095, 3276 — error SO00 results as soon as the view is renewed.**
- **An image transform is applied to an invisible segment that is then made visible, or to a segment that includes a transformed segment. Because of the way that the transforms are concatenated internally, error SO00 is generated.**

When 0004, 000A, or 000B picture processor error occurs, set the fixup level to 2 for XOR mode segments and to 4 for Set mode segments. Then change the window or specify different scale factors in the segment transform command. Finally, renew the view to erase unwanted graphics and to display the segments correctly.

Although the 0004, 000A, and 000B error types are the most common, other numbers may appear in the first position of the SO00 error message, For example, 0005 indicates a stack overflow, which may occur when the INCLUDE-COPY-OF-SEGMENT command is used to append segments repeatedly. 000C indicates a panel filling error (specifically, the 000C error occurs when there is not enough scratchpad RAM to handle a large panel-filling pattern). Other errors may occur, however, as a result of a bad op code sequence from the DMA interface, If you cannnot determine the cause of the error, contact the Tektronix field office closest to you for more information.

Error SO00 is the *only* Level 4 error, and it cannot be suppressed by setting the error threshold to 4.

SS SET-SEGMENT-DISPLAY-PRIORITY = $^{E}_{C}$SS int int

SS02 (Level 3): Out of memory while processing the command.

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.*

SS21 (Level 2): Invalid *priority-number.*

ST BEGIN-GRAPHTEXT-CHARACTER = $^{E}_{C}$ST int int

ST01 (Level 2): MOVE or DRAW is out of range (X = 0 to 4095, Y = 0 to 4095).

ST02 (Level 3): Out of memory while defining graphtext character.

ST03 (Level 2): Command is invalid at this time. (A graphtext character is currently being defined, or the pivot-point coordinate is out of range X = 0 to 4095, Y = 0 to 4095, or coordinate mode is 2 (4128 and 4129 only)).

ST10 (Level 2): The specified font does not exist (no SET-GRAPHTEXT-FONT-GRID command has been issued for *font*).

ST11 (Level 2): Invalid *font* number.

ST20 (Level 2): The *character* specified has already been defined in this *font.*

ST21 (Level 2): Invalid *character* number.

SU END-GRAPHTEXT-CHARACTER = $^{E}_{C}$SU

SU03 (Level 1): This command is invalid at this time. (No graphtext character is being defined.)

SV SET-SEGMENT-VISIBILITY = $^{E}_{C}$SV int

SV02 (Level 3): Out of memory.

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.*

SV21 (Level 2): Invalid *visibility.*

SW SET-SEGMENT-MATRIX-TRANSFORM = $^{E}_{C}$SW int int real-array xy + -array

SW00 (Level 0): Unrecognized command (the terminal is not a 4128 or 4129).

SW02 (Level 3): Out of memory while transforming segment.

SW03 (Level 2): Command is invalid at this time (coordinate mode 2 is not in effect or specified segment is currently being defined).

SW03 (Level 3): Internal transformation calculation cannot be performed.

SW09 (Level 3): Internal stack overflow.

SW10 (Level 2): Segment does not exist.

SW11 (Level 2): Invalid *segment* number.

SW13 (Level 2): Segment is not a 3D segment.

SW21 (Level 2): Invalid *xform-mode.*

SW31 (Level 2): Invalid *xform-matrix.*

SW41 (Level 2): Invalid *points* array.

SX SET-SEGMENT-POSITION = $^{E}_{C}$SX int xy

SX02 (Level 3): Out of memory.

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.*

SX13 (Level 2): The specified segment's dimensionality does not match the current coordinate mode of the terminal.

SX21 (Level 2): *Position* out of valid range.

SZ DELETE-GRAPHTEXT-CHARACTER = $^{E}_{C}$SZ 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.*

SZ20 (Level 1): The character specified does not exist in this font.

SZ21 (Level 2): Invalid *character-number.*

TB SET-BACKGROUND-COLOR = ^{E}cTB int int int

TB11 (Level 2): Invalid first parameter. (If in HLS mode, must range from –32768 to 32767. If in RGB or CMY mode, must range from 0 to 100. If in Machine RGB mode, must range from 0 to 255.)

TB21 (Level 2): Invalid second parameter (HLS, RGB, CMY must range from 0 to 100, Machine RGB must range from 0 to 255).

TB31 (Level 2): Invalid third parameter (HLS, RGB, CMY must range from 0 to 100 and 1000 to 1100, Machine RGB must range from 0 to 255 and 1000 to 1255).

TG SET-SURFACE-COLOR-MAP = ^{E}cTG int int-array

TG10 (Level 2): Surface does not exist (has not been defined with a SET-SURFACE-DEFINITIONS command).

TG11 (Level 2): Invalid *surface-number.*

TG21 (Level 2): Invalid *color-mixtures* array.

TG22 (Level 3): Out of memory while parsing parameter.

TL SET-LIGHTSOURCE = ^{E}cTL int-array int-array xy + -array real-array

TL00 (Level 0): Unrecognized command. (Terminal is not a 4128 or 4129.)

TL02 (Level 3): Insufficient memory to save the light-source parameters.

TL03 (Level 2): Command is invalid in coordinate mode 0 or 1.

TL11 (Level 2): Invalid *source-id.*

TL12 (Level 3): Out of memory while parsing *source-id* array.

TL21 (Level 2): Invalid *views-switch.*

TL22 (Level 3): Out of memory while parsing *views-switch* array.

TL31 (Level 2): Invalid *direction.*

TL32 (Level 3): Out of memory while parsing *direction* array.

TL41 (Level 2): Invalid *intensity* array.

TL42 (Level 3): Out of memory while parsing *intensity* array.

TM SET-COLOR-MODE = ^{E}cTM int int int

TM11 (Level 2): Invalid *color-specifying-mode*

TM21 (Level 2): Invalid *color-overlay-mode.*

TM31 (Level 2): Invalid *gray-mode.*

TO SELECT-OBJECT-SURFACE-COLOR = ^{E}cTO int

TO00 (Level 0): Unrecognized command. (Terminal is not a 4128 or 4129.)

TO10 (Level 2): Object-surface-color range definition does not exist (has not been defined or has been deleted).

TO11 (Level 2): Invalid *object-surface-color.*

TR SET-OBJECT-SURFACE-COLOR-RANGE = ^{E}cTR int int-array real-array

TR00 (Level 0): Unrecognized command. (Terminal is not a 4128 or 4129.)

TR11 (Level 2): Invalid *object-surface-color.*

TR21 (Level 2): Invalid *indices* array.

TR22 (Level 3): Out of memory while parsing *indices* array.

TR31 (Level 2): Invalid *reflectivity-coefficients* array.

TR32 (Level 3): Out of memory while parsing *reflectivity-coefficients* array.

UB SET-DRAW-BOUNDARY-MODE = ^{E}cUB int

UB11 (Level 2): Invalid *draw-boundary-mode.*

UC DRAW-CURVE = ^{E}cUC int xy-array

UC03 (Level 2): A graphtext character is currently being defined or the 4128 or 4129 terminal is in coordinate mode 2.

UC11 (Level 2): Invalid *curve-type.*

UC21 (Level 2): Invalid *list-of-points* array.

UD DELETE-PART-OF-SEGMENT = ^{E}cUD int int int

UD02 (Level 3): Out of memory.

UD03 (Level 2): Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.

UD10 (Level 2): Specified segment does not exist.

UD11 (Level 2): Invalid *segment-number.*

UD13 (Level 2): The specified segment has a dimensionality attribute which does not match the coordinate mode of the terminal.

UD20 (Level 2): Cannot find first *pickIDfirst.*

UD21 (Level 2): Invalid *pickIDfirst.*

UD23 (Level 2): Invalid because of transformations in the included segment.

UD30 (Level 2): Cannot find *pickIDlast.*

UD31 (Level 2): Invalid *pickIDlast.*

UD33 (Level 2): Invalid because of included segment or panel begin/end boundaries.

UE REPLACE-PART-OF-SEGMENT = ^{E}cUE int int int

UE02 (Level 3): Out of memory.

UE03 (Level 2): Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.

UE10 (Level 2): Specified segment does not exist.

UE11 (Level 2): Invalid *segment-number.*

UE13 (Level 2): The dimensionality of the specified segment does not match the current coordinate mode of the terminal.

UE20 (Level 2): Cannot find *pickIDfirst.*

UE21 (Level 2): Invalid *pickIDfirst.*

UE23 (Level 2): Invalid because of transformations in an included segment.

UE30 (Level 2): Cannot find *pickIDlast.*

UE31 (Level 2): Invalid *pickIDlast.*

UE33 (Level 2): Invalid because of included segment or panel

UF DRAW-FACETS = ^{E}cUF int-array xy + -array int-array xy + -array

UF00 (Level 0): Unrecognized command (terminal is not a 4128 or 4129).

UF03 (Level 2): Command is invalid at this time (coordinate mode 2 must be in effect).

UF11 (Level 2): Invalid *type* code in *action* array.

UF12 (Level 3): Out of memory while parsing *action.*

UF21 (Level 2): Invalid *points* array.

UF22 (Level 3): Not enough memory for *points* array.

UF31 (Level 2): Invalid *indices* array.

UF32 (Level 3): Not enough memory for *indices* array.

UF41 (Level 2): Invalid *normals* array.

UF42 (Level 3): Not enough memory for *normals* array.

UG SET-CURVE-SMOOTHNESS = ^{E}cUG real

UG11(Level 2): Invalid value for *smoothness.*

UH SET-SEGMENT-EDIT-MODE = ^{E}cUH int

UH11 (Level 2): Invalid *edit-mode.*

UI INSERT-INTO-SEGMENT = ^{E}cUI int int int

UI02 (Level 3): Out of memory.

UI03 (Level 2): Command is invalid at this time because another segment, a graphtext character, or a panel is being defined.

UI10 (Level 2): Specified segment does not exist.

UI11 (Level 2): Invalid *segment-number.*

UI13 (Level 2): The dimensionality of the specified segment does not match the coordinate mode of the terminal.

UI20 (Level 2): Cannot find *pickID-group.*

UI21 (Level 2): Invalid *pickID-group.*

UI23 (Level 2): Context error because of image transformation of the included segment.

UI31 (Level 2): Invalid *beforeorafter.*

UJ SAVE-PIXELS-TO-MEMORY = $^{E}{}_{C}$UJ int xy xy

UJ02 (Level 3): Not enough main memory available to store the specified region.

UJ11 (Level 2): Invalid *ID-number.*

UJ21 (Level 2): Invalid *first-corner.*

UJ31 (Level 2): Invalid *second-corner.*

UK RESTORE-PIXELS-FROM-MEMORY = $^{E}{}_{C}$UK int int xy

UK10 (Level 2): *ID-number* not previously saved.

UK11 (Level 2): Invalid *ID-number.*

UK21 (Level 2): Invalid *restore-flag.*

UK31 (Level 2): Invalid *lower-left* value.

UR DRAW-RECTANGLE = $^{E}{}_{C}$UR xy-array

UR03 (Level 2): Context error. (A new panel boundary cannot be added to an existing panel during segment editing. Rectangles are not allowed when the terminal is in coordinate mode 2).

UR11 (Level 2): Invalid *opposite-corners* array.

UR12 (Level 3): Out of memory while parsing parameter.

UV SET-VIEWING-TRANSFORM = $^{E}{}_{C}$UV xy + -array int-array xy + -array int int-array

UV03 (Level 2): Invalid coordinate mode (must not be 0 or 1).

UV11 (Level 2): Invalid *uvw-space.*

UV12 (Level 3): Out of memory parsing parameter.

UV21 (Level 2): Invalid *UV-window.*

UV22 (Level 3): Out of memory parsing parameter.

UV31 (Level 2): Invalid *eye-position.*

UV41 (Level 2): Invalid *projection-type.*

UV51 (Level 2): Invalid *distances.*

UV52 (Level 3): Out of memory parsing parameter.

UW SET-OVERVIEW-WINDOW = $^{E}{}_{C}$UW xy xy

UW11 (Level 2): Invalid *first-corner* parameter.

UW21 (Level 2): Invalid *second-corner* parameter.

UX SET-COORDINATE-MODE = $^{E}{}_{C}$UX int

UX11 (Level 2): Invalid *coordinate-mode.*

UX21 (Level 2): Invalid *intc-report-size.*

WA SET-VIEW-DISPLAY-ATTRIBUTES = $^{E}{}_{C}$WA int-array

WA00 (Level 0): Unrecognized command. (The terminal is not a 4128 or 4129.)

WA11 (Level 2): Invalid *attributes* array.

WA12 (Level 3): Out of memory while parsing *attributes* array.

WS SET-SECTIONING-PLANES = $^{E}{}_{C}$WS int-array xy + -array

WS00 (Level 0): Unrecognized command (terminal is not a 4128 or 4129).

WS03 (Level 2): Invalid coordinate mode (must not be 0 or 1).

WS11 (Level 2): Invalid *sectioning.*

WS12 (Level 3): Out of memory while parsing the *sectioning* parameter.

WS21 (Level 2): Invalid *plane-points.*

WS22 (Level 3): Out of memory while parsing the*plane-points* parameter.

ANSI ERROR CODES

%! SELECT-CODE = $^{E}{}_{C}$%!nt

%!00 (Level 0): Unrecognized command; terminal firmware is Version 3.

%!11 (Level 2): Invalid *command-set* (must be 0 or 1.)

[@ INSERT-CHARACTER = *CSI* Pn @

[@11 (Level 2): Invalid *number-of-characters* parameter (range is 0 to 32767).

[A CURSOR-UP = *CSI* Pn A

[A11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

[B CURSOR-DOWN = *CSI* Pn B

[B11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

[C CURSOR-FORWARD = *CSI* Pn C

[C11 (Level 2): Invalid *number-of-columns* parameter (range is 0 to 32767).

[D CURSOR-BACKWARD = *CSI* Pn D

[D11 (Level 2): Invalid *number-of-columns* parameter (range is 0 to 32767).

[f HORIZONTAL-AND-VERTICAL-POSITION = *CSI* Pn Pn f

[f11 (Level 2): Invalid *line-number* parameter (range is 0 to 32767).

[f21 (Level 2): Invalid *column-number* parameter (range is 0 to 32767).

[g TABULATION-CLEAR = *CSI* Ps g

[g11 (Level 2): Invalid *tab-clear-extent* parameter. (must be 0, 1, or 2).

[H CURSOR-POSITION = *CSI* Pn Pn H

[H11 (Level 2): Invalid *line-number* parameter (range is 0 to 32767).

[H21 (Level 2): Invalid *column-number* parameter (range is 0 to 32767)

[h SET-MODE = *CSI* Ps Ps . . . h

[h11 (Level 2): Invalid *mode* value.

[I CURSOR-HORIZONTAL-TAB = *CSI* Pn I

[I11 (Level 2): Invalid value (range is 0 to 32767).

[J ERASE-IN-DISPLAY = *CSI* Ps J

[J11 (Level 2): Invalid *erase-extent* parameter (must be 0, 1, or 2).

[K ERASE-IN-LINE = *CSI* PS K

[K11 (Level 2): Invalid *erase-extent* parameter (must be 0, 1, or 2).

[L INSERT-LINE = *CSI* Pn L

[L11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767)

[l RESET-MODE = *CSI* Ps Ps . . . l

[l11 (Level 0): Invalid *mode* value.

[M DELETE-LINE = *CSI* Pn M

[M11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

[m SELECT-GRAPHIC-RENDITION = *CSI* Ps Ps . . . m

[m11 (Level 2): Invalid *rendition* parameter (must be 0, 1, 4, 5, or 7).

[n DEVICE-STATUS-REPORT = *CSI* Ps n

[n11 (Level 2): Invalid *device* parameter (the only valid device specifier is **6**).

[P DELETE-CHARACTER = *CSI* Pn P

[P11 (Level 2): Invalid *number-of-characters* parameter (range is 0 to 32767).

[S SCROLL-DOWN = *CSI* Pn S

[S11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

[T SCROLL-DOWN = *CSI* Pn T

[T11 (Level 2): Invalid *number-of-lines* parameter (range is 0 to 32767).

[X ERASE-CHARACTER = *CSI* Pn X

[X11 (Level 2): Invalid *number-of-characters* parameter (range is 0 to 32767).

[Z CURSOR-BACKWARD-TAB = *CSI* Pn Z

[Z11 (Level 2): Invalid *number-of-tabs* parameter (range is 0 to 32767).

Appendix D

MACRO NUMBER REFERENCE TABLES

Table D-1 is arranged by keys; keys are listed roughly in the order you see them if you "read" the keyboard (left to right, top to bottom). Note that for the ASCII keys, the macro number for each standard key is the same as the ASCII decimal equivalent of that character.

Table D-2 lists the difference between the ASCII characters generated for control and control-shift key combinations by the parallel interface keyboard and by the serial interface keyboard. The serial interface keyboards were introduced with the 4120 Series Computer Display Terminals.

Table D-1

MACROS BY KEYBOARD LAYOUT

Key Group	Key Label	Un-Shift	Caps Lock	Shift	Ctrl	Ctrl-Shift
ASCII and Function Keys	{ [	91	91	123	27	27
	! 1	49	49	33	49	33
	@ 2	50	50	64	50	0
	# 3	51	51	35	51	35
	$ 4	52	52	36	52	36
	% 5	53	53	37	53	37
	^ 6	54	54	94	54	30
	& 7	55	55	38	55	38
	* 8	56	56	42	56	42
	(9	57	57	40	57	40
	) 0	48	48	41	48	41

Table D-1 (cont)

MACROS BY KEYBOARD LAYOUT

Key Group	Key Label	Un-Shift	Caps Lock	Shift	Ctrl	Ctrl-Shift
ASCII and Function Keys Cont.	__ -	45	45	95	45	31
	+ =	61	61	43	61	43
	}]	93	93	125	29	29
	RUB OUT	127	127	−34	−35	−36
	ESC	27	27	−37	−38	−39
	~		124	124	126	28
	Q	113	81	81	17	17
	W	119	87	87	23	23
	E	101	69	69	5	5
	R	114	82	82	18	18
	T	116	84	84	20	20
	Y	121	89	89	25	25
	U	117	85	85	21	21
	I	105	73	73	9	9
	O	111	79	79	15	15
	P	112	80	80	16	16
	' \	92	92	96	28	0
	BACK SPACE	8	8	−40	−41	−41
	LINE FEED	10	10	−43	−44	−45
	TAB	9	9	−46	−47	−46
	A	97	65	65	1	1
	S	115	83	83	19	19
	D	100	68	68	4	4
	F	102	70	70	6	6

Table D-1 (cont)

MACROS BY KEYBOARD LAYOUT

Key Group	Key Label	Un Shift	Caps Lock	Shift	Ctrl	Ctrl-Shift
ASCII and Function Keys Cont.	G	103	71	71	7	7
	H	104	72	72	8	8
	J	106	74	74	10	10
	K	107	75	75	11	11
	L	108	76	76	12	12
	: ;	59	59	58	59	58
	″ ,	39	39	34	39	34
	Rtn	13	13	–49	–50	–51
	Z	122	90	90	26	26
	X	120	88	88	24	24
	C	99	67	67	3	3
	V	118	86	86	22	22
	B	98	66	66	2	2
	N	110	78	78	14	14
	M	109	77	77	13	13
	,	44	44	60	44	60
	> .	46	46	62	46	62
	? /	47	47	63	47	63
	Space	32	32	–52	–53	–54
	F1	128	128	136	–2	–10
	F2	129	129	137	–3	–11
	F3	130	130	138	–4	–12
	F4	131	131	139	–5	–13
	F5	132	132	140	–6	–14
	F6	133	133	141	–7	–15
	F7	134	134	142	–8	–16
	F8	135	135	143	–9	–17
	DIALOG	–111	–111	–117	–123	–129
	SETUP	–112	–112	–118	–124	–130
	LOCAL	–113	–113	–119	–125	–131
	COPY	–114	–144	–120	–126	–132
	PAGE	–115	–115	–121	–127	–133
	BREAK	–116	–116	–122	–128	–134

Table D-1 (cont)

MACROS BY KEYBOARD LAYOUT

Key Group	Key Label	Un Shift	Caps Lock	Shift	Ctrl	Ctrl-Shift
Numeric Pad Keys	0	–55	–55	–69	–83	–97
	1	–56	–56	–70	–84	–98
	2	–57	–57	–71	–85	–99
	3	–58	–58	–72	–86	–100
	4	–59	–59	–73	–87	–101
	5	–60	–60	–74	–88	–102
	6	–61	–61	–75	–89	–103
	7	–62	–62	–76	–90	–104
	8	–63	–63	–77	–91	–105
	9	–64	–64	–78	–92	–106
	.	–65	–65	–79	–93	–107
	,	–66	–66	–80	–94	–108
	–	–67	–67	–81	–95	–109
	Ent	–68	–68	–82	–96	–110
Joystick Positions	Right	–135	–135	–139	–143	–147
	Up	–136	–136	–140	–144	–148
	Left	–137	–137	–141	–145	–149
	Down	–138	–138	–142	–146	–150
	Trig- ger	–151	–151	–152	–153	–154
Viewing and Function Keys	Zoom	–18	–18	–22	–26	–30
	Pan	–19	–19	–23	–27	–31
	Next View	–20	–20	–24	–28	–32
	View	–21	–21	–25	–29	–33
Mouse Buttons	Left Down	–155	–155	–161	–167	–173
	Left Up	–156	–156	–162	–168	–174
	Mid Down	–157	–157	–163	–169	–175
	Mid Up	–158	–158	–164	–170	–176
	Right Down	–159	–159	–165	–171	–177
	Right Up	–160	–160	–166	–172	–178

Table D-2

DIFFERENCES OF ASCII CHARACTER GENERATION

Key	CTRL	CTRL-Shift	Parallel Kbd	Serial Kbd
,<	X		28	44
. >	X		30	46
/ ?	X		31	47
E		X	21	5
I		X	25	9
O		X	31	15
P		X	0	16
A		X	17	1
D		X	20	4
F		X	22	6
G		X	23	7
H		X	24	8
J		X	26	10
K		X	27	11
L		X	28	12
C		X	19	3
B		X	18	2
N		X	30	14
M		X	29	13
,		X	28	60
. >		X	30	62
/ ?		X	31	63

Appendix E

COLOR COORDINATES

You can select which colors to use from a palate of four billion color mixtures. To select a particular color mapping, use the SET-BACKGROUND-COLOR and SET-SURFACE-COLOR-MAP commands, which are described in Section 7.

In these commands, you specify a particular color mixture using one of the following color coordinate systems:

- RGB (Red, Green, Blue)
- CMY (Cyan, Magenta, Yellow)
- HLS (Hue, Lightness, Saturation)
- Machine RGB

SET-COLOR-MODE Command

On power-up, the termimal is set to use the HLS system. You can select other coordinate systems with the SET-COLOR-MODE command (the Setup mode name for this command is **CMODE**.

SET-COLOR-MODE
= ^{E}c**TM** int:*color-specifying-mode*
int:*color-overlay-mode*
int:*gray-mode*

The first parameter, *color-specifying-mode*, is 1 to select the RGB color coordinate system, 2 to select the CMY system, 3 to select the HLS system, or 4 to select tha Machine RGB system. If this parameter is zero, the *color-specifying-mode* is left unchanged.

(For information about the other two parameters, see the description in Section 7 of the SET-COLOR-MODE command.)

RGB COORDINATE SYSTEM

In the RGB color coordinate system, you specify a color mixture as percentages of red, green, and blue, in that order. Each color coordinate is an integer in the range from 0 to 100.

For instance, one way to set the background color to red is to issue these commands:

- SET-COLOR-MODE: 1, 0, 1. Selects the RGB color coordinate system. Leave the overlay mode unchanged, but set the gray mode to "COL" to ensure that the display is in color rather than in black and white.
- SET-BACKGROUND-COLOR: 100,0,0. Sets the intensities of the red, green, and blue electron beams to 100%, 0%, and 0% of their maximum values, respectively. Likewise, you can set color index one on surface number three to "green," as follows:
- SET-COLOR-MODE: 1, 0, 1. Selects RGB color coordinates.
- SET-SURFACE-COLOR-MAP: 1,(1,0,100,0) Sets the color mixture for surface one, color index one, as follows: 0% red, 100% green, 0% blue.

In Setup mode, these latter two commands are typed as follows:

```
CMODE 1
CMAP  1  1  0  100  0
```

From the host computer, these same commands are sent as escape sequences:

SET-COLOR-MODE: 1,0,1
= ^{E}c**TM** int:1 int:0 int:1
= **^{E}cTM101**

SET-SURFACE-COLOR-MAP: 1, (1,0,100,0)>
= ^{E}c**TG** int:1 int-array:(1,0,100,0)
= ^{E}c**TG** int:1 int:4 int:1 int:0 int:100 int: 0
= **^{E}cTG1410F40**

CMY COORDINATE SYSTEM

In the CMY system, the three color coordinates are percentages percentages of cyan, magenta, and yellow pigments. Each coordinate is an integer in the range from 0 to 100.

(The additive primaries — red, green, and blue — are used when mixing lights to produce color mixtures. The subtractive primaries — cyan, magenta, and yellow — are used when mixing pigments.)

The CMY coordinates are related to the RGB coordinates as follows:

$$C = 100 - R$$
$$M = 100 - G$$
$$Y = 100 - B$$

For instance, you can use the following commands can be used to select a red background color:

- SET-COLOR-MODE: 2, 0, 1. Selects the CMY coordinate system, while leaving the overlay mode unchanged and setting the gray mode to "color" rather than "black and white."
- SET-BACKGROUND-COLOR: 0,100,100. Mixes pigments of 0% cyan, 100% magenta, and 100% yellow to produce a "red" color mixture.

In Setup mode, the operator can type these commands as follows:

CMODE CMY
$^{E}{}_{c}$TB 0 100 100

(There is no Setup mode name for the SET-BACKGROUND-COLOR command, so in Setup mode the operator must use the escape-sequence op code for that command.)

The same two commands can be sent from the host computer as escape sequences:

SET-COLOR-MODE: 2,0,0
= $^{E}{}_{c}$**TM** int:2 int:0 int:0
= $^{E}{}_{c}$**TM200**

SET-BACKGROUND-COLOR: 0,100,100
= $^{E}{}_{c}$**TB** int:0 int:100 int:100
= $^{E}{}_{c}$**TB0F4F4**

HLS COORDINATE SYSTEM

In the HLS coordinate system, the universe of possible color mixtures is represented as a double-ended cone (Figure E-1). The three coordinates are H (hue), L (lightness), and S (saturation).

Hue. The hue coordinate runs around the cone, from 0 to 360 degrees:

Hue Coordinate	Color Name
0	Blue
60	Magenta
120	Red
150	Orange (red-yellow)
180	Yellow
240	Green
300	Cyan

Lightness. The lightness coordinate runs up the cone, from black at the bottom (0% lightness) to white at the top (100% lightness).

Saturation. The saturation coordinate expresses the degree to which a color mixture differs from a shade of gray. This coordinate runs radially outward from the axis of the HLS cone. It is expresses as a percentage of the maximum saturation that is possible at a given lightness level. The most fully saturated color mixtures are at the 50% lightness level, where the double-ended cone is widest.

In the HLS coordinate system, all "red" color mixtures have the same hue angle. For instance, "dark red," "fully saturated red," and "light red" differ only in the lightness coordinate:

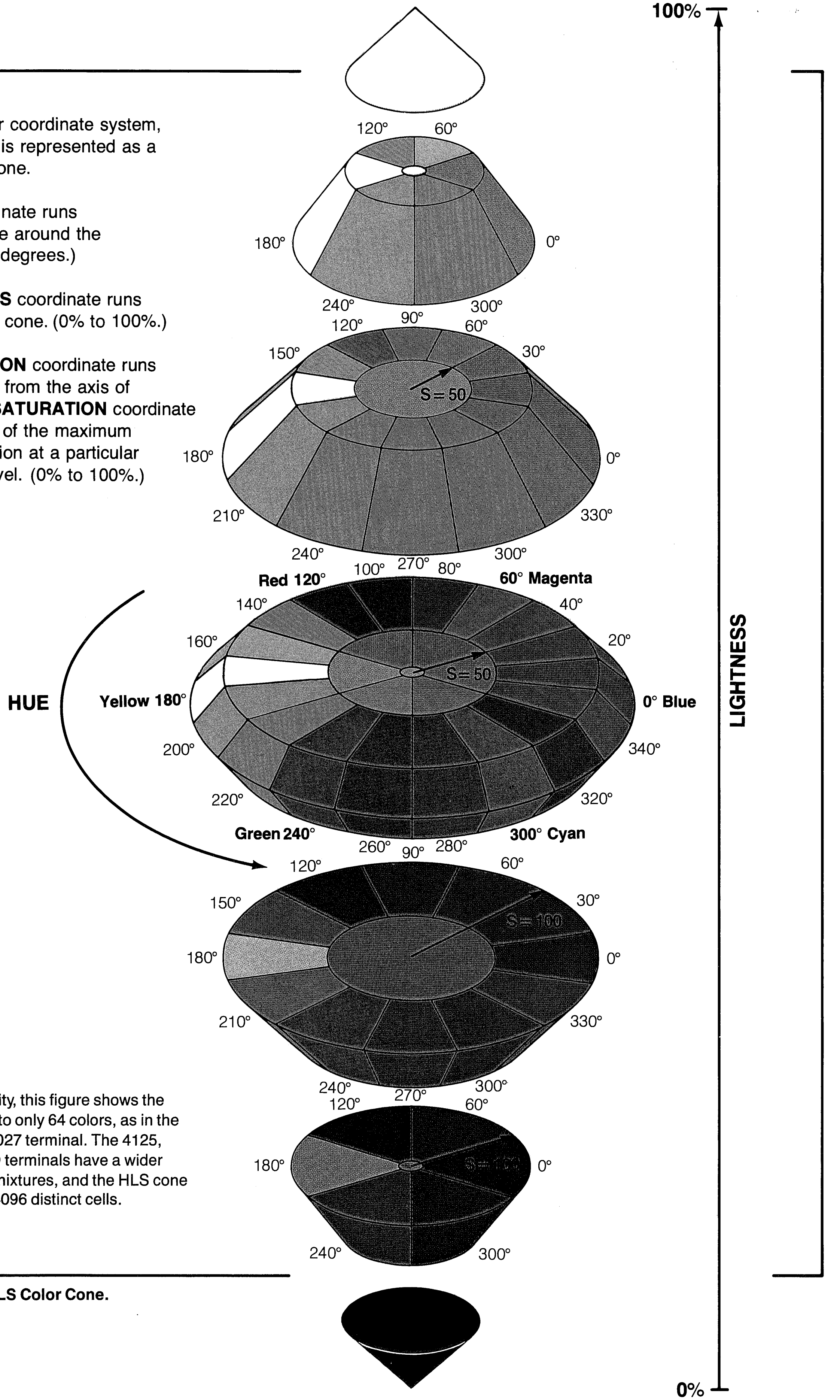
Color Name	H	L	S
Dark Red	120	33	100
Fully Saturated Red	120	50	100
Light Red	120	67	100

In the **HLS** color coordinate system, the color space is represented as a double-ended cone.

The **HUE** coordinate runs counterclockwise around the cone. (0 to 360 degrees.)

The **LIGHTNESS** coordinate runs vertically up the cone. (0% to 100%.)

The **SATURATION** coordinate runs radially outward from the axis of the cone. The **SATURATION** coordinate is a percentage of the maximum possible saturation at a particular **LIGHTNESS** level. (0% to 100%.)



NOTE: For clarity, this figure shows the cone divided into only 64 colors, as in the TEKTRONIX 4027 terminal. The 4125, 4128, and 4129 terminals have a wider range of color mixtures, and the HLS cone is divided into 4096 distinct cells.

Figure E-1. HLS Color Cone.

TEKTRONIX COLOR STANDARD

Overview:

The world of color is filled with ambiguous terminology, i.e. intensity, purity, value, etc. Many color users feel that "color theory" is a prerequisite to operating color systems; T.V., Videotaping, Photography, Computer Graphics.

In order to end this confusion, Tektronix has developed a color language and function based on human engineering, rather than machine engineering. Below is a description of this system, which will provide a clear and concise means for understanding how color is defined and how our syntax was derived.

Color Concepts:

Color selection is specified by hue, lightness and saturation which is the HLS method. The definitions are as follows:

Hue: The characteristic associated with a color name such as red, yellow, green, blue, etc. Hue is a gradation of color advanced by degrees, thus represented as an angle from 0 to 360.

Lightness: The characteristic that allows the color to be ranked on a scale from dark to light. Lightness is expressed as a parameter ranging from 0 to 100% with black being 0 (bottom of cone) and white being 100% (top of cone).

Saturation: The characteristic which describes the extent to which a color differs from a gray of the same lightness. Saturation is expressed as percentage, ranging from 0% (maximum white content at that lightness level) to 100% (full saturated).

Geometrically, colors can be described in terms of a double cone.
Variations in lightness are represented along the axis, with white at the apex of the cone and black at the opposite apex. Variations in saturation are represented by radial distances from the lightness axis, in constant lightness planes. Hue is represented as an angular quantity from a known reference point.

Likwise, you can get light-colored mixtures of different hues by setting the lightness coordinate to a relatively large value and varying only the hue coordinate:

Color Name	H	L	S
Light Red	120	67	100
Light Orange	150	67	100
Light Yellow	180	67	100
Light Green	240	67	100
Light Blue	0	67	100

Again, different mixtures of a given color with gray can be achieved by varying only the saturation coordinate:

Color Name	H	L	S
50% Gray	120	50	0
Grayish Red	120	50	50
Fully Saturated Red	120	50	100

For instance, to set the color index 2 on surface 1 to a light shade of green, you could issue these commands:

```
SET-COLOR-MODE: 3,0,1
```

Select the HLS color coordinate system, while leaving the overlay mode unchanged and setting the gray mode so that the display is in color rather than in black and white.

```
SET-SURFACE-COLOR-MAP: 1,(2,240,67,100)
```

Set the color mixture for surface one, color index two, as follows: a green hue (H = 240), of a light shade (L = 67), with the maximum saturation possible at that lightness level (S = 100).

For more information on the SET-COLOR-MODE and SET-SURFACE-COLOR-MAP commands, see their descriptions in Section 7.

MACHINE RGB COORDINATE SYSTEM

Machine RGB behaves like regular RGB mode, except instead of being limited to 100 units in each color direction (red, green, and blue), 256 units are available in each direction.

Since there are 256 colors allowed at any one time on the display, the finer divisions allow you to tune more finely the colors.

Table E-1 shows the default color index values in RGB coordinates.

Table E-1

MACHINE RGB DEFAULT COLORS

INDEX	R	G	B	INDEX	R	G	B
0	0	0	0	50	60	100	17
1	100	100	100	51	45	100	25
2	100	0	0	52	17	90	45
3	0	100	0	53	17	83	55
4	0	0	100	54	17	67	67
5	0	100	100	55	17	55	75
6	100	0	83	56	17	40	90
7	100	100	0	57	40	25	90
8	100	50	0	58	55	17	90
9	60	83	0	59	83	17	75
10	0	100	60	60	100	17	67
11	0	50	100	61	100	25	60
12	55	0	83	62	100	40	50
13	100	0	55	63	100	50	40
14	33	33	33	64	100	83	9
15	67	67	67	65	75	100	0
16	100	100	33	66	60	100	13
17	75	100	45	67	40	100	17
18	45	100	60	68	13	100	33
19	17	100	75	69	13	83	45
20	25	83	83	70	17	75	50
21	50	67	83	71	13	60	67
22	75	55	83	72	13	50	75
23	100	33	90	73	13	40	83
24	67	90	45	74	17	25	90
25	40	90	55	75	17	17	100
26	17	90	67	76	33	9	100
27	17	67	90	77	55	9	90
28	17	50	100	78	75	0	83
29	45	33	100	79	90	0	75
30	75	17	100	80	90	75	9
31	90	9	100	81	67	90	9
32	83	100	17	82	55	90	13
33	67	100	25	83	25	90	25
34	45	100	33	84	13	90	33
35	17	100	50	85	17	75	40
36	17	83	60	86	13	67	50
37	17	67	75	87	13	55	60
38	17	55	83	88	13	45	67
39	25	45	90	89	17	33	75
40	40	33	90	90	17	25	83
41	67	17	90	91	33	17	83
42	83	17	83	92	55	13	75
43	90	33	67	93	75	13	67
44	83	45	60	94	90	17	55
45	83	60	50	95	83	25	50
46	90	67	40	96	83	75	0
47	90	67	25	97	60	83	9
48	100	90	9	98	45	90	9
49	83	100	9	99	33	90	13

Table E-1 (cont)

MACHINE RGB DEFAULT COLORS

INDEX	R	G	B	INDEX	R	G	B
100	13	90	25	150	9	40	40
101	17	83	25	151	17	33	40
102	17	75	33	152	17	25	45
103	17	60	45	153	25	17	50
104	13	50	55	154	25	13	55
105	13	45	60	155	33	13	50
106	17	33	67	156	45	13	45
107	33	25	67	157	55	13	40
108	45	25	60	158	67	13	33
109	67	25	50	159	83	13	25
110	83	25	40	160	50	45	9
111	100	25	33	161	40	50	9
112	75	67	0	162	33	60	0
113	55	75	9	163	17	60	9
114	40	83	9	164	13	67	9
115	25	90	9	165	13	60	13
116	17	90	13	166	13	50	17
117	17	83	17	167	9	45	25
118	13	75	25	168	13	40	25
119	13	67	33	169	13	33	33
120	13	60	40	170	9	25	40
121	13	45	50	171	9	17	50
122	13	40	55	172	9	13	55
123	25	33	55	173	13	0	60
124	45	33	45	174	33	0	50
125	55	33	40	175	50	0	40
126	67	33	33	176	75	33	0
127	83	33	25	177	67	40	0
128	67	60	0	178	55	40	9
129	50	67	9	179	33	40	17
130	40	67	13	180	25	33	25
131	25	67	17	181	25	25	33
132	13	67	25	182	25	17	40
133	9	60	33	183	33	13	40
134	9	50	40	184	50	13	33
135	13	40	45	185	60	13	25
136	13	33	50	186	75	9	25
137	9	17	67	187	83	0	25
138	17	13	67	188	100	0	17
139	40	13	55	189	100	13	13
140	50	13	50	190	90	17	9
141	60	13	45	191	83	25	9
142	75	13	40	192	60	25	0
143	90	13	33	193	45	40	0
144	55	50	9	194	25	40	13
145	45	60	9	195	25	33	17
146	33	60	13	196	17	25	25
147	17	60	17	197	17	17	33
148	9	55	25	198	25	13	33
149	13	45	33	199	33	9	33

Table E-1 (cont)

MACHINE RGB DEFAULT COLORS

INDEX	R	G	B	INDEX	R	G	B
200	50	9	25	250	17	0	9
201	67	9	17	251	25	0	9
202	83	0	17	252	33	0	9
203	90	0	13	253	45	9	0
204	100	0	9	254	25	9	0
205	100	13	9	255	13	9	0
206	83	17	0				
207	75	25	0				
208	50	17	0				
209	40	33	0				
210	13	45	9				
211	9	40	13				
212	9	33	17				
213	9	25	25				
214	13	17	33				
215	17	9	40				
216	25	0	33				
217	45	0	25				
218	60	0	17				
219	75	0	13				
220	83	0	9				
221	75	9	9				
222	67	13	9				
223	60	17	0				
224	40	17	0				
225	17	33	0				
226	9	33	0				
227	9	25	9				
228	9	17	13				
229	9	13	25				
230	9	13	33				
231	9	0	33				
232	13	0	25				
233	17	0	17				
234	33	0	17				
235	33	0	13				
236	40	0	9				
237	45	9	9				
238	45	13	0				
239	17	9	0				
240	25	13	0				
241	13	13	0				
242	9	13	0				
243	9	13	9				
244	9	13	13				
245	0	13	13				
246	9	9	13				
247	13	0	13				
248	13	0	17				
249	17	0	13				

Appendix F

DISPLAY LIST OP CODES

The terminal contains a special-purpose microprocessor called a *picture processor*. The picture processor excutes a list of instructions (a display list) to display a picture on the screen. *Op codes* tell the picture processor what operation to perform.

A picture processor instruction consists of an op code, which is always one byte long, followed by any required operands (see *Format of A Picture Processor Instruction* later in this appendix). Some instructions do not require operands, making the total instruction only one byte long. The longest instruction, a 3CATFORM instruction, is 37 bytes long (an op code followed by 36 operand bytes).

Only certain op codes are allowed in a display list by the devices DS: and SG:. In the dictionary part of this appendix, the allowed op codes for these devices are listed in alphabetical order according to the op code mnemonic. Under each op code definition, the following information is given:

- a two-digit hexadecimal equivalent for the op code
- the operation
- the length of the total instruction
- a description of what the total instruction does
- the parameters (operands) required

FORMAT OF A PICTURE PROCESSOR INSTRUCTION

A picture processor instruction must always consist of a number of 8-bit bytes. The first byte is always the op code. The parameters for an instruction immediately follow the op code byte and are packed one after another so that there are no unused bits between parameters. Figure F-1 shows the format of a picture processor instruction.

TERMS USED IN OP CODE DEFINITIONS

(Bx,By) — The coordinates of the current beam position.

(Xabs,Yabs) — The coordinates for an absolute move or draw.

(Xrel,Yrel) — The coordinates for a relative move or draw.

Signed(n) — a field of n bits that are specified in 2′s complement form in the range –2<n–1> to +2<n–1>–1.

SignedPair(m,n) — A signed(m) preceded by a signed(n), resulting in a signed(m + n) field. When the SignedPair(m,n) appears in a data string, the m-bit subfield contains the low order bits of the string and the n-bit subfield contains the high order bits. Figure F-2 shows the data format of a SignedPair(8,8) field.

Unsigned(n) — a field of n bits that specify an unsigned integer in the range 0 to +2<n>–1.

UnsignedPair(m,n) — An unsigned(m) preceded by an unsigned(n), resulting in an unsigned(m + n) field. When UnsignedPair(m,n) appears in a data string, the m-bit subfield contains the low order bits of the string and the n-bit subfield contains the high order bits (similar to the SignedPair(8,8) shown in Figure F-2).

OP CODE DICTONARY

ADRW32

Op Code: 21 hex

Operation: 32-bit Absolute Draw

Total Length (op code + operands): 9 bytes

Description: This instruction draws from (Bx,By) to (Xabs,Yabs). Since the draw begins at the current beam position, only the ending coordinates of the draw need to be specified as a parameter.

Parameter1: (Xabs,Yabs) = SignedPair(32,32).

AMOV16

Op Code: 1F hex

Operation: 16-bit Absolute Move

Total Length (op code + operands): 5 bytes

Description: This instruction moves the beam postion to (Xabs,Yabs). Only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xabs,Yabs) = SignedPair(16,16).

AMOV32

Op Code: 20 hex

Operation: 32-bit Absolute Move

Total Length (op code + operands): 9 bytes

Description: This instruction moves the beam position to (Xabs,Yabs). Only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xabs,Yabs) = SignedPair(32,32).

BPANEL

Op Code: 2B hex

Operation: Begin Panel

Total Length (op code + operands): 1 byte

Description: Begin a panel starting at (Bx,By). No parameters are needed for this instruction, since the current beam position is used as a starting point. To end (close) the panel, an EPANEL instruction must be given.

CANGLE

Op Code: 09 hex

Operation: Concatenate Angle

Total Length (op code + operands): 13 bytes

Description: This instruction rotates the coordinate space by angle A for any subsequent relative moves and draws. Angle A is specified by its sine and cosine (where SA and CA are parameters for this instruction):

SIN(A) = (SA)(2<-46>)
COS(A) = (CA)(2<-46>)

When doing a CANGLE instruction, the current beam position (Bx,By) acts as the invariant point and does not change screen position. The new transform stays in effect until it is again changed (or the terminal is turned off).

Parameter1: CA = Signed(48)
Parameter2: SA = Signed(48)

CQUADR

Op Code: 08 hex

Operation: Concatenate Quadrant

Total Length (op code + operands): 2 bytes

Description: This instruction contains special cases of the CANGLE and CSCALE instructions. One of eight special cases may be defined by the parameter *newcase*. The parameter values of the special cases are:

0: Rotate the coordinate space 0 degrees (this instruction is essentially a no operation instruction).

1: Negate the x-coordinates (this causes a reflection about the y-axis).

2: Negate the y-coordinates (this causes a reflection about the x-axis).

3: Rotate the coordinate space +180 degrees.

4: Rotate the coordinate space +90 degrees, then negate the x-coordinates.

5: Rotate the coordinate space –90 degrees.

6: Rotate the coordinate space +90 degrees.

7: Rotate the coordinate space –90 degrees, then negate the x-coordinates.

When doing a CQUADR instruction, the current beam position (Bx,By) acts as the invariant point and does not change screen position. The new transform stays in effect until it is again redefined (or until the terminal is turned off).

Parameter1: Newcase = Unsigned(8)

CSCALE

Op Code: 07 hex

Operation: Concatenate Scale

Total Length (op code + operands): 7 bytes

Description: This instruction scales the x-axis and y-axis for any subsequent relative moves or draws. When these axes are scaled, the current beam position (Bx,By) acts as the invariant point and does not change screen position.

The x-axis is scaled by a factor of (a)(2<na>) times its current scale. The y-axis is scaled by a factor of (d)(2<nd>) times its current scale. Negative numbers can be used when specifying the scaling factors. The new transform remains in effect until the scale is again changed (or until the terminal is turned off).

Parameter1: a = Signed(16)
Parameter2: d = Signed(16)
Parameter3: na = Signed(8)
Parameter4: nd = Signed(8)

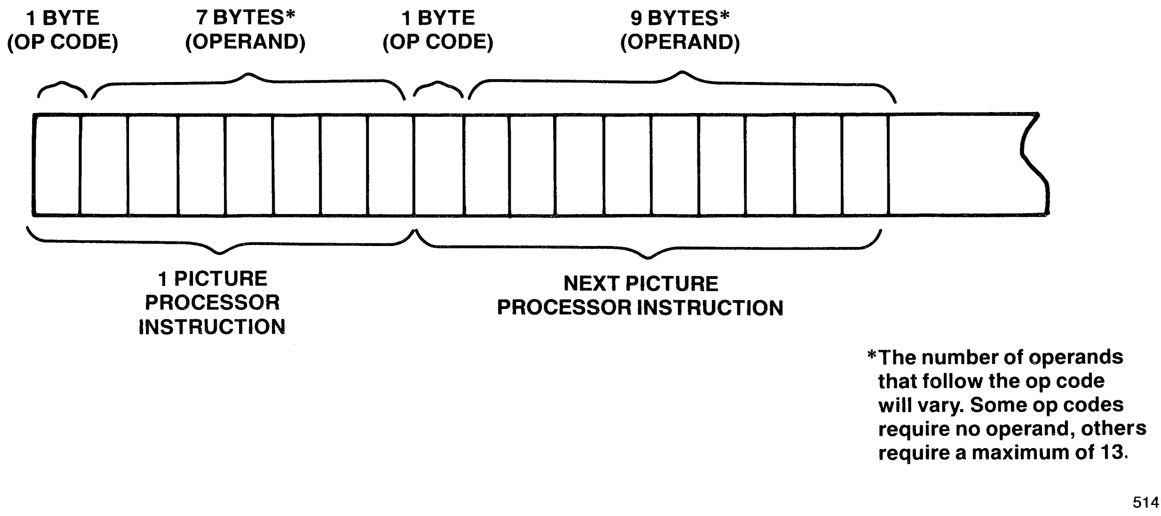

1 BYTE (OP CODE)
7 BYTES* (OPERAND)
1 BYTE (OP CODE)
9 BYTES* (OPERAND)
1 PICTURE PROCESSOR INSTRUCTION
NEXT PICTURE PROCESSOR INSTRUCTION
*The number of operands that follow the op code will vary. Some op codes require no operand, others require a maximum of 13.
5141-11

Figure F-1. Format of a Picture Processor Instruction.

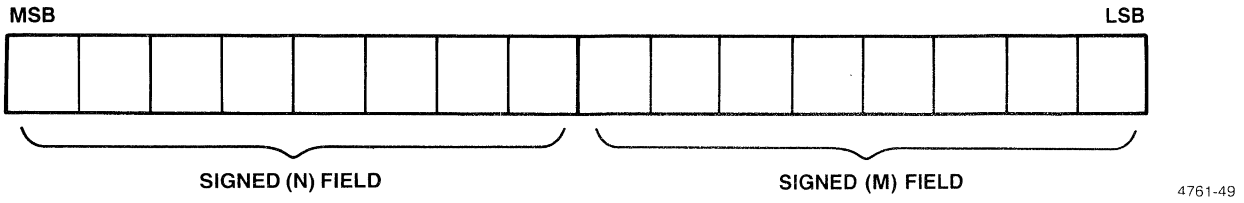


Figure F-2. A SignedPair(8,8) Field.

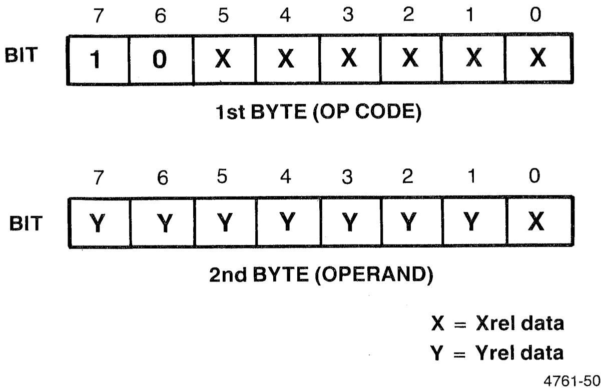
DRAW7

Op Code: 80 — BF hex

Operation: Short Relative Draw

Total Length (op code + operands): 2 bytes

Description: This instruction does a short relative draw from (Bx,By) to (Bx + Xrel,By + Yrel). The instruction is in two bytes, the first being the op code, the second the parameter. However, part of the op code is used to specify Xrel (see Figure F-3).



BIT	7	6	5	4	3	2	1	0
	1	0	X	X	X	X	X	X

1st BYTE (OP CODE)

BIT	7	6	5	4	3	2	1	0
	Y	Y	Y	Y	Y	Y	Y	X

2nd BYTE (OPERAND)

X = Xrel data
Y = Yrel data

Figure F-3. Data Format for DRAW7.

Parameter1: See Figure F-3

DRAW8

Op Code: 17 hex

Operation: 8-bit Relative Draw

Total Length (op code + operand): 3 bytes

Description: This instruction does a relative draw from (Bx,By) to (Bx + Xrel,By + Yrel). Since the draw begins at the current beam position, only the ending coordinates of the draw need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(8,8)

DRAW16

Op Code: 18 hex

Operation: 16-bit Relative Draw

Total Length (op code + operand): 5 bytes

Description: This instruction does a relative draw from (Bx,By) to (Bx + Xrel,By + Yrel). Since the draw begins at the current beam position, only the ending coordinates of the draw need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(16,16)

DRAW32

Op Code: 19 hex

Operation: 32-bit Relative Draw

Total Length (op code + operand): 9 bytes

Description: This instruction does a relative draw from (Bx,By) to (Bx + Xrel,By + Yrel). Since the draw begins at the current beam position, only the ending coordinates of the draw need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(32,32)

DRAW40

Op Code: 1A hex

Operation: 40-bit Relative Draw

Total Length (op code + operand): 11 bytes

Description: This instruction does a relative draw from (Bx,By) to (Bx + Xrel,By + Yrel). Since the draw begins at the current beam position, only the ending coordinates of the draw need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(40,40)

EPANEL

Op Code: 2C hex

Operation: End the Panel

Total Length (op code + operand): 1 byte

Description: This instruction ends a panel. No parameters are needed for this instruction.

The legal sequence for a simple panel is:

Move or draw to set beam position
BPANEL
Draws (no moves)
EPANEL (draws back to starting beam position and resets beam position to that value)

Draw instructions must be used even if the boundary of the panel will not be visible (see PANSTL). Relative draws are perferred in a segment because they allow the segment to be repositioned.

ESBPAN

Op Code: 2D Hex
Operation: End Subpanel

Total Length (op code + operands): 1 byte

Description: This instruction ends a subpanel. No parameters are needed for this instruction.

The legal sequence using subpanels is:

Move or draw to set beam position
BPANEL
Draws (no moves)
ESBPAN (draw back to starting beam postion and resets beam
position to that value)
BPANEL
Draws (no moves)
ESBPAN
Move to start of subpanel
BPANEL
Draws (no moves)
EPANEL

Note that the last subpanel must end with an EPANEL, not an ESBPAN.

FILLPT

Op Code: 0A Hex
Operation: Fill Pattern

Total Length (op code + operands): 4 bytes

Description: This instruction defines the pattern that is used to fill a panel. There are two operands for this instruction. The first parameter consists of two bytes, which must contain the fill pattern number. The fill pattern number must be 1 to 32767 if specifying a user-defined pattern, or the twos complement form of 0 to –255 to indicate a solid fill pattern of that color index. The second parameter must always be a zero.

If Parameter 1 contains a number of a pattern that is not defined, or a negative number less than –255, the instruction defaults to pattern 0 (transparent)

Parameter1:	Fill Pattern (1 to 32767) = Unsigned(16) Fill Pattern (0 to –255) = Signed16)
Parameter2:	Always a 0

LINGAP

Op Code: 05 Hex
Operation: LineGap Index

Total Length (op code + operands): 3 bytes

Description: This instruction specifies the color index of the gap between lines when a particular line style is drawn. The possible indices are:

–2	specifies the viewport index
–1	specifes transparent as the index
0 to 255	specifies the corresponding color index

The new line gap index remains in effect for any subsequent draws until it is again changed (unless the terminal is turned off).

Parameter1: New Line Gap Index = Signed(16)

LININD

Op Code: 04 Hex
Operation: Line Index

Total Length (op code + operand): 3 bytes

Description: This instruction specifies the color index of the line that is drawn. The possible indices are:

–2	specifies the viewport index
0 to 255	specifies the corresponding color index

The new line index remains in effect for any subsequent draws until it is again changed (unless the terminal is turned off).

Parameter1: New Line Index = Signed(16)

LINSTL

Op Code: 06 Hex
Operation: Line Style

Total Length (op code + operand): 2 bytes

Description: This instruction specifies the number corresponding to the type of line style to be drawn. All lines drawn after this instruction remain at the specified style until the style is changed (unless the terminal is turned off).

Parameter1: New Line Style = Signed(8)

MOVE8

Op Code: 1B Hex
Operation: 8-bit Relative Move

Total Length (op code + operand): 3 bytes

Description: This instruction moves beam from (Bx,By) to (Bx + Xrel,By + Xrel). Since the move originates at the current beam position, only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(8,8)

MOVE16

Op Code: 1C Hex
Operation: 16-bit Relative Move

Total Length (op code + operand): 5 bytes

Description: This instruction moves beam from (Bx,By) to (Bx + Xrel,By + Xrel). Since the move originates at the current beam position, only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(16,16)

MOVE32

Op Code: 1D Hex
Operation: 32-bit Relative Move

Total Length (op code + operand): 9 bytes

Description: This instruction moves beam from (Bx,By) to (Bx + Xrel,By + Xrel). Since the move originates at the current beam position, only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(32,32)

MOVE40

Op Code: 1E Hex
Operation: 40-bit Relative Move

Total Length (op code + operand): 11 bytes

Description: This instruction moves beam from (Bx,By) to (Bx + Xrel,By + Xrel). Since the move originates at the current beam position, only the ending coordinates of the move need to be specified as a parameter.

Parameter1: (Xrel,Yrel) = SignedPair(40,40)

MRKDRW

Op Code: 27 Hex
Operation: Marker Draw

Total Length (op code + operand): 2 bytes

Description: This instruction draws a marker centered at the current beam position in 2D coordinate mode. If the terminal is in 3D coorindate mode, the marker is drawn at the current 3D beam position. When the marker has been drawn, the beam position is at the center of the marker (unchanged from before the draw). The parameter specifies the corresponding number of the marker to be drawn.

Parameter1: Marker = Unsigned(8)

NOOPER

Op Code: 16 Hex
Operation: No operation

Total Length (op code + operand): l byte

Description: This program does not perform any operation. It does, however, increment the program counter by one. This instruction does not have any operands.

PANSTL

Op Code: 0C Hex
Operation: Panel Style

Total Length (op code + operand): 3 bytes

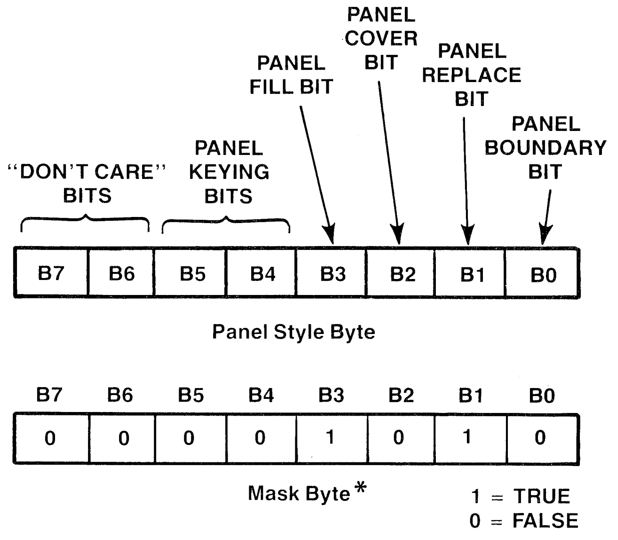
Description: This instruction defines the style of panel to be drawn. There are two operands, each eight bits long, to this instruction. The first parameter specifies the panel style. The second parameter is a mask byte.

Certain bits in the panel style byte define characteristics of of the panel (see Figure F-4):

- Panel Boundry bit — If this bit is TRUE, a boundry is drawn around the pattern. If this bit is FALSE, no boundry is drawn.
- Panel Replace bit — If this bit is TRUE, then the zeros in the fill pattern definition are placed in the bit planes. If this bit is FALSE, no zeros are placed in the bit planes.
- Panel Cover bit — If this bit is TRUE, then the interior and boundry of the panel is filled with the fill-pattern data. If this bit is FALSE, then only the interior of the panel, and not the boundry, is to be filled with fill-pattern data.
- Panel Fill bit — If this bit is TRUE, then fill the panel. If this bit is FALSE, then do not fill the panel.
- Panel Keying bits — If bits 4 and 5 are equal to a zero, then data is keyed to the lower-left corner of the viewport. If bits 4 and 5 are a one, then data is keyed to the lower-left corner of the screen. If bits 4 and 5 are a two, then the fill-pattern data is keyed to the panel being drawn.

The Mask byte is used to specify what bits to change. If the mask is set to zero for a specific bit, that bit cannot be changed (no matter what state the corresponding bit in the panel style byte is in). If a bit is to be changed in the panel style byte, the corresponding mask bit must be set to one.

Parameter1: New Panel Style = Unsigned(8)
Parameter2: Mask = Unsigned(8)



B7	B6	B5	B4	B3	B2	B1	B0

B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	1	0	1	0

*This setting of the mask byte would only allow bits 2 and 4 of the panel style byte to change.

4761-52

Figure F-4. Panel Style and Mask Bytes.

PICKID

Op Code: 03 Hex
Operation: PickID is set equal to newpickID

Total Length (op code + operand): 3 bytes

Description: This instruction sets pickID equal to newpickID. The pickID is equal to zero, nothing satisfies the pick conditions. NewpickID should always be greater than zero.

Parameter1: NewPickID = Signed (16)

POPBC

Op Code: 12 Hex
Operation: Pop Beam Context

Total Length (op code + operand): 1 byte

Description: Pop the current transform, the current beam position (Bx,By), and associated primitive attributes. If a panel is popped, redisplay the panel on the screen (see PUSHBC). If the panel was open when it was pushed, POPBC continues as if the panel was never pushed.

The actual transform, beam position, and attributes popped depend on the current coordinate mode:

- In 2D mode, those popped are the 2D transform, beam position, and the attributes text index, text size, text background index, ALU mode, dash pattern, pickID, and viewport coordinates.
- In 3D mode, those popped are the 3D transform, 3D beam coordinates (Bx,By,Bz), the 2D attributes previously listed, and the 3D attributes object color and translucency.

PUSHBC

Op Code: 11 Hex
Operation: Push Beam Context

Total Length (op code + operand): 1 byte

Description: Push the current transform, the current beam position (Bx,By), and associated primitive attributes. If an open panel is pushed, save all information about that panel and continue as if no panel had been open.

The actual transform, beam position, and attributes pushed depend on the current coordinate mode:

- In 2D mode, those pushed are 2D transform, beam position, and the attributes text index, text size, text background index, fill pattern, panel style, marker size, line index, ALU mode, dash pattern, pickID, and viewport coordinates.
- In 3D mode, those pushed are 3D transform, 3D beam coordinates (Bx,By,Bz), the 2D attributes previously listed, and the 3D attributes object color and translucency pattern.

RCT8

Op Code: 28 Hex
Operation: 8-bit Panel Rectangle

Total Length (op code + operand): 3 bytes

Description: Draw a panel of a specified width and height starting at (Bx,By). Since the panel begins at the current beam position, only the width (W) and height (H) need to be given as operands. Leaves the current beam postion at (Bx + W, By + H).

Parameter1: (W,H) = UnsignedPair(8,8)

RCT16

Op Code: 29 Hex
Operation: 16-bit Panel Rectangle

Total Length (op code + operand): 5 bytes

Description: Draw a panel of a specified width and height starting at (Bx,By). Since the panel begins at the current beam position, only the width (W) and height (H) need to be given as operands. Leaves the current beam postion at (Bx + W, By + H).

Parameter1: (W,H) = UnsignedPair(16,16)

RCT32

Op Code: 2A Hex
Operation: 32-bit Panel Rectangle

Total Length (op code + operand): 9 bytes

Description: Draw a panel of a specified width and height starting at (Bx,By). Since the panel begins at the current beam position, only the width (W) and height (H) need to be given as operands. Leaves the current beam postion at (Bx + W, By + H).

Parameter1: (W,H) = UnsignedPair(16,16)

RESATR

Op Code: 10 Hex
Operation: Reset Attributes

Total Length (op code + operand): 1 byte

Description: If the the terminal is not in highlighted mode, then reset the graphic primitive attributes to their default values. When the terminal is in 3D coordinate mode, the 3D primitive attributes are reset.

POPAX

Op Code: 30 Hex
Operation: Pop attributes, transform, and current beam position

Total Length (op code + operand): 1 byte

Description: Pop the current transform, the current beam position (Bx, By), and any associated attributes.

POPPS

Op Code: 34 Hex
Operation: Pop current beam position

Total Length (op code + operand): 1 byte

Description: Pop the current beam position (Bx,By).

POPXP

Op Code: 32 hex

Operation: Pop transform and current beam position

Total Length (op code + operand): 1 byte

Description: Pop current transform and current beam position (Bx,By).

PUSHAX

Op Code: 2F Hex
Operation: Push attributes, current transform, and current beam position

Total Length (op code + operand): 1 byte

DESCRIPTION: Push current transform, the current beam position (Bx, By), and associated attributes.

PUSHPS

Op Code: 33 Hex
Operation: Push beam position

Total Length (op code + operand): 1 byte

Description: Push current beam position (Bx,By).

PUSHXP

Op Code: 31 Hex
Operation: Push current transform and current beam position.

Total Length (op code + operand): 1 byte

DESCRIPTION: Push current transform and current beam

position (Bx, By).

RETURN

Op Code: 13 Hex
Operation: Return

Total Length (op code + operand): 1 byte

Description: This instruction marks the end of a segment definition. It is valid only for pseudo device SG:.

STMENT

Op Code: 35 Hex
Operation: Enter stroke text mode

Total Length (op code + operand): 1 byte

Description: This instruction saves current attributes, sets line style (LINSTL) for a solid line (0), and sets lineindex (LININD) to textindex (TXTIND). This op code should be used immediately proceeding a sequence of moves and draws that are stroke text.

STMEXI

Op Code: 36 Hex
Operation: Restores attributes

Total Length (op code + operand): 1 byte

Description: This command restores all attributes to the values saved by the STMENT command. This command should immediately follow a sequence of moves and draws that are stroke text.

SUBROU

Op Code: 2E Hex
Operation: Calls a segment

Total Length (op code + operand): 4 bytes

Description: This command calls the specified segment. If the segment does not exist, this command is a noop.

Use this command in a sequence of op codes like this (optional entries are enclosed in []).

[PICKID (allows user editing of the transform)]

<PUSHAS, PUSHXP, or PUSHPS> (Use PUSHPS if no CONCAT op codes are used. PUSHAX is always used if the attributes are to be preserved.)

RELMOV from C.P. to requested position for this call.

[CANGLE or CQUADR (from segment –5)]

[CSCALE (from segment –5)]

[RESATR]

SUBROU (segment number)

<POPAX, POPXP, or POPPS> (to match the above push)

Parameter1: Segment number to call = signed(16)
Parameter2: filler byte = 0

TXTBAC

Op Code: 26 Hex
Operation: Text Background Index

Total Length (op code + operand): 3 bytes

Description: This instruction defines the color of the background index for alphatext character cells. The possible indices are:

–2	specifies the viewport index
–1	specifes transparent as the index
0 to 255	specifies the corresponding color index

The new background index remains at the set color until the index is again changed (or until the terminal is turned off).

Parameter1: New Background Index = Signed(16)

TXTCHR

Op Code: 22 Hex
Operation: Draw Alpha Text Character

Total Length (op code + operand): 2 bytes

Description: This instruction draws an alphatext character in the character cell. The parameter specifies the ASCII code for the character to be drawn. The standard ascii character set is used if the value is between 0 and 127 (inclusive), and the alternate character set is used if the value is between 128 and 255 (inclusive).

The effect on 2D coordinate mode beam position is as follows:

- The initial beam position is left unchanged; any relative move, relative draw, or rectangle command calculates where to go based on teh beam position *before* the character.
- An internl *pixel-level position* is changed to the next character cell. The next TXTCHR operates from that pixel-level position and appears as expected.
- Even though the beam position does not change, a drae after a TXTCHR draws the visible line from the last pixel-level position to a point calculated from the beam position.

In 3D coordinate mode the beam position is not changed.

Parameter1: Character = Unsigned(8)

TXTIND

Op Code: 25 Hex
Operation: Text Index

Total Length (op code + operand): 3 bytes

Description: This instruction sets the color index for alphatext characters. The possible indices are:

–2	specifies the viewport index (this can be used to erase a character)
0 to 255	specifies the corresponding color index

The new text index remains at the set color until it is again changed (or until the terminal is turned off).

Parameter1: New Text Index = Signed(16)

TXTMOV

Op Code: 23 Hex
Operation: Alphatext Position Move

Total Length (op code + operand): 3 bytes

Description: This instruction moves the beam position by character cells. For this instruction, one Xrel or Yrel coordinate is equal to one character cell. When using this command, the new beam position is established at (Bx + Xrel,By + Yrel). In 3D coordinate mode, the beam position is unchanged.

For example, to backspace one character cell, the (Xrel,Yrel) coordinates would be specified as (–1,0). Or, as another example, to move down one character cell, the (Xrel,Yrel) coordinates would be specified as (0,–1)

Parameter1: Xrel = Signed(8)
Parameter2: Yrel = Signed(8)

If you plan to save the segment with a SAVE command (^{E}cJV), then use only thses three movements:

(–1,0)	Backspace one character
(0,–1)	Linefeed
(0,1)	Vertical tab

Parameter1:Xrel = Signed(8)
Parameter2:Yrel = Signed(8)

TXTSIZ

Op Code: 24 Hex
Operation: Set Alphatext Size

Total Length (op code + operand): 2 bytes

Description: This instruction changes the size of alphatext by specifying the amount the × and y-axis of a character cell is multiplied. The parameter for this instruction consists of an 8-bit word, with the highest four bits specifying how much to multiply the y-axis, and the lowest four bits specifying the x-axis multiplication factor.

Text size may be multiplied by a factor of 16 on each axis. A size of (1,1), which is the default character size, specifies a character cell 8 pixels wide and 16 scan lines high.

The new text size remains at the new size until it is again changed (or the terminal is turned off).

Parameter1: New Text Size = Unsigned(8)

3ADRW24

Op Code: 4A Hex
Operation: 24-bit absolute draw

Total length (op code + operand): 10 bytes

Description: This instruction draws from the 3D beam (3Bx,3By,3Bz) to (Xabs,Yabs,Zabs)

Parameter1: (Xabs,Yabs,Zabs) = SignedTriplet(24,24,24)

3AMOV16

Op Code: 48 Hex
Operation: 16-bit absolute move

Total length (op code + operand): 7 bytes

Description: This instruction moves the 3D beam to (Xabs,Yabs,Zabs).

Parameter1: (Xabs,Yabs,Zabs) = SignedTriplet(16,16,16)

3AMOVE24

Op Code: 49 Hex
Operation: 24-bit absolute move

Total length (op code + operand): 10 bytes

Description: This instruction moves the 3D beam to (Xabs,Yabs,Zabs).

Parameter1: (Xabs,Yabs,Zabs) = SignedTriplet(24,24,24)

3CATFORM

Op Code: 4F Hex
Operation: concatenate transform

Total length (op code + operand): 37 bytes

Description: This instruction specifies relative rotation and scale factors for subsequent 3D moves, draws, and edges of 3d panels and facets. The transform is in the form of a 3x3 matrix specified in row-major order. Each element of the matrix consists of a signed, 24-bit mantissa and an unsigned and biased 8-bit exponent of 2. The mantissa is assumed to be in floating point format with the binary point between bits 23 and 22 as shown:

sT.TTTTTTTTTTTTTTTTTTTTTT

where s is the sign bit, and the T's are the other 23 bits in the mantissa. The byte containing the low-order bits of the mantissa must be sent as the first byte of the mantissa, and the byte containing the sign bit must be sent as the last byte of the mantissa. The actual exponent is calculated by subtracting the exponent portion of the element from 40.

Parameter1: transform matrix = 9*[Signed(24),Unsigned(8)]

3DRAW8

Op Code: 40 Hex
Operation: 8-bit relative draw

Total length (op code + operand): 4 bytes

Description: This instruction does a short relative draw from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(8,8,8)

3DRAW16

Op Code: 41 Hex
Operation: 16-bit relative draw

Total length (op code + operand): 7 bytes

Description: This instruction does a relative draw from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(16,16,16)

3DRAW24

Op Code: 42 Hex
Operation: 24-bit relative draw

Total length (op code + operand): 10 bytes

Description: This instruction does a relative draw from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(24,24,24)

3DRAW32

Op Code: 43 Hex
Operation: 32-bit relative draw

Total length (op code + operand): 13 bytes

Description: This instruction does a relative draw from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(32,32,32)

3FCR

Op Code: 4E Hex
Operation: fill color range

Total length (op code + operands): 4 bytes

Description: This instruction determines the object-color-number which will be used for subsequent shading of object surfaces (panels and facets). There are tow operands for this instruction. The first parameter contains the object-color-number. This number must be in the range 1 to 255 to specify a user-defined color number, or –4095 to 0 to specify an extended-range color-map index for constant shading. The second parameter must always be a byte of zero.

Parameter1: object-color-number (–4095 to 255) = Signed(16)
Parameter2: 0 = Unsigned(8)

3MESH

Op Code: 56 Hex
Operation: draw quadrilateral mesh

Total length (op code + operand): 5 bytes

Description: This instruction sets up the picture processor to receive a series of RowCount × ColumnCount 3D draw instructions. Interspersed with the draw instructions may be one or more 3VI8, 3VI16, and 3VRTNRM instructions. See the DRAW-FACETS command for a detailed description of meshes. The mesh definition is completed when the last draw is sent. OutlineMode specifies which of the mesh lines will be drawn in wireframe, or outlined in shaded-image mode as follows:

bit 0 — first row boundary edges (LSB)
bit 1 — last column boundary edges
bit 2 — last row boundary edges
bit 3 — first column boundary edges
bit 4 — all inner row edges
bit 5 — all inner column edges

Parameter1: OutlineMode = Unsigned(8)
Parameter2: ColumnCount = Unsigned(8) (range 2 to 255)
Parameter3: RowCount = Unsigned(16) (range 2 to 32767)

3MOVE8

Op Code: 44 Hex
Operation: 8-bit relative move

Total length (op code + operand): 4 bytes

Description: This instruction moves the 3D beam from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(8,8,8)

3MOVE16

Op Code: 45 Hex
Operation: 16-bit relative move

Total length (op code + operand): bytes

Description: This instruction moves the 3D beam from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(16,16,16)

3MOVE24

Op Code: 46 Hex
Operation: 24-bit relative move

Total length (op code + operand): 10 bytes

Description: This instruction moves the 3D beam from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(24,24,24)

3MOVE32

Op Code: 47 Hex
Operation: 32-bit relative move

Total length (op code + operand): 13 bytes

Description: This instruction moves the 3D beam from (3Bx,3By,3Bz) to (3Bx + Xrel,3By + Yrel,3Bz + Zrel).

Parameter1: (Xrel,Yrel,Zrel) = SignedTriplet(32,32,32)

3QLST

Op Code: 5A Hex
Operation: draw quadrilateral list

Total length (op code + operand): 4 bytes

Description: This instruction sets up the picture processor to receive a series of four times Count 3D draw instructions. The count parameter is the number of quadrilaterals. Interspersed with the draw instructions may be one or more 3VI8, 3VI16, and 3VRTNRM instructions. See the DRAW-FACETS command for a detailed description of quadrilateral lists. The quadrilateral list definition is completed when the last draw is sent. OutlineMode specifies which of the quadrilateral list lines will be drawn in wireframe, or outlined in shaded-image mode as follows:

bit 0 — first edge (LSB)
bit 1 — second edge
bit 2 — third edge
bit 3 — fourth edge

Parameter1: OutlineMode = Unsigned(8)
Parameter2: Count = Unsigned(16)   (range 1 to 32767)

3SPOKE

Op Code: 57 Hex
Operation: draw triangle spoke

Total length (op code + operand): 4 bytes

Description: This instruction sets up the picture processor to receive a series of Count 3D draw instructions. Count is the number of points in the spoke. Interspersed with the draw instructions may be one or more 3VI8, 3VI16, and 3VRTNRM instructions. See the DRAW-FACETS command for a detailed description of spokes. The spoke definition is completed when the last draw is sent. OutlineMode specifies which of the spoke lines will be drawn in wireframe, or outlined in shaded-image mode as follows:

bit 0 — first spoke (1st to 2nd vertex)(LSB)
bit 1 — boundary edges (2nd through count vertex)
bit 2 — last spoke (1st to count vertex)
bit 3 — inner spokes (1st to [3rd through count−1] vertex)

Parameter1: OutlineMode = Unsigned(8)
Parameter2: Count = Unsigned(16)   (range 3 to 32767)

3STRIP

Op Code: 58 Hex
Operation: draw triangle strip

Total length (op code + operand): 4 bytes

Description: This instruction sets up the picture processor to receive a series of Count 3D draw instructions. Count is the number of points in the strip. Interspersed with the draw instructions may be one or more 3VI8, 3VI16, and 3VRTNRM instructions. See the DRAW-FACETS command for a detailed description of strips. The strip definition is completed when the last draw is sent. OutlineMode specifies which of the strip lines will be drawn in wireframe, or outlined in shaded-image mode as follows:

bit 0 — 1st row (between odd numbered vertices)
bit 1 — last edge (between last two vertices)
bit 2 — 2nd row (between even numbered vertices)
bit 3 — first edge (between first two vertices)
bit 4 — all inner edges

Parameter1: OutlineMode = Unsigned(8)
Parameter2: Count = Unsigned(16)   (range 3 to 32767)

3TLST

Op Code: 59 Hex
Operation: draw triangle list

Total length (op code + operand): 4 bytes

Description: This instruction sets up the picture processor to receive a series of three times Count 3D draw instructions. The count parameter is the number of triangles. Interspersed with the draw instructions may be one or more 3VI8, 3VI16, and 3VRTNRM instructions. See the DRAW-FACETS command for a detailed description of triangle lists. The triangle list definition is completed when the last draw is sent. OutlineMode specifies which of the triangle list lines will be drawn in wireframe, or outlined in shaded-image mode as follows:

bit 0 — first edge (LSB)
bit 1 — second edge
bit 2 — third edge

Parameter1: OutlineMode = Unsigned(8)
Parameter2: Count = Unsigned(16)   (range 1 to 32767)

3STRANP

Op Code: 5E Hex
Total Length (op code + operand): 2 bytes

Description: This instruction selects the screen-door translucency pattern for subsequent 3D panels and facets for when they are displayed in shaded image mode. The low-order 4 bits of this parameter signify which translucency pattern to select for subsequent shaded 3D panels and facets. The next three bits can have three values as follows:

000 — solid shading (pattern selection ignored)

001 — pattern all floowing 3D panels and facets with the specified pattern

100 — pattern followed by 3D panels and facets which have been sectioned

Parameter1: Pattern Type = Unsigned(8)

3VI16

Op Code: 4C Hex
Operation: vertex index

Total length (op code + operand): 3 bytes

Description: This instruction assigns an extended-range color-map index to and facet vertices until either a new index is specified (using a 3VI8 or 3VI16 instruction), or the lighting model is directed to calculate a new index (by using a 3VRTNRM instruction).

Parameter1: Vertex index = Unsigned(16)

3VI8

Op Code: 4B Hex
Operation: vertex index

Total length (op code + operand): 2 bytes

Description: This instruction assigns a color-map index to following facet vertices until either a new index is specified (using a 3VI8 or 3VI16 instruction), or the lighting model is directed to calculate a new index (by using a 3VRTNRM instruction). The index specified is actually the extended-range color map index divided by 16. This is therefore a short version of the 3VI16 instruction.

Parameter1: Vertex index = Unsigned(8)

3VRTNRM

Op Code: 4D Hex
Operation: vertex normal

Total length (op code + operand): 4 bytes

Description: This instruction sets the object-surface normal vector for subsequent panel or facet vertices. The three unit-vector components (Nx,Ny,Nz) are multiplied by 127 and passed as 8-bit integers. The normal vector is used in conjunction with the local lighting model to determine the color map index at each vertex.

Parameter1: (Nx,Ny,Nz) = SignedTriplet(8,8,8)

Appendix G

SHADED SURFACES

A major feature of the 4129 terminal is that it can shade 3D surfaces. Shaded surfaces are made up of 3D facets. 3D vectors, markers, and alphatext are not "surfaces", but are included in a shaded image if they are in 3D segments. The shading process begins when a view which contains 3D segments and whose object-surface-display attribute is "shaded-surface" or "hidden-line" is renewed. First, for each graphic primitive in each 3D segment, the shading process:

1. determines the Extended Range Color Indices (ERCI's) of the primitive
2. pixelates the primitive
3. puts the image into a z-buffer

After all segments have been processed, the intensity values in the z-buffer are sent to the frame buffer.

This appendix describes these steps, with four substeps given special attention: the Local Lighting Model, Gouraud Shading, Halftoning, and Dithering. The descriptions given here are from a "functional" point of view and may not exactly represent the internal workings of the terminal.

EXTENDED RANGE COLOR INDICES (ERCI'S)

Extended Range Color Indices (ERCI's) are 12-bit unsigned integers, which gives them the range of 0 to 4095. There are four ways to specify ERCI's:

1. Directly with the *indices-array* parameter of the DRAW-FACETS command regardless of the object-surface-color.
2. Directly with a negative or zero object-surface-color (or one whose range has been deleted after the segment was defined), when there are no elements in the *indices-array* parameter of the DRAW-FACETS command, or when normals are specified for the DRAW-FACETS command.
3. Indirectly through the Local Lighting Model (described below), when the current object-surface-color is positive and has been defined, and either no normals or indices are specified in the DRAW-FACETS command (triangles and quadrilateral types only), or one or more normals are specified in the DRAW-FACETS command.
4. Indirectly with line-index for vectors and markers, gap-index for dashed vectors, and text-index and text-background-index for alphatext. (The low-order index bits are moved into the high-order bits of the ERCI, with the remaining low-order ERCI bits set to 0; the number of bits moved is the number of bit planes assigned to the current view's surface.)

USAGES OF DIFFERENT ERCI SPECIFYING METHODS

1. If no indices or normals are used, constant shading is generated from the terminal's Local Lighting Model and based on the triangle and quadrilateral points themselves. The constant color of each triangle or quadrilateral depends on the direction of the light sources. An example is a group of flat panels, such as the walls of a room.
2. If one index per command is used, constant shading is generated for a spoke, strip, or mesh, or for all the triangles and quadrilaterals in the command, where the index is computed by the host computer. An example is a flat panel, such as a checkerboard.
3. If one index per facet is used, constant shading is generated for groups of triangles and quadrilaterals where each facet's index is computed by the host computer. An example is a group of flat panels, such as a mechanical part.
4. If one index per vertex is used, smooth surface shading is generated based on the indices computed by the host computer. An example is a realistic image of curved surfaces, such as an automobile fender.
5. If one normal per command is used, constant shading is generated for spokes, strips, and meshes, or for all the triangles or quadrilaterals in the command, where the index is computed through the Local Lighting Model. An example is a flat panels, such as a checkerboard.
6. If one normal per facet is used, constant shading is generated for groups of triangles and quadrilaterals where the indices are computed through the Local Lighting Model. An example is a group of flat panels, such as a mechanical part.

7. If one normal per vertex is used, smooth surface shading is generated where the indices are computed through the Local Lighting Model. An example is a realistic image of a curved surface, such as an automobile fender.

Note that 2, 3, and 4 do not use the Local Lighting Model, while items 1, 5, 6, and 7 do. Table G-1 summarizes this information.

Table G-1

DIFFERENT ERCI SPECIFYING METHODS

	Results In Constant Shading	Results In Smooth Shading
Host Lighting Model (indices)	one index/command one index/facet[a]	one index/vertex
Local Lighting Model (normals)	no indices or normals[a] one normal/command one normal/facet[a]	one normal/vertex

[a] triangles and quadrilaterals only

LOCAL LIGHTING MODEL

The Local Lighting Model derives an ERCI from an object-surface-color-range, a normal vector, and the current lightsources, using Lambert's cosine law. When the Local Lighting Model derives an ERCI for each vertex in a facet, the result is sometimes called "cosine shading." When an ERCI is derived for a triangle or quadrilateral, the result is "constant shading."

First, an intensity in the range 0 to 1 is computed from the normal vector and the lightsources, using this procedure:

```
Given: Ir is the resulting intensity
       Ia is the ambient intensity
       Ka is the object-surface-color ambient reflectivity
         coefficient
       Ln is the direction vector of lightsource n
       N is the surface normal vector
       D is the dot product of Ln and N
       In is the intensity of lightsource n
       Kd is the object-surface-color diffuse reflectivity
         coefficient
Ir = IaKa
for each lightsource turned on in the current view:
   D = Ln dot N
   if D < 0 then D = 0
   Ir = Ir + In*Kd*D
if Ir > .FFFFh then Ir = .FFFFh (.FFFFh = $(2^{16}-1)/2^{16}$)
```

This intensity is then mapped through the object-surface-color-range onto the ERCI by this formula:

```
if color-smoothing is NONE
then ERCI = Ir * (max-index – min-index + 1)
   + min-index
else ERCI = Ir * (max-index - min-index) + min-index
```

The Local Lighting Model is applied as the segments containing facets are redrawn during a renew-view operation. This makes the segments independent of the lightsources, so that you can transform the segments and/or modify the lightsources and redisplay a segment to achieve a different appearance. For example, a segment of the globe of the earth could be rotated between renew-views, with the sun and moon supplying (possibly moving) lightsources and the stars providing a low ambient lightsource.

PIXELATION OF PRIMITIVES

To display the various graphic primitives, the terminal "pixelates" them and sends the resulting pixels to the z-buffer (or the frame buffer in wireframe mode). "Pixelate" means "to turn into pixels"; a "pixel" (short for "picture element") is a point with an index. The z-buffer and frame buffer are, from this point of view, just arrays of pixels.

Vectors are pixelated by setting the indices of the pixels closest to the center line of the vector to the current line- or gap-index. Markers and alphatext have 8x16 dot matrices stored in memory for each different marker type and character; they are pixelated by copying the appropriate dot matrix with the proper current indices.

Facet boundaries are pixelated as vectors (without dash patterns), while the interiors of facets are pixelated using Gouraud Shading (see below).

GOURAUD SHADING

The 4129 terminal uses the linear interpolation part of the Gouraud Shading algorithm to determine the ERCI of each pixel of a facet, based on the vertices and ERCI's of a given triangle. The given triangle is derived from each facet, as described in the DRAW-FACETS command description in this manual.

Linear interpolation is used to determine the ERCI of each pixel on the boundary of the triangle. Then, as each row of pixels of the facet is scanned, each pixel's ERCI is determined by linear interpolation between the triangle boundary points.

Linear interpolation means that the ratio of "distances" between ERCI's is the same as the ratio of distances between pixels. In other words, given a pixel at X, a row of pixels starting at M and ending at N, with ERCI's Ix, Im and In at those pixels, the ratios of pixel and index "distances" are equal and are:

(In–Ix) / (In–Im) = (N–X)/(N–M)

From this equation, Ix is computed as:

Ix = In – (In–Im)∗(N–X) / (N–M)

Gouraud Shading is always applied, whether the ERCI's of the given triangle are different or all the same. If they are the same, the result is that all pixels have the same ERCI (constant shading). If they are different, the result is that intermediate pixels have intermediate ERCI's (smooth shading).

One undesirable result of Gouraud Shading is called "Mach banding", where the edges of adjacent triangles seem to be brighter than the centers of the triangles. This is actually due to the human eye's response to color changes. One way to reduce Mach banding is to use dithering; another is to use more facets to specify the surface.

Another undesirable result of Gouraud Shading is "color banding", where the resulting image has distinct bands of each color index resulting from the linear interpolation. To smooth the color bands out, enable halftoning or dithering, described below.

This manual uses the term "Gouraud Shading" to mean only linear interpolation between indices to achieve both constant and smooth shading. This usage of "Gouraud Shading" does not include the derivation of normals, the averaging of normals, or the calculation of vertex intensities, as described in some literature.

Z-BUFFER

The z-buffer consists of an array of 24-bit words which are divided between z values and intensity values, depending on the color-smoothing view display attribute, as follows:

Table G-2

Z-BUFFER BITS

Z Bits	Intensity Bits	Color Smoothing
16	8	1 no dithering or halftoning
14	10	2 2x2 dithering
14	10	3 halftoning with 2x2 dithering
12	12	4 4x4 dithering
12	12	5 halftoning with 4x4 dithering

When a view is renewed, the z-buffer is first initialized with all z bits set to one and all intensities values set to the wipe index for the view. Then as each pixel is received by the z-buffer (with x, y, z, and ERCI information for each pixel), if the new z value is less than the current z value, the old z is replaced by the new and the old intensity is replaced by the top 8, 10 or 12 bits of the new ERCI. If the new z value is greater than or equal to the old, the old values are left in the z-buffer.

This process removes pixels that are "behind" other pixels, so that all hidden surfaces and other hidden primitives are suppressed.

When the pixels are moved from the z-buffer, the 12, 10, or 8 bits of the intensity values are shifted into the high bits of a 12-bit ERCI for halftoning and dithering, with the remaining low 0, 2, or 4 bits set to 0.

Z-BUFFER TO FRAME-BUFFER

The ERCI bits in the z-buffer are sent to the bit planes in the frame buffer assigned to the current view's surface. Since 0 to 8 bit planes can be assigned to a view's surface, only the top bits from the z-buffer are actually sent. As each pixel is sent, it may be halftoned and/or dithered, as described below.

HALFTONING

Since halftoning involves expanding pixels by two in both directions, the first thing the terminal does when renewing a view with halftoning enabled is to reduce both the pixel width and height of the view's viewport by half. Then it pixelates the segments' primitives, applies Gouraud Shading, and sends the results to the z-buffer.

Then, as the intensities in the z-buffer are sent to the frame buffer, each z-buffer pixel is expanded into four frame buffer pixels in a 2x2 pattern. This makes the resulting image fill the view's viewport. If the viewport was an odd number of pixels high or wide, the extra row or column is not displayed.

The effect of halftoning is to speed up the renew-view significantly, due to the one-quarter effort needed for the Gouraud Shading and z-buffering. However, all vectors are doubled in thickness, small-size text is rendered illegible, and the apparent resolution of the display is reduced.

Note that when halftoning is enabled, either 2x2 dithering or 4x4 dithering is also enabled.

DITHERING

The whole point of using 12-bit ERCI's while the terminal has at most 8 bit planes per surface is to provide "fraction" bits for use in dithering. The dithering algorithm decides which of two adjacent color map indices to map an ERCI onto, based on the ERCI's fraction bits and the location of the pixel. The "fraction" bits are the four bits just below the "bit plane bits" in the ERCI. For example, for a surface with 5 bit planes, the high-order five bits of the ERCI are the "bit plane bits" and represent one of the two possible color map indices; the other possible color map index is the one immediately higher. The next four lower bits are the "fraction" bits, while the bottom three bits are stripped and ignored.

The terminal dithers each pixel transferred from the z-buffer to the frame buffer by these steps:

1. It shifts the ERCI right by 8 minus the number of bit planes on the view surface. (Shifting a number right one bit is the same as dividing it by two and ignoring the remainder.)

2. It adds a dither value to the ERCI. The dither value comes from the dither table, below, depending on the low 2 bits of the pixel's x and y. For color smoothing mode 1 (none), this step is skipped.

3. It shifts the ERCI right 4 bits (dividing it by 16) and sends it to the bit planes in the frame buffer assigned to the view surface.

The following table gives the dither values for the color smoothing modes:

Table G-3

DITHER TABLE

Binary Values					Decimal Values				
11	1111	0111	1101	0101	3	15	7	13	5
10	0011	1011	0001	1001	2	3	11	1	9
01	1100	0100	1110	0110	1	12	4	14	6
00	0000	1000	0010	1010	0	0	8	2	10
Y +					Y +				
X	00	01	10	11	X	0	1	2	3

Note that ERCI's generated for vectors, markers, and alphatext always have zeros in the bottom four bits, so the result of dithering is always just the input index. Note also that when color smoothing 1 (none) is used, the bottom 4 bits of the ERCI are set to zeros by the z-buffer bit shifting actions. Also, when color smoothing 2 or 3 is enabled, the bottom 2 bits are always zeros, with a somewhat coarser dithering result than when color smoothing 4 or 5 is enabled.

GLOSSARY OF SHADING TERMS

ambient light — non-directional light applied to all object surfaces.

color banding — parallel bars of color

cosine shading — the ERCIs used in shading are derived by the Local Lighting Model

diffuse light — directional light applied to object surfaces via Lambert's cosine law

dithering — a way to reduce the effects of color banding

ERCI (Extended Range Color Index) — a 12-bit color index

facet — a 3D graphic primitive made up of triangles and capable of being shaded

Gouraud Shading — a smooth shading technique using linear interpolation

halftoning — a way to speed up shading at the cost of lower resolution

Host Lighting Model — the application program technique of deriving ERCI's

object-surface — one or more facets

object-surface-color — an ERCI or a pointer to a range of colors

z-buffer — a 24-bit-deep frame buffer for intensities and z values