1976-03

Users manual for the vector general graphics display unit

Thorpe, Lloyd Allen

Monterey, California. Naval Postgraduate School

NPS-72Rr76031

NAVAL POSTGRADUATE SCHOOL

Monterey, California

USERS MANUAL FOR THE VECTOR GENERAL GRAPHICS DISPLAY UNIT

by

L. A. Thorpe

G. M. Raetz

March 1976

Approved for public release; distribution unlimited.

Prepared for: Naval Electronics Systems Command
(ELEX 320)
Washington, D. C.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS-72Rr76031	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): USERS MANUAL FOR THE VECTOR GENERAL GRAPHICS DISPLAY UNIT	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s): L. A. Thorpe G. M. Raetz	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: N0003975WR 59051 PDM 000320
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Electronics Systems Command (ELEX 320) Washington, D. C. 20360	12. REPORT DATE: 15 March 1976 13. NUMBER OF PAGES: 85
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Computer Science Group (Code 72) Naval Postgraduate School Monterey, California 93940	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Real time
Interactive graphics
Graphics

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is the description of a users manual for an interactive graphics package designed to allow programmatic control of the Vector General Graphics Display Unit from a Digital Equipment Corporation PDP-11/50 computer. The manual requires a knowledge of the C-programming language and a general familiarity with graphics terminology. Included is a brief description of the Vector General Graphics Display Unit, a

20. (cont.)

description of the interface routines, and a description of the Vector General graphics display instructions.

ABSTRACT

This report is the description of a users manual for an interactive graphics package designed to allow programmatic control of the Vector General Graphics Display Unit from a Digital Equipment Corporation PDP-11/50 computer. The manual requires a knowledge of the C-programming language and a general familiarity with graphics terminology. Included is a brief description of the Vector General Graphics Display Unit, a description of the interface routines, and a description of the vector General graphics display instructions.

TABLE OF CONTENTS

I. INTRODUCTION

This manual is a user oriented description of a C-callable interactive graphics package designed to allow programmatic control of the Vector General Graphics Display Unit (vector general). A knowledge of the C-programming language and a general familiarity with graphics terminology are assumed.

The vector general is an interactive graphics display system interfaced with the PDP-11/50 computer. The display interacts with a PDP-11 user by displaying pictoral information on the surface of a cathode ray tube and by accepting inputs from external control devices. The inputs are requested and processed by PDP-11 computer programs that alter and maintain the output picture being presented to the user.

With the permission of Vector General Corporation, portions of the material in sections II, III, and V have been reproduced from the Vector General Graphics Display Unit Reference Manual (VG 10105b).

II. VECTOR GENERAL DISPLAY SYSTEM

The vector general contains the following features for interactive interactive graphics display:

Interface unit

Display controller

Digital to analog converter

Vector generator

Display monitor

Character generator

Circle/Arc generator

Three dimension coordinate transform generator

Four interactive control devices are connected to the system. The devices are:

Alphanumeric keyboard

Thirty-two lighted function switches with manual interrupt

Ten control dials

Light pen

The computer and display controller communicate by way of the interface unit through three types of channels. These channels are:

Data Channel - Direct memory access channel. Used by the vector general to directly access memory without interferring with the operation of the PDP-11 central processor.

Programmed Input/Output Channel - Used to start the controller and acknowledge interrupts.

Interrupt Channel - Used by the display to interrupt the PDP-11 computer.

The display controller processes all display functions while running asynchronously with the PDP-11 central processor. The controller also receives inputs from the external control devices.

The digital to analog convertor converts the digital values from the display controller into analog signals for use in the vector generator.

The cathode ray tube generates an electron beam that shows as a spot of light on the face of the tube. An electromagnetic deflection system causes the spot to move in a direction on the tube face in response to signals from the vector generator. An input from the vector generator causes the brightness of the spot to vary.

The method of controlling the movement of the spot is called the random scan method. This involves steering the spot in a straight line between two points on the display screen. A series of these straight lines constitute an image. To present a clear image, the pattern traced on

the tube must be repeated approximately thirty to forty times per second. Each repetition is called a frame and the frequency at which it is generated is called the refresh rate. The default rate is forty hertz but can be varied from thirteen to one hundred twenty hertz by the routine vaclock().

A. VISIBLE SPACE

The rectangular portion of the CRT which can be viewed by a user is called the visible space. The visible space is limited by an opaque mask with a rectangular cutout. See Figure 2-1.

The maximum picture space is larger than the visible space. This permits limited zooming but primarily allows fully visible objects to be rotated and positioned to the extreme limits of the visible space and yet draw any remaining visible portions without distortion.

The picture being generated is adjusted in size (scaled) to present the desired output by means of two controls:

a. The manually adjustable gain controls on the CRT deflection hardware.

b. The program controlled picture scale transformation hardware. See pscal() routine in section VI.

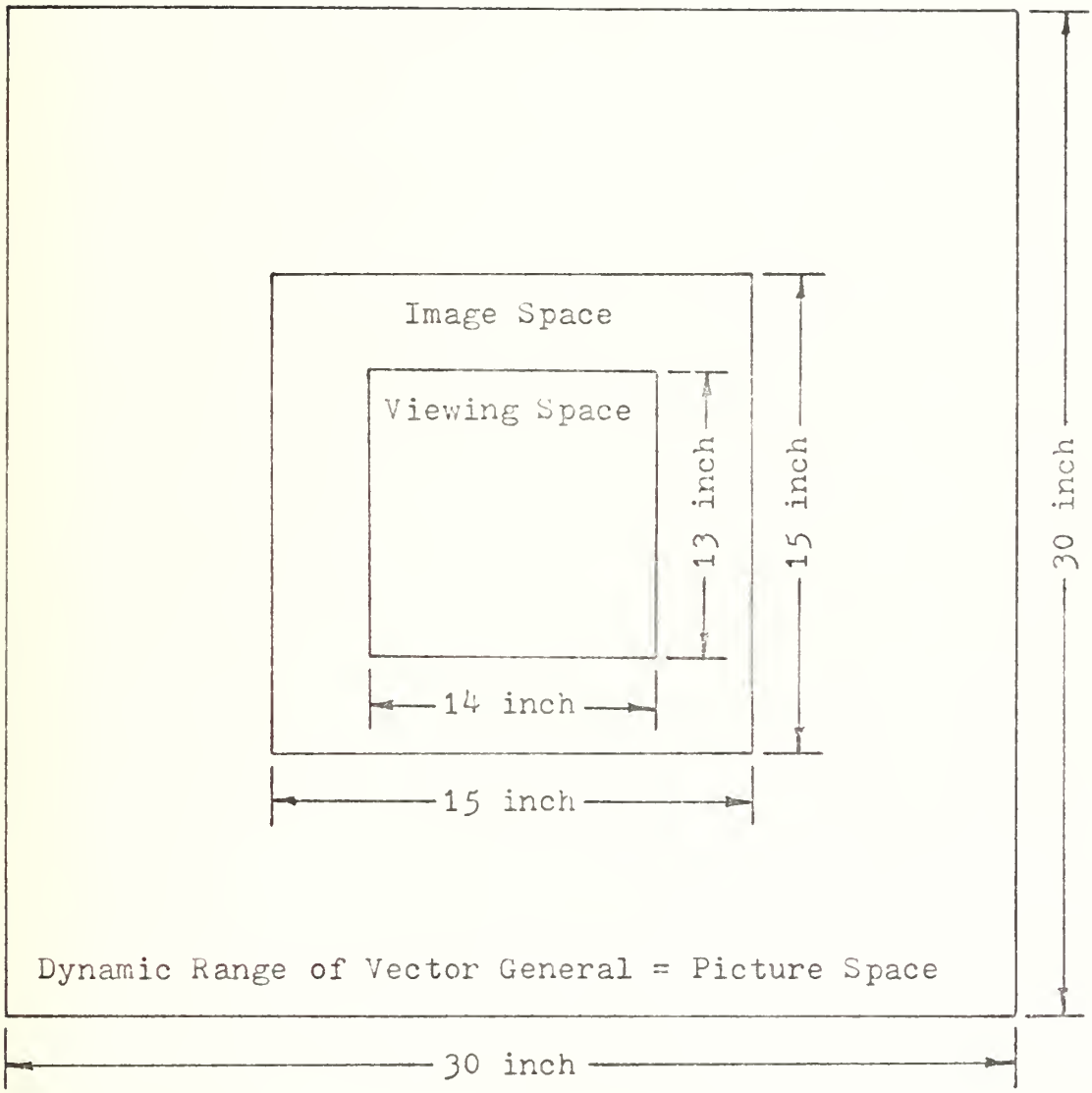


Figure 2-1. Image Areas

The picture can be generated on a picture space coordinate system and scaled for viewing through the visible space. See vgpost() and vgpscal() routines in section VI.

With the gain control knobs at standard midrange calibrated settings, the maximum picture space (over which the vector generator accurately reproduces images) is a thirty inch by thirty inch plane of which the visible space is a thirteen inch by fourteen inch rectangle in the center. See Figure 2-1.

B. PICTURE SPACE

The transformation hardware permits the coordinates defining an element to be transformed prior to display generation. For the input coordinates (X,Y,Z) the output transformed X and Y are used to generate the element's horizontal and vertical picture space position respectively. Thus, the picture space is the X-Y projection of the transformed element definition.

If no transformation is performed, or for zero rotation, zero displacement, and full scale size transformation, an element coordinate (X,Y,Z) will correspond directly to the picture space (X,Y), with the positive X term horizontal towards the right of a viewer and positive Y being vertical. With the gain knobs at the calibrated settings and the picture scale set to maximum

(1.0), a plus full scale X element coordinate value (+2047) transforms into an X picture space coordinate value which corresponds to a horizontal displacement of seven and one-half inches to the right of center or one-half to the right of the visible space. Similarly, for no transformation and maximum picture scale, a full scale Y element coordinate value (+2047) corresponds to a picture space position seven and one-half inches up from the center.

To view a centered two-dimensional object defined over the entire X-Y coordinate range (such as a page of text), the picture scale can be set to .92 or the gain knobs turned down. To view an entire centered three-dimensional object which is defined over the entire (X, Y, Z) image space, a 0.577350 factor is needed to view the maximum length of the projected diagonals of the image space. The picture scale is defined as 1.0 unless explicitly changed by the routine vapscal().

Since the picture space is larger than the visible space, each element may be positioned out of the viewing area in any direction without distorting any remaining visible portions. This capability is termed the hardware scissoring facility.

C. IMAGE SPACE

Prior to transformation and projection onto the picture space an element is defined in a coordinate system referred to as the image space. All separately transformed objects of a displayed picture are defined in their respective untransformed image spaces.

The image space coordinate system is defined with positive X horizontal to the right of the viewer and Y being vertical. The Z axis is perpendicular to the X-Y plane and intersects the X and Y axes at the zero point. Positive Z is toward the viewer. The image space is a fifteen inch by fifteen inch rectangle. See Figure 2-1.

To exploit maximum use of transformation ranges and coordinate resolution, all elements should be defined as large as possible. Elements are defined primarily in terms of generated visual elements: Vectors and Characters.

D. THREE-DIMENSIONAL DISPLAY

Three-dimensional presentation involves a third, or Z, axis that is perpendicular to the face of the screen and intersects the X and Y picture space axis at the zero point. The Z axis represents depth into and out of the display screen. The illusion of depth may be achieved by

varying the light intensity of the fluorescent spot in proportion to the value of the Z coordinate. The intensity increases exponentially with the intensity value over the range of minus full-scale intensity to one-half full-scale intensity. Maximum intensity is at the face of the screen.

E. PICTURE CONTROL

The picture control is used for picture transformation after all transformations of individual objects have been completed. The registers used for this feature are the twelve bit intensity offset register, the twelve bit intensity scale register, and the twelve bit picture scale register. The value in the picture scale register is multiplied by each of the transformed X, Y, and Z coordinates to establish the final picture size. This scaling applies also to characters in the picture.

The intensity scale register is used in conjunction with the intensity offset register to provide depth cueing, or shading of the intensity of the picture according to the values of the Z coordinate. The hardware calculations of the spot intensity at any instant can be represented by the following C-programming language routine:

```
int is;          intensity scale register value
int io;          intensity offset register value
int i;           resulting intensity
int imax;        maximum intensity
int z;           transformed Z coordinate value
int z1;
int k;           constant
int e;           2.7182818284

z1 = 0.5 * abs(is) * z + io;
if (is >= 0 && z1 <= .02) i = imax * e ** (k * z1);
if (is >= 0 && z1 > .02) i = saturation;
if (is < 0 && z1 <= .02) i = imax * e ** (k * z1);
if (is < 0 && is > .02) i = 0;
```

The intensity cutoff plane is established by the value in the intensity offset register. Within the depth range of an image, the intensity is blanked between the viewer and the screen. The intensity is at its maximum at the face of the screen and decreases exponentially with decreasing values of Z toward the back of the image. Figure 2-3A and Figure 2-3B show the effect of intensity offset variation. As the value in the intensity offset register is changed, the element moves forward or backward through the intensity range, to vary the section that is intensified and the part that is blanked out.

The intensity range, or apparent depth of the image, is determined by the value in the intensity scale register. If the value is 1.0, the maximum intensity is achieved. If the value is zero, the intensity is constant and the element has no depth-cueing. Figure 2-4A and Figure 2-4B show how a variation in intensity scale changes the depth of the element.

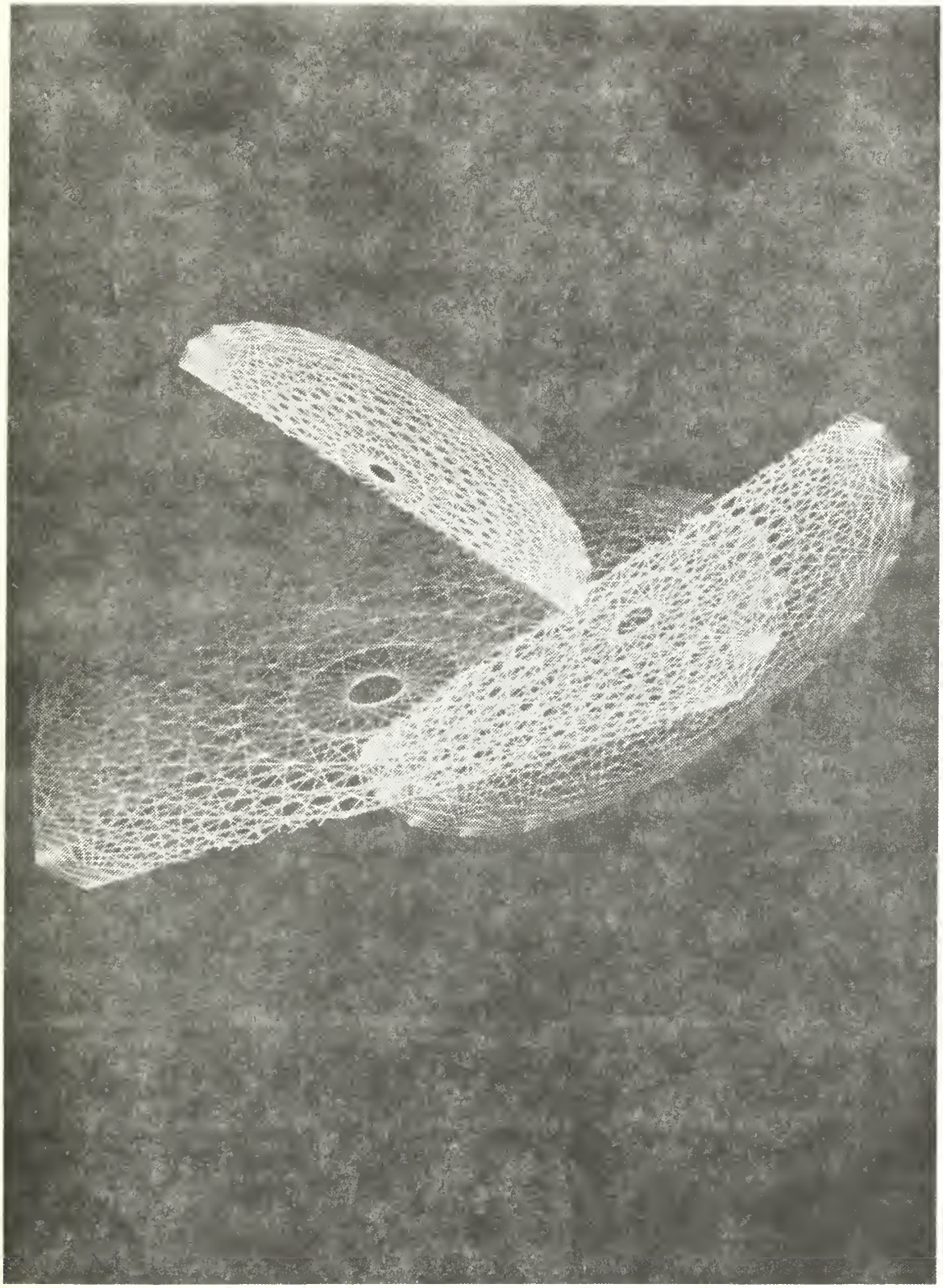


Figure 2-3A. Intensity Offset Variation

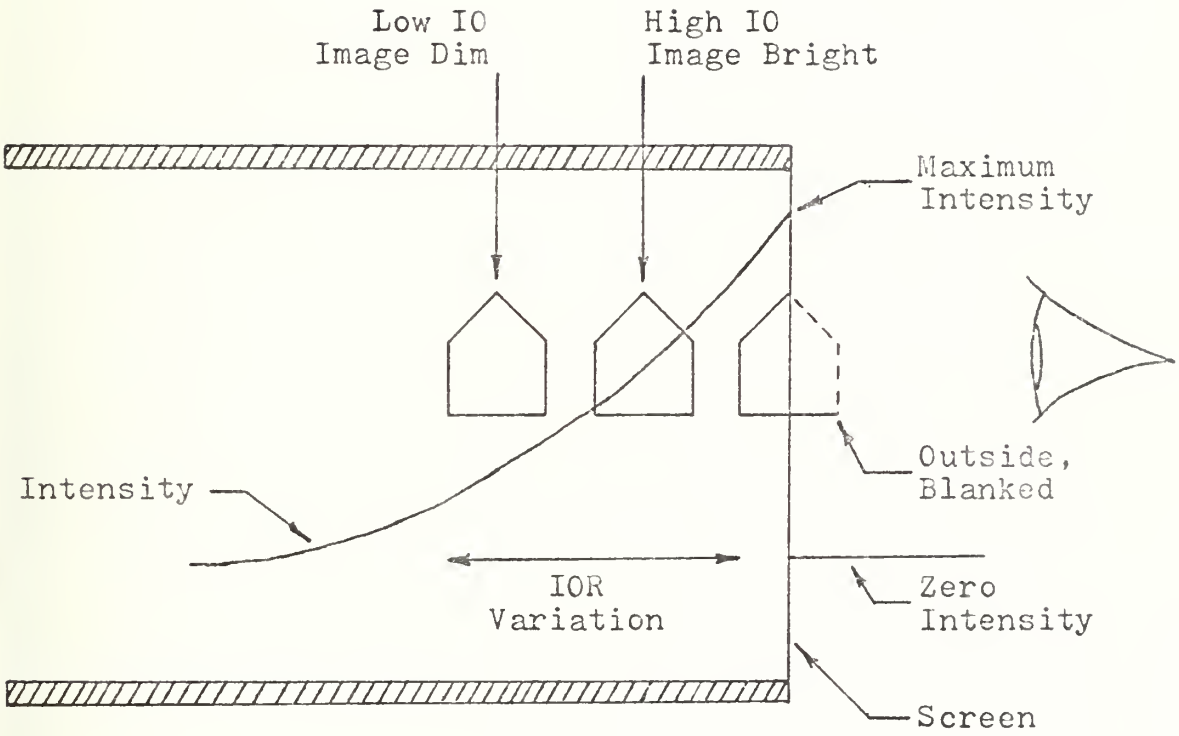
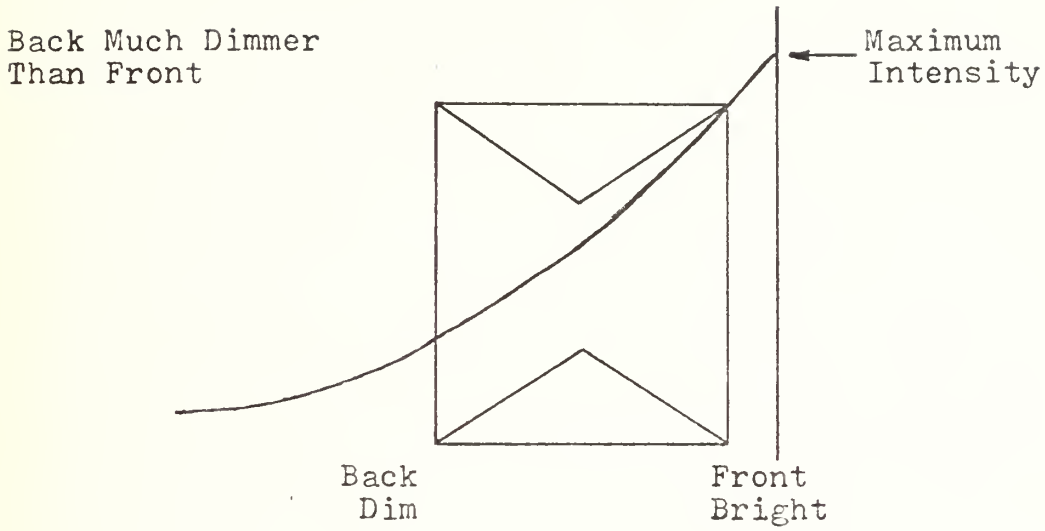


Figure 2-3B. Effect of Intensity Offset Variation

Figure 2-4A. Intensity Scale Variation

Large Intensity Scale:



Small Intensity Scale:

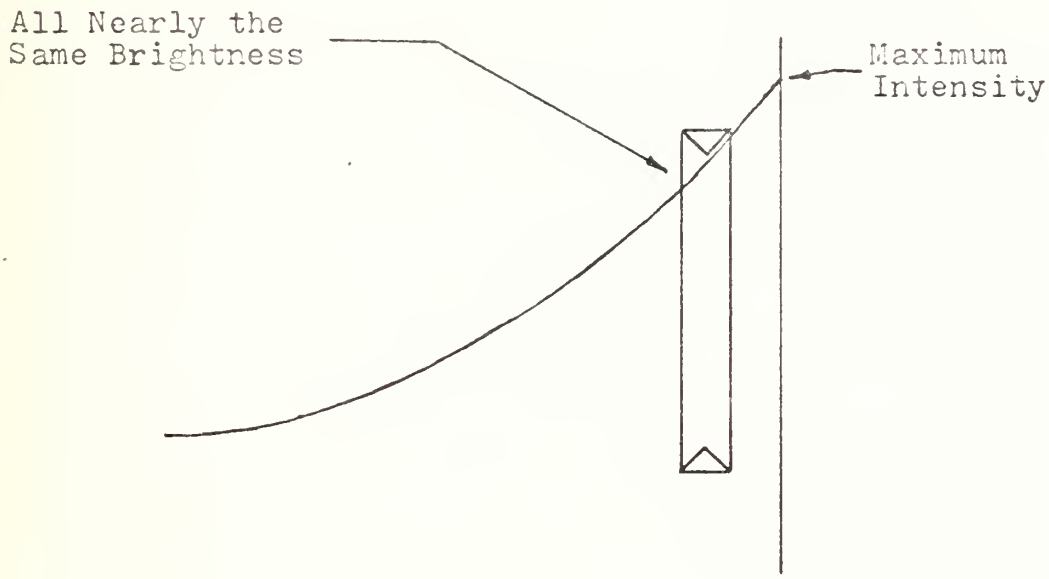


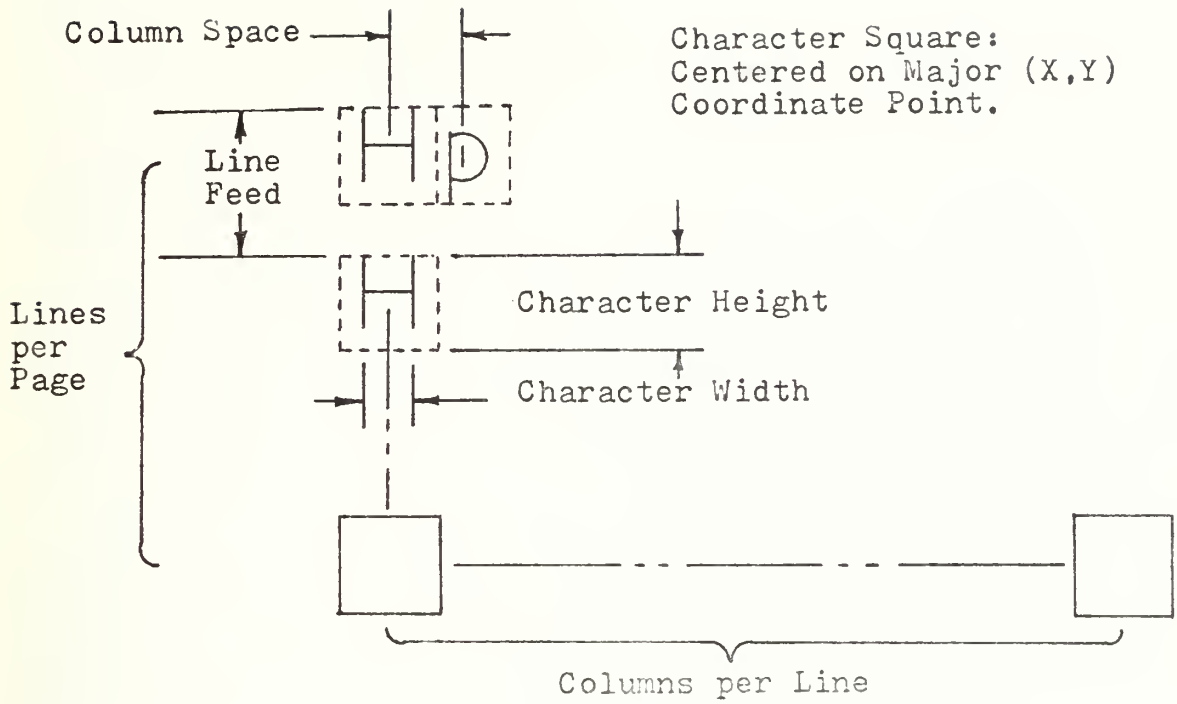
Figure 2-4B. Effect of Intensity Scale Variation

F. CHARACTER GENERATION

The character generator accepts coded inputs from the display list and produces text strings composed of ASCII characters and special characters. Characters are drawn on the screen as a series of short vectors and curves. Unlike the vector generator, however, the character generator draws are generated automatically by the character generator each time a character code is received.

The program can select one of four character sizes. The intensity is determined by the intensity scale register. Continuous character scaling allows the picture scale and coordinate scale to scale the image and characters proportionally. The program can also specify whether the text lines are to be displayed horizontally on the screen or are to be positioned as if on a page that has been rotated ninety degrees clockwise. One of the characters is a cursor. The cursor differs from other displayed characters in that the character following the cursor is drawn in the same place without a column feed. This permits the cursor to be moved over the screen as desired with manual inputs. A hardware feature causes the cursor to blink twice per second.

The dimensions for character generator outputs in number space units are given in Figure 2-5. The standard character set font is shown in Figure 2-6. The codes for each character can be found in Appendix A.



Char. Size	Cols./ Line	Lines/ Page	Col. space Size
0= (1x)	120	60	34
1= (1.5x)	81	41	50
2= (2X)	60	30	68
3= (4x)	32	16	128

Char. Size	Linefeed Size	Char Height	Char Width
0= (1X)	68	34	23
1= (1.5)	100	50	34
2= (2X)	136	68	45
3= (4x)	256	128	85

Figure 2-5. Character Generator Outputs

Figure 2-6. Character Generator Display

G. CONTROL CHARACTERS

Fourteen codes in the character set are used for control purposes only and do not cause a display on the screen. The control characters and their functions are listed in section V.

III. CONTROL DEVICE DESCRIPTION

A. ALPHANUMERIC KEYBOARD

The alphanumeric keyboard is used as an entry device for manual input to the display system. Pressing a key on the keyboard enters an eight bit ASCII character code into the keyboard character queue. The character entered in the keyboard character queue does not directly affect the display on the screen. The program can read the keyboard character via vgetcar() and use the information in its operation. Holding any key down will maintain the correct code and, after an initial delay, will repeat the character. Appendix A lists the codes generated by the keyboard for shifted and unshifted key combinations.

B. LIGHTED FUNCTION SWITCHES

This device contains thirty-two function switches plus a manual interrupt switch. The function switch registers in the display controller have one bit corresponding to each function switch. While any function switch is depressed, the corresponding bit in the function switch register is set. The program can then read the contents of the registers via vgetfsw(). The function switch lamps can be lighted via valamps().

The manual interrupt switch can be used to cause an interrupt. This feature allows the operator to intercept the program at any desired point. When the manual interrupt switch is pressed, the manual interrupt counter, vgmanint, is incremented. The program may reset this counter when desired.

C. LIGHT PEN

The light pen can be used to point at an element of a display or to create information by drawing on the display. The light pen, a wand containing a photocell, is held over the face of the cathode ray tube by the viewer. When the light pen is held over a line or point on the display, the light pen interrupt flag, vglpflag, is set to indicate a light pen interrupt condition. If the light

pen switch is activated, the light pen sense switch interrupt flag, valpsflg, is set to indicate this condition.

A hardware delay feature permits the light pen interrupt state of the Vector General to be obtained prior to additional display processing. The values retained at interrupt time can be obtained by calling vgpetlpn().

D. CONTROL DIALS

Ten control dials may be used to send digital numerical information to the computer for any purpose specified by the program. Each dial is associated with a twelve bit dial input register in the display controller. As a dial is turned, the corresponding register is updated to reflect the dial value. These values may be read at any time by calling vgdial().

IV. USING THE VECTOR GENERAL

The vector general interface design concept is to define high level constructs which the interface routines convert into vector general commands. There are three classes of constructs defined: objects, elements, and the picture. An object is the lowest level construct which

can be displayed alone. Each object is independently rotatable, scalable, and translatable into any portion of the thirty inch by thirty inch picture space. An object can be as large as fifteen inches by fifteen inches and be rotated or positioned to the extreme limits of the picture space without distortion to any of the remaining visible portion. Each object is composed of one or more independently light pen hookable elements. An element is composed of a series of user-drawn images or characters relative to the untransformed image space of its object. An object can be defined unrotated in such a way as to fill the entire object space and then be scaled, rotated, and moved so that the image space is the appropriate size, is viewed from the appropriate aspect, and is in the appropriate area of the picture. The picture defines the picture scale and screen coordinates for all objects.

The user is responsible for the generation and content of each element. Prior to its inclusion within the display list, the user must fill each element with the necessary draw and move commands. In addition, the user must provide three unused words succeeding the draw-move commands. These three words are needed by the interface routines for proper display list termination.

The generation and contents of all objects and the picture is the responsibility of the interface software. A set of routines are provided to link elements to objects and objects to the picture. Dynamic modification of

objects and picture parameters is also provided. The routines for manipulation and modification of the display data structure are:

vaaddele(obp,num,size)	vainit()
vablink(type,num,action)	vaiofset(num,val)
vaclock(rate)	vaiscal(num,val)
vacoord(num,x,y,z)	valamps(obp,action)
vacsr(num,val)	valpen(type,num,action)
vadelele(num)	vamkobj()
vadeleobj(num)	vapicture()
vadial(obp)	vapost(px,py)
vagetcar()	vapscal(val)
vagetfsw(obp)	varotate(num,x,y,z)
vagetlpn(obp)	vaterm()

All picture and object display parameters are initialized to default values. The user may then modify these parameters as desired. The default parameters are:

Display Blink - The picture, all objects, and all elements are defined as non-blink. vablink() can set or clear the blink mode on any of the constructs.

Coordinate Position - The X, Y, and Z coordinates of each object are (0,0,0). The routine vacoord() can move an object to any area of the image space by varying these coordinate values.

Picture Position - The location of the picture coordinates is defined as the center of the screen

(0,0,0). The entire picture can be moved on the X-Y plane by calling vapost().

Coordinate Scale - The default is one, the maximum size. This parameter may be be scaled between zero and one by calling vacsr().

Picture Scale - The default is one. Reducing the picture scale will shrink all of the display. vapscal() modifies this parameter.

Intensity Offset - The maximum intensity offset, value of one, is set as the default parameter. A call to vaiofset() can reduce this value.

Intensity Scale - Initially set to a one, this maximum value can be reduced by the routine vaiscal().

Refresh Rate - A refresh rate of forty hertz has been assigned as the default value. vaclock() can modify this value.

Light Pen Enable - The light pen is initially disabled. A call to valpen() will enable it.

Function Switch Lamps - All function switch lamps are extinguished at display time and can be set by the routine valamps().

In order to compile a program using the display subroutines, global names, and structures the user should invoke the following statement:

cc -f -O filename -lv

This will cause the lexical scope of the vector general display system to extend through the program.

Certain global names and routines are not visible to the user. To avoid colliding with them, the user should refrain from defining external variables and routines that start with the letters "vg".

The user must preface his display actions by a call to vginit() and should call vgterm() before process termination.

V. DISPLAY INSTRUCTIONS

Instructions used in the display system fall into two main types: control instructions (used to control the display operation) and output instructions (used to generate image vectors or characters on the display screen).

A description of the control instructions and their effect on the vector general can be found in the Vector General Graphics Display Unit Reference Manual (VG 101056).

The following paragraphs contain functional descriptions of the various display-list instruction configurations processed by the display system. Each instruction discussion includes a format diagram, a listing of the octal codes for the instruction variations, a definition

of the applicable code, and a description of the purpose of the instruction.

The octal code given assumes the instruction to be an eighteen bit instruction with the first two bits zero.

Operation of the display system consists of processing data words in accordance with their instructions. Instructions that draw lines or manipulate text strings process the data words following the instruction command. The data words give the end point coordinates of the lines or character codes of the text.

Data words are transmitted in a string or block following the applicable instruction. The last data word in the string must contain a coded terminate bit, field, or character to indicate that it is the last word for that instruction.

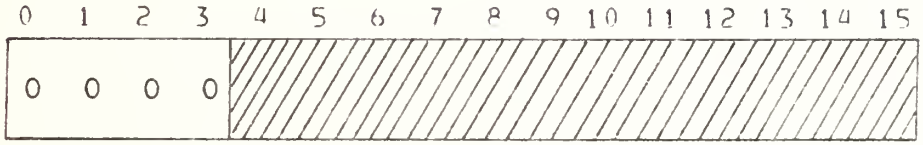
A. WORD FORMATS

The display system uses as its basic informational element a sixteen bit word with the bit positions numbered zero through fifteen as shown in the following diagram. Bit zero is the most significant bit.

B. CONTROL DISPLAY INSTRUCTIONS

1. No Operation

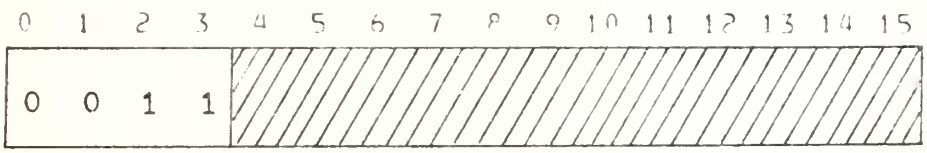
000000



The NOOP instruction may be used to hold data or addresses. The information in bits 4-15 is not interpreted by the vector general.

2. Halt

030000

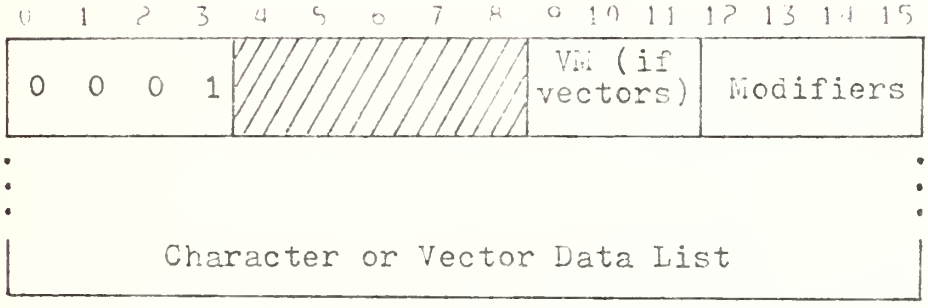


The HLT instruction causes the display system to cease all operations. No further instructions or data words are

accepted. The display system state is set to not-run, not-wait. This command does not disable any of the vector general interrupts.

C. DISPLAY WRITE INSTRUCTIONS

These display instructions and their following data are output as lists over the direct memory access channel to generate visual display elements. The basic word format is as shown in the following diagram.



The image generation instructions are used to present display elements consisting of solid lines, dashed lines, dotted lines, or dash-dot-dashed lines between two positions on the display screen.

The modifier bits (12-15) of the image generation instruction specify if the data words that follow it are to be used for characters, absolute or relative vectors, X, Y, or Z auto-incrementing, or 2D or 3D incremental vectors. The instruction also indicates the type of display (normal, dashed, dot, or point) and the incremental

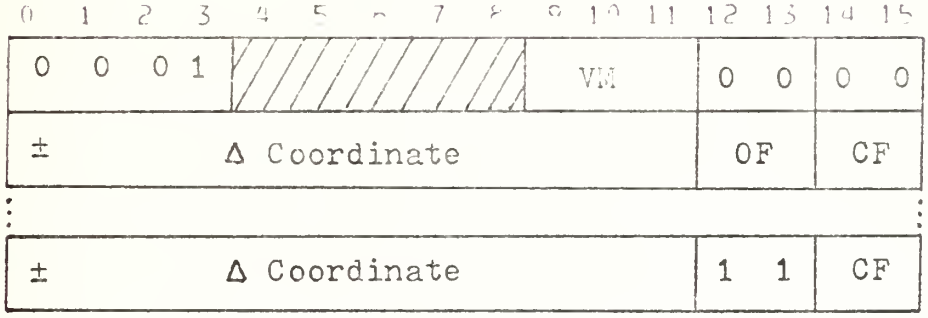
resolution or character scaling to be used.

The character generation instruction indicates the size of the characters to be displayed and whether they are to be displayed horizontally or vertically.

The following descriptions are given for no transformations imposed on the generated image prior to display. The user must specify any desired transformation prior to processing any display generating instructions whose output is to be affected.

1. Vector Relative

010000



The display instruction for relative vectors generates a vector display whose coordinates are relative to the initial contents of the coordinate registers, XR, YR, and ZR. The type of display generated by the moving beam is specified by the vector mode field (VM).

VM	VECTOR MODE
000	Line
001	Dashed Line
010	Dotted Line
011	End Point
101	Dash-Dot-Dash
110	Dash-Dot-Dash

The operation field (OF) of each data word specifies if the beam is to be moved to a new position held in the coordinate registers; also, when moving the beam, it specifies if a vector type (VM) is to be drawn. The OF field also specifies the end of the data list.

OF	OPERATION FIELD
00	Load Register
01	Load, then draw vector
10	Load, move beam (no draw)
11	Load, draw, terminate

Each data word has a signed twelve bit coordinate increment to be added to a coordinate register or to the auto-increment register (AIR). The coordinate field (CF) of each data word specifies which register is to be updated by the coordinate increment.

CF	COORDINATE FIELD
00	Auto-increment register (AIR)

01	X-coordinate register (XR)
10	Y-coordinate register (YR)
11	Z-coordinate register (ZR)

To draw an arc with the circle arc generator use the following procedure:

1. Move the beam or draw to the start point of the arc.
2. Load the end point into the coordinate registers.
3. Issue an arc command. The arc commands are:

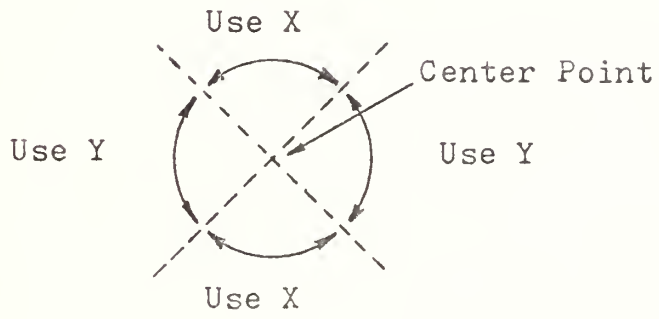
	UF - CF fields
Clockwise arc	000004
Counter clockwise arc	000010

4. Issue a draw or draw-terminate command to the center point of the arc. This defines the radius of the arc.

A circle may be drawn by either using the procedure described above and specifying the same value for both the start point and end point of the arc or by leaving step two of the procedure described above out entirely and just moving or drawing to the start point before issuing the arc command.

If the radius to start and to the end point are not equal, the start point will be used and the circle arc

generator will use either the X or Y coordinate for terminating the draw in accordance with the following diagram. A straight line will be drawn from the terminating point on the radius and the specified end point.



2. Vector Relative Auto-X

010001

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	1	///	///	///	///	///	///	VM		0	0	0	1
±	Δ Coordinate											OF		CF	
⋮															⋮
±	Δ Coordinate											1	1	CF	

The display instruction for vector relative auto-X is processed as a relative vector. Each coordinate value is added to the register designated by CF; then the vector generator performs any function specified by VM and CF. With each move or draw operation, the X-coordinate register (XR) is incremented by the value in the auto-increment

register (AIR) following the load instruction but preceding the move or draw command of the operation.

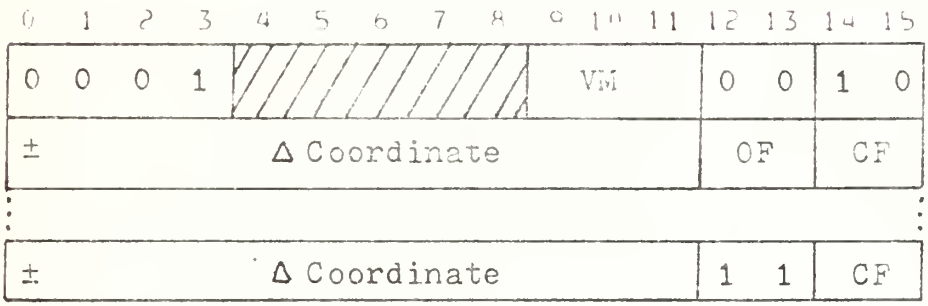
The type of vectors generated is specified by VM as described above.

Control of beam motion and blanking or list termination is specified by OF as described for relative vectors.

Specification of the register to be incremented by the coordinate value is given by CF as described above.

3. Vector Relative Auto-Y

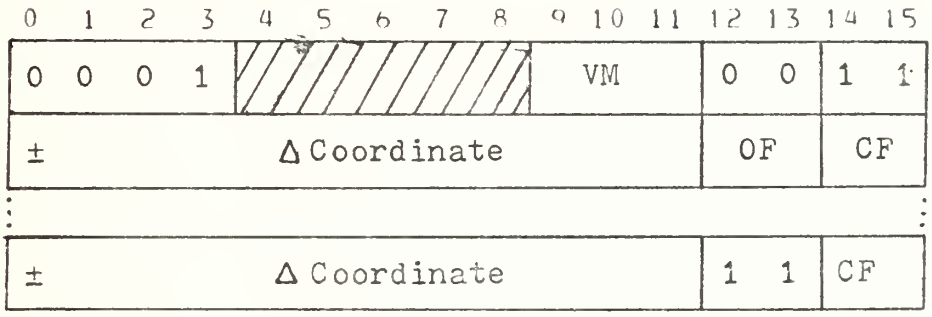
C1002?



The display instruction for vector relative auto-Y is processed as a relative vector. Each coordinate value is added to the register designated by CF; then the vector generator performs any function specified by VM and OF. With each move or draw operation the Y-coordinate register (YR) is incremented by the value in the auto-increment register (AIR) following the load operation but preceding the move or draw portion of the operation. The VM, OF, and CF fields are as described for relative vectors.

4. Vector Relative Auto-Z

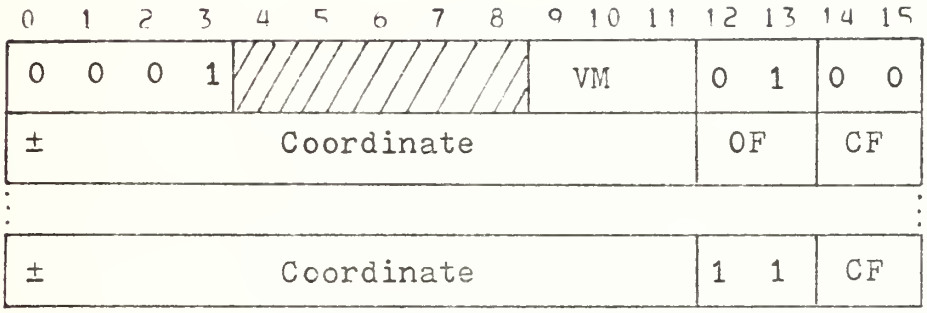
010003



The display instruction for vector relative auto-Z is processed as a relative vector. Each coordinate value is added to the register designated by CF; then the vector generator performs any function specified by VM and OF. With each move or draw operation, the Z coordinate register (ZR) is incremented by the value in the auto-increment register (AIR) following the load portion but preceding the move or draw portion of the operation. The VM, OF, and CF fields are as described for relative vectors.

5. Vector Absolute

010004

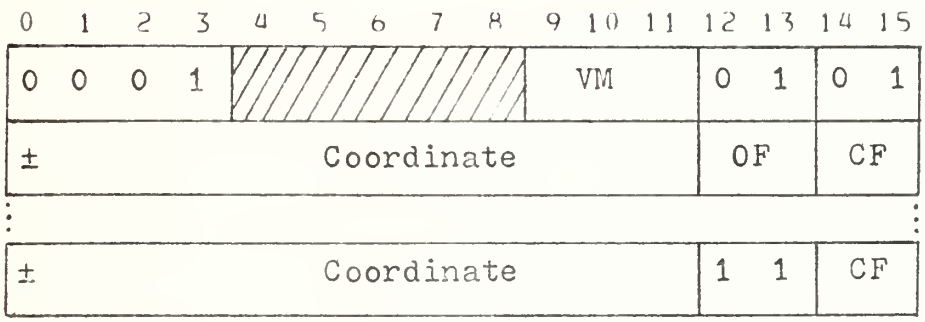


The vector absolute display instruction loads the coordinate value from each of its data words directly into the register specified by CF, replacing the previous contents. The beam position is moved if called for by OF and a vector of type VM is drawn if required by CF. The VM, OF, and CF fields for absolute vectors are the same as described for vector relative.

To draw an arc using the circle arc generator use the procedure described for vector relative.

6. Vector Absolute Auto-X

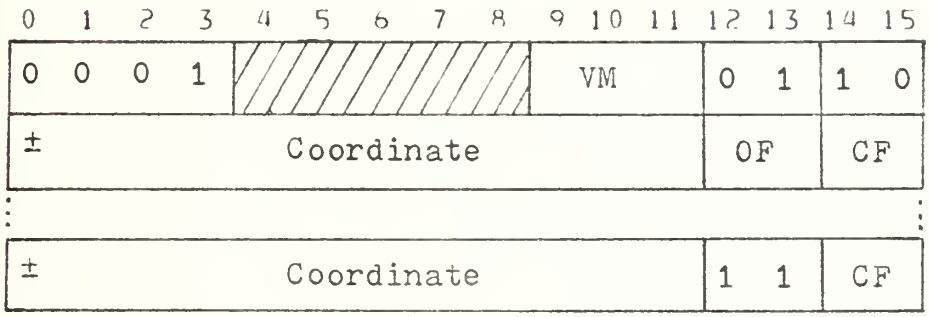
010005



The display instruction for vector absolute auto-x processes its data list as absolute vectors. Each coordinate value is loaded into the register designated by CF; then the vector generator performs any move or VM type draw operation if called for by OF. With each draw or move operation, the x-coordinate register (XR) is incremented by adding the value from the auto-increment register (AIR) following the load portion but preceding the move or draw portion of the operation. The VM, OF, CF fields are used as described for vector relative.

7. Vector Absolute Auto-Y

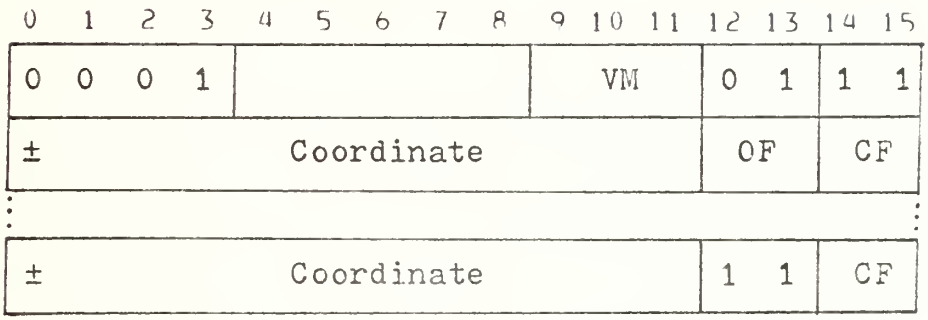
010006



The display instruction for vector absolute auto-Y processes its data list as absolute vectors. Each coordinate value is loaded into the register designated by CF; then the vector generator performs any move or VM type draw operation if called for by OF. With each draw or move operation, the Y-coordinate register (YR) is incremented by adding the value from the auto-increment register (AIR) following the load portion but preceding the move or draw portion of the operation. The VM, OF, CF fields are used as described for vector relative.

8. Vector Absolute Auto-Z

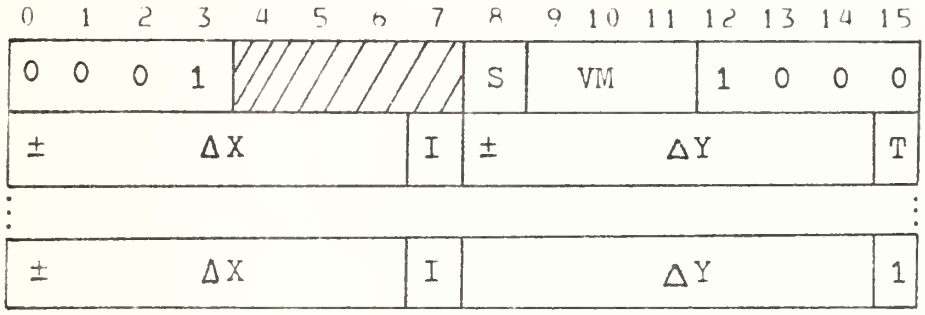
010007



The display instruction for vector absolute auto-Z processes its data list as absolute vectors. Each coordinate value is loaded into the register designated by CF; then the vector generator performs any move or VI type draw operation if called for by OF. With each draw or move operation, the Z-coordinate register (ZR) is incremented by adding the value from the auto-increment register (AIR) following the load portion but preceding the move or draw portion of the operation. The VM, OF, CF fields are used as described for vector relative.

9. Incremental Vectors, 2D

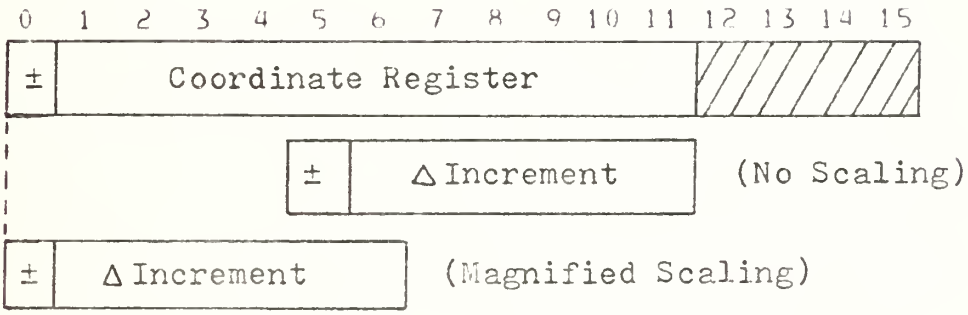
010010



The 2D incremental vector display instructions generate an XY vector display whose coordinates are relative to the initial contents of the coordinate registers. Also, the maximum possible data rate has been doubled and the storage requirements halved (over those of relative vectors). This is done by reducing the coordinate data field by seven-twelfths and packing two values per data word. This performance increase can be exploited where the lower resolution data is adequate and the processing of packed values is not detrimental. The applicability of incremental vectors is enhanced by the scale field (S) which permits the data values to be applied as increments over a coarse or fine grid.

S	INCREMENT SCALE
0	No magnification: add to seven low-order bits
1	Magnified: add to seven high-order bits

by specifying magnification, the coordinate increments are added to the high-order bits of the register being updated; otherwise the increment is sign-extended and added to the low-order bits:



The type of display generated by the moving beam is specified by the vector mode (VM) field.

The I-field of the incremental vector data word controls beam blanking for processing of the entire data word.

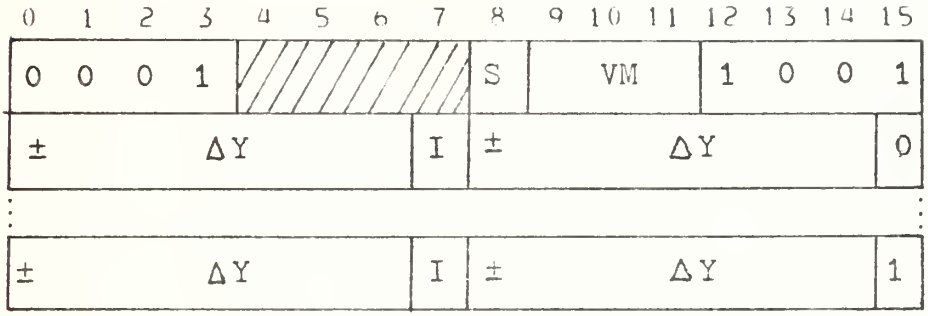
I	INTENSITY FIELD
0	Move beam with no intensification
1	Move beam and draw VM-type vector

The last bit of an incremental vector data-list word is used to flag the end of the data list.

T	TERMINATE FIELD
0	Continue data list
1	Last word of data

10. Incremental Vectors, 2D Auto-X

010011



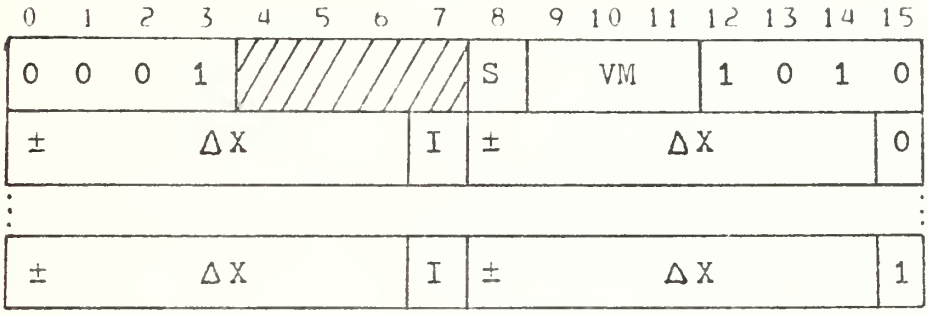
The 2D auto-X display instruction generates a two-dimensional, relative vector display from packed data increments; but the data words supply only the Y increments. The corresponding X-increments are taken as the constant held in the auto-increment register (AIR). This further doubles the possible vector rate and halves the memory requirements for the displays such as graphs where one coordinate is stepped by a constant.

Each data word supplies two Y-increments and, therefore, is used to generate two vectors.

The S, VM, I, and I fields are coded and used as described for incremental vector 2D, but the I-field applies to both vectors generated from its data word, and both vectors are generated from the final data word.

11. Incremental Vectors, 2D Auto-Y

010012



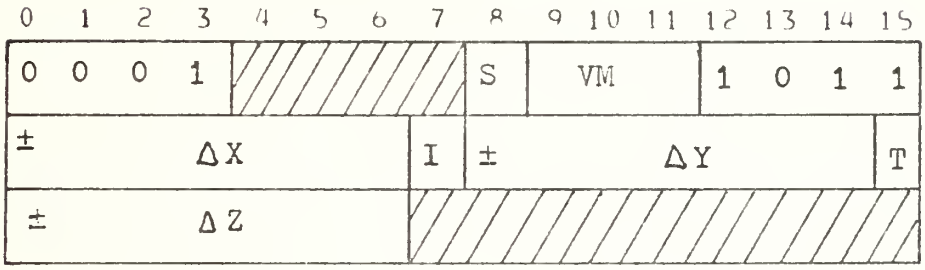
The 2D auto-Y display instruction generates a two-dimensional, relative vector display from packed data increments; but the data words supply only the X increments. The corresponding Y-increments are taken as the constant held in the auto-increment register (AIR). This further doubles the possible vector rate and halves the memory requirements for the displays such as graphs where one coordinate is stepped by a constant.

Each data word supplies two X-increments and, therefore, is used to generate two vectors.

The S, VM, I, and I fields are coded and used as described for incremental vector 2D, but the I-field applies to both vectors generated from its data word, and both vectors are generated from the final data word.

12. Incremental Vectors, Three Dimensional

010013



The 3D incremental vector display instructions generate an XYZ vector display whose coordinates are relative to the initial contents of the coordinate registers. Also, the maximum possible data rate has been increased and the storage requirements reduced (over those of relative vectors). This is done by shortening the coordinate data field width seven-twelfths and packing up to two values per data word. This performance increase can be exploited where the lower resolution data is adequate and the processing of packed values is not detrimental. The applicability of incremental vectors is enhanced by the scale field (S) which permits the data values to be applied as increments over a coarse or fine grid.

One vector is generated for every two data words processed.

The S, VM, I, and T fields are coded and used as

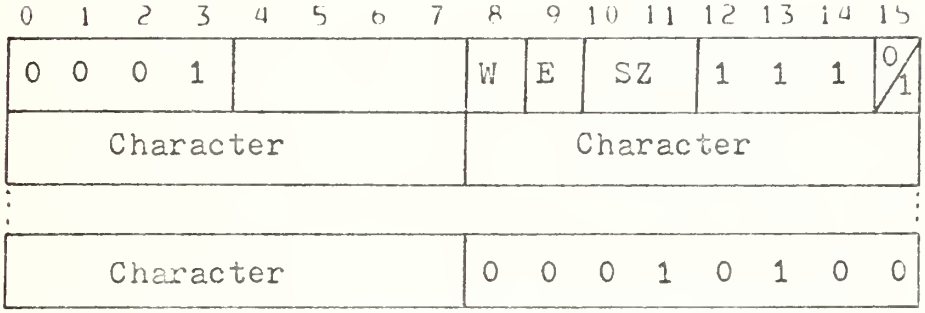
described for incremental vectors, 2D.

13. Character Generation

010017

Bit 15

1 - No slant
0 - slant



The character generation display instruction processes its data as a string of extended ASCII character codes packed two per word.

Each successive character displays a symbol or performs a control function until a terminate character, ASCII code DC4, is processed signaling the end of the instruction's data list.

The symbols available include all of the ninety-six ASCII graphics, plus a standard set of ninety-six additional sysmbols (programming, math, greek, etc.), and an optional set of 32 user-specified special symbols.

The standard symbols and their codes are given in Appendix A.

The direction field (W) when set causes the characters to be displayed as if on a page which has been rotated

ninety degrees counter clockwise.

W	CHARACTER WRITE-DIRECTION
0	Write characters horizontally
1	Write characters vertically

The size field (SZ) is used to specify one of the four available string-controlled character sizes. The size-enable bit (E) causes the contents of the SZ field to be instated as the new character size for the subsequent character generation.

E	SZ	CHARACTER SIZE CONTROL
0	xx	Use previous character size
1	00	Set size to one hundred columns by sixty lines
1	01	Set size to eighty-one columns by fifty-one lines
1	10	Set size to sixty columns by thirty lines
1	11	Set size to thirty-two columns by sixteen lines

CONTROL CHARACTERS

CHARACTER	FUNCTION
DELETE	No display is generated and the text is not stepped to the next character.
NULL	Same as DELETE.

BACKSPACE	Causes the positioning to revert to the previous character position.
LINE FEED	Causes the current line poistion to be increased by one line.
FORM FEED	Instates the current character positioning at the first character of line one.
CARRIAGE RETURN	Resets the current column position to position one, the left margin, and increases the current line position by one line.
DC1	Reduces the current line position by one line.
DC2	Decreases the current character size by one size. Permits subscript and superscript sizes to be encoded in text. Size zero is changed to size three.
DC3	Increases the current character size by one size. Size three is changed to size zero.
DC4	When encoded in a display list, this code terminates the data associated with a character generation instruction. If input via the keyboard, it is interpreted as a process termination code. It has the same affect as

	the "rubout" key on a DATAMEDIA terminal.
DLE	When encoded in a display list, this code is ignored. When input from the keyboard, it is interpreted as a command to clear all characters from the keyboard character queue.
ESC	When encoded in a display list, this code is ignored. When input from the keyboard, it is interpreted as an escape code for the next character. When ESC is depressed, the next character will be passed as data regardless of the content. This allows the DC4 code to be input as data from the keyboard.
HORIZONTAL TAB	Resets the current column position to horizontal center and increases the current line position by one line.
VERTICAL TAB	Instates the current character positioning to horizontal center of line one.

VI. USER ROUTINES

The description of the user routines are described in the same format as the routine descriptions given in the Unix Reference Manual.

NAME:

vgaddobj - add object

SYNOPSIS:

```
vgaddobj(num)
int num;
```

DESCRIPTION:

The object indicated by num is added to the active display list.

Normal return is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative, zero, or greater than the maximum permitted in the system.

-2 The object doesn't exist

-3 The picture cannot link any more objects

NAME:

vgaddele - add element

SYNOPSIS:

```
vgaddele(abp,num,size)
int *abp;
int num,size;
```

DESCRIPTION:

abp is a pointer to a display list with all draw-move commands defined. The byte count of the display list is the parameter size. This value includes the three words (six bytes) that must be added after the draw-move commands. The display list is linked to the object, num.

The value returned is the element number of the display list and should be retained for subsequent element identification.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number provided is negative or zero.
-2 The object has not been defined by vgmkobj().
-3 The object cannot link additional elements.
-4 The system has generated the maximum number of elements permitted.
-5 The display list is less than six bytes long.
-6 The display list buffer pointer is zero.

SEE ALSO:

vgmkobj(), vgdelele()

NAME:

vgblink - set or clear display blink mode

SYNOPSIS:

```
vgblink(type,num,action)
char type,action;
int num;
```

DESCRIPTION:

The display blink mode of the entire picture, a single object, or a single element may be set or cleared. The following parameter values apply:

TYPE:
- 0 - Modify the entire picture.
- 1 - Modify the object, num.
- 2 - Modify the element, num.

ACTION:
- 0 - Clear the display blink mode
- 1 - Set the display blink mode

Modifying the blink mode of an object affects all elements linked to that object. Likewise, modification of the picture affects all objects comprising the picture.

The normal return value is a zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 Object number is negative or zero.

-2 The object or element does not exist.

-4 The element number is negative, zero, or greater than the maximum number of elements permitted in the entire system.

NAME:

vaclock - set refresh rate

SYNOPSIS:

```
vaclock(rate)
int rate;
```

DESCRIPTION:

Set the refresh rate of the display. The parameter, rate, is in hertz. The value should be an integer divisor of one hundred twenty. The default value at display initialization is forty hertz.

NAME:

vgcoord - modify object X-, Y-, and Z-coordinates

SYNOPSIS:

```
vgcoord(num,x,y,z)
int num,x,y,z;
```

DESCRIPTION:

The image space X,Y,Z coordinate values of the object, num, are set to the values of x,y,z respectively. Each coordinate point of all elements linked to the object is translated by the x,y,z values at display time. The contents of the original display list remain unchanged.

The range of values is -2047 through +2047. The system default is zero for the X, Y, and Z coordinates.

The normal return value is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative or zero.

-2 The object is not defined.

NAME:

vgcsr - modify object coordinate scale

SYNOPSIS:

```
vgcsr(num,val)
int num;
double val;
```

DESCRIPTION:

Change the image space coordinate scale of the object, num, to val. Each coordinate of all elements linked to the object is scaled to val at display time. The contents of the original display list remain unchanged.

The range of val is zero to one. The scale factor is one, the maximum scale.

The normal return value is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are listed:

-1 The object number is negative or zero.

-2 The object is not defined.

SEE ALSO:

vgpscal()

LIMITATIONS:

Negative input values cause the coordinates to be inverted. This gives the appearance of viewing the object from the negative Z axis.

NAME:

vqdelele - delete element

SYNOPSIS:

```
vqdelele(num)
int num;
```

DESCRIPTION:

Remove the element, num, from the display system. The contents of the display list remain unchanged; however, the display cannot be accessed by the display system until again linked to some object.

The normal return value is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-2 The element does not exist.

-4 The element number is negative, zero, or greater than the maximum number of elements permitted in the system.

SEE ALSO:

vqaddele(), vqaddobj()

NAME:

vgdelobj - delete object

SYNOPSIS:

```
vgdelobj(num)
int num;
```

DESCRIPTION:

Remove the object, num, from the display system. This also deletes all elements linked to the object. The contents of the display lists linked to the object remain unchanged; however, the display cannot access any of the elements until again linked to some object.

The normal return value is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative or zero.

-2 The object has not been defined.

SEE ALSO:

vgdelele(), vgmkobj()

NAME:

vgdial - get analog dial values

SYNOPSIS:

```
vgdial(abp)
int *abp;
```

DESCRIPTION:

Obtain the ten vector general dial values and return them to the caller beginning at the ten word buffer pointed to by abp.

DIAGNOSTICS:

The only error code is -6. This means the buffer pointer is zero.

NAME:

vqgetcar - get keyboard character

SYNOPSIS:

vqgetcar()

DESCRIPTION:

An eight bit ASCII character is returned to the caller. If no character has been input a -1 is returned.

A four character queue is maintained by the system. This permits some variation in processing time without character loss. The queue can be cleared of all previous characters by a DLE character from the vector general keyboard.

NAME:

vqgetfsw - get function switch values

SYNOPSIS:

```
vqgetfsw(abp)
int *abp;
```

DESCRIPTION:

Read the thirty-two function switches and return them to the caller starting at the two word buffer pointed to by abp. Each bit represents one function switch. If a bit is set, the function switch is depressed.

DIAGNOSTICS:

A -6 is returned if the buffer pointer is zero.

SEE ALSO:

valarms()

NAME:

vgaetlpn - get light pen interrupt values

SYNOPSIS:

```
vgaetlpn(abp)
int *abp;
```

DESCRIPTION:

Seven light pen interrupt values are read into a seven word buffer beginning at the buffer pointed to by abp.

The meaning of each word is:

Word 0 - The element number of the display list being displayed at the time of the interrupt.

Word 1 - Instruction word being executed at the time of the interrupt.

Word 2 - The number of words executed from the beginning of the display.

Word 3,4,5 - The X,Y,Z coordinates respectively at the time of the interrupt.

Word 6 - Pen resolution count.

DIAGNOSTICS:

A return of -6 indicates a buffer pointer of zero.

SEE ALSO:

vglpen()

LIMITATIONS:

Word 2 does not provide any useful information to the user. Its value is the total number of instructions executed from the start of the picture. It has no relation to the instruction count of an element.

NAME:

vginit - vector general display initialization

SYNOPSIS:

vginit()

DESCRIPTION:

This routine performs all display initialization and default parameter assignment. The vector general is accessed, cleared, and reset for display operations.

This routine must be called before any display actions.

DIAGNOSTICS:

If the initialization cannot be completed, an error message is printed and the process is terminated. Error messages usually occur because another process is a real-time process or because the vector general has been accessed by another user.

NAME:

vaiofset - modify intensity offset

SYNOPSIS:

```
vaiofset(num,val)
int num;
double val;
```

DESCRIPTION:

Set the maximum level of intensity permitted for the object, num. val, between zero and one, is a measure of the maximum intensity of the object. The intensity is maximum at the face of the screen and decreases exponentially toward the back of the image. If val is negative, a screen cut-off is imposed. If the transformed values of the Z coordinate axis are greater than 0.02, they are blanked.

The normal return value is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative or zero.

-2 The object does not exist.

SEE ALSO:

vaiscal()

NAME:

vgiscal - modify object intensity scale

SYNOPSIS:

```
vgiscal(num,val)
int num;
double val;
```

DESCRIPTION:

The intensity range of the object, num, is determined by val. If val is one, the maximum intensity range is achieved. If the value is zero, the intensity is constant and the image has no depth-cueing.

Normal return is zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative or zero.
-2 The object does not exist.

SEE ALSO:

vgiofset()

NAME:

vglamps - light function switch lamps

SYNOPSIS:

```
vglamps(abp)
int *abp;
```

DESCRIPTION:

Light the function switch lamps of those function switches with a bit set in the two word buffer pointed to by abp. Each bit position corresponds to one of the thirty-two function switch lamps.

DIAGNOSTICS:

If the buffer pointer is zero, a -6 is returned.

SEE ALSO:

vggetfsw()

NAME:

valpen - set or clear light pen hookability

SYNOPSIS:

```
valpen(type,num,action)
char type;
int num,action;
```

DESCRIPTION:

The light pen hookability of the entire picture, a single object, or a single element may be set or cleared. The following parameter values apply

TYPE
0
Modify the entire picture.
1
Modify the object, num.
2
Modify the element, num.

ACTION
0
Clear the light pen hookability.
1
Set the light pen hookability.

The normal return value is a zero.

DIAGNOSTICS:

All errors are indicated by negative return values. The error values and their meanings are as follows:

-1 The object number is negative or zero.

-2 The object or element does not exist.

-4 The element number is negative, zero, or greater than the maximum number of elements permitted in the entire system.

NAME:

vgmkobj - make object

SYNOPSIS:

vgmkobj()

DESCRIPTION:

Create an empty object structure and return the object number to the caller.

DIAGNOSTICS:

A -3 is returned if all available objects have been previously assigned.

NAME:

vgpicture - display picture

SYNOPSIS:

vgpicture()

DESCRIPTION:

Send the vector general display list and start the display.

NAME:

vgpost - modify the picture coordinate axis

SYNOPSIS:

```
vgpost(px,py)
int px,py;
```

DESCRIPTION:

Modify the transformed X,Y coordinate points of each image in the picture by px and py respectively. The original display lists remain unchanged. The range of values are -2047 through +2047.

SEE ALSO:

vgcoord()

NAME:

vapscal - modify picture scale

SYNOPSIS:

```
vapscal(val)
double val;
```

DESCRIPTION:

Scale each coordinate of all elements by val. The original display list remains unchanged. A value of one gives the largest picture. A value of zero the smallest.

NAME:

vgrotate - rotate object

SYNOPSIS:

```
vgrotate(num,x,y,z)
int num;
double x,y,z;
```

DESCRIPTION:

Rotate the object, num, about the X, Y, and Z axis. The input parameters are radian measure rotation about the axis indicated. All elements linked to the object are affected. The original display lists remain unaffected.

NAME:

vgterm - terminate vector general display

SYNOPSIS:

vgterm()

DESCRIPTION:

Terminate access to the vector general and release all system resources. This should be called at the conclusion of all display operations.

APPENDIX A

ASCII CHARACTER CODES

Table A-1 lists the ASCII codes used by the display system for the various general and special characters. The codes are given in octal notation. The octal codes are given as though there were eighteen bits in the data word instead of sixteen bits. Since two characters can be given in each data word, the octal codes are given for the right half-word and the left half-word. The left half-word code is given as though there were no character in the right half-word. To obtain the complete code for the two characters in a word, the user must add the two codes together. For example, if the character C is to be in the left half-word and the character A is to be in the right half-word, the code would be:

```
C      041400
A        0101
------------
CA     041501
```

TABLE A-1

OCTAL LEFT	OCTAL RIGHT	CHARACTER GENERATOR SYMBOL	KEYBOARD SYMBOL
000000	000	NULL (ignored)	@ ctrl
000400	001	SO (ignored)	A ctrl
001000	002	STX (ignored)	B ctrl
001400	003	ETX (ignored)	C ctrl
002000	004	EOT (ignored)	D ctrl
002400	005	ENQ (ignored)	E ctrl
003000	006	ACK (ignored)	F ctrl
003400	007	BEL (ignored)	G ctrl
004000	010	BS	BS
004400	011	HT (LF, cent)	I ctrl
005000	012	LF	LF
005400	013	VT (top, cent)	K ctrl
006000	014	FF (top, left)	L ctrl
006400	015	NL (CR, LF)	CR
007000	016	SF (ignored)	N ctrl
007400	017	SI (ignored)	O ctrl
010000	020	DLE (clear queue)	P ctrl
010400	021	DC1 (-LF)	Q ctrl
011000	022	DC2 (-SZ)	R ctrl
011400	023	DC3 (+SZ)	S ctrl
012000	024	DC4 (term)	T ctrl
012400	025	NAK (ignored)	U ctrl
013000	026	SYN (ignored)	V ctrl
013400	027	ETB (ignored)	W ctrl
014000	030	CAN (ignored)	X ctrl
014400	031	EM (ignored)	Y ctrl
015000	032	SUB (ignored)	Z ctrl
015400	033	ESC (esc)	[ctrl
016000	034	FS (ignored)	ctrl
016400	035	GS (ignored)	] ctrl
001700	036	RS (ignored)	ctrl
017400	037	US (ignored)	
020000	040	Space	sp bar
020400	041	!	1 shift
021000	042	"	2 shift
021400	043	#	3 shift
022000	044	$	4 shift
022400	045	%	5 shift
023000	046	&	6 shift
023400	047	'	7 shift
024000	050	(	8 shift
024400	051	)	9 shift
025000	052	*	: shift
025400	053	+	; shift
026000	054	,	,
026400	055	-	-
027000	056	.	.

TABLE A-1

027400	057	/	/
030000	060	0	0
030400	061	1	1
031000	062	2	2
031400	063	3	3
032000	064	4	4
032400	065	5	5
033000	066	6	6
033400	067	7	7
034000	070	8	8
034400	071	9	9
035000	072	:	:
035400	073	;	;
036000	074	<	, shift
036400	075	=	- shift
037000	076	>	. shift
037400	077	?	?
040000	100	@	@
040400	101	A	A shift
041000	102	B	B shift
041400	103	C	C shift
042000	104	D	D shift
042400	105	E	E shift
043000	106	F	F shift
043400	107	G	G shift
044000	110	H	H shift
044400	111	I	I shift
045000	112	J	J shift
045400	113	K	K shift
046000	114	L	L shift
046400	115	M	M shift
047000	116	N	N shift
047400	117	O	O shift
050000	120	P	P shift
050400	121	Q	Q shift
051000	122	R	R shift
051400	123	S	S shift
052000	124	T	T shift
052400	125	U	U shift
053000	126	V	V shift
053400	127	W	W shift
054000	130	X	X shift
054400	131	Y	Y shift
055000	132	Z	Z shift
055400	133	[	[
056000	134	\	\
056400	135	]	]
057000	136	^	^
057400	137	(sun-, superscript)	
060000	140	`	a shift

TABLE A-1

060400	141	a		A
061000	142	b		B
061400	143	c		C
062000	144	d		D
062400	145	e		E
063000	146	f		F
063400	147	g		G
064000	150	h		H
064400	151	i		I
065000	152	j		J
065400	153	k		K
066000	154	l		L
066400	155	m		M
067000	156	n		N
067400	157	o		O
070000	160	p		P
070400	161	q		Q
071000	162	r		R
071400	163	s		S
072000	164	t		T
072400	165	u		U
073000	166	v		V
073400	167	w		W
074000	170	x		X
074400	171	y		Y
075000	172	z		Z
075400	173	{		[shift
076000	174	!		\ shift
076400	175	}		] shift
077000	176	~		^ shift
077400	177	del		DEL
100000-	200-			
117400	237			
120000	240	□		space spec
120400	241	↓		1 shft spec
121000	242	‖		2 shft spec
121400	243	○		3 shft spec
122000	244	£		4 shft spec
122400	245	∨		5 shft spec
123000	246	∫	(centered)	6 shft spec
123400	247	∧		7 shft spec
124000	250	⊂		8 shft spec
124400	251	⊃		9 shft spec
125000	252	10	(subscript)	: shft spec
125400	253	÷		; shft spec
126000	254	≤		, spec
126400	255	≡		- spec
127000	256	≥		. spec
127400	257	∫		/ spec
130000	260	°		0 spec

TABLE A-1

130400	261	↑		1 spec
131000	262	∀		2 spec
131400	263	□		3 spec
132000	264	∝		4 spec
132400	265	∧	(centered)	5 spec
133000	266	∂		6 spec
133400	267	∠		7 spec
134000	270	∪		8 spec
134400	271	∩		9 spec
135000	272	•	(center dot)	: spec
135400	273	×		; spec
136000	274	←		, shft spec
146400	315	⊞		M shft spec
147000	316	η		N shft spec
147400	317	Ω		O shft spec
150000	320	π		P shft spec
150400	321	∩		Q shft spec
151000	322	⊅		R shft spec
151400	323	Σ		S shft spec
152000	324	∂		T shft spec
152400	325	∣		U shft spec
153000	326	∫		V shft spec
153400	327	∧		W shft spec
154000	330	⊦		X shft spec
154400	331	⌐		Y shft spec
155000	332	⌐		Z shft spec
155400	333	⌊		[shft spec
156000	334	⇒		spec
156400	335	⊢		] shft spec
157000	336	∣ *		^ spec
157400	337	. ∘ *		_ spec
160000	340	⌊	(blinking)	@ shft spec
160400	341	∝		A shft spec
161000	342	β		B shft spec
161400	343	∇		C shft spec
162000	344	δ		D shft spec
162400	345	∈		E shft spec
163000	346	∅		F shft spec
163400	347	∂		G shft spec
164000	350	⊥		H shft spec
164400	351	ι		I shft spec
165000	352	/		J shft spec
165400	353	\ *		K shft spec
166000	354	λ		L shft spec
166400	355	μ		M shft spec
167000	356	ν		N shft spec
167400	357	ω		O shft spec
170000	360	π		P shft spec
170400	361	Б		Q shft spec
171000	362	ρ		R shft spec

TABLE A-1

171400	363	σ	S shft spec
172000	364	τ	T shft spec
172400	365	υ	U shft spec
173000	366	φ	V shft spec
173400	367	χ	W shft spec
174000	370	ψ	X shft spec
174000	371	ω	Y shft spec
174400	372	ζ	Z shft spec
175000	373	[	[spec shft
175400	374	|	spec shft
176000	375	]	] spec shft
176400	376	~ *	spec shft
177000	377	▥	DEL spec

* Superscript

APPENDIX B
SAMPLE PROGRAM

The sample program described here is a box containing a zigzag line and the word box. This program is not intended to illustrate all of the features of the vector general interface but to illustrate one way of developing a display picture.

```
 1  #
 2  #define MKOBJ  0          //  make object
 3  #define ADELE  1          //  add element
 4  #define ADOBJ  2          //  add object
 5
 6  // Display list describing a box
 7  int box [10]
 8      {
 9       010004,              //  vector absolute inst
10      0100001,              //  load X coordinate
11      0100012,              //  load Y coord. and move
12       077765,              //  load X coord. and draw
13       077766,              //  load Y coord. and draw
14      0100005,              //  load X coord. and draw
15      0100016,              //  load Y coord., draw & term.
16      0000000,              //  Noop
17      0000000,              //  Noop
18      0000000               //  Noop
19      };
20
21  // Character display list spelling the word box
22  int word [7]
23      {
24      010157,               //  char generation inst
25      010440,               //  neg. line feed & space
26      041157,               //  ASCII bytes, B and O
27      074024,               //  ASCII bytes, X & term.
28      000000,               //  Noop
29      000000,               //  Noop
30      000000                //  Noop
31      };
32
```

```
33  // Zigzag line display list
34  int zig[12]
35      {
36      010011,             //  2D vector incremental,
37                          //  X auto-increment
38      077176,             //  move Y
39      077602,             //  increment X, draw Y
40      077602,             //  increment X, draw Y
41      077602,             //  increment X, draw y
42      077602,             //  increment X, draw Y
43      077602,             //  increment X, draw Y
44      077602,             //  increment X, draw Y
45      077603,             //  increment X, draw Y & term.
46      000000,             //  Noop
47      000000,             //  Noop
48      000000              //  Noop
49      };
50
51
52
53  main()
54      {
55      int ele1, ele2, ele3;
56      int obj1, obj2, obj3;
57
58
59      vainit();   // initiate vector general
60
61  //Make three objects
62      if ((obj1=vamkobj()) < 0) error(MKOBJ);
63      if ((obj2=vamkobj()) < 0) error(MKOBJ);
64      if ((obj3=vamkobj()) < 0) error(MKOBJ);
65
66  //Add the display lists to the objects
67      if ((ele1=vaaddele(box,obj1,20)) < 0)
68                          error(ADELE);
69      if ((ele2=vaaddele(word,obj2,14)) < 0)
70                          error(ADELE);
71      if ((ele3=vaaddele(zig,obj3,24)) < 0)
72                          error(ADELE);
73
74  //Add the objects to the picture
75      if (vaaddobj(obj1) < 0) error(ADOBJ);
76      if (vaaddobj(obj2) < 0) error(ADOBJ);
77      if (vaaddobj(obj3) < 0) error(ADOBJ);
78
79      vapicture();    // start display of picture
80      sleep(50);      // wait 50 seconds
81      vaterm();       // terminate display operations
82      }
```

```
83
84
85  error(msg)
86      int msg;
87      {
88      switch(msg)
89          {
90          case MKOBJ:
91              {
92              printf("make object error#");
93              break;
94              }
95          case ADELE:
96              {
97              printf("add element error#");
98              break;
99              }
100         case ADOBJ:
101             {
102             printf("add object error#");
103             break;
104             }
105         }
106     vaterm();
107     exit();
108     }
```

BIBLIOGRAPHY

1. Howard, J. E. and Thorpe, L. A., Proposed Design specification Manual for the Vector General Graphics Display Unit, paper presented at Naval Postgraduate School CS4204 Class, June 1975

2. Howard, J. E. and Thorpe, L. A., Proposed Users Manual for the Vector General Graphics Display Unit, paper presented at the Naval Postgraduate School CS4202 Class, June 1975

3. Ritchie, D. E. and Thompson, K., The UNIX Timesharing System, Communications of the ACM, v. 17, no. 7, p 365-375, July, 1974.

4. Naval Postgraduate School Technical Report NPS72Rr76u302, Design Manual for the Interactive Graphics Display Unit, by L. A. Thorpe and G. M. Raetz, March 1976

5. Naval Postgraduate School Technical Report NPS72Rr76031, Users Manual for the Vector General Graphics Display Unit, by L. A. Thorpe and G. M. Raetz, March 1976

6. Vector General Inc., Alphanumeric Keyboard KB1 Option Reference Manual, Vector General, Inc., Woodland Hills, California May 1974

7. Vector General Inc., Analog Devices Option Reference Manual, Vector General Inc., Woodland Hills, California, August 1974

8. Vector General Inc., Circle-Arc Generator Reference Manual, Vector General Inc., Canoga Park, California, February 1974

9. Vector General Inc., Function Switch Option, Vector General, Inc., Canoga Park, California, April 1974

10. Vector General Inc., Graphics Display System Reference Manual, Vector General Inc., Woodland Hill, California, August 1974

11. Vector General Inc., Graphics Display System Technical Manual, Vector General Inc., Woodland Hills, California, June 1974

12. Vector General Inc., Light Pen LP3 Option Reference Manual, Vector General Inc., Canoga Park, California, May 1974

13. Vector General Inc., PDP-11 Interface Option (with Sub-Stack) Reference Manual, Vector General Inc., Woodland Hills, California, August 1974

INITIAL DISTRIBUTION LIST

	Copies
Dean of Research Code 023 Naval Postgraduate School Monterey, California 93940	1
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
Library (Code 0212) Naval Postgraduate School Monterey, California 93940	2
G. R. Church Computer Center Naval Postgraduate School Monterey, California 93940	1
Naval Electronics Systems Command (ELEX 320) Attn: Capt. A. Miller Department of the Navy Washington, D. C. 20360	1
Naval Electronics Laboratory Center Library 271 Catalina Boulevard San Diego, California 92152	1
Ltjg. Gary W. Raetz, USN (Code 72Rr) Naval Postgraduate School Monterey, California 93940	10
Lt. Lloyd A. Thorre, USN 915 24th Street West Billings, Montana 59102	1