



**PLEASE CHECK FOR CHANGE INFORMATION
AT THE REAR OF THIS MANUAL.**

**4041R01
Graphics ROM**

INSTRUCTION MANUAL

**Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077**


Serial Number _____

070-4440-00
Product Group 76

First Printing OCT 1982
Revised MAY 1985

Copyright © 1982 Tektronix, Inc. All rights reserved.
Contents of this publication may not be reproduced in any
form without the written permission of Tektronix, Inc.

Products of Tektronix, Inc. and its subsidiaries are covered
by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are
registered trademarks of Tektronix, Inc. TELEQUIPMENT
is a registered trademark of Tektronix U.K. Limited.

Printed in U.S.A. Specification and price change privileges
are reserved.

INSTRUMENT SERIAL NUMBERS

Each instrument has a serial number on a panel insert, tag,
or stamped on the chassis. The first number or letter
designates the country of manufacture. The last five digits
of the serial number are assigned sequentially and are
unique to each instrument. Those manufactured in the
United States have six unique digits. The country of
manufacture is identified as follows:

B000000	Tektronix, Inc., Beaverton, Oregon, USA
100000	Tektronix Guernsey, Ltd., Channel Islands
200000	Tektronix United Kingdom, Ltd., London
300000	Sony/Tektronix, Japan
700000	Tektronix Holland, NV, Heerenveen, The Netherlands

MANUAL REVISION STATUS

PRODUCT: 4041R01 Graphic ROM

This manual supports the following versions of this product: Version 1.0

REV DATE	DESCRIPTION
10/82	Original Issue

CONTENTS

Page

Section 1	GENERAL DESCRIPTION	
	Introduction	1-1
	Graphics Concepts	1-2
	User Data Units	1-2
	Graphic Display Units	1-2
	Window	1-3
	Viewport	1-3
	Current Position	1-3
	Move and Draw	1-3
	Rmove and Rdraw	1-4
	Graphic Input	1-4
	Specifications	1-5
	Installation and Removal	1-6
Section 2	REPLACEABLE PARTS	
Section 3	SYSTEM ENVIRONMENTAL ROMCALLS	
	Introduction	3-1
	The GDEVICE Romcall	3-2
	The GINIT Romcall	3-3
Section 4	GRAPHIC ENVIRONMENTAL ROMCALLS	
	Introduction	4-1
	The COLOR and ASK COLOR Romcalls	4-2
	The LINSTYLE and ASK LINSTYLE Romcalls	4-3
	The TEXTSIZE and ASK TEXTSIZE Romcalls	4-4
	The VIEWPORT and ASK VIEWPORT Romcalls	4-6
	The WINDOW and ASK WINDOW Romcalls	4-10
Section 5	GRAPHIC PRIMITIVES	
	Introduction	5-1
	The DRAW Romcall	5-2
	The GIN Romcall	5-5
	The GTEXT Romcall	5-6
	The HARDCOPY Romcall	5-7
	The HOME Romcall	5-8
	The MOVE Romcall	5-9
	The PAGE Romcall	5-10
	The POINTER Romcall	5-11
	The RDRAW Romcall	5-13
	The RMOVE Romcall	5-16
Appendix A	ERROR MESSAGES	
Appendix B	INITIALIZATION ACTIONS TAKEN IN GINIT	
Index		

ILLUSTRATIONS

Fig. No.		Page
1-1	Cartesian Coordinate System for Tektronix Graphic Devices	1-3
1-2	Location of Rom Carrier	1-6
1-3	Rompack and Rom Carrier	1-7
4-1	Specifying a Portion of the Drawing Surface as the Viewport	4-7
4-2	Defining a Portion of the Viewport Off the Physical Drawing Surface	4-8
4-3	Inverting a Graph by Means of the VIEWPORT Romcall	4-9
4-4	Mapping a Window Onto a Viewport	4-11
4-5	Clipping Vectors at Window Boundaries	4-12
4-6	Inverting a Graph by Means of the WINDOW Romcall	4-13
5-1	Use of RCALL "DRAW"	5-3
5-2	Output from RCALL "DRAW" of Array	5-4
5-3	Use of RCALL "MOVE"	5-9
5-4	Use of RCALL "RDRAW"	5-13
5-5	Output from RCALL "RDRAW" of Array	5-14
5-6	Use of RCALL "RMOVE"	5-17

TABLES

Table No.		Page
3-1	Devices and Options Supported by 4041 Graphics Rom	3-4
4-1	Color Indices for Tektronix Devices	4-2
4-2	Linestyles Generated by RCALL "LINESTYLE"	4-3
4-3	Character Heights Available with Tektronix Terminals	4-4
5-1	One-Dimensional Array DRAW	5-3
5-2	Two-Dimensional Array DRAW	5-4
5-3	One-Dimensional Array RDRAW	5-15
5-4	Two-Dimensional Array RDRAW	5-15

Section 1

GENERAL DESCRIPTION

INTRODUCTION

The 4041R01 Graphics Rompack gives the 4041 the capability to generate graphics with any device that uses TEKTRONIX 4010-, 4025-, or 4662- compatible graphics codes.

Included in the R01 Graphics Rompack are system environmental romcalls, graphic environmental romcalls, and graphic primitive romcalls.

System environmental romcalls set up the 4041 to communicate with a graphic device. The system environmental romcalls are:

- GINIT: used to set up the 4041 to communicate with Tektronix graphic devices.
- GDEVICE: used to set up the 4041 to communicate with other graphic devices.

Graphic environmental romcalls define the graphic environment. The graphic environmental romcalls are:

- COLOR: sets a color or gray index (terminals) or chooses which pen will be used for drawing (plotters).
- LIFESTYLE: selects the style of line to be drawn.
- TEXTSIZE: selects the maximum size of text characters.
- VIEWPORT: selects the physical area of the graphic device on which to draw.
- WINDOW: selects the "window" in user data space to map onto the viewport.
- ASK COLOR: returns the latest requested and current color/gray index or pen index.
- ASK LIFESTYLE: returns the current linestyle index.
- ASK TEXTSIZE: returns the latest requested and current character sizes.

- ASK VIEWPORT: returns the current viewport parameters.
- ASK WINDOW: returns the current window parameters.

Graphic primitive romcalls perform the basic operations required to draw a picture or accept input from a graphic device. The graphic primitive romcalls are:

- DRAW: draws a line from the current position to a specified point.
- GIN: records the current position of the terminal cursor or plotter pen.
- GTEXT: writes characters into the graphic area.
- HARDCOPY: issues a COPY command to an attached Hard Copy unit.
- HOME: moves the cursor or plotter pen to the "home" position (one character's height below the upper-left hand corner of the graphic device).
- MOVE: moves the cursor or plotter pen to a specified point, without drawing a line.
- PAGE: moves the cursor or pen to the "HOME" position. On terminals, clears the screen. On plotters with paper advance, advances the paper.
- POINTER: allows the user to move the cursor or plotter pen in order to input its position.
- RDRAW: draws a line from the current position to a new position, whose coordinates are given relative to the current position.
- REMOVE: moves the cursor or plotter pen to a new position, whose coordinates are given relative to the current position.

GRAPHICS CONCEPTS

The following is intended as an overview of the terms used in creating graphics programs for the 4041. More detailed explanations, illustrations, and examples are found in Sections 3 through 5.

User Data Units

The term "user data units" refers to the units of measure in which the user defines a particular graphing application. Part of the beauty of the Tektronix graphics system is that it allows you to work in your own units of measure.

Suppose, for example, that you were plotting voltage vs. time. In this case, the units of measure might be volts (along the y-axis) and seconds (along the x-axis). If the data warranted, you might prefer the units of measure to be millivolts vs. milliseconds, or microvolts vs. nanoseconds, or any combination of the above.

Whatever the application, the Graphics rompack allows you to specify the units of measure you want to work with, and "talk" to the 4041 in those units. The rompack automatically converts data from your data units to graphic data units for plotting.

Graphic Display Units

A graphic display unit (GDU) is an internal unit defined as 1/100 of the shortest axis on the drawing surface. Internally, the Graphics Rompack converts all coordinate values specified in user data units to graphic display units before addressing points on the screen or the graphic surface of an external peripheral device.

The physical length of a GDU is device dependent. This means that the GDUs on one graphic device may be physically longer or shorter than the GDUs on another graphic device.

The graphic display unit concept frees graphic data from being device dependent. It provides automatic scaling, which allows the same set of data points to be plotted on any graphic device without changing the numeric values to conform to the physical dimensions of the plotting surface.

A Word About Resolution

Since the coordinates of a data point can be specified using any decimal value, the GDU concept allows for practically infinite screen resolution. For example, the data point with the coordinates (2.0003675,4.630087) can be addressed, if so desired. In practice, however, the hardware limitations of a graphic device determine the actual resolution.

The Cartesian Coordinate System

In order to understand computer graphics, an understanding of the Cartesian Coordinate System is essential, so let's review it. The Cartesian Coordinate System provides a way to locate any point on a two dimensional plane. The plane is divided into four quadrants by a horizontal and vertical axis.

The horizontal axis is called the X axis and the vertical axis is called the Y axis. The point at which the axes cross is called the point of origin. The position of each point on the plane is located by specifying two coordinates (X,Y). X represents the point's horizontal distance from the point of origin and Y represents the point's vertical distance from the point of origin.

All graphic surfaces represent a cartesian coordinate plane. The location of a point on that plane is specified by its X coordinate value and its Y coordinate value.

How Graphic Devices Fit into a Cartesian Coordinate System

Internally, the drawing surface of any graphic device represents the first quadrant in a cartesian coordinate system (Fig. 1-1).

The lower-left corner of the graphic device (display) represents the point of origin. The vertical axis (Y) runs along the left side of the display and is marked off in Graphic Display Units. The horizontal axis runs along the bottom of the display and is also marked off in Graphic Display Units. The physical size of the drawing surface covers an area 100 GDUs high by 130 GDUs wide. Points on the drawing surface are specified by their X and Y coordinates. In Figure 1-1, the point (65,50) denotes the center of the screen; the point (0,100) denotes the upper-left corner; the point (130,100) denotes the upper-right corner; the point (130,0) denotes the lower-right corner; and the point (0,0) is the lower-left corner (the point of origin). Specifying the drawing surface boundaries with the VIEWPORT romcall is based on this concept.

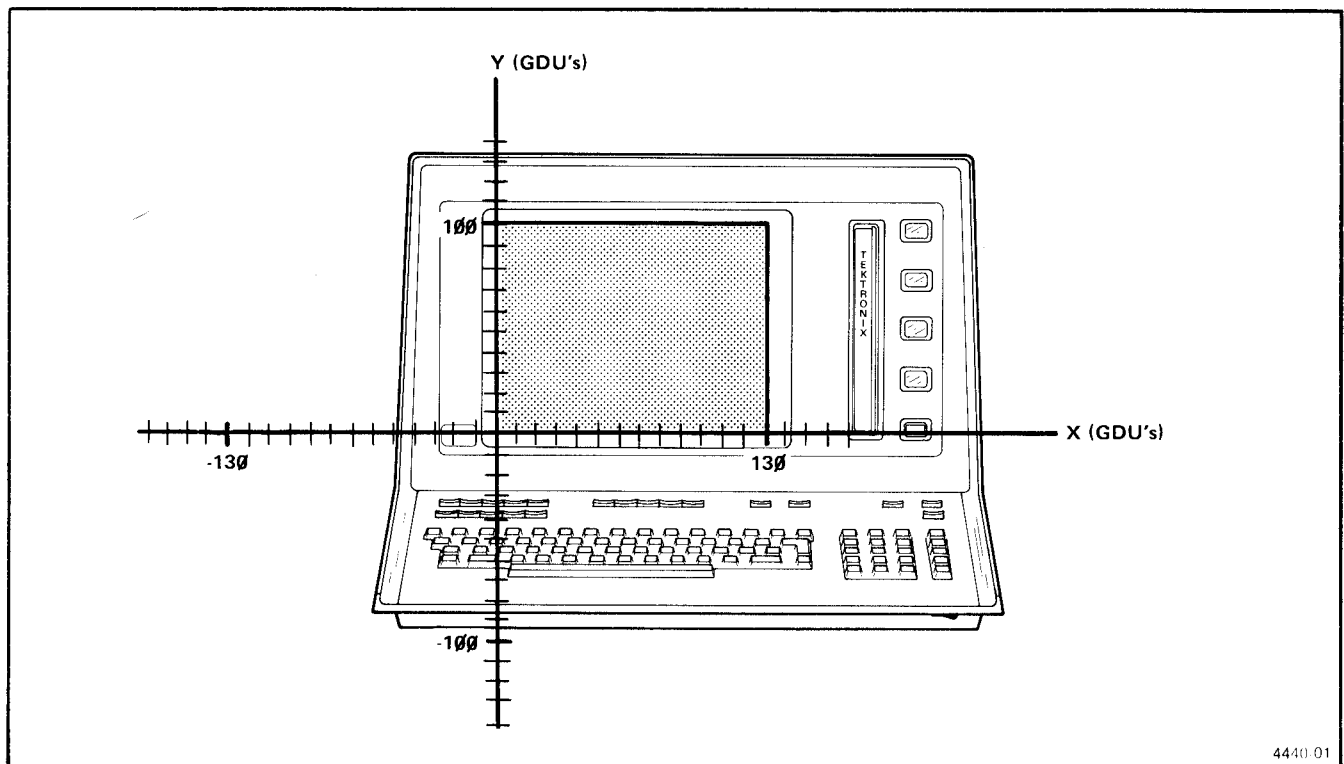


Figure 1-1. Cartesian coordinate System for Tektronix Graphic Devices.

Window

The window can be thought of as a rectangle on the user data plane. The WINDOW romcall defines the range that user data values may take along both the X- and Y-axes. More importantly, the window specifies what portion of the user's data space fits onto the viewport, and allows the coordinates of different points in that space to be specified in user data units (inches, feet, miles, dollars, pounds, volts, seconds, etc.).

Viewport

The viewport is the image of the window upon the drawing surface. The VIEWPORT romcall establishes the boundaries of the drawing surface on the graphic device. This surface can be any rectangular shape, and can be located anywhere on (i.e., consist of any portion of) the graphic device's surface. Parameters for the VIEWPORT romcall are given in graphic display units.

Current Position

The current position is the location of the terminal cursor or plotter pen in user data units.

MOVE and DRAW

The MOVE romcall establishes a new current position. In effect, the MOVE romcall moves the cursor or pen to a given position without drawing a line.

The DRAW romcall draws a line from the current position to a specified position, and makes that the new current position.

In both romcalls, the user specifies an X- and a Y-coordinate to move/draw to. Arguments for the MOVE and DRAW romcalls are given in user data units.

Thus, if the user wishes to draw a line from the point (100 volts, 2 seconds) to the point (350 volts, 3 seconds), the user enters the statements

```
Recall "move",100,2
Recall "draw",350,3
```

and the 4041 automatically translates the arguments into the appropriate coordinates in graphic display units. These coordinates are then transmitted to the graphic device to execute the moves and draws.

RMOVE and RDRAW

Where the MOVE and DRAW romcalls use the absolute coordinates of the destination, the RMOVE and RDRAW romcalls use the coordinates of the destination relative to the current position. Thus, the DRAW romcall in the previous example could have been replaced by an RDRAW romcall, as follows:

```
Recall "move",100,2  
Recall "rdraw",250,1
```

The RDRAW romcall draws a line 250 user data units to the right and 1 user data unit above the current position, i.e., to the point (350,3).

Graphic Input

The GIN and POINTER romcalls input the current coordinates of the graphic cursor or plotter pen. These romcalls effectively allow the user to communicate with the 4041 by means of the graphic device. The POINTER romcall allows the user to move the graphic cursor or plotter pen prior to inputting its coordinates; the GIN romcall simply inputs the coordinates at the current location.

SPECIFICATIONS

Power Requirements

The power requirements for rompacks are included in the base instrument power requirements for the 4041. These requirements are:

Input Power: 120 Watts maximum

Output Power: 80 Watts maximum

Line Voltage Limits:

130V Range: 90-132 Volts

230V Range: 180-250 Volts

Line Frequency: 48-66 Hz

Line Fuse:

Low Range: 2.5A fast blow

High Range: 1.6A slow blow

Temperature

Operating: 32 to 130° F (0 to 55° C)

Non-Operating: -40 to 165° F (-40 to 75° C)

Altitude

Operating: 15,000 ft (4.5 km)

Non-Operating: 50,000 ft (15 km)

Humidity

Operating: 95% max

Non-Operating: 95% max at 150° F (65° C)

Static Immunity

Installed: 15 kV

Non-Installed: No immunity.

CAUTION

4041 Rom packs are static-sensitive when not installed. DO NOT TOUCH THE ROM LEADS ON THE UNDERSIDE OF THE ROM CARRIERS WITH AN INSTRUMENT OR FINGER — YOU MAY DAMAGE THE ROM PACK.

Vibration

Less than 0.025 inch (0.64 mm) p-p amplitude.

Shock

50 G's

Packaged Transportation

Meets NSTA requirements for packaged shock and vibration.

EMI

Meets FCC Part 15, Subpart J, Class A requirements, and VDE 0871, Class B requirements.

Physical Specifications (with latchbar)

Length: 3.5 in. (8.89 cm)

Width: 1.05 in. (2.67 cm)

Height: 0.35 in. (0.89 cm)

Weight: 0.564 oz. (16.1 g)

Memory Requirements

The Graphics rompack uses 1500 bytes of random access memory.

INSTALLATION AND REMOVAL

NOTE

When the 4041 is turned on, it automatically starts a series of self-tests. Part of this self-testing determines whether any rompacks are installed and checks that they are functioning correctly. Therefore, rompacks must be installed before turning on the 4041. The system does not recognize any rompacks unless they are correctly installed before the 4041 is turned on.

Removing the Rom Carrier

CAUTION

Turn the 4041 power off before removing the rom carrier. Never remove the rom carrier while the power is on. Removal with the power on can cause power fluctuations that can damage rom chips.

The rom carrier is a tray located behind the grill on the front of the 4041 (Figure 1-2).

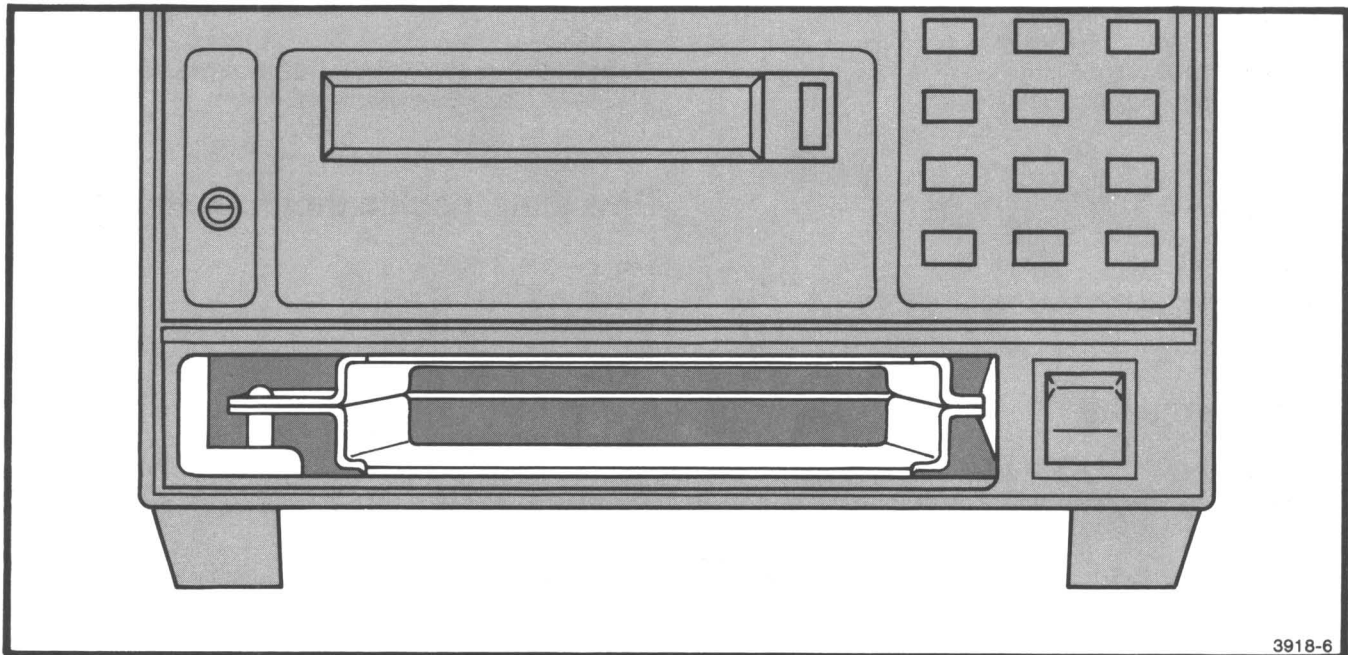
To remove the carrier (if it is already installed), first remove the grill. The grill has a horizontal coin slot at its top; pry out the grill using a coin in the cutout. Pull the carrier out of its compartment using the carrier's plastic strap.

Putting the Rompacks Into the Carrier

CAUTION

Do not touch the metal leads on the underside of the rom packs. The rompacks are static-sensitive and could be damaged by static charges from fingers or tools. Hold the rompacks by the plastic holders.

Figure 1-3 shows a rompack and the rom carrier. The individual rompacks are placed into the carrier, and the carrier is slid into the compartment of the 4041.



3918-6

Figure 1-2. Location of Rom Carrier.

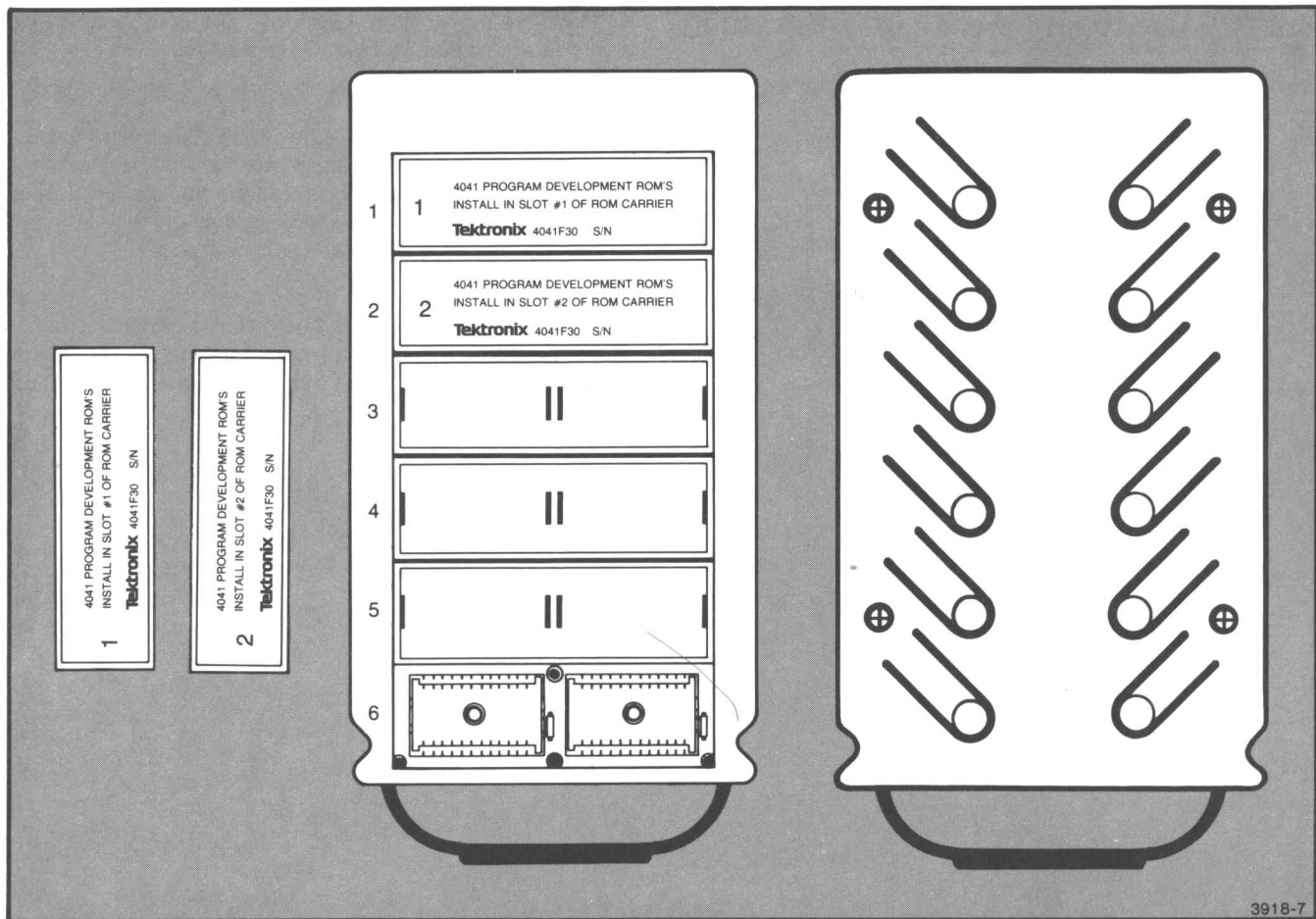


Figure 1-3. Rompack and Rom Carrier.

The rom carrier holds up to six rompacks. The individual rompacks specify a particular slot in the carrier where they must be placed. Match the numbers on the rompacks with the numbers on the carrier.

To insert rompacks into the rom carrier, place the rompack right side up over the desired slot and gently press the pack into position. The rompacks are keyed and will only go in one way. When properly inserted, the top of the rompack should be flush with the top of the rom carrier. Be gentle.

Removing Rompacks from the Rom Carrier

Figure 1-3 also shows the underside of the rom carrier. To remove rompacks from the carrier, turn the carrier over. Gently press the two indentations over the rompack that must be removed; the rompack will pop out of the carrier.

Putting the Rom Carrier Into the 4041

The rom carrier is placed in the compartment covered by the grill on the front panel of the 4041.



The 4041 power must be turned off before the rom carrier is installed. Inserting the rom carrier with the power on could cause power fluctuations that could damage the rom chips.

If the 4041's power is on, turn it off. Snap out the grill by prying at the coin slot at the top of the grill.

Slide the rom carrier into position in its compartment with the rom-pack-side up and the carrier handle last. Press the carrier firmly into place to securely seat the rom connectors. Make certain that the carrier is seated all the way in the compartment and is securely held in place.

Replace the front panel grill. The grill should easily snap back into place; if it does not, check that the rom carrier is inserted all the way. The grill's coin slot should be at the top.

Section 2

REPLACEABLE PARTS

PARTS ORDERING INFORMATION

Replacement parts are available from or through your local Tektronix, Inc. Field Office or representative.

Changes to Tektronix instruments are sometimes made to accommodate improved components as they become available, and to give you the benefit of the latest circuit improvements developed in our engineering department. It is therefore important, when ordering parts, to include the following information in your order: Part number, instrument type or number, serial number, and modification number if applicable.

If a part you have ordered has been replaced with a new or improved part, your local Tektronix, Inc. Field Office or representative will contact you concerning any change in part number.

Change information, if any, is located at the rear of this manual.

SPECIAL NOTES AND SYMBOLS

X000 Part first added at this serial number
 00X Part removed after this serial number

FIGURE AND INDEX NUMBERS

Items in this section are referenced by figure and index numbers to the illustrations.

INDENTATION SYSTEM

This mechanical parts list is indented to indicate item relationships. Following is an example of the indentation system used in the description column.

```

1 2 3 4 5           Name & Description
Assembly and/or Component
Attaching parts for Assembly and/or Component
    ---*---
Detail Part of Assembly and/or Component
Attaching parts for Detail Part
    ---*---
Parts of Detail Part
Attaching parts for Parts of Detail Part
    ---*---
  
```

Attaching Parts always appear in the same indentation as the item it mounts, while the detail parts are indented to the right. Indented items are part of, and included with, the next higher indentation. The separation symbol ---*--- indicates the end of attaching parts.

Attaching parts must be purchased separately, unless otherwise specified.

ITEM NAME

In the Parts List, an Item Name is separated from the description by a colon (:). Because of space limitations, an Item Name may sometimes appear as incomplete. For further Item Name identification, the U.S. Federal Cataloging Handbook H6-1 can be utilized where possible.

ABBREVIATIONS

"	INCH	ELCTRN	ELECTRON	IN	INCH	SE	SINGLE END
#	NUMBER SIZE	ELEC	ELECTRICAL	INCAND	INCANDESCENT	SECT	SECTION
ACTR	ACTUATOR	ELCTLT	ELECTROLYTIC	INSUL	INSULATOR	SEMICOND	SEMICONDUCTOR
ADPTR	ADAPTER	ELEM	ELEMENT	INTL	INTERNAL	SHLD	SHIELD
ALIGN	ALIGNMENT	EPL	ELECTRICAL PARTS LIST	LPHLDR	LAMPHOLDER	SHLDR	SHOULDERED
AL	ALUMINUM	EQPT	EQUIPMENT	MACH	MACHINE	SKT	SOCKET
ASSEM	ASSEMBLED	EXT	EXTERNAL	MECH	MECHANICAL	SL	SLIDE
ASSY	ASSEMBLY	FIL	FILLISTER HEAD	MTG	MOUNTING	SLFLKG	SELF-LOCKING
ATTEN	ATTENUATOR	FLEX	FLEXIBLE	NIP	NIPPLE	SLVG	SLEEVING
AWG	AMERICAN WIRE GAGE	FLH	FLAT HEAD	NON WIRE	NOT WIRE WOUND	SPR	SPRING
BD	BOARD	FLTR	FILTER	OBD	ORDER BY DESCRIPTION	SQ	SQUARE
BRKT	BRACKET	FR	FRAME or FRONT	OD	OUTSIDE DIAMETER	SST	STAINLESS STEEL
BRS	BRASS	FSTNR	FASTENER	OVH	OVAL HEAD	STL	STEEL
BRZ	BRONZE	FT	FOOT	PH BRZ	PHOSPHOR BRONZE	SW	SWITCH
BSHG	BUSHING	FXD	FIXED	PL	PLAIN or PLATE	T	TUBE
CAB	CABINET	GSKT	GASKET	PLSTC	PLASTIC	TERM	TERMINAL
CAP	CAPACITOR	HDL	HANDLE	PN	PART NUMBER	THD	THREAD
CER	CERAMIC	HEX	HEXAGON	PNH	PAN HEAD	THK	THICK
CHAS	CHASSIS	HEX HD	HEXAGONAL HEAD	PWR	POWER	TNSN	TENSION
CKT	CIRCUIT	HEX SOC	HEXAGONAL SOCKET	RCPT	RECEPTACLE	TPG	TAPPING
COMP	COMPOSITION	HLCPS	HELICAL COMPRESSION	RES	RESISTOR	TRH	TRUSS HEAD
CONN	CONNECTOR	HLEXT	HELICAL EXTENSION	RGD	RIGID	V	VOLTAGE
COV	COVER	HV	HIGH VOLTAGE	RLF	RELIEF	VAR	VARIABLE
CPLG	COUPLING	IC	INTEGRATED CIRCUIT	RTNR	RETAINER	W/	WITH
CRT	CATHODE RAY TUBE	ID	INSIDE DIAMETER	SCH	SOCKET HEAD	WSHR	WASHER
DEG	DEGREE	IDENT	IDENTIFICATION	SCOPE	OSCILLOSCOPE	XFMR	TRANSFORMER
DWR	DRAWER	IMPLR	IMPELLER	SCR	SCREW	XSTR	TRANSISTOR

REPLACEABLE PARTS

CROSS INDEX—MFR. CODE NUMBER TO MANUFACTURER

Mfr. Code	Manufacturer	Address	City, State, Zip
80009	TEKTRONIX, INC.	P O BOX 500	BEAVERTON, OR 97077

Fig. 1 EXPLODED VIEW

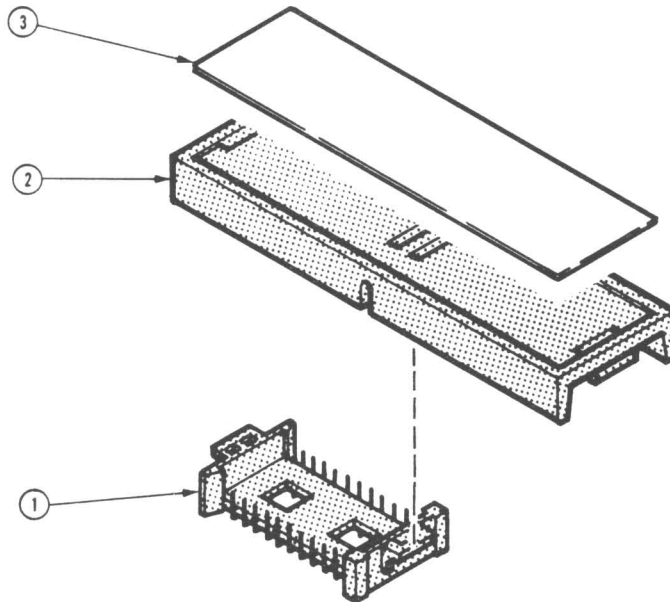


Fig. & Index No.	Tektronix Part No.	Serial/Model No. Eff	Dscont	Qty	1	2	3	4	5	Name & Description	Mfr Code	Mfr Part Number
1-1	119-1556-00	B010100	B039999	1						ROM CARRIER:W/ROM 160-1755-00	80009	119-1556-00
	119-1556-01	B040000		1						ROM CARRIER:W/ROM 160-1755-01	80009	119-1556-01
	-----			-						(U5;LOW NUMBERED SLOTS,DESIGNATED UXX)		
	119-1557-00	B010100	B039999	1						ROM CARRIER:W/ROM 160-1756-00	80009	119-1557-00
	119-1557-01	B040000		1						ROM CARRIER:W/ROM 160-1756-01	80009	119-1557-01
	-----			-						(U10;HIGH NUMBERED SLOTS,DESIGNATED UXXX)		
-2	105-0897-00			1						LCH BAR,CHIP CA:POLYPROPYLENE	80009	105-0897-00
-3	334-4915-00			1						MARKER,IDENT:MDK 4041 GRAPHIC ROM	80009	334-4915-00
										STANDARD ACCESSORIES		
	070-4440-00			1						MANUAL,TECH:INSTR,4041 GRAPHIC ROM PACK	80009	070-4440-00
	070-4441-00	B010100	B010112	1						CARD,INFO:REFERENCE,4041 R01	80009	070-4441-00
	070-4490-00	B010113		1						MANUAL,TECH:PRGM REF GUIDE,4041R01/4041R02	80009	070-4490-00

Section 3

SYSTEM ENVIRONMENTAL ROMCALLS

INTRODUCTION

System environmental romcalls set up the 4041 to communicate with a graphic device. The system environmental romcalls are:

— GDEVICE: used to set up the 4041 to communicate with non-Tektronix graphic devices.

— GINIT: used to set up the 4041 to communicate with Tektronix graphic devices.

The GDEVICE Romcall

Syntax Form: [Line-no.] RCALL "GDEVICE", numexp, numexp, numexp, numexp, numexp, numexp, numexp

Descriptive Form: [Line-no.] RCALL "GDEVICE", logical-unit, min-x, max-x, min-y, max-y, char-width, char-height

PURPOSE

The GDEVICE romcall initializes all the global variables in the graphics rompack, and informs the rompack of the size of the writing surface when a non-Tektronix graphics device is used for graphics.

EXPLANATION

The GDEVICE romcall performs the same functions as GINIT, except it performs them for non-Tektronix devices. A romcall to GINIT or GDEVICE must precede any other romcall to the graphics rompack. Devices connected to a GPIB interface port are sent GPIB 4662 plotter codes, while devices connected to an RS-232-C interface port are sent 4010-style codes.

All arguments in the GDEVICE romcall must be numeric scalar expressions.

The first argument is the logical unit you wish to do your graphics to. This logical unit must be opened before GDEVICE is called, and must be left open until all graphics operations are complete.

The next four arguments are values representing the minimum and maximum x- and y-coordinates of the graphics device in device-dependent data units. The "minimum" values must be less than their respective "maximum" values. The minimum value cannot be less than 0, and the maximum value cannot be greater than 4095.

The sixth and seventh arguments are scalar values representing the width and height, respectively, of characters in device space (i.e., on the scale defined by the x- and y-ranges just given). These arguments must evaluate to numbers greater than zero. The widths and heights specified include the actual width and height of the characters, plus any extra space required for readability.

GDEVICE creates a default window in the graphics device of 0 to 100 in the shorter axis and 0 to a proportional number in the longer axis. It creates a default viewport that represents the entire writing surface of the device. GDEVICE also makes the home position the current position.

EXAMPLE:

```
1500 Rcall "gdevice",1,0,1024,0,756,10,14
```

This statement designates the device associated with logical unit 1 as the graphics device. The minimum x-coordinate (in device space) of this device is 0, and the maximum x-coordinate is 1024. The minimum y-coordinate is 0, and the maximum y-coordinate is 756. Characters are 10 device units wide, and 14 device units high.

The GINIT Romcall

Syntax Form: [Line-no.] RCALL "GINIT", numexp, numexp, numexp

Descriptive Form: [Line-no.] RCALL "GINIT", logical-unit, model-no., option-no.

PURPOSE

The GINIT romcall initializes all the global variables in the graphics rompack.

EXPLANATION

The GINIT romcall is used with Tektronix graphics devices. To do graphics with non-Tektronix devices, use the GDEVICE romcall.

GINIT or GDEVICE must be called before calling any other romcall from the graphics rompack.

The first argument in the GINIT romcall is the logical unit you wish to send all your graphics to. This logical unit must be opened before GINIT is called, and must be left open until all graphics operations are complete.

The second argument is the model number of the graphics device you will be using. The graphics rompack supports the terminals listed in Table 3-1 as well as the GPIB versions of the 4662 and 4663 plotters.

The third argument specifies what options the graphics device includes. Option numbers are also listed in Table 3-1.

GINIT creates a default window in the graphics device of 0 to 100 in the shorter axis and 0 to a proportional number in the longer axis. It creates a default viewport that represents the entire writing surface of the device. GINIT also makes the home position the current position.

The GINIT romcall can also be used to reset the graphics environment (i.e., return to default window and viewport, set the current position to home, return graphics environmentals to default values, etc.).

EXAMPLE:

```
1000 Rcall "ginit",6,4025,2
```

This statement specifies the device associated with logical unit 6 as the graphics device, and identifies this device as a TEKTRONIX 4025 terminal with the graphic input modification.

Table 3-1
DEVICES AND OPTIONS SUPPORTED BY 4041 GRAPHICS ROM

Model Number	Option Number	Meaning	Default Viewport Dimensions
4006,4010,4012,4013	1	standard 4006,4010,4012,4013	133.4 × 100
4012,4013	2	4012,4013 with small character option	133.4 × 100
4014,4015	1	standard 4014,4015	133.4 × 100
4014,4015	2	4014,4015 with enhanced graphics module	133.4 × 100
4016	1	standard 4016	133.4 × 100
4016	2	4016 with 4014 character set	133.4 × 100
4025*	1	standard 4025	152.5 × 100
4025*	2	4025 with graphic input modification	152.5 × 100
4027*	1	standard 4027	152.5 × 100
4051,4052	1	standard 4051,4052 emulating 4012	133.4 × 100
4054	1	standard 4054 emulating 4014	133.4 × 100
4112,4113	1	standard 4112,4113	150.9 × 100
4114	1	standard 4114	133.4 × 100
4662	1	standard 4662	150.0 × 100
4662	2	4662 with 8-pen carousel	150.0 × 100
4663**	1	standard 4663	135.5 × 100
4663**	2	4663 with paper advance	135.5 × 100

* Note: The Graphics rompack uses the "back-quote" character (shift-backslash, ASCII 96) as the command character when the graphic device is a 4020-Series terminal. The command character on the terminal should be set to back-quote before the GINIT romcall is executed.

** Note: The 4663 must be equipped with Version 5 or higher firmware in order to run correctly with the 4041R01 Graphics Rompack.

Section 4

GRAPHIC ENVIRONMENTAL ROMCALLS

INTRODUCTION

Graphic environmental romcalls define the graphic environment. These romcalls include:

- COLOR: sets a color or gray index (for terminals) or chooses which pen will be used for drawing (plotters).
- ASK COLOR: returns the latest requested and assigned color/gray index or selected pen index.
- LINSTYLE: selects the style of line to be drawn.
- ASK LINSTYLE: returns the current linestyle index.
- TEXTSIZE: selects the maximum size of text characters.
- ASK TEXTSIZE: returns the latest requested and assigned character sizes.
- VIEWPORT: selects the physical area of the graphic device on which to draw.
- ASK VIEWPORT: returns the parameters of the current viewport.
- WINDOW: selects the "window" in user data space to map onto the viewport.
- ASK WINDOW: returns the parameters of the current window.

The COLOR Romcall

Syntax Form: [Line-no.] RCALL "COLOR" [, numexp]

Descriptive Form: [Line-no.] RCALL "COLOR" [, color-index]

The ASK COLOR Romcall

Syntax Form: [Line-no.] RCALL "ASK COLOR", {numvar} {numvar}
 {num-array} ' {num-array}

Descriptive Form: [Line-no.] RCALL "ASK COLOR", value-asked-for, actual-value

PURPOSE

The COLOR romcall specifies a color index or gray index to use in drawing.

The ASK COLOR romcall returns the latest color index value asked for and the current value of the color index.

EXPLANATION

If the graphics rompack was initialized with a GDEVICE romcall, the Graphics Rompack assumes the device does not have color capability.

If the graphics device is a Tektronix terminal, the COLOR romcall selects an index into a color map or gray map. If the graphics device is a Tektronix plotter, the COLOR romcall selects which pen to use. Table 4-1 lists the indices allowed for different Tektronix devices and option numbers.

If the color index is not in the range allowed by the device, or if the graphics device does not have color capability, then the COLOR romcall stores the color index asked for, but has no other effect. (In this case, an ASK COLOR romcall will show different values for value-asked-for and value-assigned.)

If COLOR is called with no argument, the color index reverts to its default value (1).

See "Caution Concerning ASK Romcalls" in the description of the VIEWPORT romcall (Section 4) for a note concerning the use of ASK COLOR.

Table 4-1
COLOR INDICES FOR TEKTRONIX DEVICES

Model Number	Option Number	Indices Allowed
4027	1	0-7
4112	1	0-7
4113	1	0-15
4662	2	1-8
4663	1-2	1-2

EXAMPLE

```
2100 Recall "color",3
```

This statement sets the color index to the value 3.

```
2200 Recall "ask color",asked,got
```

This statement stores the last color index value requested in numeric variable Asked, and stores the actual color index value in numeric variable Got.

The LINESTYLE Romcall

Syntax Form: [Line-no.] RCALL "LINESTYLE" [, numexp]

Descriptive Form: [Line-no.] RCALL "LINESTYLE" [, linestyle-no.]

The ASK LINESTYLE Romcall

Syntax Form: [Line-no.] RCALL "ASK LINESTYLE", {numvar}
 {num-array}

Descriptive Form: [Line-no.] RCALL "ASK LINESTYLE", current-linestyle-value

PURPOSE

The LINESTYLE romcall specifies the line pattern that the graphics rompack is to draw.

The ASK LINESTYLE romcall returns the current setting of the linestyle parameter.

EXPLANATION

If no argument is included in the LINESTYLE romcall, the linestyle reverts to its default value (solid line).

The linestyles generated by different argument values are listed in Table 4-2.

When the LINESTYLE romcall is executed, the graphics rompack checks to see if the device is capable of generating different linestyles. If so, it checks to see if the linestyle being asked for matches a linestyle in the device's repertoire. If it does, the graphics rompack sets the device's linestyle parameter to the requested linestyle.

Otherwise, the graphics rompack emulates the linestyle being asked for by executing a series of moves and draws on the graphics device. (This is a slower process, so it pays to use the device's linestyles whenever possible.)

If the graphics rompack was initialized by a GDEVICE romcall instead of a GINIT romcall, all linestyles are generated by sequences of moves and draws.

Table 4-2
LINESTYLES GENERATED BY RCALL "LINESTYLE"

Argument	Line Style
0	No line
1	Solid line
2	Dotted line (.....)
3	Dash-dot line (_ . _ . _)
4	Medium dashed line (_ _ _ _ _)
5	Long dashed line (_ _ _ _ _)
6	Long-dash-2-dots (_ _ _ _ _)

EXAMPLE

```
1800 Rcall "linestyle",2
```

This statements sets the graphics rompack to produce a dotted line.

```
1900 Rcall "linestyle"
```

This statement sets the graphics rompack to produce a solid line.

```
2000 Rcall "ask linestyle",cur_ls
```

This statement stores the current value of the linestyle parameter in numeric variable Cur_Ls.

The TEXTSIZE Romcall

Syntax Form: [Line-no.] RCALL "TEXTSIZE" [, numexp]
Descriptive Form: [Line-no.] RCALL "TEXTSIZE" [, max-char-height-in-GDU's]

The ASK TEXTSIZE Romcall

Syntax Form: [Line-no.] RCALL "ASK TEXTSIZE", {numvar} {numvar}
 {num-array} ' {num-array}'
Descriptive Form: [Line-no.] RCALL "ASK TEXTSIZE", height-asked-for, height-assigned

PURPOSE

The TEXTSIZE romcall sets the maximum height of characters on the graphics device in graphic data units.

The ASK TEXTSIZE romcall returns the character height asked for and the actual character height assigned in two numeric variables.

EXPLANATION

The argument in the TEXTSIZE romcall specifies the maximum height to be taken up by a character, including both the height of the character itself and any extra space required for readability.

If the graphics rompack was initialized with a GDEVICE romcall, the only character size available is the character size specified in the GDEVICE argument list.

Otherwise, if the graphics device has different character sizes available, TEXTSIZE selects the largest hardware font on the device that is smaller than or equal to the height specified in the TEXTSIZE romcall (in graphic data units). This font becomes the font used for characters written with the GTEXT romcall.

Table 4-3 lists the text sizes available with various Tektronix terminals. (Plotters have an "infinite" variety of text sizes, limited only by the physical dimensions of the plotter.) If no font is smaller than the height specified, TEXTSIZE selects the smallest font available.

Table 4-3
CHARACTER HEIGHTS AVAILABLE
WITH TEKTRONIX TERMINALS

Terminal	Textsizes
4006	Option 1: 2.86832
4010	Option 1: 2.86832
4012	Option 1: 2.86832
4012	Option 2: 1.62973, 2.86832
4013	Option 1: 2.86832
	Option 2: 1.62973, 2.86832
4014	Option 1: 1.56454, 1.72751, 2.67275, 2.86832
	Option 2: 1.56454, 1.72751, 2.67275, 2.86832
4015	Option 1: 1.56454, 1.72751, 2.67275, 2.86832
	Option 2: 1.56454, 1.72751, 2.67275, 2.86832
4016	Option 1: 1.1734, 1.56454, 2.67275, 2.86832
	Option 2: 1.56454, 1.72751, 2.67275, 2.86832
4025	Option 1: 3.34129
	Option 2: 3.34129
4027	Option 1: 3.34129
4051	Option 1: 2.86832
4052	Option 1: 2.86832
4054	Option 1: 1.56454, 1.72751, 2.67275, 2.86832
4105	Option 1: 3.3344
4112	Option 1: 3.30383
4113	Option 1: 3.30383
4114	Option 1: 1.56454, 1.72751, 2.67275, 2.86832

If TEXTSIZE is called with no argument, the textsize parameter reverts to the default font size, which is the one the device powered up with.

See "Caution Concerning ASK Romcalls" in the description of the VIEWPORT romcall (Section 4) for a note concerning the use of ASK TEXTSIZE.

EXAMPLE

```
2500 Recall "textsize",2.5
```

This statements sets the maximum height of graphic text characters to 2.5 graphic data units.

```
2600 Recall "ask textsize",asked,got
```

This statement stores the character height asked for by the last TEXTSIZE romcall in numeric variable Asked, and stores the actual character height assigned in numeric variable Got.

The VIEWPORT Romcall

Syntax Form: [Line-no.] RCALL "VIEWPORT"[, numexp, numexp, numexp, numexp]

Descriptive Form: [Line-no.] RCALL "VIEWPORT"[, minimum horizontal value in GDUs , maximum horizontal value in GDUs , minimum vertical value in GDUs , maximum vertical value in GDUs]

The ASK VIEWPORT Romcall

Syntax Form: [Line-no.] RCALL "ASK VIEWPORT",
 {numvar} {numvar} {numvar} {numvar}
 {numarray} ' {numarray} ' {numarray} ' {numarray}

Descriptive Form: [Line-no.] RCALL "ASK VIEWPORT",
 left-edge-value, right-edge-value, bottom-edge-value, top-edge-value

PURPOSE

The VIEWPORT romcall defines the boundaries of the drawing surface on the graphic device. The VIEWPORT parameters are automatically set to device-dependent values upon execution of a "GINIT" romcall.

The ASK VIEWPORT romcall returns the current values of the viewport parameters.

EXPLANATION

Within the 4041, the graphic device represents the first quadrant in a cartesian coordinate system. Both axes are marked in graphic display units (GDUs).

The VIEWPORT romcall controls the boundaries of the drawing surface. These boundaries are specified as follows: minimum horizontal (X) value, maximum horizontal (X) value, minimum vertical (Y) value, maximum vertical (Y) value.

The VIEWPORT parameters are automatically set to device-dependent default values after execution of a GINIT romcall. Executing a VIEWPORT romcall with no parameters specified also returns the VIEWPORT parameters to their default values. The default values represent the entire writing surface.

Specifying a Smaller Viewport

Sometimes it's necessary to reduce the size of the drawing surface to leave room for printed messages. This is done by changing the VIEWPORT parameters.

Example

```
2700 Rcall "viewport",50,100,50,100
```

In Figure 4-1, the viewport occupies an area in the center of the upper half of the drawing surface.

This VIEWPORT romcall reduces the drawing surface to the area shown by the shaded portion of the figure. Graphic output can be displayed in this area. All graphic output is automatically scaled to fit this area. The viewport can be changed as many times as desired throughout the course of a program. This allows the graphic rompack to draw several plots in different areas on the screen.

The viewport can be defined to be any shape or size. Whatever the size, all graphic output occurs within the defined area. A zero width or height is not allowed.

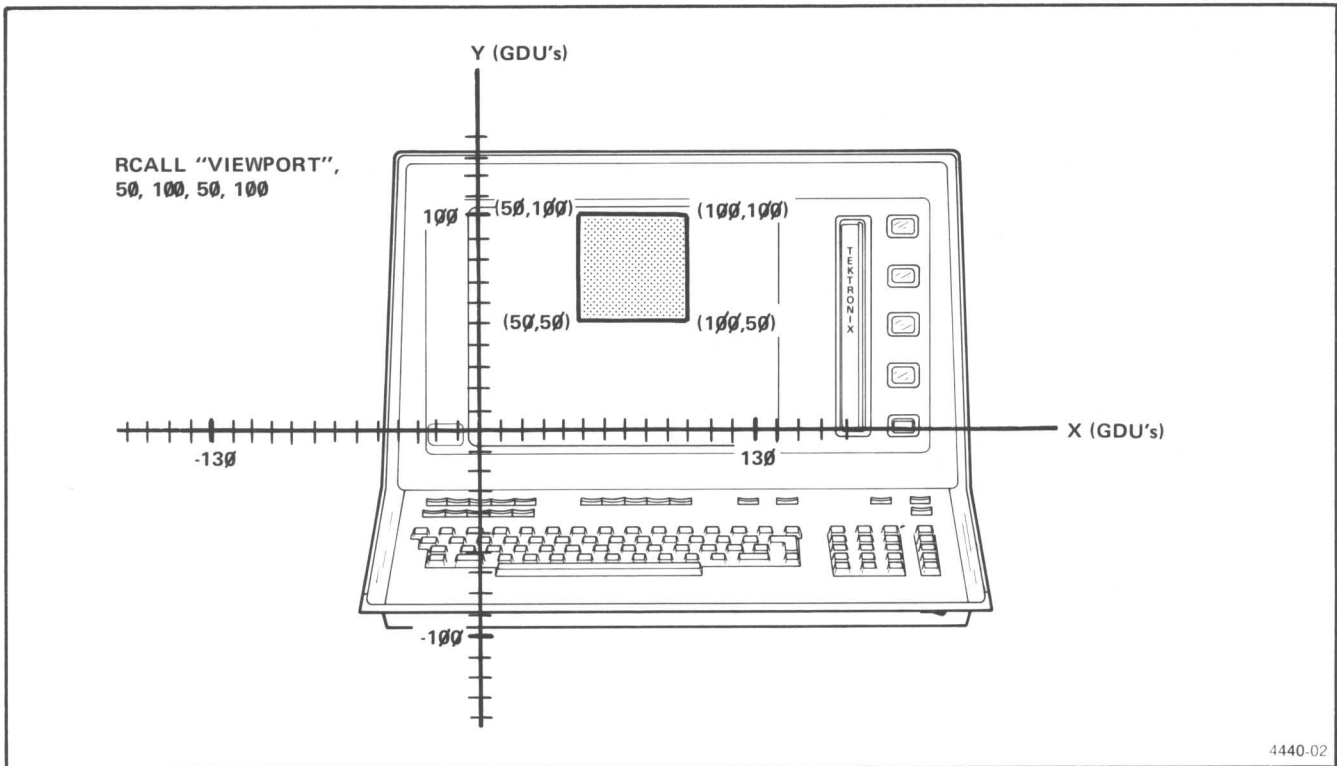


Figure 4-1. Specifying a Portion of the Drawing Surface as the Viewport.

Specifying a Viewport Larger than the Graphic Device

Specifying a viewport larger than the size of the graphics device can be done; however, it's not recommended. Figure 4-2 illustrates what happens when a portion of the viewport is defined outside the physical drawing surface.

Example

```
2800 Rcall "viewport",-65,65,-50,50
```

Inversion

The VIEWPORT romcall can be used to "invert" data, i.e., make data values decrease from left to right or from bottom to top. However, it is usually less confusing to do this with the WINDOW romcall. See the description of the WINDOW romcall for more information about inversion.

Example

```
1990 Rcall "window",0,20,0,100
2000 Rcall "viewport",130,0,100,0
```

These statements set up the following mappings:

UDUs	GDUs	
0	130	(right-side of the viewport)
20	0	(left-side of the viewport)
0	100	(top of the viewport)
100	0	(bottom of the viewport)

These mappings are illustrated in Figure 4-3.

Caution Concerning ASK Romcalls

The 4041 evaluates list elements for assignments from right-to-left. This can cause confusion when assigning values to arrays within a list.

For example, when the following statements are executed

```
1000 Dim x(4)
1010 Rcall "ask viewport",x,x(2),x(3),x(4)
```

each element of array X contains the value assigned to the left edge of the viewport.

This caveat applies to all the "ask" romcalls in the Graphics Rompack that assign values to more than one element. These include "ask viewport", "ask window", "ask color", and "ask textsize".

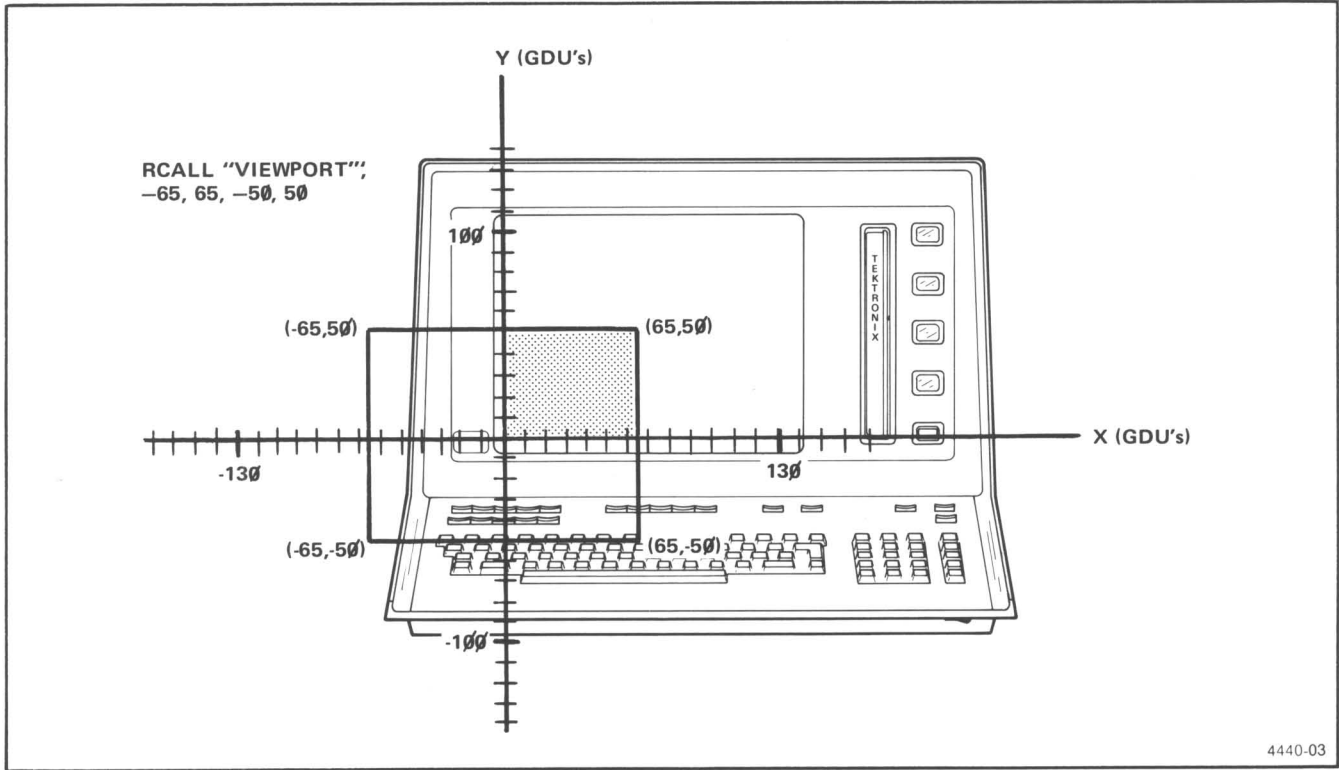


Figure 4-2. Defining a Portion of the Viewport Off the Physical Drawing Surface.

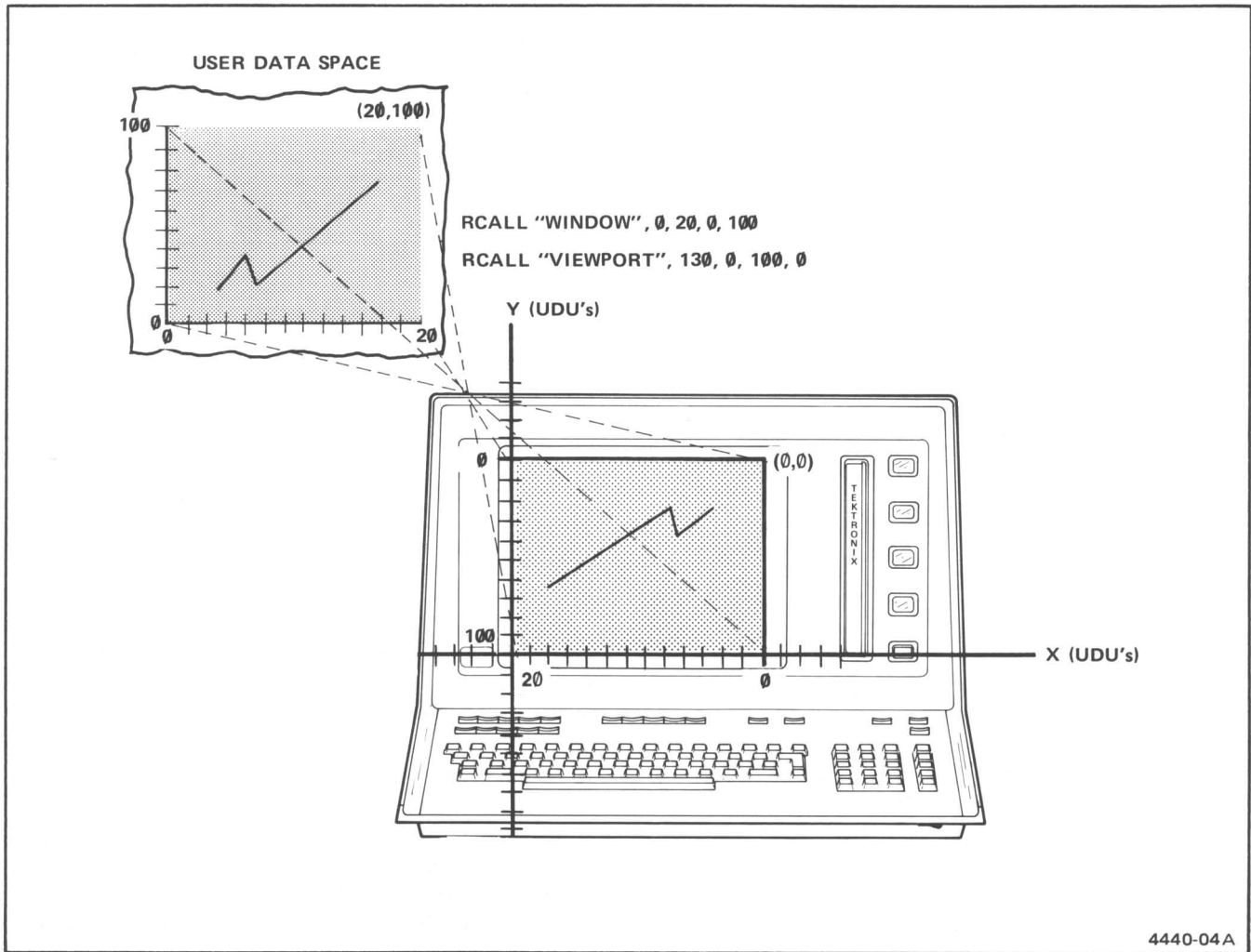


Figure 4-3. Inverting a Graph by Means of the VIEWPORT Romcall.

The WINDOW Statement

Syntax Form: [Line no.] RCALL "WINDOW"[, numexp, numexp, numexp, numexp]

Descriptive Form: [Line no.] RCALL "WINDOW"[, minimum horizontal (X) value in user data units, maximum horizontal (X) value in user data units, minimum vertical (Y) value in user data units, maximum vertical (Y) value in user data units]

The ASK WINDOW Romcall

Syntax Form: [Line-no.] "ASK WINDOW", {numvar} {numvar} {numvar} {numvar}
{numarray} ' {numarray} ' {numarray} ' {numarray}

Descriptive Form: [Line-no.] RCALL "ASK WINDOW", left-edge-value, right-edge-value, bottom-edge-value, top-edge-value

PURPOSE

The WINDOW romcall specifies how many user data units fit inside the viewport. The user data units can be any units of measure you wish to work with (inches, miles, dollars, years, etc.) The WINDOW parameters are automatically set to device-dependent default values after execution of a GINIT romcall.

The ASK WINDOW romcall returns the current values of the WINDOW parameters.

EXPLANATION

The "window" is defined as the image of the user data space cast onto the viewport.

The default values for the WINDOW parameters are the same as for the VIEWPORT parameters. The WINDOW parameters are set to their default values when a GINIT or GDEVICE romcall is executed.

Executing a WINDOW romcall with no parameters specified returns the WINDOW parameters to their default values.

See "Caution Concerning ASK Romcalls" in the description of the VIEWPORT romcall for a note concerning the use of ASK WINDOW.

In Figure 4-4 a portion of a sales graph is mapped into a viewport that is defined in the upper-right corner of a display terminal screen. The statements setting up the WINDOW and VIEWPORT are as follows:

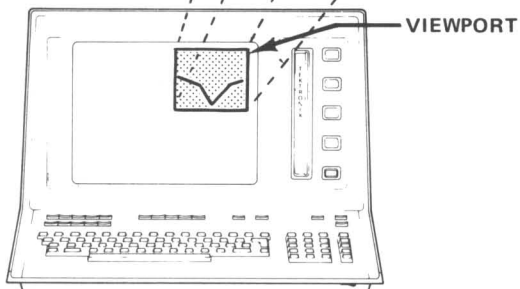
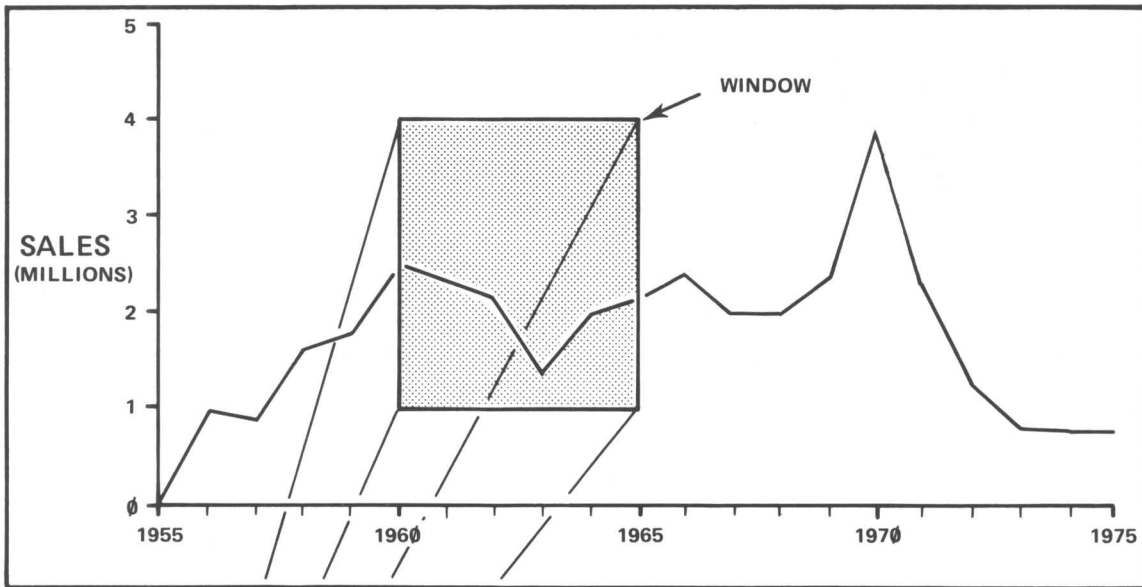
```
2010 Rcall "viewport",70,120,60,100  
2020 Rcall "window",1960,1965,1,4
```

Notice that the viewport parameters are specified in graphic display units. This defines the physical size and location of the drawing area. The viewport is then marked up in user data units with the WINDOW romcall. In this case, the width of the viewport starts at the year 1960 and ends in the year 1965. The vertical axis of the viewport is labeled in millions of dollars starting at 1 million and extending to 4 million.

Now, all you have to do is specify the year and the amount of sales dollars in a MOVE or DRAW romcall. For example, to draw a portion of the graph, you might specify the following:

```
2030 Rcall "draw",1963,1.4
```

This statement draws a line from the current position to the point (1963, 1.4) in user data units.



RCALL "WINDOW"
1960, 1965, 1, 4

RCALL "VIEWPORT",
70, 120, 60, 100

4440-05

Figure 4-4. Mapping a Window Onto a Viewport.

Clipping

If the parameters of a MOVE, RMOVE, DRAW, or RDRAW romcall specify a destination point outside the window, the Graphics Rompack executes a "theoretical" move or draw to that point. If a DRAW is executed, then only that portion of the vector that lies inside the window is drawn. The portion of the vector lying outside the window is clipped (chopped off) at the edge of the viewport. Figure 4-5 illustrates clipping action.

In this example, the viewport is defined as an area in the middle of the screen and the window covers a section of the sales graph on the right. The appropriate statements to set this up are:

```
2040 Rcall "viewport",20,100,20,80
2050 Rcall "window",1967,1972,1,3
2060 Rcall "move",1967,2
2070 Rcall "draw",1968,2
2080 Rcall "draw",1969,2.75
2090 Rcall "draw",1970,4
2100 Rcall "draw",1971,2.5
2110 Rcall "draw",1972,1.33
```

Because the vertical axis of the window only extends to 3 million dollars, the sales figure for 1970 is clipped. Notice that the Graphics Rompack goes through the motions of drawing the complete graph even though the portion of the graph lying outside the window is not drawn.

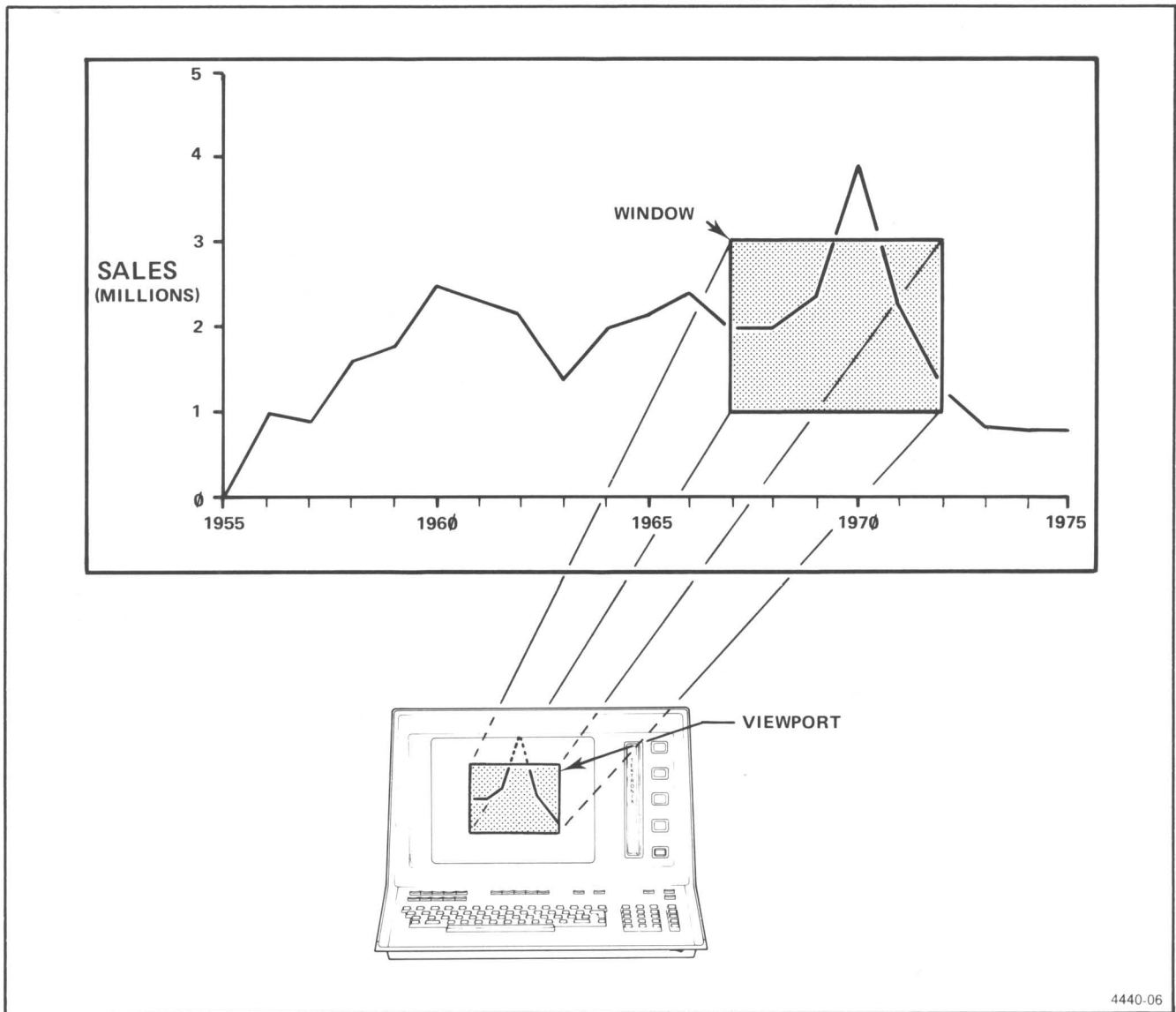


Figure 4-5. Clipping Vectors at Window Boundaries.

Inversion

The WINDOW romcall can be used to "invert" data, i.e., make data values decrease from left to right or from bottom to top.

The WINDOW romcall "links" each WINDOW parameter with a corresponding VIEWPORT parameter. Thus, if a WINDOW romcall inverts the "normal" data values for the graphic environment, and a VIEWPORT romcall inverts them back, the final WINDOW-to-VIEWPORT mapping will be "normal" again, i.e., x-values will increase from left to right and y-values will increase from bottom to top.

Example:

2000 Rcall "window",50 -50, 25, 0

This command sets up the graphic environment so that an x-value of 50 in user data units is mapped to the LEFT side of the viewport, and an x-value of -50 in user data units is mapped to the RIGHT side of the viewport. A y-value of 0 in user data units is mapped to the top of the viewport, and a y-value of 25 in user data units is mapped to the bottom of the viewport. (See Figure 4-6.)

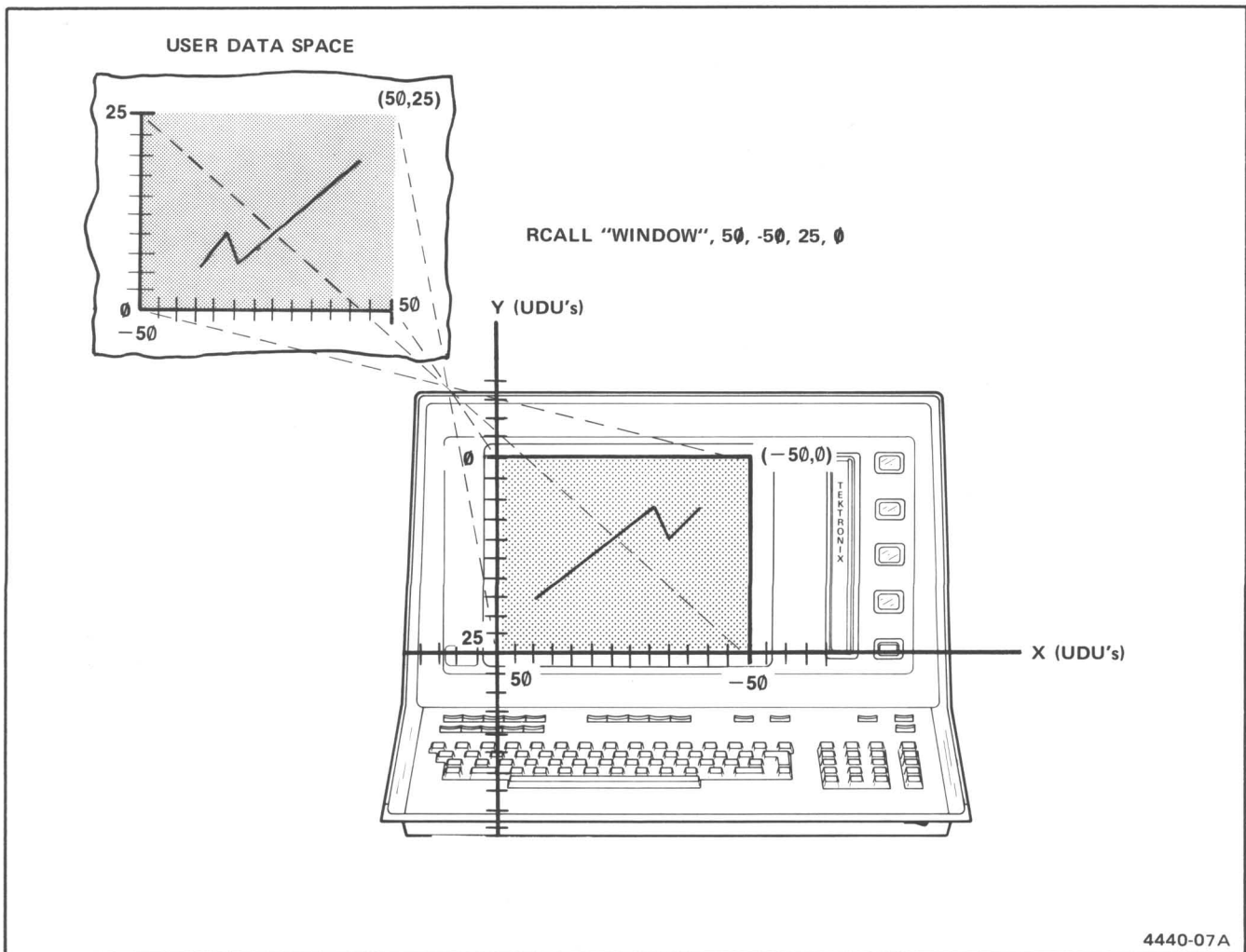


Figure 4-6. Inverting a Graph by Means of the WINDOW Romcall.

Section 5

GRAPHIC PRIMITIVES

INTRODUCTION

Graphic primitive romcalls perform the actual graphic operations. Graphic primitives include:

- DRAW: draws a line from the current position to a specified point.
- GIN: records the current position of the cursor or plotter pen.
- GTEXT: writes characters into the graphic area.
- HARDCOPY: issues a COPY command to an attached Hard Copy unit.
- HOME: moves the cursor or plotter pen to the "home" position (one character's height below the upper-left hand corner of the graphic device).
- MOVE: moves the cursor or plotter pen to a specified point, without drawing a line.
- PAGE: clears the screen, if the graphic device is a terminal, and moves the cursor or plotter pen to the "HOME" position. On TEKTRONIX 4663 plotters with paper advance, this romcall also advances the paper.
- POINTER: allows the user to move the cursor or plotter pen in order to input its position.
- RDRAW: draws a line from the current position to a new position, whose coordinates are given relative to the current position.
- RMOVE: moves the cursor or plotter pen to a new position, whose coordinates are given relative to the current position.

The DRAW Romcall

Syntax Form: [Line-no.] RCALL "DRAW", {numexp} {numexp}
{numarray} ' {numarray}

Descriptive Form: [Line-no.] RCALL "DRAW", X coordinates in user data units, Y coordinates in user data units

PURPOSE

The DRAW romcall reduces the specified numeric expressions to numeric constants, interprets the numeric constants as the X and Y coordinates of a graphic data point in user data units, then draws a vector (a line) on the graphic device from the present position of the cursor to the specified graphic data point.

EXPLANATION

The DRAW romcall draws a vector from the present position of the cursor to the specified data point. The coordinates of the data point are specified in user data units (refer to the User Data Unit concept explanation in Section 1 for a definition of user data units). The following program illustrates the execution of a DRAW romcall (see Figure 5-1).

```
90 Open #1:"comm:"
100 Rcall "ginit",1,4051,1
110 Rcall "page"
120 Rcall "move",40,70
130 Rcall "draw",90,40
140 End
```

Line 100 in this program initializes the Graphics Rompack's global variables and sets the VIEWPORT and WINDOW parameters to their default values. Line 110 clears the screen and line 120 moves the cursor to the coordinates (40,70). This positions the cursor at the starting point of the vector. Line 130 then draws a vector from the present position of the cursor to the specified coordinates (90,40). The first coordinate (90) specifies the absolute horizontal position of the destination point. The second coordinate (40) specifies the absolute vertical position of the destination point. Both values are specified in user data units. If the point lies outside the current WINDOW, the Graphics Rompack attempts to draw to the specified coordinates; however, clipping occurs at the boundary of the VIEWPORT. (Refer to the WINDOW romcall in Section 4 for an explanation of clipping.)

Array DRAW

Several vectors can be drawn with one statement by specifying two arrays as parameters in a DRAW romcall. The following program illustrates the execution of this kind of DRAW. Output from this program is shown in Fig. 5-2. The DRAW romcall in this program is equivalent to executing four separate DRAW romcalls.

Example

```
90 Open #1:"comm:"
100 Rcall "ginit",1,4051,1
110 Rcall "page"
120 Dim x(4),y(4)
130 Data 105,105,65,65,50,90,90,50
140 Read x,y
150 Rcall "move",65,50
160 Rcall "draw",x,y
170 End
```

In line 120, the variables X and Y are dimensioned as one dimensional integer arrays with four elements each. The values for each element are stored in a DATA statement (line 130) and assigned to each element with a READ statement (line 140). The first element in array X is assigned the value 105, the second element is assigned the value 105, the third element is assigned the value 65, and so on. (Refer to section 8, "Input/Output", in the 4041 Programmer's Reference manual for a complete explanation of the READ and DATA statements.)

Line 150 moves the cursor to the center of the viewport (65,50) and the DRAW romcall in line 160 causes the 4041 to draw the square as shown in the illustration. This DRAW romcall produces the same result as four separate DRAW romcalls because array X and array Y contain four elements each. The first vector is drawn using X(1) as the horizontal coordinate and Y(1) as the vertical coordinate; the second vector is drawn immediately afterwards using X(2), Y(2) as coordinates, and so on. This process continues until the last element in each array is used to draw a vector or until the elements in one array run out. Table 5-1 summarizes the effect of the DRAW.

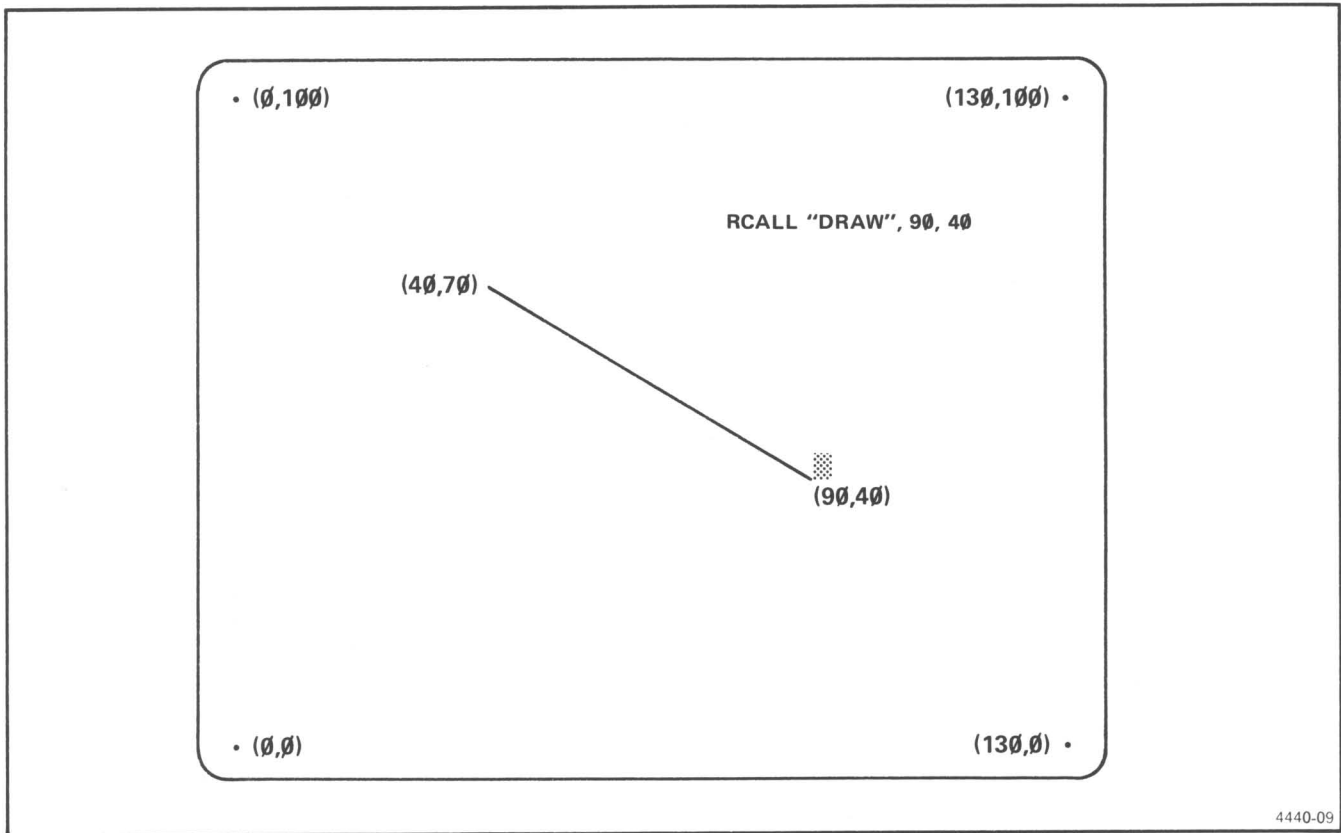
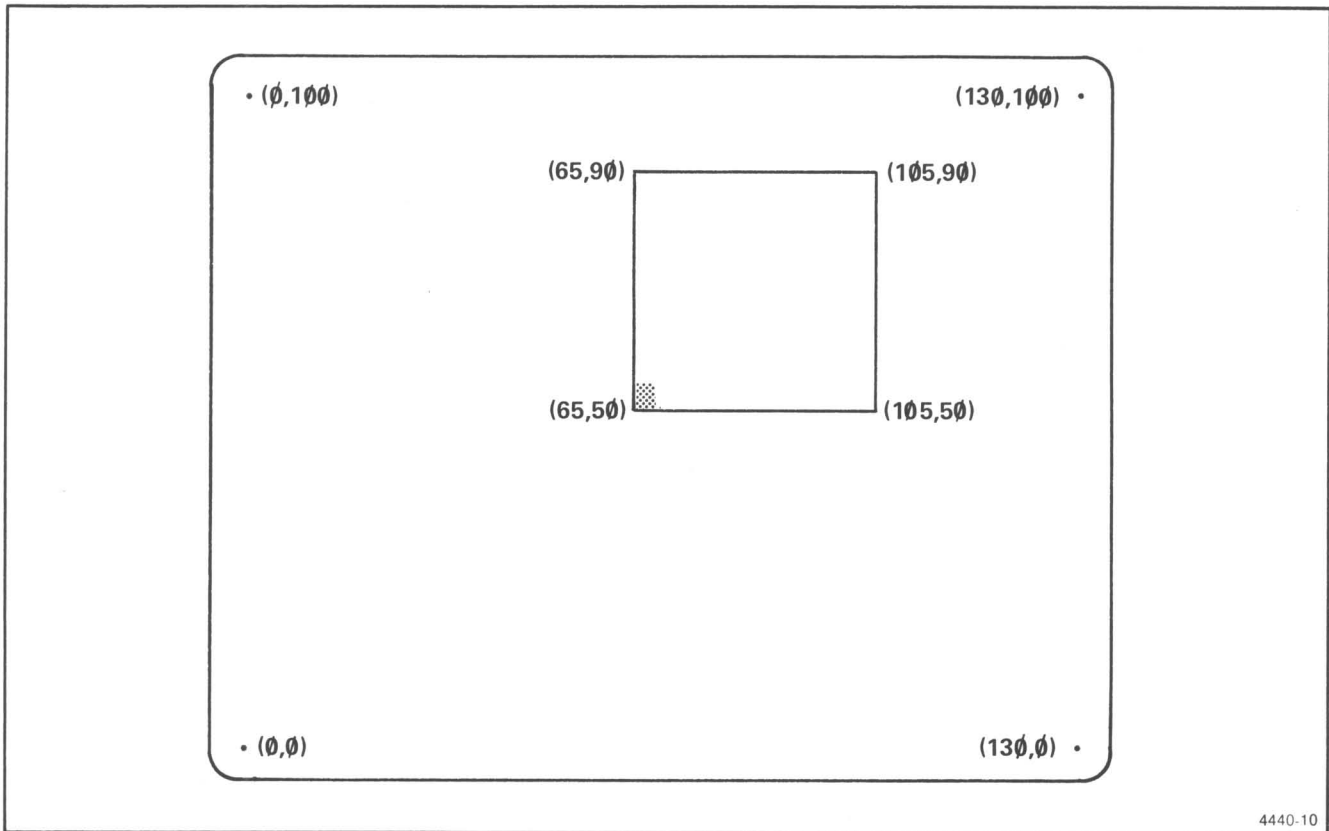


Figure 5-1. Use of RCALL "DRAW".

Table 5-1
One-Dimensional Array DRAW

VECTOR	ARRAY X	ARRAY Y	ROMCALL EQUIVALENT
1st Vector	X(1) = 105	Y(1) = 50	RCALL "DRAW",105,50
2nd Vector	X(2) = 105	Y(2) = 90	RCALL "DRAW",105,90
3rd Vector	X(3) = 65	Y(3) = 90	RCALL "DRAW",65,90
4th Vector	X(4) = 65	Y(4) = 50	RCALL "DRAW",65,50



4440-10

Figure 5-2. Output from RCALL "DRAW" of Array.

If a two-dimensional array is specified as either the X or Y coordinate, the elements are read in row-major order. Table 5-2 summarizes the results of a DRAW when X is a two-

dimensional array. The result is the same as the DRAW just described.

Table 5-2
Two-Dimensional Array DRAW

VECTOR	ARRAY X	ARRAY Y	ROMCALL EQUIVALENT
1st Vector	X(1,1) = 105	Y(1) = 50	RCALL "DRAW",105,50
2nd Vector	X(1,2) = 105	Y(2) = 90	RCALL "DRAW",105,90
3rd Vector	X(2,1) = 65	Y(3) = 90	RCALL "DRAW",65,90
4th Vector	X(2,2) = 65	Y(4) = 50	RCALL "DRAW",65,50

The GIN Romcall

Syntax Form: [Line-no.] RCALL "GIN", {numarray} {numarray}
{numvar} , {numvar}

Descriptive Form: [Line no.] RCALL "GIN", target variable for X coordinate, target variable for Y coordinate

PURPOSE

The GIN (Graphic Input) romcall records the current position of the cursor or graphic writing tool on the graphic device.

EXPLANATION

The GIN romcall is primarily useful as a program development/debugging tool. It is especially useful in immediate mode, to allow the user to ascertain the coordinates of the current position at different points during program execution.

GIN can also be helpful when using the GTEXT romcall, as for example if the user wishes to write a message in the graphic area without affecting the current position. The GIN romcall can be used to save the current position before writing the message. Then, the MOVE romcall can be used to return to the original position.

GIN vs. POINTER

The GIN romcall is used to determine where the cursor or pen is; the POINTER romcall is used to input values by means of the graphic device. (See the description of the POINTER romcall, later in this section, for more information.)

WARNING

When using a TEKTRONIX 4014 terminal as the graphics device, the 4014's TC-2 board must be strapped to send <cr> AND NOT <eot> in order to use the "GIN" or "POINTER" romcalls.

Example

```
1500 Rcall "gin",x,y
```

When this statement is executed, the X coordinate of the current position is assigned to the first variable specified in the GIN romcall (the variable X in this case). The Y coordinate of the position is assigned to the second variable specified in the GIN romcall (the variable Y). Both values are recorded in user data units.

When using a Tektronix 4050-Series Graphic System as the graphics device, the program must execute the following statement

```
CALL "GRAFIN",CHR(13),0,0
```

before calling "TERMIN" to enter terminal emulation mode, if the "GIN" or "POINTER" romcalls are to be used.

NOTE: When communicating with the graphic device over a COMM port, users who wish to execute a PRINT statement after a GIN or POINTER romcall should set the COMM port's ECHO parameter to "NO". Otherwise, the cursor might be moved before the PRINT statement is executed.

EXAMPLE

```
1000 Rcall "gin",oldx,oldy
1010 Rcall "gtext","your message appears here"
1020 Rcall "move",oldx,oldy
```

Line 1000 saves the coordinates of the current position in numeric variables Oldx and Oldy.

Line 1010 prints a message in the graphic area and moves the current position to the end of the message.

Line 1020 returns the current position to where it was before the message was printed.

The GTEXT Romcall

Syntax Form: [Line-no.] RCALL "GTEXT",strexp

Descriptive Form: [Line-no.] RCALL "GTEXT",string

PURPOSE

The GTEXT romcall writes a text string in the graphics area starting at the current position, then moves the current position to the end of the string.

EXPLANATION

GTEXT uses the character size specified by the last TEXTSIZE romcall, or the device's default character size if TEXTSIZE has not been called.

Output from a GTEXT romcall is always clipped to the viewport.

On terminals having both a graphics area and a dialogue area (e.g., Tektronix 4020-Series terminals), the GTEXT romcall writes text in the graphic area, while the PRINT statement writes text in the dialogue area.

Printable vs. Non-Printable ASCII Characters

When the string to be written with GTEXT contains only printable characters (i.e., characters whose ASCII codes range from 32 to 126), the current position after the string is written into the graphic area will always be at the end of the string.

Non-printable ASCII characters (ASCII codes 0 to 31 and 127) are printed in different ways, depending upon the par-

ticular graphic device being used. THE CURRENT POSITION MAY NOT BE WHERE YOU EXPECT IT if you use the GTEXT romcall to print a string containing non-printable ASCII characters. Only trial-and-error will tell you the effect of non-printable characters on the current position when using the GTEXT romcall.

Differences Between PRINT Statement and GTEXT Romcall

The differences between the PRINT statement and the GTEXT romcall are as follows:

- (1) GTEXT updates the current position to the end of the text; PRINT does not affect the current position.
- (2) On Tektronix 4020-Series and 4100-Series terminals, text written with a GTEXT romcall goes to the graphic area, while text written with a PRINT statement goes to the dialogue area.

EXAMPLE

```
200 Recall "gtext","HELLO THERE"
```

This statement prints the message "HELLO THERE" on the graphic device, and updates the current position to a point immediately following the final "E".

The HARDCOPY Romcall

Syntax and Descriptive Form: [Line-no.] RCALL "HARDCOPY" [, numexp]

Descriptive Form: [Line-no.] RCALL "HARDCOPY" [, wait-time]

PURPOSE

The HARDCOPY romcall causes an attached Hard Copy Unit to make a paper copy of information on the graphic device.

If a wait time is given, the 4041 waits the specified number of seconds before sending data to the graphic device. The default wait time is 10 seconds. This is necessary because some terminals lose commands sent while they are making a hardcopy.

EXPLANATION

The HARDCOPY romcall can be used with any Tektronix terminal that has a hardcopy option.

When the HARDCOPY romcall is executed, a MAKE COPY control signal is sent to a Hard Copy Unit (if attached).

EXAMPLE

```
3000 Recall "hardcopy"
```

When this statement is executed under program control, program execution stops while an attached Hard Copy Unit scans the terminal screen. Processing continues as soon as the scan is completed.

The HOME Romcall

Syntax and Descriptive Forms: [Line-no.] RCALL "HOME"

PURPOSE

The HOME romcall returns the cursor or plotter pen to the HOME position. The HOME position is one character position down from the upper-left corner of the viewport.

EXAMPLE

4000 Recall "home"

The MOVE Romcall

Syntax Form: [Line-no.] RCALL "MOVE", numexp , numexp

Descriptive: [Line-no.] RCALL "MOVE", X coordinate in user data units , Y coordinate in user data units

PURPOSE

The MOVE romcall reduces the specified numeric expression to numeric constants, interprets the numeric constants as the coordinates of a graphic data point in user data units, then moves the cursor or plotter pen to the specified graphic data point and makes that point the current position.

EXPLANATION

The MOVE romcall can move the current position to any point in the viewport. The following program illustrates the execution of a MOVE statement. (Output from the program is shown in Fig. 5-3.)

EXAMPLE

```

90 Open #1:"comm:"
100 Recall "ginit",1,4010,1
110 Recall "move",40,70
120 End

```

Line 100 in this program sets the VIEWPORT and WINDOW parameters to their default values (for ease of illustration). Line 110 then moves the 4010 cursor to the coordinates (40,70); 40 is the horizontal coordinate (X) specified in user data units; 70 is the vertical coordinate (Y) specified in user data units. Moves are normally used to position the cursor at the starting point of a vector, and are usually executed just prior to a DRAW or RDRAW romcall.

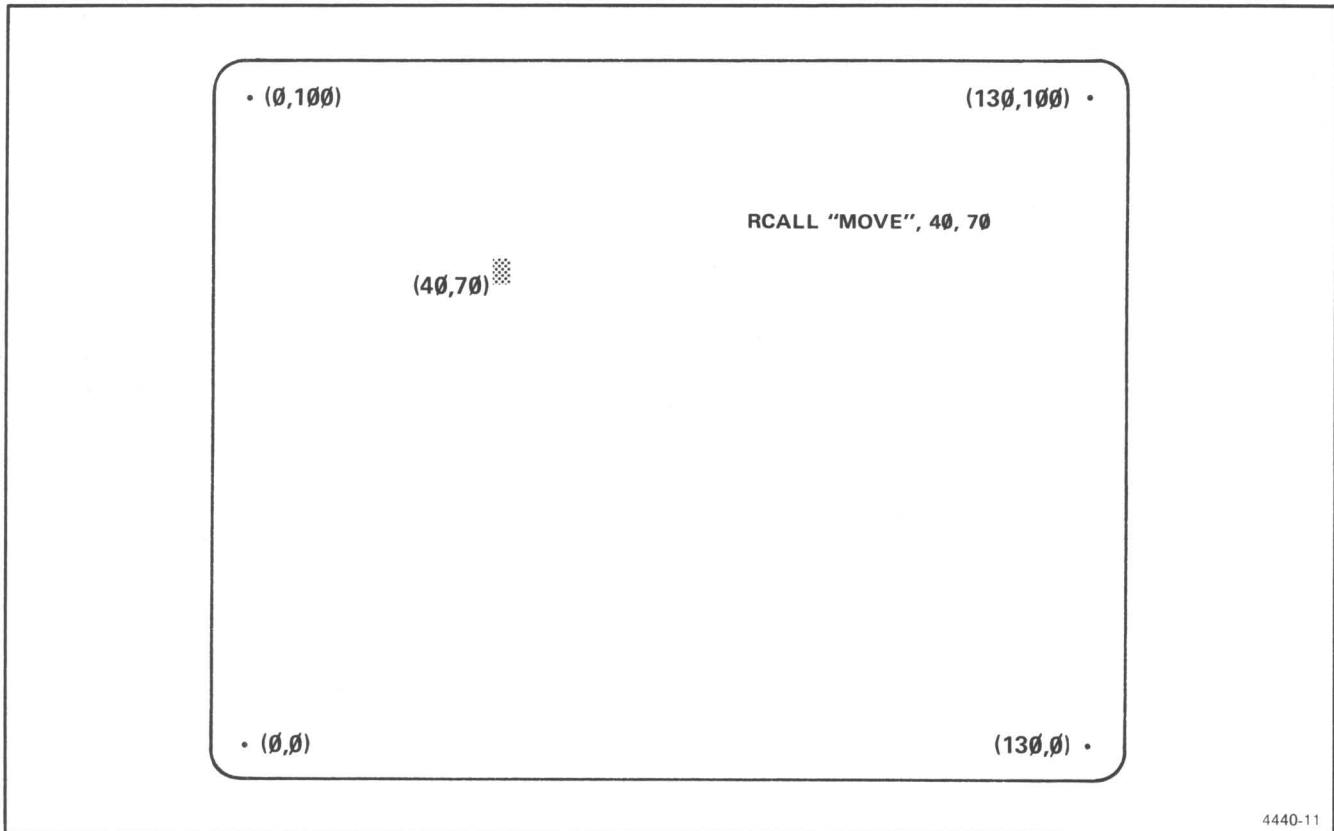


Figure 5-3. Use of RCALL "MOVE".

The PAGE Romcall

Syntax and Descriptive Forms: [Line-no.] RCALL "PAGE"

PURPOSE

If the graphic device is a terminal, the PAGE romcall erases the screen and returns the cursor to the HOME position. If the graphic device is a plotter with paper advance, PAGE advances the paper, else it is equivalent to HOME.

EXAMPLE

2500 Recall "page"

The POINTER Romcall

Syntax Form: [Line-no.] RCALL "POINTER", {numarray} {numarray} {strarray}
 {numvar} {numvar} {strvar}

Descriptive Form: [Line-no.] RCALL "POINTER", target variable for X coordinate of graphic point in user data units, target variable for Y coordinate of graphic point in user data units, target variable to record the key pressed to end the entry

PURPOSE

The POINTER romcall:

- (1) causes the graphic cursor to be displayed (if the graphic device is a terminal),
- (2) allows the user to maneuver the terminal cursor or plotter pen, and
- (3) records the X- and Y-coordinates of the graphic cursor or pen when the user presses any key (if the graphic device is a terminal), the "CALL" switch (if the graphic device is a 4662 plotter), or the "POINT" switch (if the graphic device is a 4663 plotter).

EXPLANATION

Using POINTER With Graphic Terminals

When the graphic device is a terminal, the POINTER romcall places the graphic cursor on the display screen.

When the graphic device is a plotter, the POINTER romcall enables the plotter to input graphic data.

Example

```
2600 Rcall "pointer",xpos,ypos,c$
```

When this statement is executed, the graphic cursor is placed on the screen. The tip of the arrow or the crosshair on the graphic cursor marks the current location of the graphic point. (This point is normally used as a reference to draw graphic vectors).

Completing the POINTER Operation

After the graphic cursor is positioned on the screen with the graphic input device, a key is pressed on the terminal. When a key is pressed, the X and Y coordinates of the current location of the graphic point are recorded. The X coordinate is assigned to the first target variable specified in the POINTER romcall, and the Y coordinate is assigned to the second target variable specified in the POINTER romcall. Both the X and Y coordinates are measured in user data units. The third target variable records which key is pressed to terminate the operation. The key symbol is recorded, but not echoed on the terminal. (The character returned for plotters is the null string.) After the key is pressed, program execution continues with the next statement.

Applications for the POINTER Romcall

The POINTER romcall provides the user with a method to communicate with the system via the graphic device. For example, assume that 1000 data values are input into memory from an external peripheral device over the GPIB. Assume also that this data is processed by the 4041 and displayed on the terminal as a point plot graph. The POINTER romcall and the graphic input devices allow the keyboard operator to "talk" to the system about the graph. In this case, program subroutines can be placed in memory so that the keyboard operator can place the graphic cursor over a particular data point on the graph and press a key to execute a predefined function. Pressing the "Y" key might mean "display the numeric value of this data point"; pressing the "D" key might mean "delete this data point from memory."

WARNING

When using a TEKTRONIX 4014 terminal as the graphics device, the 4014's TC-2 board must be strapped to send <cr> AND NOT <eot> in order to use the "GIN" or "POINTER" romcalls.

When using a TEKTRONIX 4050-Series Graphic System as the graphics device, the program must execute the following statement

CALL "GRAFIN",CHR(13),0,0

before calling "TERMIN" to enter terminal emulation mode, if the "GIN" or "POINTER" romcalls are to be used.

NOTE

For TEKTRONIX 4020-Series terminals the terminator character must be in the ASCII range 32-126 to work properly.

The RDRAW Romcall

Syntax Form: [Line-no.] RCALL "RDRAW", {numexp}, {numexp}
 {numarray}, {numarray}

Descriptive Form: [Line-no.] RCALL "RDRAW", X increment in user data units , Y increment in user data units

PURPOSE

The RDRAW romcall reduces the specified numeric expressions to numeric constants, interprets the numeric constants as the increments of a graphic data point relative to the present position of the cursor, then draws a vector from the present position of the cursor to the specified data point.

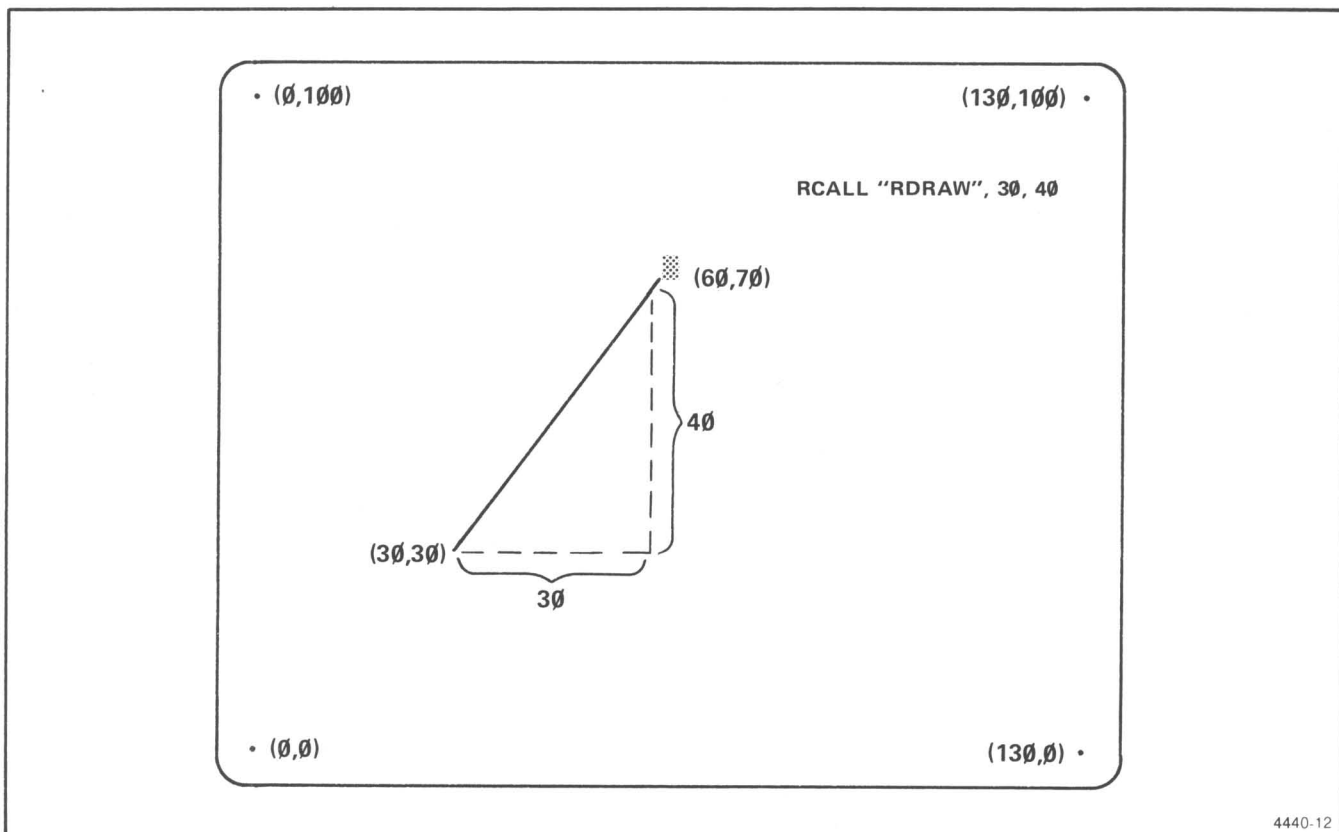
40 units above the present position of the cursor, for example, then these increments are specified in an RDRAW romcall. The Graphics rompack figures out the absolute coordinates of the destination point and draws the vector to that point.

The first parameter in the RDRAW romcall specifies the horizontal increment in user data units. The second parameter specifies the vertical increment in user data units. All movement is relative to the current position and clipping occurs if the boundary of the viewport is crossed.

EXPLANATION

The RDRAW romcall frees the programmer from having to figure out the absolute coordinates of each data point. If the next vector to be drawn is a point 30 units to the right and

The following example illustrates the execution of an RDRAW romcall. The output produced by this example is shown in Fig. 5-4.



4440-12

Figure 5-4. Use of RCALL "RDRAW".

Example

```

90 Open #1:"comm:"
100 Rcall "ginit",1,4010,1
110 Rcall "page"
120 Rcall "move",30,30
130 Rcall "rdraw",30,40
140 End
    
```

Line 100 in this program sets the VIEWPORT and WINDOW parameters to their default values (for ease of illustration). Line 110 clears the screen. Line 120 moves the cursor to the absolute coordinates (30,30). Line 130 then draws a vector from the current position (30,30) to a position 30 units to the right and 40 units above. The result is a vector drawn to the absolute coordinates (60,70). The same result can be obtained by executing the statement RCALL "DRAW", 60,70.

Array RDRAW

Several relative draws, like several absolute draws, can be executed in one statement by specifying arrays for coordinates. The following examples illustrate the execution of this kind of RDRAW. Output from this example is shown in Fig. 5-5.

Example

```

90 Open #1:"comm:"
100 Rcall "ginit",1,4010,1
110 Rcall "page"
120 Dim x(4),y(4)
130 Data 40,0,-40,0,0,40,0,-40
140 Read x,y
150 Rcall "move",65,50
160 Rcall "rdraw",x,y
170 End
    
```

In this program, the variables X and Y are defined as one dimensional arrays with four elements each (line 120). The values for the elements are stored in a DATA statement in line 130 and assigned to each element with the READ statement in line 140. The first element in array X is assigned the value 40, the second element is assigned the value 0, and so on. (Refer to Section 8, "Input/Output" in the 4041 Programmer's Reference manual for a complete explanation of the READ statement and the DATA statement.)

Line 150 moves the cursor to the center of the viewport (65,50) and the RDRAW romcall in line 160 draws the square as shown in the illustration. This RDRAW romcall produces the same result as four RDRAW romcalls,

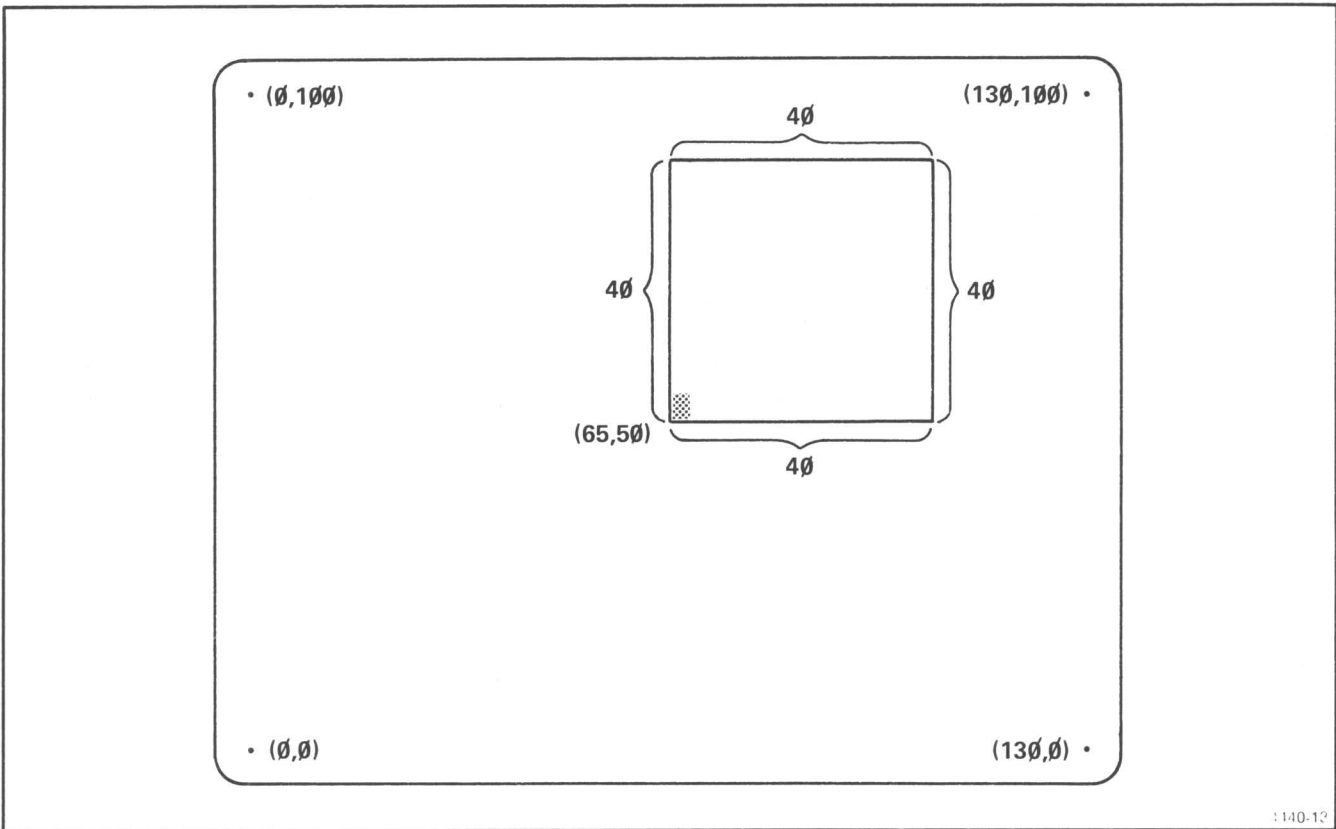


Figure 5-5. Output from RCALL "RDRAW" of Array.

because array X and array Y contain four elements each. The first vector is drawn using X(1) as the horizontal increment and Y(1) as the vertical increment; the second vector is drawn immediately afterwards using X(2) as the horizontal increment and Y(2) as the vertical increment, and so on. This process continues until the last element in each array is used to draw a vector or until one of the arrays run out of elements. Table 5-3 summarizes the effect of the RDRAW.

If a two dimensional array is specified, then the elements are read in row major order. Table 5-4 summarizes the results of an RDRAW when X and Y are two dimensional arrays. The result is the same as the RDRAW just described.

Table 5-3
One-Dimensional Array RDRAW

VECTOR	ARRAY X	ARRAY Y	RDRAW EQUIVALENT
1st Vector	X(1) = 40	Y(1) = 0	RCALL "RDRAW",40,0
2nd Vector	X(2) = 0	Y(2) = 40	RCALL "RDRAW",0,40
3rd Vector	X(3) = -40	Y(3) = 0	RCALL "RDRAW",-40,0
4th Vector	X(4) = 0	Y(4) = -40	RCALL "RDRAW",0,-40

Table 5-4
Two-Dimensional Array RDRAW

VECTOR	ARRAY X	ARRAY Y	ROMCALL EQUIVALENT
1st Vector	X(1,1) = 40	Y(1,1) = 0	RCALL "RDRAW",40,0
2nd Vector	X(1,2) = 0	Y(1,2) = 40	RCALL "RDRAW",0,40
3rd Vector	X(2,1) = -40	Y(2,1) = 0	RCALL "RDRAW",-40,0
4th Vector	X(2,2) = 0	Y(2,2) = -40	RCALL "RDRAW",0,-40

The RMOVE Romcall

Syntax Form: [Line-no.] RCALL "RMOVE", numexp , numexp

Descriptive Form: [Line-no.] RCALL "RMOVE", X increment in user data units , Y increment in user data units

PURPOSE

The RMOVE romcall reduces the specified numeric expressions to numeric constants, interprets the numeric constants as the increments of a graphic data point relative to the current position of the cursor, then moves the current position to that point.

EXPLANATION

The RMOVE romcall frees the programmer from having to figure out the absolute coordinates of each data point. If the cursor is to be moved to a point 30 units to the right and 40 units above its present position, for example, then these increments are specified in an RMOVE romcall. The Graphics rompack figures out the absolute coordinates of the destination point and moves the cursor to that point.

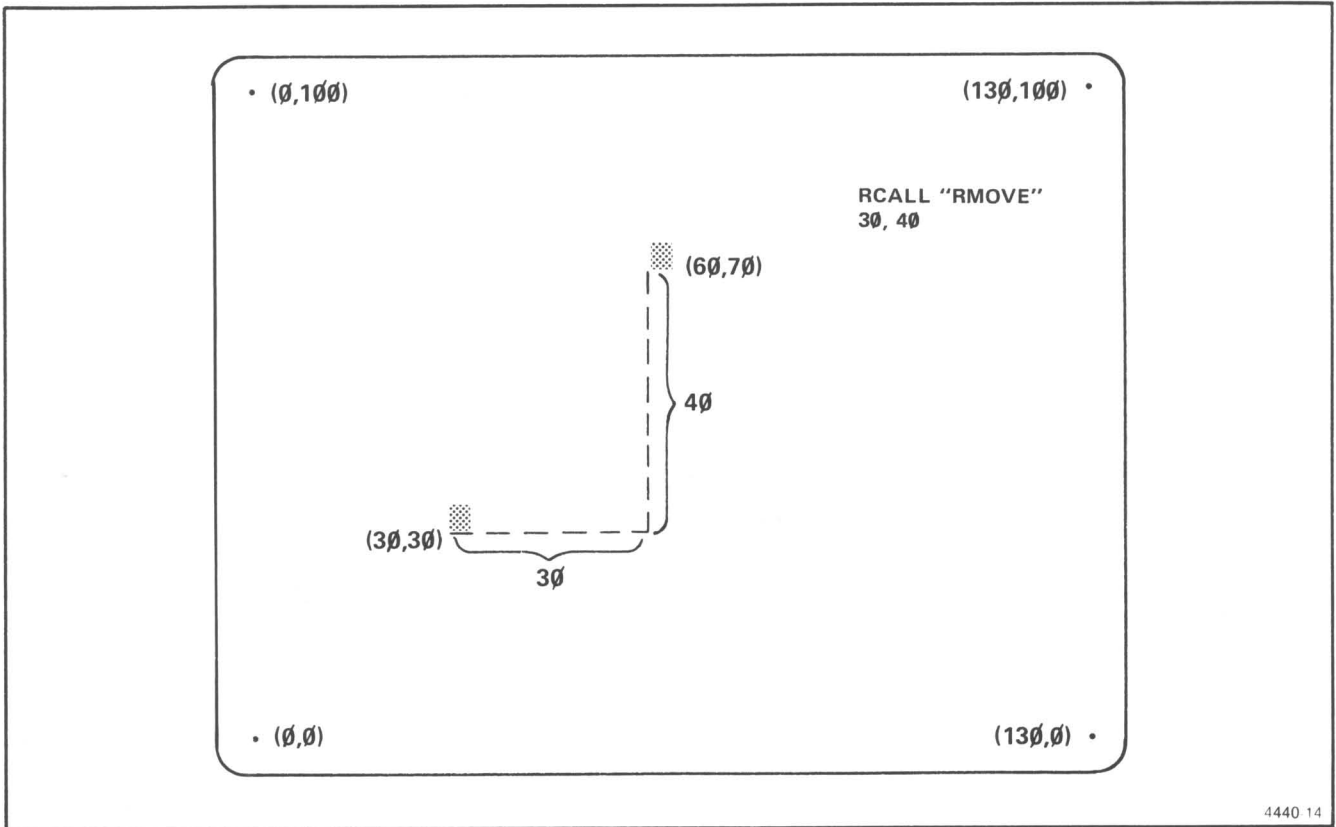
The first parameter in the RMOVE romcall specifies the horizontal increment (X) in user data units. The second parameter specifies the vertical increment (Y) in user data units. All movement is relative to the present position of the cursor.

The following example illustrates the execution of an RMOVE romcall. Output from this example is shown in Fig. 5-6.

EXAMPLE

```
 90 Open #1:"comm:"  
100 Rcall "ginit",1,4010,1  
110 Rcall "page"  
120 Rcall "move",30,30  
130 Rcall "rmove",30,40  
140 End
```

Line 100 in this program sets the VIEWPORT and WINDOW parameters to their default values (for ease of illustration). Line 110 clears the screen. Line 120 moves the cursor to position (30,30). Line 130 then moves the cursor to a position 30 units to the right and 40 units above the present position of the cursor. The result is a move to the absolute coordinates (60,70). The same result can be achieved by executing the statement RCALL "MOVE", 60,70.



4440 14

Figure 5-6. Use of RCALL "RMOVE".

Appendix A

ERROR MESSAGES

Error Number	Meaning
1	Rompack not initialized — need to execute GINIT or GDEVICE romcall
2	Wrong number of arguments
10	Invalid model number
11	Invalid option to model number
12	Graphic device will not work through logical unit specified (e.g., 4662 plotter on logical unit associated with COMM driver, or 4014 on logical unit associated with GPIB).
20	In GDEVICE argument list, minimum value less than 0 or maximum value greater than 4095 was specified.
30	Two values that must be different are equal or too close to be differentiated.
32	Value that must be greater than 0 is not.
33	Value that must be greater than or equal to 0 is not.
34	Minimum values must be less than maximum values.
40	Viewport specified is completely off the surface of the graphic device.
50	Line style index out of range
60	Device does not support GIN
61	Device does not support POINTER

Note: For errors that pertain to rom packs, the actual error number returned by the 4041 is equal to the rompack error number (i.e., 1, 2, etc.) plus an offset that depends on the rompack's slot position in the rom carrier.

ASK\$("ROMPACK") returns the names of all rompacks recognized by the system, along with the error-number offset for each rompack.

Appendix B

INITIALIZATION ACTIONS TAKEN IN GINIT

This appendix describes the action taken by the 4041 in executing a GINIT romcall with different Tektronix devices.

Model/ Option	Initialization actions
4006/1 4010/1 4012/1 4013/1 4051/1 4052/1	None
4012/2 4013/2 4014/1 4015/1	Set large character size
4014/2 4015/2 4016/1 4016/2 4054/1	Set large character size Set solid line style
4025/1 4025/2	Make workspace 30 lines Make graphic area 30 lines All communication in the monitor No shrink mode for graphics No buffered mode No form fill mode Set solid line style
4027/1	Make workspace 30 lines Make graphic area 30 lines All communication in the monitor No shrink mode for graphics No buffered mode No form fill mode Set solid line style Set color to color 1

Model/ Option	Initialization actions
4112/1 4113/1	Set dialog area characters to 80 Set dialog lines to 4 Set dialog area buffer to 20 lines Set dialog area position to 0,0 Enable dialog area Make dialog area visible Set graphic text precision to string precision Set window to 0,0,4095,2712 Set viewport to 0, 359, 4095, 3071 Set linestyle to solid Set text and graphic color to color 1
4114/1	Set dialog area characters to 80 Set dialog lines to 4 Set dialog area buffer to 20 lines Set dialog area position to 0,0 Enable dialog area Make dialog area visible Set graphic text precision to string precision Set large character size Set linestyle to solid
4663/1 4663/2	Set command format to 2 Use GDU's Use absolute coordinates Set window to 0,0 135.48,100 Set viewport same as window Set font to the default font Set linestyle to solid Use pen 1 (color 1)
4662/1	Set font to default size
4662/2	Set font to default size Set color to color 1 (pen 1)