EK-VT48-TM-001

VT48
display processor
unit
technical manual

digital equipment corporation • maynard, massachusetts

1st Edition, November 1976

Printed in U.S.A.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

CONTENTS

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

ILLUSTRATIONS (Cont)

ILLUSTRATIONS (Cont)

TABLES

CHAPTER 1
GENERAL DESCRIPTION

1.1 INTRODUCTION

This manual describes the VT48 Display Processor Unit (DPU) shown in Figure 1-1. The VT48 is a high-speed, interactive graphics processor. It is a major component of the VS60 Vectoring System, used to drive the display monitor VR48 Table and Scope Assembly. Instructions are received by the VT48 Display Processor Unit from a PDP-11.

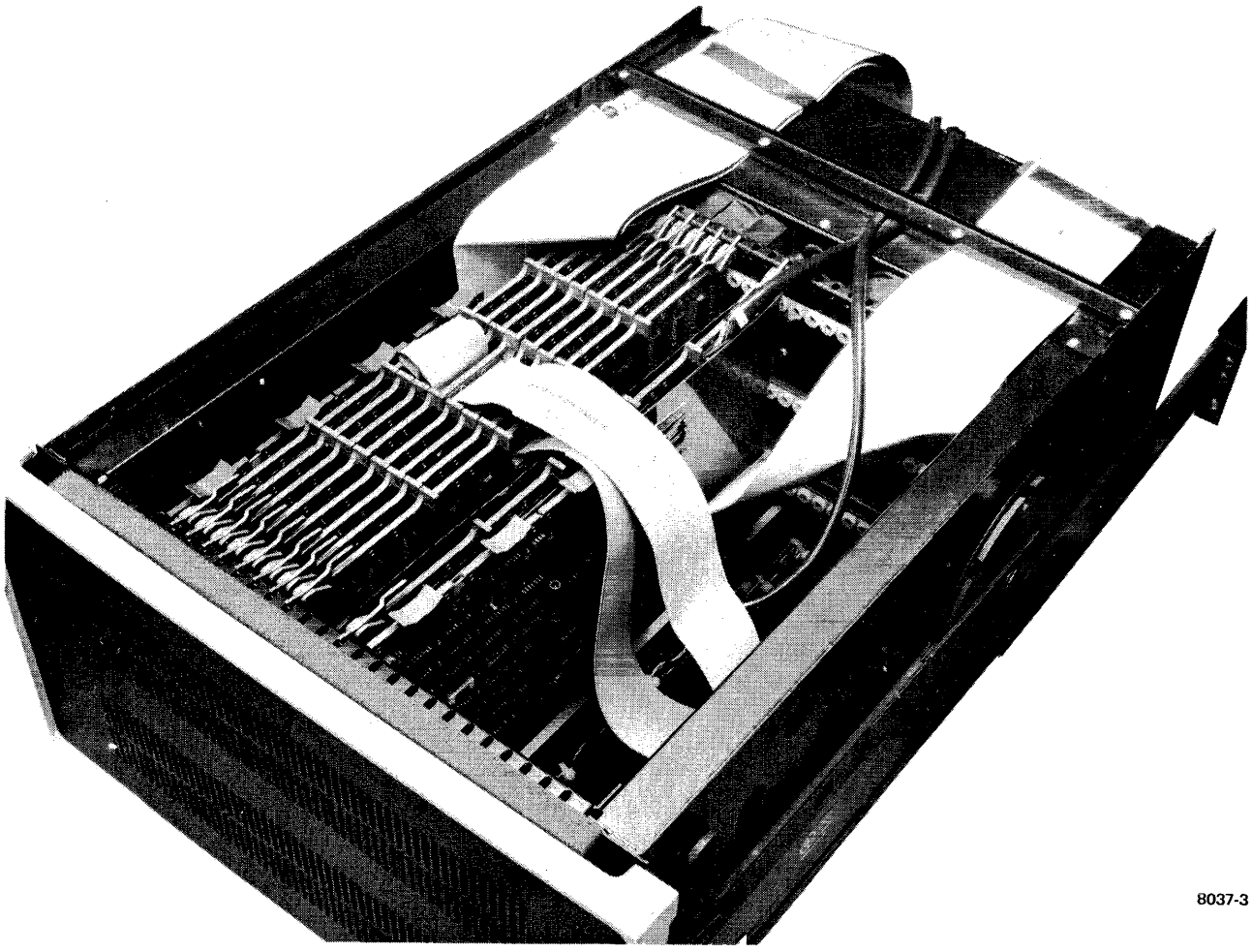


Figure 1-1 VT48 Display Processor Unit in Extended Position

1.2 PHYSICAL DESCRIPTION

The VT48 is contained in a standard H967 cabinet assembly. DC operating voltages are supplied from an H7070 Power Supply mounted at the base of the cabinet. The VT48 is contained in a BA11-K unit assembly that houses 11 printed circuit modules. Access to the printed circuit modules is attained by drawing out the unit on its slide rails and removing the top cover (Figure 1-1). Table 1-1 lists the printed circuit modules and the circuitry contained on each.

Table 1-1 Printed Circuit Modules

BA11-K Position	Type No.	Size	Circuitry
13	M7059	Quad	Unibus Control Logic
12	M7058	Hex	Display Instruction Control
11	M7057	Hex	Display Status Registers
10	M7056	Hex	Stack/Silo Memory
9	M7055	Hex	Read Status Multiplexer
8	M7054	Hex	Stack and Silo Control
7	M7053	Hex	Scissoring and Tangent Control*
6	M7052	Hex	Dual 12 by 12 Multiplier*
5	M7051	Hex	Register and Multiplexer Board*
2	A322	Hex	Vector Generator
1	A321	Quad	Character Generator

*Part of Graphics Calculation Logic.

Access to the VT48 backplane is gained by pulling on both slide handles (Figure 1-2) and simultaneously lifting the unit. Three different tilt positions are available to service the backplane area.

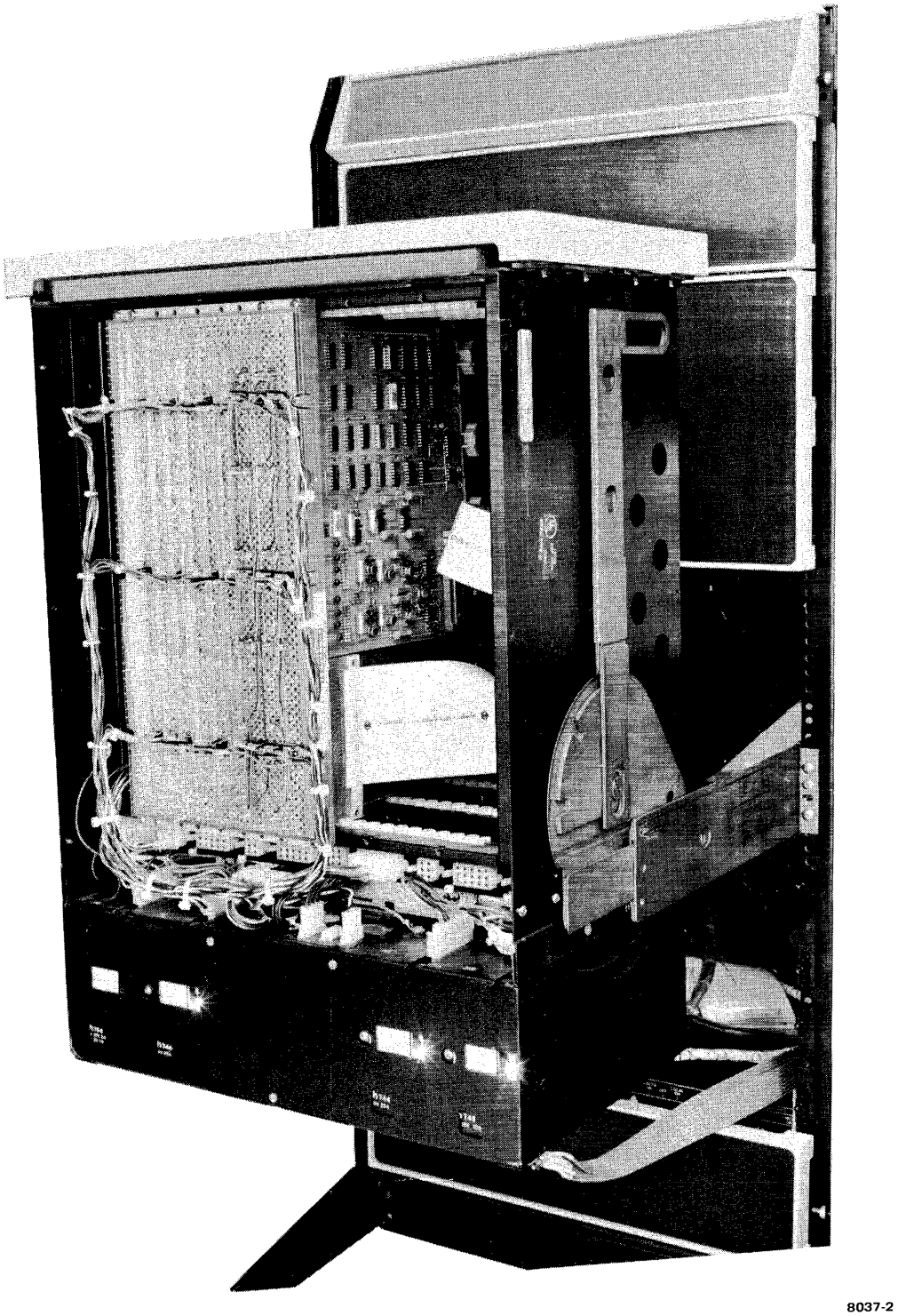


Figure 1-2 VT48 Display Processor Unit Tilted to Access Backplane

1.3 FUNCTIONAL DESCRIPTION

The VT48 Display Processor Unit is a high-speed interactive graphics processor used to drive a VR48 Table and Scope Assembly display monitor. The VT48 detects and executes 25 separate instructions in processing a display file to produce and refresh images at the display monitor CRT (Cathode Ray Tube). The source of the display file is a host processor that can be any of the various PDP-11 processors. For example, in a typical application, the GT62 Graphics Terminal host processor is a PDP-11/34.

The VT48 contains its own program counter (called the DPC for Display Program Counter). The DPC is updated every instruction to access sequentially-stored instructions from PDP-11 memory. To access each instruction, the Unibus Control Logic executes an NPR (nonprocessor request) cycle that fetches the instruction from PDP-11 memory. Each accessed instruction is interpreted by the display instruction control. The instruction determines the course of action taken. There are three main categories of instructions: load status, graphic entity, and branch instructions. The basic processing activities undertaken for each of these instruction types is summarized as follows:

1. Load Status Instructions – These instructions are used to set up internal conditions within the VT48 before executing a string of graphic entity instructions. That is, they are used to enable certain interrupts, set up vector or character scale factors, assign name values, etc. before displaying a set of graphic symbols.

2. Graphic Entity Instructions – There are ten graphic entity instructions. These instructions generate and refresh images at the VR48 CRT. Instructions that display points, vectors, and characters are executed in sequence to display complex images and provide text messages.

3. Branch Instructions – These instructions are used primarily to branch to different areas (for example, subroutines/macros) within the display file. In general, their execution affects the content of the stack memory and display program counter within the VT48.

The VT48 is also equipped with 16 addressable registers that both implement instruction processing and provide status information to the PDP-11 processor during interrupts.

The display instruction control interprets each command and determines routing of the information conveyed by each command. For load status instructions, parameter data is steered to the Display Status registers. This is done for both control and status sampling purposes. Subsequent routing of status information to the silo memory occurs when image generating (graphic entity) instructions are being processed.

Entry into silo memory serves two purposes:

1. Makes control information (intensity level, blink, menu select, etc.) available to the vector/character generator at the time the image is being drawn.

2. Makes all status information available to the read status multiplexer when interrupts occur.

When the display instruction control detects graphic entity instructions, it routes the accompanying delta X/Y values to the graphics calculation logic. This circuitry can carry out its processing activities independent of the display instruction control. Therefore, when the graphics calculation logic accepts the delta X/Y data, the display instruction control is free to fetch the next instruction from the PDP-11.

The graphics calculation logic carries out numerous arithmetic operations related to vector generator beam movement. For example, on every vector processed, this logic must determine whether all, some segment, or none of the vector falls within the display surface and therefore is to be drawn or ignored.

The graphics calculation logic must also determine the amount to step the beam between each character. Still further calculations are carried out to detect point coordinates on light pen hits and edge transitions.

Once all calculations attendant to a graphic entity instruction have been executed, the graphics calculation logic commands the vector/character generator to draw the vector/character.

As their names imply, the vector and character generators physically draw images on the VR48 Monitor CRT. Once either generator begins the draw process, it frees the graphics calculation logic to accept a new graphic entity instruction. Hence, the display instruction control, graphics calculation logic, and vector/character generator can all operate independently (asynchronously) with respect to one another.

The stack memory within the VT48 is used when executing branch instructions. Such instructions allow one to jump to different areas in the display file to take advantage of the efficiencies inherent in subroutines and macros. The stack memory stores status information and provides a convenient means of nesting subroutines under certain display processing conditions.

1.4 RELATED DOCUMENTS

Documents required in addition to this manual or containing information related to the descriptions herein are:

1. GT62 Field Maintenance Print Set
2. *VS60 Display Processor Subsystem User's Guide*
3. *VR48 Table and Scope Assembly Technical Manual,* EK-VR48-TM
4. *GT62 Graphics Terminal System Manual,* EK-GT62-TM
5. *LK11 Pushbutton Option Technical Manual,* EK-LK11-TM.

1.5 SPECIFICATIONS

Specification data for the VT48 Display Processor Unit is summarized below.

Instruction Word Length	16 bits
Number of Instructions	25, including 10 graphic entity modes
Display Parameters	
Refresh Rate	30 frames per second, 40 frames per second, or external sync
Intensity Level	8 Programmable
Line Types	4, solid, long dash, short dash or dot dash
Hardware Blink	Approximately 2 cps
Vector Scaling	15 scale factors from 1/4 to 3-3/4
Character Scaling	4 scale factors from 1/2 to 2
Virtual Display Area	12 bits × 12 bits
Subroutine Nested Levels	8
Environmental	
Temperature	16° to 35°C (60° to 95°F)
Relative Humidity	20% to 80% (noncondensing)
VT48 Mounting	15-3/4 in. cabinet space
Power Source Requirement	115 Vac, 60 Hz at 12 A nom.
Cable Length (VT48-to-VR48)	30 ft (9.1) m)

Vector Generator	
Type	Analog Stroke
Writing Time	
Full screen vector	26 μs
8-inch vector	17 μs
2-inch vector	6 μs
1/2-inch vector	3.5 μs
Character Generator	
Type	Directed Analog Stroke
Character Set	Full ASCII plus 31 special symbols
Fonts	Standard and italics. Both can be rotated 90° CCW
Writing Time	9 μs average

CHAPTER 2
VT48 DISPLAY PROCESSOR UNIT INSTALLATION

Complete system installation procedures are provided in the appropriate system documentation, (such as the *GT62 Graphics Terminal System Manual).* The following procedures apply specifically to the VT48 Display Processor Unit when it is to be installed as an add-on to an existing PDP-11 system installation.

2.1 UNPACKING

No special unpacking procedures are required other than to check the shipping list to ensure that all components have been shipped in the carton(s).

2.2 H7070 POWER SUPPLY MOUNTING PROCEDURE

Use supplied hardware to mount the H7070 Power Supply in the 5-1/4 inch space at the base of the cabinet assembly.

NOTE
In some configurations, it may be desirable to mount the power supply above the VT48 Display Processor Unit. This is permissible.

After installing the H7070 Power Supply, connect the power cable to the outlet on the 861-C Power Control Unit.

2.3 VT48 DISPLAY PROCESSOR UNIT MOUNTING

To install the VT48, proceed as follows:

1. Use supplied hardware to install slide rails in the cabinet assembly so that the VT48 Display Processor Unit mounts in the 10-1/2 inch space above the H7070 Power Supply.

2. After mounting the slide rails, mount the VT48 into the slide rails; then check that the VT48 slides freely in and out of cabinet.

3. Connect the power harness from the VT48 to the H7070 Power Supply.

4. Connect the Unibus "in" cable to slot AB01.

5. Connect the Unibus "out" cable to slot AB13.

2.4 VT48 DISPLAY PROCESSOR UNIT CHECKOUT

With power applied, the VT48 is now ready for checkout. Follow the procedures called for in the subsequent paragraphs.

2.4.1 Instruction Processing Checkout

The VT48 instruction processing checkout is executed through the use of diagnostic programs. Conduct all instruction diagnostic tests specified in the VT48 Diagnostics, Preventive Maintenance and Alignment Procedures, Chapter 5, Paragraphs 5.2.1.1 through 5.2.1.3.

2.4.2 VT48 Analog (Vector Generator and Chapter Generator) Checkout

After successful completion of the instruction diagnostic tests, the VT48 analog modules are adjusted. Proceed as follows:

1. Connect VT48 output cables to the connector on the VR48 Monitor.

2. Apply power to the VR48 Monitor.

3. Load the visual test diagnostic program in accordance with the procedures specified in the VT48 Diagnostics, Preventive Maintenance and Alignment Procedures, Chapter 5, Paragraph 5.2.

4. Conduct all vector generator and character generator adjustments specified in the VT48 Diagnostics, Preventive Maintenance and Alignment Procedures, Chapter 5, Paragraphs 5.4.3 and 5.4.5. This completes the VT48 installation procedure.

CHAPTER 3
VT48 OPERATION AND PROGRAMMING

3.1 INTRODUCTION

This chapter describes the VT48 power up procedure and the formats of the various instructions executable by the VT48 Display Processing Unit. Examples of programming sequences are also given to indicate typical instruction usage within a display file.

3.2 POWER APPLICATION

Power is applied to the VT48 Display Processor Unit when the PDP-11 computer is activated. System initialization and clearing all registers and flip-flops in the VT48 is effected at this time.

3.3 VT48 INSTRUCTION SET

The VT48 is capable of executing 25 different instructions. As a group, the instructions can be divided into two basic categories:

1. Graphic Entity Instructions – There are ten instructions of this type, all generating visible graphic symbols on the face of the VR48 Monitor.

2. Control Instructions – Instructions falling in this category are used to set up status/parameter conditions within the VT48 and also to branch to different locations within the display file. Control instructions can be divided further into three categories:

   a. Load Status Instructions – These instructions are used to set up internal conditions within the VT48 before executing a string of graphic entity instructions. That is, they are used to enable certain interrupts, set vector or character scale, assign name values, etc. before displaying a set of graphic symbols.

   b. Set Graphic Mode Instruction – This instruction always precedes graphic entity instructions and defines the type of graphic entity instructions that follow. It also sets up certain conditions within the VT48, i.e., beam intensity level, line type, beam blank/unblank, and light pen interrupt enable.

   c. Branch Instructions – These instructions are used primarily to branch to different areas (e.g., subroutines/macros, etc.) within the display file. In general, their execution affects the content of the display program counter within the VT48.

Though all VT48 instructions can be categorized as either graphic or control, they can also be looked at from another viewpoint. That is the way or sequence in which they most normally fall within the display file processed by the VT48.

The display file constitutes the sum total of all instructions, control and graphic, executed by the VT48 in a single refresh frame. The display file is contained in PDP-11 memory. To initiate sequencing through the display file, the PDP-11 addresses the VT48 and passes the display file starting address to

the VT48 display program counter. After this (disregarding interrupts), sequencing is under control of the VT48 on a nonprocessor request (NPR) basis.

Depending on application, the display file may contain hundreds of graphic entity strings for display on the VR48 Monitor. However, it is likely that load status instructions are inserted preceding each string. This is because the programmer may wish that light pen interrupts be activated on certain graphic symbols but not on others. He/she may wish a different intensity level for upcoming vectors or decide to select the menu area for displaying a string of characters. Further, he/she may wish that they are displayed in an italicized rather than normal format. For all such reasons, the usual sequence followed in structuring the display file is as follows:

1. Load Status Instructions – Executed first to set up desired parameters.

2. Set Graphic Mode Instructions – Defines graphic instructions to follow and also sets up certain beam status parameters.

3. Graphic Entity Instructions – Processed through the VT48 to draw the specified graphic symbols.

4. Branch Instructions – Used at times where it is practical to jump to subroutine or macro locations within the display file.

The above order defines the sequence in which the VT48 instruction formats are described.

3.4 LOAD STATUS INSTRUCTIONS

The six instructions used to set up status/parameter conditions within the VT48 are described in this paragraph. The instructions and illustrations showing instruction formats are as follows:

1. Load Status A Instruction (Figure 3-1) – This instruction is used to select: stop interrupt, light pen hit bright down of a picked graphic entity, normal or italics font, refresh rate, and display menu area.

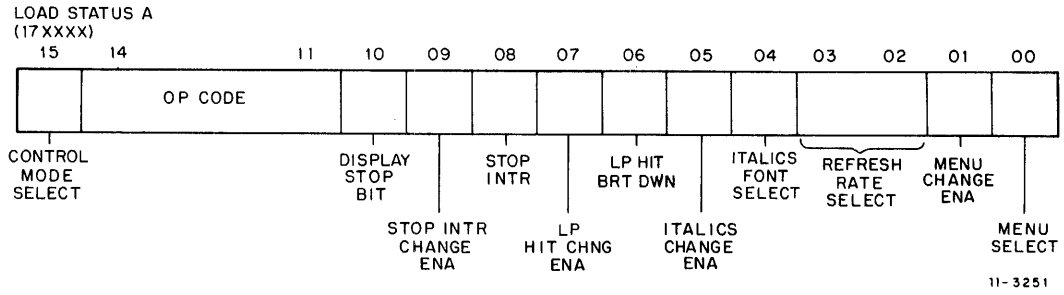
2. Load Status B Instruction (Figure 3-2) – This instruction is used to select color (color option) and also the graph plot increment value for use with the graph plot X/graph plot Y graphic entity instructions.

3. Load Status BB Instruction (Figure 3-3) – This instruction is used to enable processing of Z-axis data (depth queue option), interrupts on a vector traversing the display frame edge (edge transition), and popping stack memory on character string escape.

4. Load Status C (Figure 3-4) – This instruction is used to select vector scale, character scale, and the character rotate capability.

5. Load Scope Selection (Figure 3-5) – This instruction is used to select either the master or slave (optional) consoles. Also, it serves to blank/unblank the selected console and to enable/inhibit interrupts on the various light pen operations.

6. Name Instruction (Figure 3-6) – The name instruction is used to assign an 11-, 8-, or 4-bit name tag to the string of graphic entities about to be processed. A matchup between the name tag value and that loaded into the associative name register (described later) causes an interrupt.



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
1110_2

DISPLAY STOP BIT
1 = Stop display

STOP INTR CHANGE ENA
0 = Disable bit 8, 1 = Enable bit 8

STOP INTR
0 = Inhibit interrupt on stop
1 = Generate interrupt on stop

LP HIT CHNG ENA
0 = Disable bit 6, 1 = Enable bit 6

LP HIT BRT DWN
0 = Enable bright down of graphic entity
1 = Inhibit bright down of graphic entity

ITALICS CHANGE ENA
0 = Disable bit 4, 1 = Enable bit 4

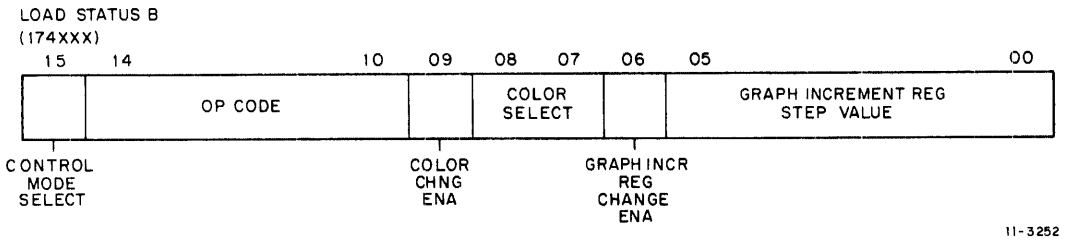
ITALICS FONT SELECT
0 = Normal font, 1 = Italics font

REFRESH RATE SELECT
00 = No operation
01 = 30 Frames per sec
10 = 40 Frames per sec
11 = External option sync

MENU CHANGE ENA
0 = Disable bit 0, 1 = Enable bit 0

MENU
0 = Graphic data refreshed in major screen area
1 = Graphic data refreshed in menu area

Figure 3-1 Load Status A



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
11110_2

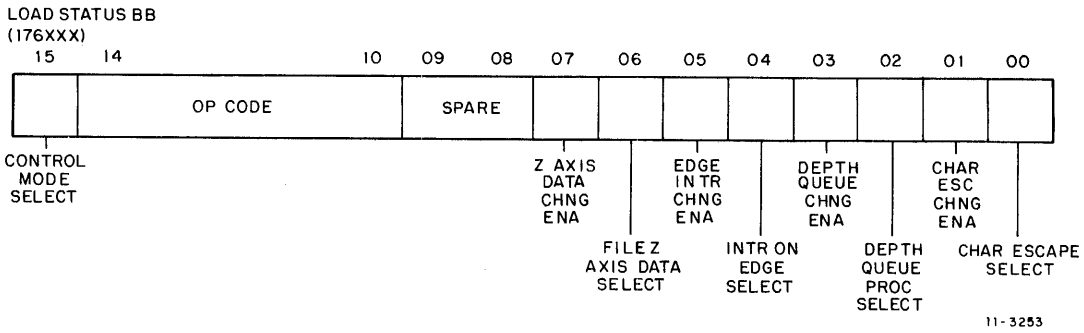
COLOR CHNG ENA
0 = Disable bits 7 and 8, 1 = Enable bits 7 and 8

COLOR SELECT
00 = Green, 01 = Yellow, 10 = Orange, 11 = Red

GRAPH INCR REG CHANGE ENA
0 = Disable change of increment register
1 = Write contents of bits 00–05 into increment register

GRAPH INCR REG STEP VALUE
0 to 63 unit step value inserted prior to executing Graphplot X/Graphplot Y instructions

Figure 3-2 Load Status B



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
11111_2

SPARE
Not used

Z AXIS DATA CHNG ENA
0 = Disable bit 6, 1 = Enable bit 6

FILE Z AXIS DATA SELECT
0 = File Z axis data displayed, 1 = File Z axis data not displayed

EDGE INTR CHNG ENA
0 = Disable bit 4, 1 = Enable bit 4

INTR ON EDGE SELECT
0 = Inhibit interrupt on edge transition, 1 = Generate interrupt on edge transition

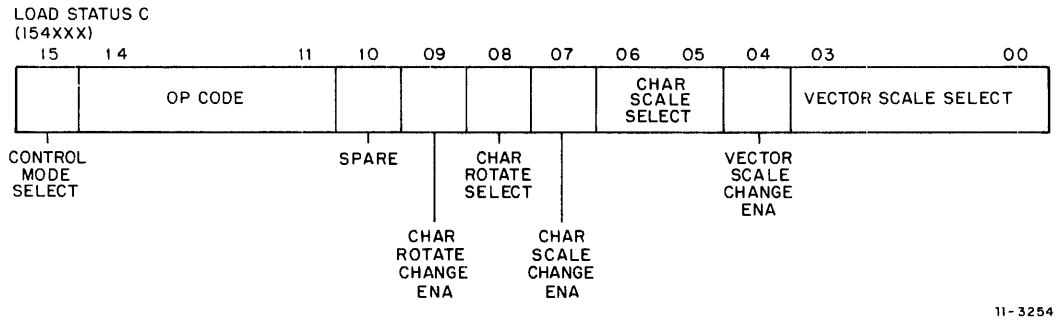
DEPTH QUEUE CHNG ENA
0 = Disable bit 2, 1 = Enable bit 2

DEPTH QUEUE PROC SELECT
0 = Inhibit Z – axis processing, 1 = Enable Z – axis processing

CHAR ESC CHNG ENA
0 = Disable bit 0, 1 = Enable bit 0

CHAR ESCAPE SELECT
0 = Inhibit pop on character string terminate match
1 = Enable pop on character string terminate match

Figure 3-3 Load Status BB



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
1011_2

SPARE
Not used

CHAR ROTATE CHANGE ENA
0 = Disable bit 8, 1 = Enable bit 8

CHAR ROTATE SELECT
0 = Character displayed in normal position
1 = Character displayed 90 degrees counterclockwise

CHAR SCALE CHANGE ENA
0 = Disable bits 5 and 6
1 = Write bits 5 and 6 into Character Scale register

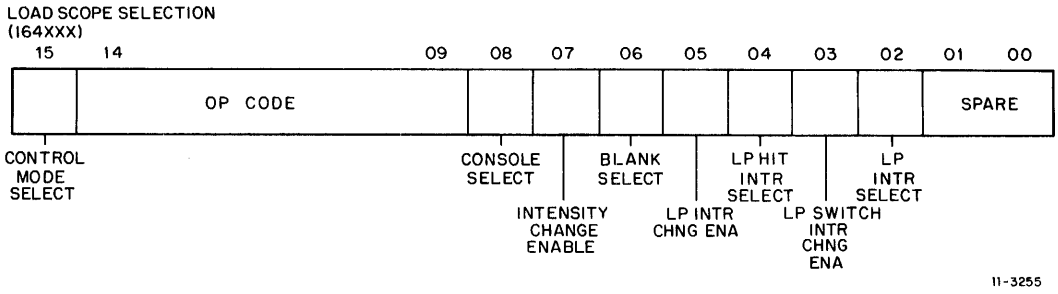
VECTOR SCALE SELECT

0000 = Invisible (vector disappears)
0001 = 1/4 X Normal size
0010 = 1/2 X Normal size
0011 = 3/4 X Normal size
0100 = 1 X Normal size
0101 = 1-1/4 X Normal size
0110 = 1-1/2 X Normal size
0111 = 1-3/4 X Normal size
1000 = 2 X Normal size
1001 = 2-1/4 X Normal size
1010 = 2-1/2 X Normal size
1011 = 2-3/4 X Normal size
1100 = 3 X Normal size
1101 = 3-1/4 X Normal size
1110 = 3-1/2 X Normal size
1111 = 3-3/4 X Normal size

CHAR SCALE SELECT
00 = 1/2 normal size, 01 = normal size, 10 = 1 and 1/2 normal size, 11 = twice normal size

VECTOR SCALE CHANGE ENA
0 = Disable bits 0 through 3
1 = Write contents of bit 0 through 3 into Vector Scale register

Figure 3-4 Load Status C



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110100_2

CONSOLE SELECT
0 = Console 0 selected, 1 = Console 1 selected

INTENSITY CHANGE ENABLE
0 = Disable bit 6, 1 = Enable bit 6

BLANK SELECT
0 = Blank display for all graphic data
1 = Unblank display for all graphic data

LP INTR CHNG ENA
0 = Disable bit 4, 1 = Enable bit 4

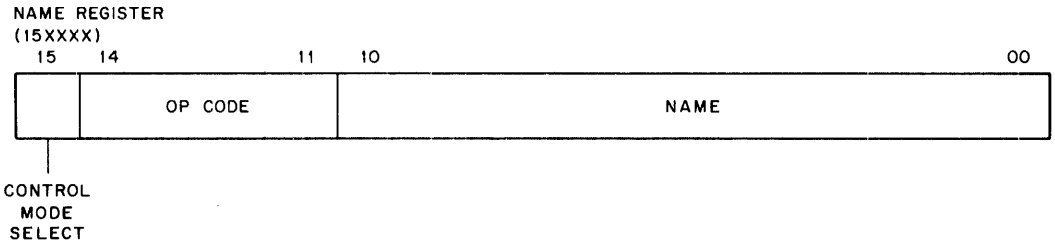
LP HIT INTR SELECT
0 = Inhibit interrupt on light pen hit
1 = Generate interrupt on light pen hit

LP SWITCH INTR CHNG ENA
0 = Disable bit 2, 1 = Enable bit 2

LP SWITCH INTR SELECT
0 = Inhibit interrupt on light pen switch change
1 = Generate interrupt on light pen switch change

SPARE
Not used

Figure 3-5 Load Scope Selection



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
1010_2

NAME TAG
11 Bit label inserted into Name register prior to processing a graphics string

Figure 3-6 Load Name Register

3.5 SET GRAPHIC MODE INSTRUCTION

The format for this instruction is shown in Figure 3-7. Note that there are ten different graphic mode code values. Each different value defines the type of graphic entity instruction (or string thereof) to follow the set graphic mode instruction. The low order bits are used to set up beam and light pen parameters as indicated.

3.6 GRAPHIC ENTITY INSTRUCTIONS

There are ten basic graphic entity instruction codes as shown in Figure 3-7. However, some codes (e.g., point/offset) define two different instructions. Distinction between the two is made by the set/reset status of a bit conveyed in the graphic entity instruction itself. For example, during a point/offset instruction, a 0 in bit 12 selects the point mode. A 1 in bit 12 selects the offset mode.

The following is a breakdown of the graphic entity instructions:

1. Character Instruction (Figure 3-8) – The character instruction conveys two 7-bit ASCII characters for display on the VR48 Monitor.

2. Short Vector Instruction (Figure 3-9) – This is a one-word instruction[1] that conveys 6-bit delta X and 6-bit delta Y values for generating vectors up to 64 units in length.

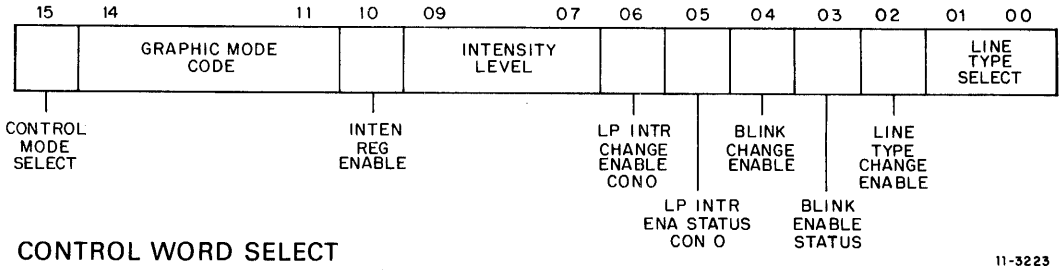
3. Long Vector Instruction (Figure 3-10) – This is a two-word instruction[2]. The first word conveys 10-bit delta X data and the second word conveys 10-bit delta Y data. When both delta values convey all 1s, a full length vector (lower left to upper right corner) is displayed on the VR48 Monitor major area.

4. Point/Offset Instruction (Figure 3-11) – Each of these is a two-word instruction[3]. The first word conveys X point/offset data and the second word conveys Y point/offset data. Bit 12 is used to distinguish between the two instructions. The point instruction is used to define point positions within the virtual display area. The offset instruction is used to "window" to different areas in the virtual display file for refresh at the VR48 console.

[1]It can be two words if depth queue option is included.

[2]It can be 3 words if depth queue option is included.

[3]Each can be a three word instruction if the depth queue option is included.



CONTROL WORD SELECT
1 Indicates control word

GRAPHIC MODE CODE
0000 = Character mode
0001 = Short vector mode
0010 = Long Vector mode
0011 = Point/offset mode
0100 = Graphplot X/basic long vector mode
0101 = Graphplot Y/basic long vector mode
0110 = Relative point mode
0111 = Basic short vector
1000 = Circle/arc mode
1001 = Absolute vector mode

INTEN REG ENABLE
1 = Enable bits 7 through 9 into Intensity register

INTENSITY LEVEL
000 = Dimmest, 111 = Brightest

LP INTR CHANGE ENABLE CON. 0
1 Enables bit 5 into Light Pen Interrupt Enable register for console 0

LP INTR ENA STATUS, CON 0
0 = Light pen hit interrupt disabled
1 = Light pen hit interrupt enabled

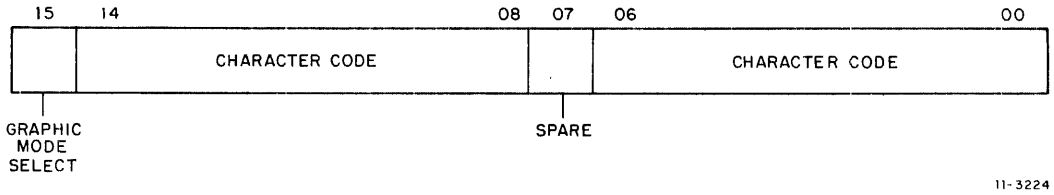
BLINK CHANGE ENABLE
1 = Enable bit 3 into Blink register

BLINK ENABLE STATUS
0 = Blink off, 1 = Blink on

LINE TYPE CHANGE ENABLE
1 = Enable bits 0 and 1 into Line Type register

LINE TYPE SELECT
00 = Solid line
01 = Long dash line
10 = Short dash line
11 = Dot dash line

Figure 3-7 Set Graphic Mode

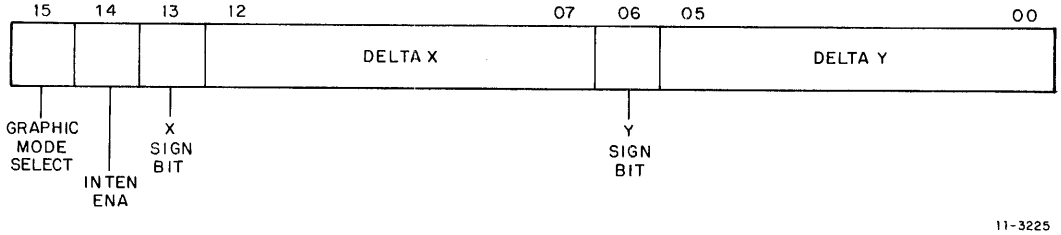


GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

CHARACTER CODE
7-bit ASCII character

SPARE
Not used

Figure 3-8 Character Mode



GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

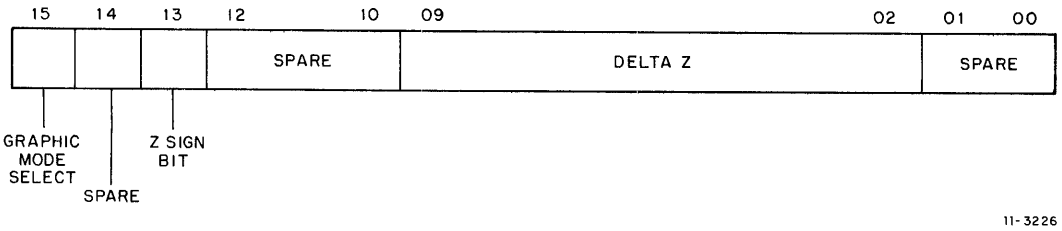
X SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA X
6-bit delta X vector component

Y SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA Y
6-bit delta Y vector component

NOTE
Short vector is a 2-word instruction when depth-queue option is used.



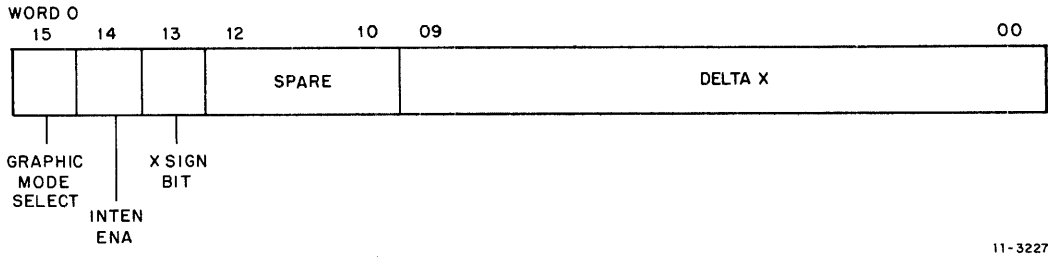
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

Z SIGN BIT
0 = Positive direction (brighter)
1 = Negative direction (dimmer)

DELTA Z
8 bits of depth data entered into high order bits of Z register

Figure 3-9 Short Vector



GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

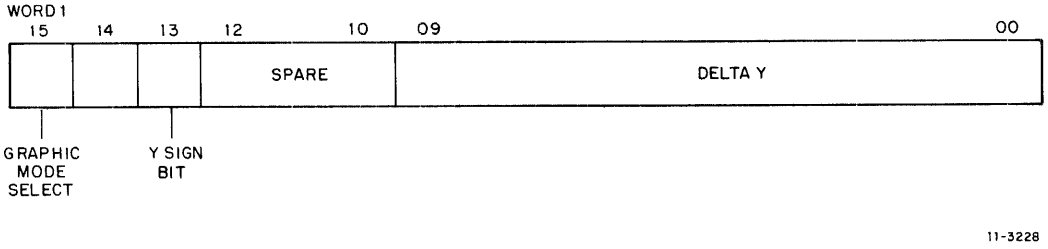
INTEN ENA
0 = Beam off, 1 = Beam on

X SIGN BIT
0 = Positive direction, 1 = Negative direction

SPARE
Not used

DELTA X
10-bit delta X vector component

NOTE
Long vector is a 2-word instruction, or a 3-word instruction when depth-queue option is used.



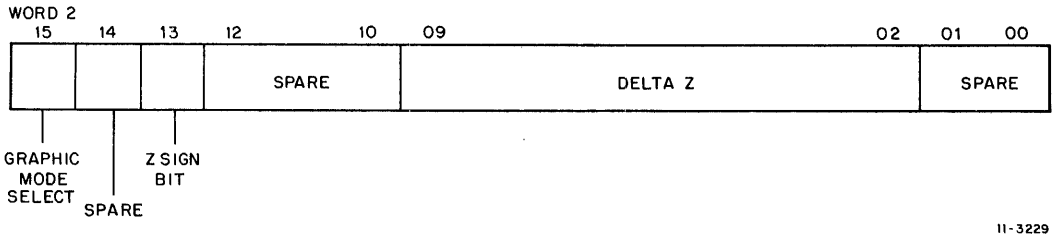
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

Y SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA Y
10-bit delta Y vector component

Figure 3-10 Long Vector (Sheet 1 of 2)



GRAPHIC MODE SELECT

0 Indicates graphic mode instruction

SPARE

Not used

Z SIGN BIT

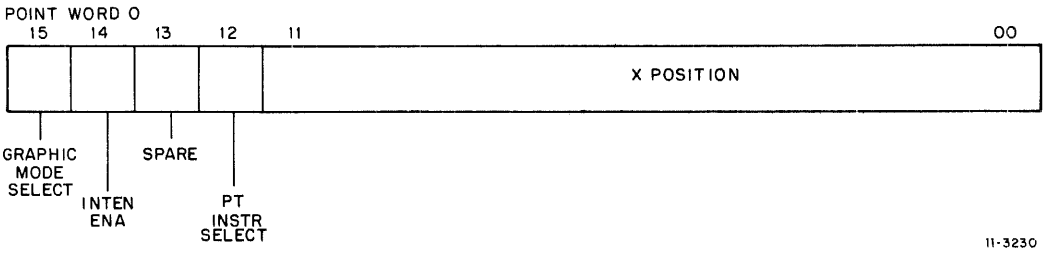
0 = Positive direction (brighter)

1 = Negative direction (dimmer)

DELTA Z

8 bits of data entered into high order bits of Z register

Figure 3-10 Long Vector (Sheet 2 of 2)



GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

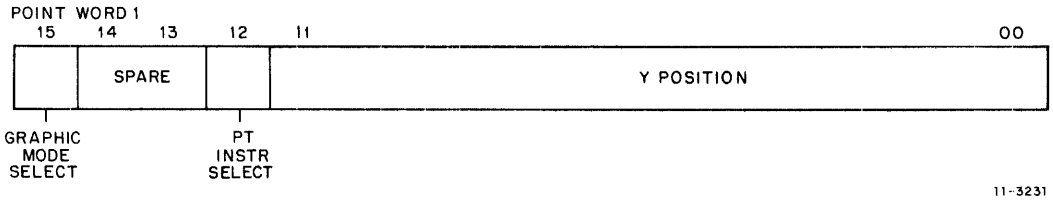
INTEN ENA
0 = Beam off, 1 = Beam on

PT INSTR SELECT
0 = Selects point rather than offset instruction

SPARE
Not used

X POSITION
10-bit absolute X position

NOTE
Point/offset is a 2-word instruction, or a 3-word instruction when depth-queue option is used.



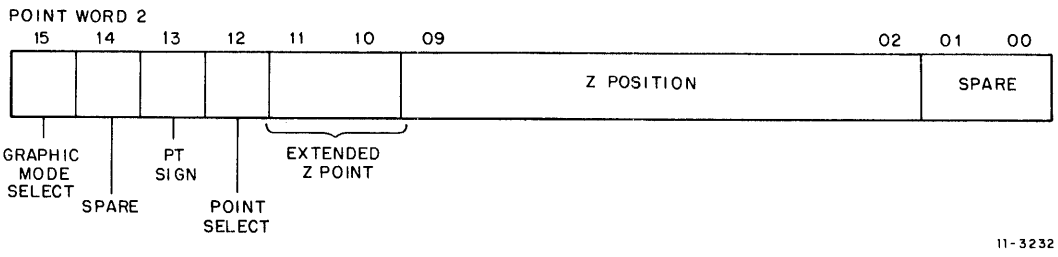
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

PT INSTR SELECT
0 Selects point rather than offset instruction

Y POSITION
10-bit absolute Y position

Figure 3-11 Absolute Point/Offset (Sheet 1 of 4)



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

SPARE
Not used

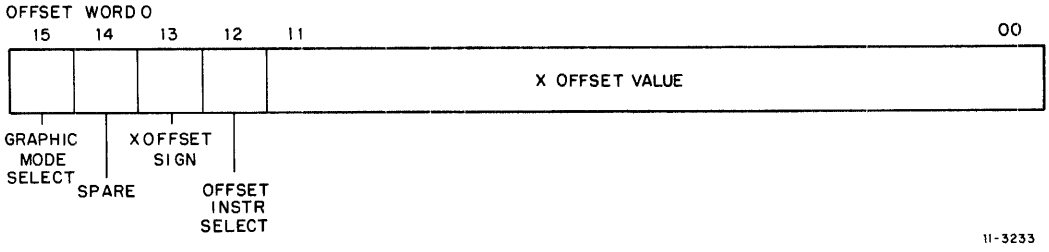
PT SIGN
0 = Positive, 1 = Negative

POINT SELECT
0 = Point instruction

EXTENDED Z POINT

Z POSITION
8-bit absolute Z position data

Figure 3-11 Absolute Point/Offset (Sheet 2 of 4)



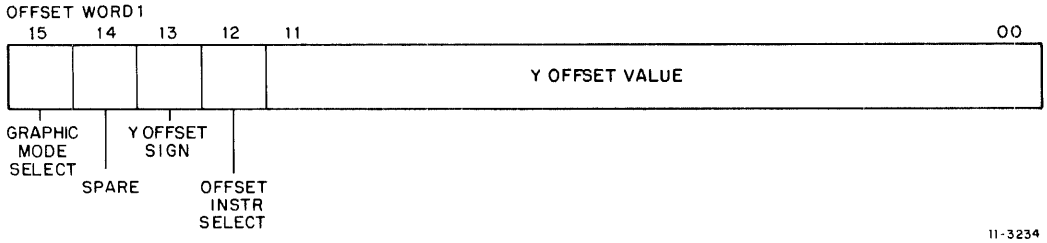
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

X OFFSET SIGN
0 = Add to X absolute points
1 = Subtract from X absolute points

OFFSET INSTR SELECT
1 Selects offset rather than point instruction

OFFSET VALUE
12-bit X offset position data



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

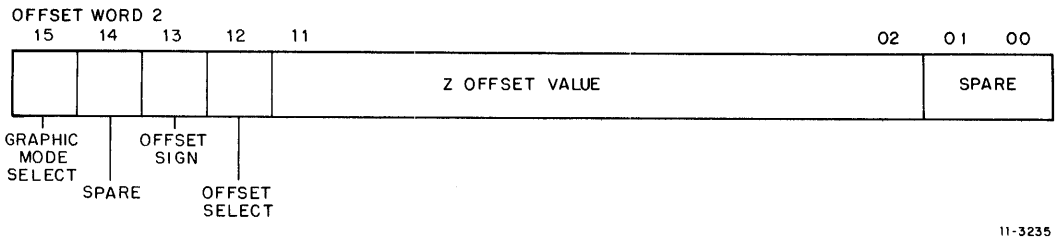
SPARE
Not used

Y OFFSET SIGN
0 = Add to Y absolute points
1 = Subtract from Y absolute points

OFFSET INSTR SELECT
1 Selects offset rather than point instruction

OFFSET VALUE
12-bit Y offset position data

Figure 3-11 Absolute Point/Offset (Sheet 3 of 4)



GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

OFFSET SIGN
0 = Add to absolute Z points
1 = Subtract from absolute Z points

OFFSET SELECT
1 = Offset instruction

Z OFFSET VALUE
10-bit Z offset position data

Figure 3-11 Absolute Point/Offset (Sheet 4 of 4)

5. Graph plot X/Basic Long Vector Instruction (Figure 3-12) – Each of these is a one-word instructions. Bit 10 is used to distinguish between the two instructions. The graph plot X instruction conveys the X delta length when it is desired to draw a graph on the VR48 Monitor. The Y component (graph increment) of the graph must be loaded through use of the load status C instruction before executing the graph plot X instruction.

   The basic long vector instruction can be used to generate basic vectors as defined in Figure 3-12, part 2.

6. Graph plot Y/Basic Long Vector (Figure 3-13) – These are both one-word instructions with bit 10 distinguishing between the two. The basic long vector instruction is identical to that described in Paragraph 5 above. Graph plot Y instructions are used when it is desired to draw a graph with vertical rather than horizontal lines. Again, the X component value must be loaded through use of the load status C instruction prior to executing the graph plot Y instruction.

7. Relative Point Instruction (Figure 3-14) – This instruction can be used to generate a point at a position relative to the last addressed beam position. This is a one-word instruction[1]. When processed, the VT48 simply updates the X/Y position data relative to the last addressed beam position.

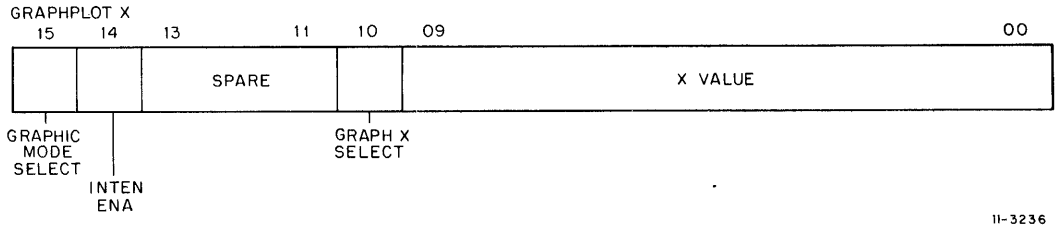
8. Basic Short Vector (Figure 3-15) – This is a one-word instruction that can be used to generate the basic vectors shown on the accompanying illustration. It is similar to basic long vector, except that vector length is limited to 16 units.

9. Circle/Arc Intruction (Figure 3-16) – This is a multiword instruction that is used to generate circles and arcs in those systems where the circle/arc option is incorporated.

10. Absolute Vector (Figure 3-17) – This a two-word instruction which conveys absolute point data[2]. The VT48 reacts to this instruction by calculating vector length (between last addressed beam position and absolute beam position conveyed by the instruction), and then drawing the vector on the VR48 Monitor.

---

[1]It is a two-word instruction when the depth queue option is used.
[2]It is a three-word instructionn when the depth queue option is used.



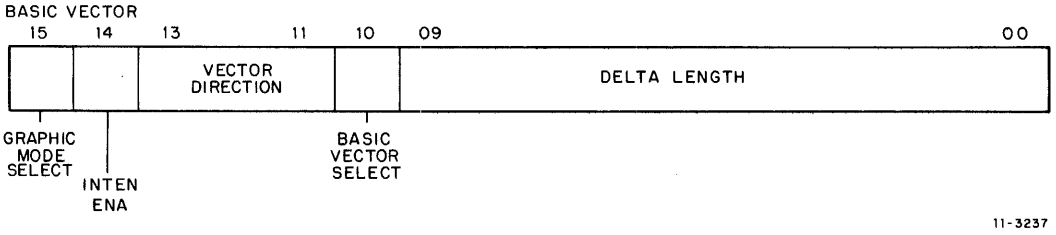
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

SPARE
Not used

GRAPH X SELECT
0 Selects graphplot X rather than basic vector instruction

X VALUE
10-bit absolute X value

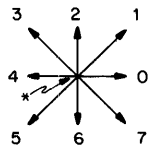


GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

VECTOR DIRECTION:

000 = 0 Path
001 = 1 Path
010 = 2 Path
011 = 3 Path
100 = 4 Path
101 = 5 Path
110 = 6 Path
111 = 7 Path

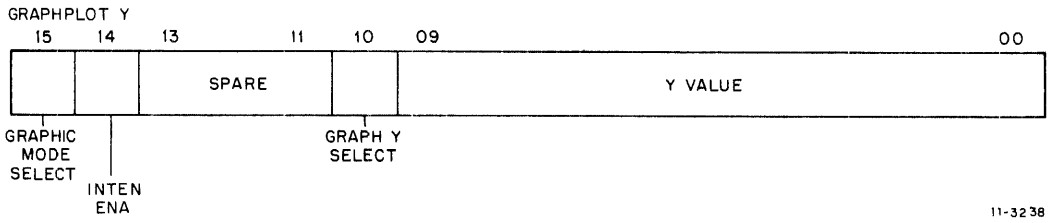


*: Starting point of vector

BASIC VECTOR SELECT
1 Selects basic vector rather than graphplot X instruction

DELTA LENGTH
10-bit delta length of vector

Figure 3-12 Graphplot X/Basic Long Vector



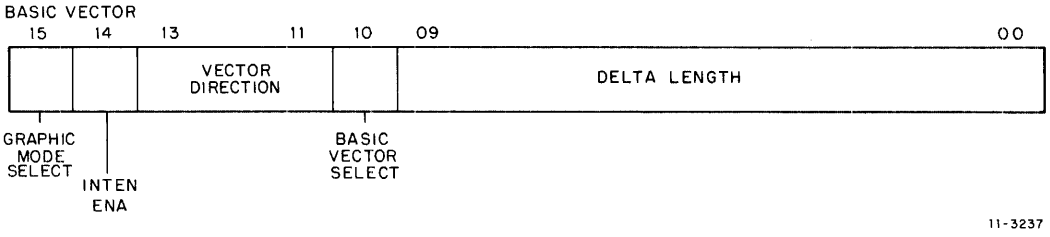
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

SPARE
Not used

GRAPH Y SELECT
0 Selects graphplot Y rather than basic vector instruction

Y VALUE
10-bit absolute Y value

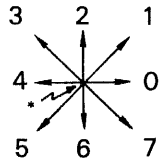


GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

VECTOR DIRECTION

000= 0 Path
001 = 1 Path
010 = 2 Path
011 = 3 Path
100 = 4 Path
101 = 5 Path
110 = 6 Path
111 = 7 Path

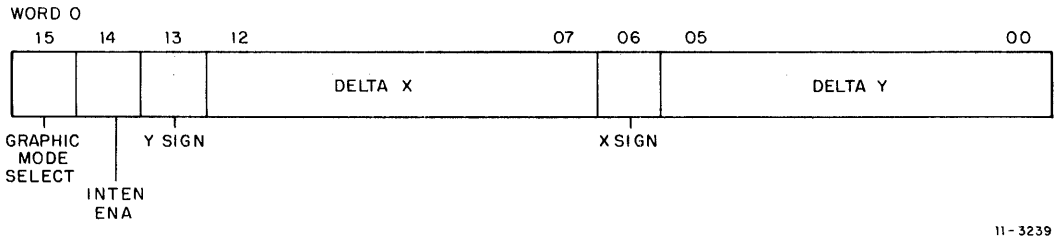


*: Starting point of vector

BASIC VECTOR SELECT
1 Selects basic vector rather than graphplot X instruction

DELTA LENGTH
10-bit delta length of vector

Figure 3-13 Graphplot Y/Basic Long Vector



GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

X SIGN
0 = Add to X absolute points
1 = Subtract from X absolute points

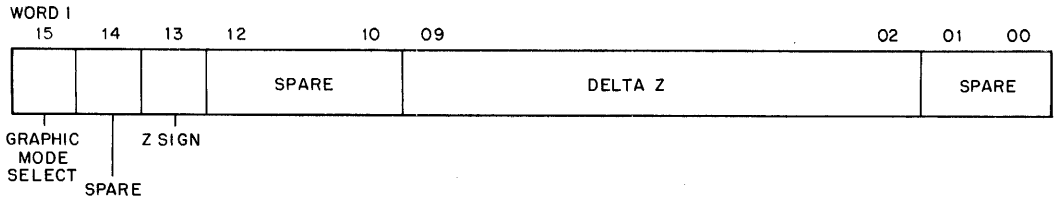
DELTA X
6-bit delta X component

Y SIGN
0 = Add to Y absolute points
1 = Subtract from Y absolute points

DELTA Y
6-bit delta Y component

NOTE

Relative point is a 2-word instruction when depth-queue is used.



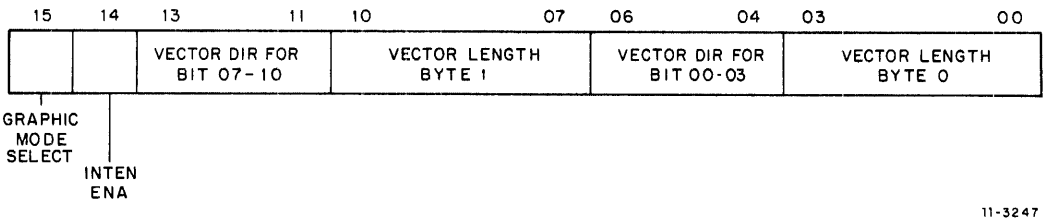
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

Z SIGN BIT
0 = Positive direction (brighter)
1 = Negative direction (dimmer)

DELTA Z
8 bits of depth data entered into high order bits of Z register

Figure 3-14 Relative Point



GRAPHIC MODE SELECT

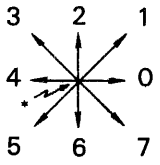
0 Indicates graphic mode instruction

INTEN ENA

0 = Beam off, 1 = Beam on

VECTOR DIRECTION FOR BIT 07–10

000 = 0 Path
001 = 1 Path
010 = 2 Path
011 = 3 Path
100 = 4 Path
101 = 5 Path
110 = 6 Path
111 = 7 Path



*: Starting point of vector

VECTOR LENGTH, BYTE 1

4-bit delta vector length

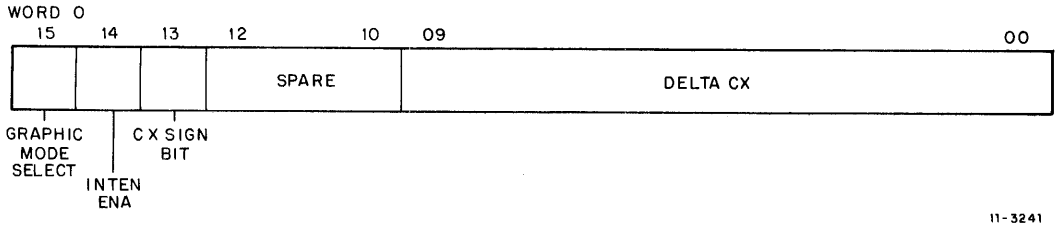
VECTOR DIR FOR BIT 00–03

Same as bits 11 through 13

VECTOR LENGTH BYTE 0

4-bit delta vector length

Figure 3-15 Basic Short Vector

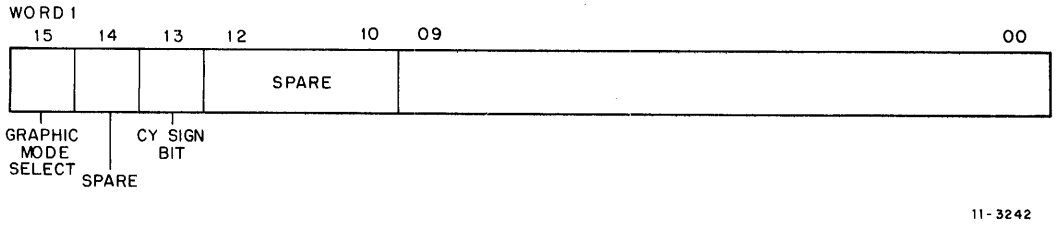


GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

INTEN ENA
0 = Beam off, 1 = Beam on

CX SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA CX
Center of circle in X coordinate

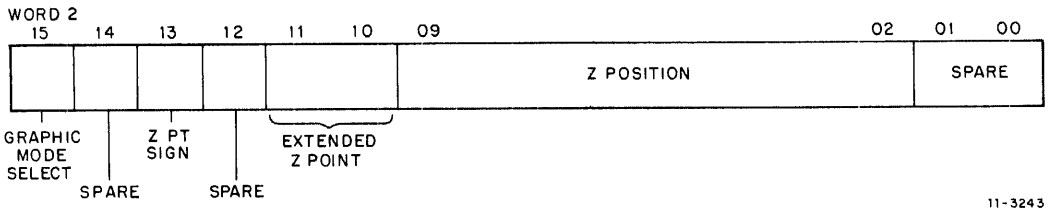


GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

SPARE
Not used

CY SIGN BIT
0 = Positive direction, 1 = Negative direction

Figure 3-16 Circle/Arc, Optional (Sheet 1 of 3)



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

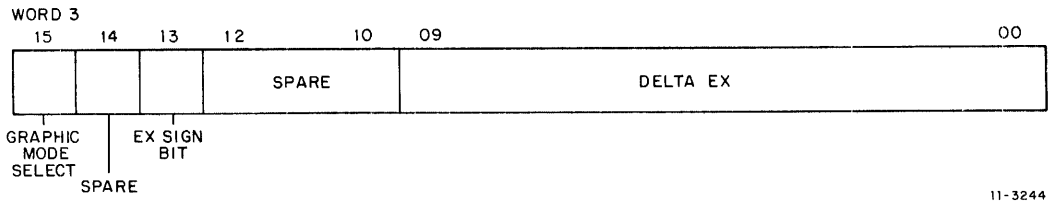
SPARE
Not used

Z PT SIGN
0 = Positive, 1 = Negative

EXTENDED Z POINT

Z POSITION
8-bit absolute Z position data

NOTE
Circle/arc is a 4-word instruction, 6-word when depth-queue option is used.



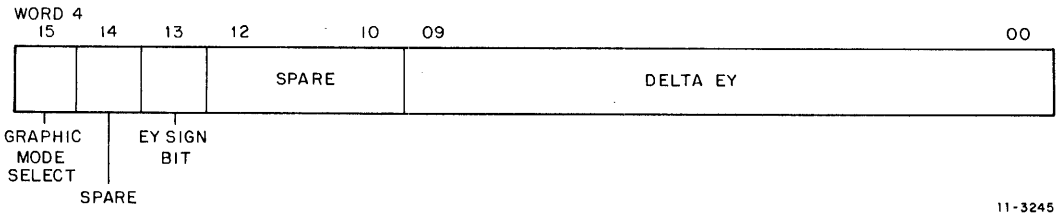
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

EX SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA EX
10-bit end point of arc in X coordinate

Figure 3-16 Circle/Arc, Optional (Sheet 2 of 3)

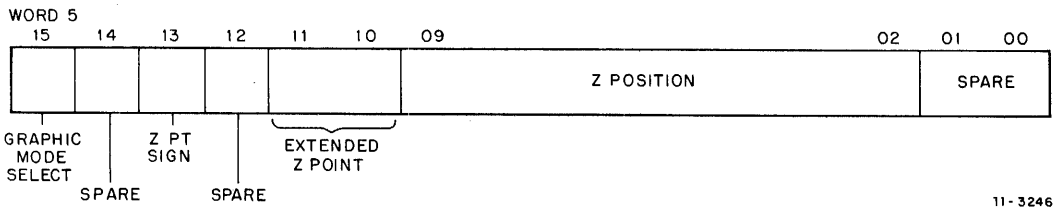


GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

EY SIGN BIT
0 = Positive direction, 1 = Negative direction

DELTA EY
10-bit end point of arc in Y coordinate



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

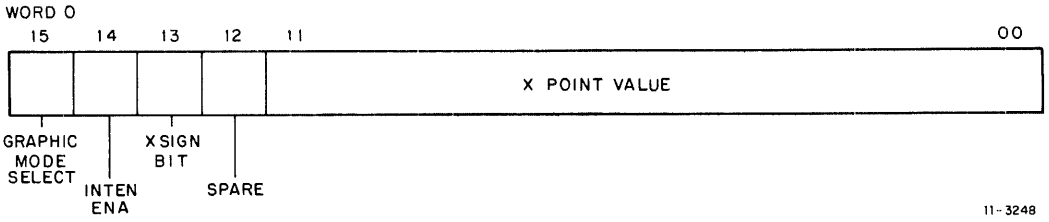
SPARE
Not used

Z PT SIGN
0 = Positive, 1 = Negative

EXTENDED Z POINT

Z POSITION
8-bit absolute Z position data

Figure 3-16 Circle/Arc, Optional (Sheet 3 of 3)



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

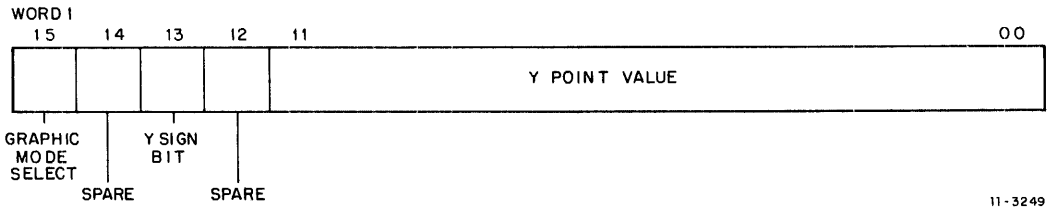
INTEN ENA
0 = Beam off, 1 = Beam on

X SIGN BIT
0 = Positive direction, 1 = Negative direction

SPARE
Not used

X POINT VALUE
12-bit absolute X point data

NOTE
Absolute vector is a 2-word instruction, 3-word when depth-queue option is used.



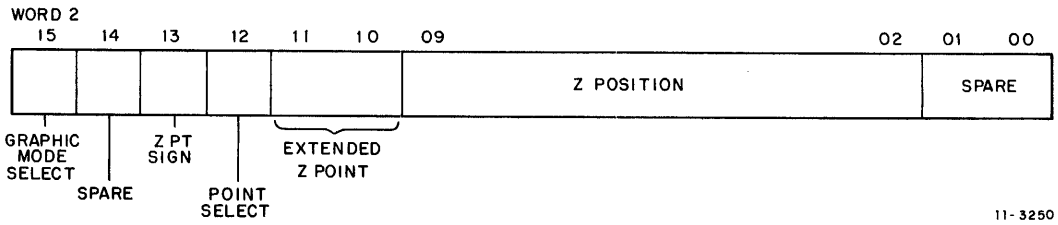
GRAPHIC MODE SELECT
0 Indicates graphic mode instruction

SPARE
Not used

Y SIGN BIT
0 = Positive direction, 1 = Negative direction

Y POINT VALUE
12-bit absolute Y point data

Figure 3-17 Absolute Vector (Sheet 1 of 2)



GRAPHIC MODE SELECT
1 Indicates graphic mode instruction

SPARE
Not used

Z PT SIGN
0 = Positive, 1 = Negative

POINT SELECT
0 = Point instruction

EXTENDED Z POINT

Z POSITION
8-bit absolute Z position data

Figure 3-17 Absolute Vector (Sheet 2 of 2)

3.7 BRANCH INSTRUCTIONS

The VT48 is capable of executing eight different branch instructions. Individual instruction uses and formats are described below:

1. Display JUMP Absolute Instruction (Figure 3-18) – This is a two-word instruction used to branch to a new area in the display file. The branch address is 15 bits in length and can, therefore, accommodate a jump anywhere within 32K of memory.

2. Display JUMP Relative Instruction (Figure 3-19) – This is a one-word instruction that can be used to effect limited branching within the display file. The relative address conveyed by the instruction is 8 bits in length. Therefore, branching is limited to 256 locations forward and 255 locations backward. The JUMP direction is defined by the sign bit, i.e., + forward, – backward.

3. Display JUMP to Subroutine (JSR) Absolute (Figure 3-20) – This is a two-word instruction that can also be used to branch to a different area in the display file; that is, to the location of a subroutine. The difference between this instruction and Display JUMP Absolute is that the Display JSR Absolute takes advantage of the VT48 stack memory. On detecting this instruction, the VT48 pushes all status/parameter data (attendant to the present display file address) onto one of eight stack memory levels, prior to entering the branch address into the display program counter. Hence, when a POP restore instruction is executed at the end of the called subroutine, processing is re-initiated at the same area of the display file with all status/parameters (intensity, blank scale factor, etc.) re-established. The absolute address is 15-bits in length allowing a jump within 32K of memory and is conveyed in the second word of the instruction.

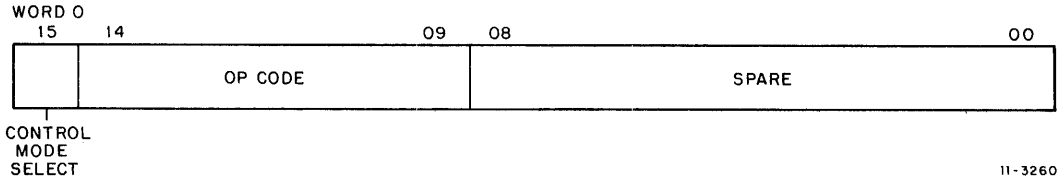
4. Display JUMP to Subroutine (JSR) Relative (Figure 3-21) – This is a one-word instruction that also takes advantage of the VT48 stack memory. The relative address is 8 bits in length and consequently, branching is limited (256 locations forward and 255 locations backward). A POP restore instruction executed at the end of the subroutine re-initiates processing in the old display file area and restores all status/parameters temporarily stored in stack memory.

5. Display NOP (Figure 3-22) – This is a one-word instruction. When decoded by the display processor unit, the VT48 simply increments the display program counter by 2 and then initiates another nonprocessor request (NPR) cycle.

6. Display POP Not Restore (Figure 3-23) – Each execution of this instruction causes POPping of one read location in VT48 stack memory. It also restores the display program counter (DPC). It does not, however, restore the values previously entered into the stack via the JSR instruction to the status/parameter registers.

7. Display POP Restore (Figure 3-24) – This is a one-word instruction which both POPs the stack memory by one location and restores all status/parameters and DPC addresses (entered onto stack level during prior JSR instruction) to the appropriate registers. Taken together, the JSR and POP restore instructions provide a convenient means of jumping around in PDP-11 memory, and also eliminate the need of reloading status/parameter data on return to pre-branch display file addresses.

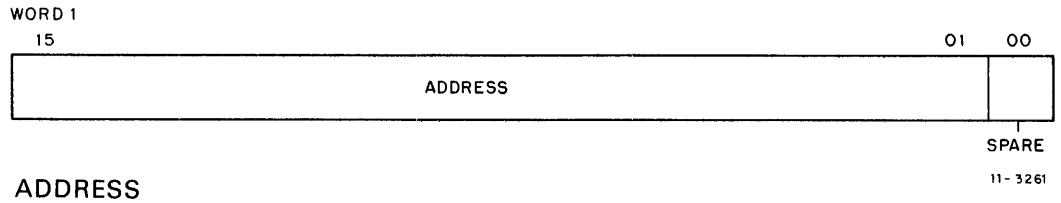
8. Display Stop Instruction (Figure 3-25) – This is a one-word instruction that can be used to stop the VT48 Display Processor Unit.



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110000_2

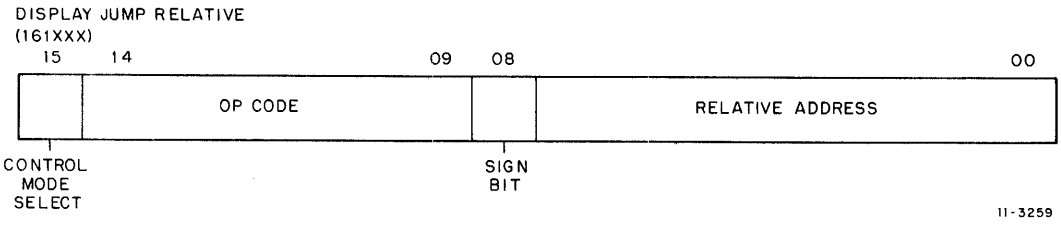
SPARE
Not used



ADDRESS
15-bit absolute address of subroutine

SPARE
Not used

Figure 3-18 Display Jump Absolute



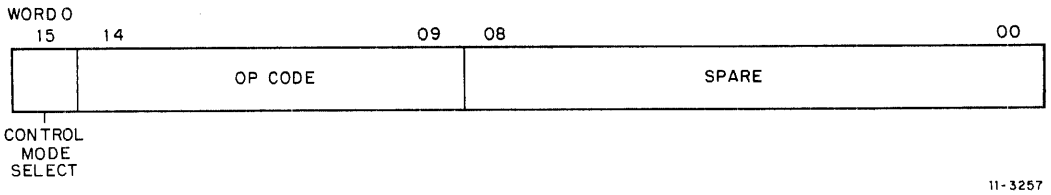
CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110001_2

SIGN BIT
0 = Add relative address to DPC
1 = Subtract relative address from DPC

RELATIVE ADDRESS
8-bit relative address value

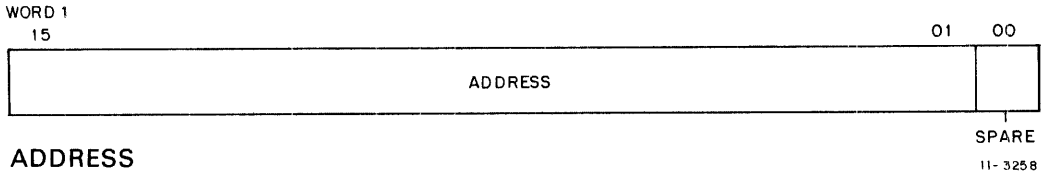
Figure 3-19 Display Jump Relative



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110010_2

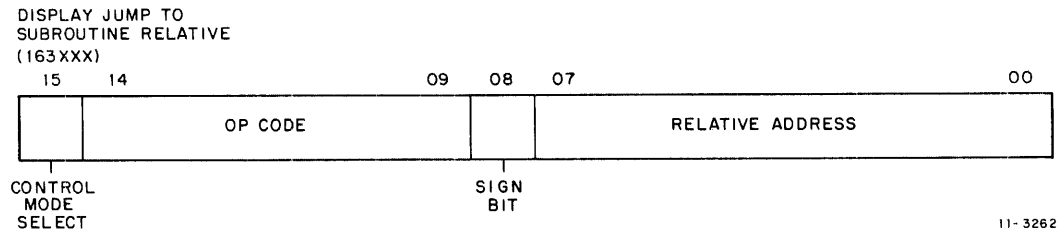
SPARE
Not used



ADDRESS
15-bit absolute address

SPARE
Not used

Figure 3-20 Display Jump to Subroutine Absolute



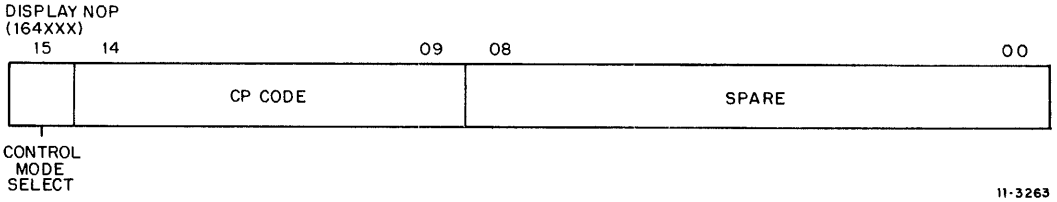
CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110011_2

SIGN BIT
0 = Add relative address to DPC
1 = Subtract relative address from DPC

RELATIVE ADDRESS
8-bit relative address of subroutine

Figure 3-21 Display Jump to Subroutine Relative

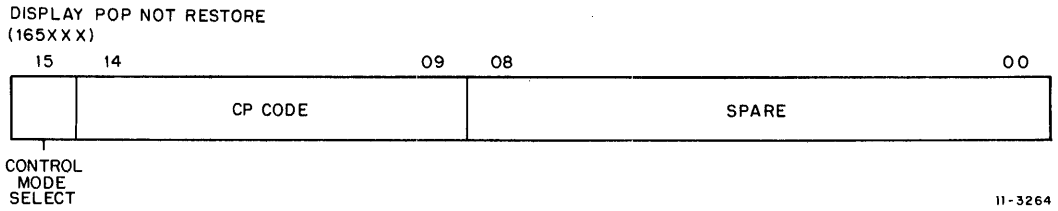


CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110100_2

SPARE
Not used

Figure 3-22 Display NOP

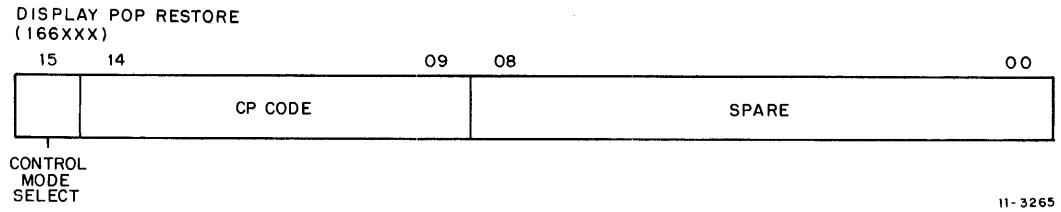


CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110101_2

SPARE
Not used

Figure 3-23 Display POP Not Restore

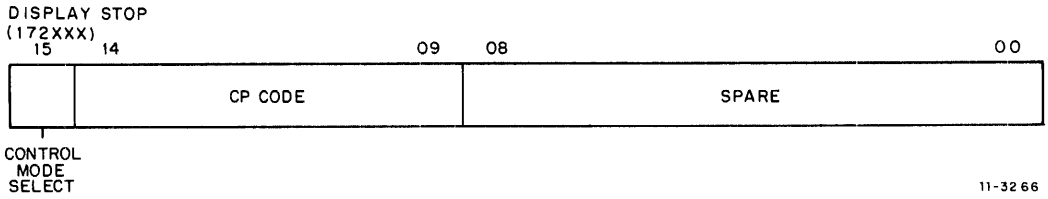


CONTROL MODE SELECT
1 Indicates control mode

OP CODE
110110_2

SPARE
Not used

Figure 3-24 Display POP Restore



CONTROL MODE SELECT
1 Indicates control mode

OP CODE
11101_2

SPARE
Not used

Figure 3-25 Display Stop

3.8 VT48 ADDRESSABLE REGISTERS

This paragraph describes the VT48 addressable register formats. Single illustrations are provided to describe each format and to indicate whether the register is readable/writeable or both. A breakdown of the addressable registers and related illustrations is given below:

1. Display Program Counter (Figure 3-26) – This register contains the current display file address. It is also used to reinitiate (resume) processing following interrupts.

2. Mode Parameter Register (Figure 3-27) – This conveys the graphic mode code, intensity levels, and miscellaneous parameters when sampled by the PDP-11.

3. Graph Plot and X-Position Register (Figure 3-28) – This conveys the assigned graph plot increment value and addressed X position when sampled by the PDP-11.

4. Character and Y Position Register (Figure 3-29) – This conveys the current character and addressed Y position when sampled by the PDP-11.

5. Relocate Register (Figure 3-30) – When written, the Relocate register can be used to branch within core memory in systems exceeding 32K of memory and ranging up to 128K of memory.

6. Status Parameter Register (Figure 3-31) – When sampled by the PDP-11, this register conveys various status/parameters to the CPU.

7. X/Y Offset Registers (Figure 3-32 and 3-33) – When written, these registers can be used to "window" within the virtual display area. This feature is discussed later in this section. When read, the X/Y Position register's high order bits are returned along with the X/Y offset values.

8. Associative Name (Figure 3-34) – This register (write only) is loaded with an assignable unique bit pattern up to 11 bits in length. Once loaded, the bit pattern or associative name is constantly compared with other name values situated in the display file (usually just prior to a string of graphic entities). When that portion of the display file having a like bit pattern is processed through the VT48, a matchup is detected and an interrupt occurs. Thus, the associative name register can be used to search the display file (data base) for a desired graphics entity string.

9. Slave Console/Color Register (Figure 3-35) – This register is used to read interrupt conditions on console 0 and console 1 (slave console), if included in the system. It is also used to select color if the color option is included.

10. Name Register (Figure 3-36) – This register is read only and conveys the name value from the dislay file (i.e., inserted via the name instruction) when sampled by the PDP-11.

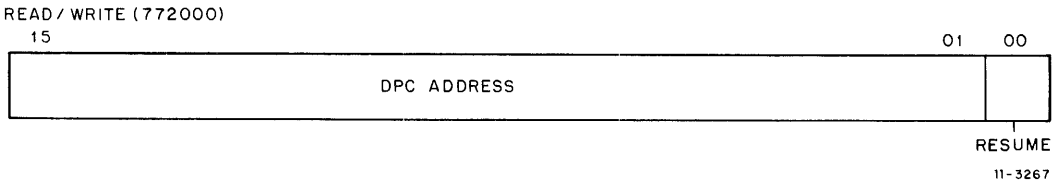
11. Stack Data Register (Figure 3-37) – This register conveys 16 bits of stack memory data as addressed by the Stack Address/Maintenance register.

12. Character String Terminate Register (Figure 3-38) – This register is used to enable the character string escape capability that forces a POP restore instruction on character matchup.

13. Stack Address/Maintenance Register (Figure 3-39) – This register is used to set up diagnostic conditions and to address 16-bits of stack memory.

14. Z Position Register (Figure 3-40) – This register is used with the depth queue option to allow status sampling of the current Z position.

15. Z Offset Register (Figure 3-41) – This register is used with the depth queue option to convey the Z offset status.



DPC ADDRESS

Read Considerations

Bits 01–15 convey low order 15 bits of VT48 Display Program Counter. The address value read represents the summation of the DPC and the 9 low order bits of the Relocate register. Hence, a read of the DPC directly after a write differs by an amount equivalent to the value in the Relocate register. Bit 00 has no meaning during read operations.

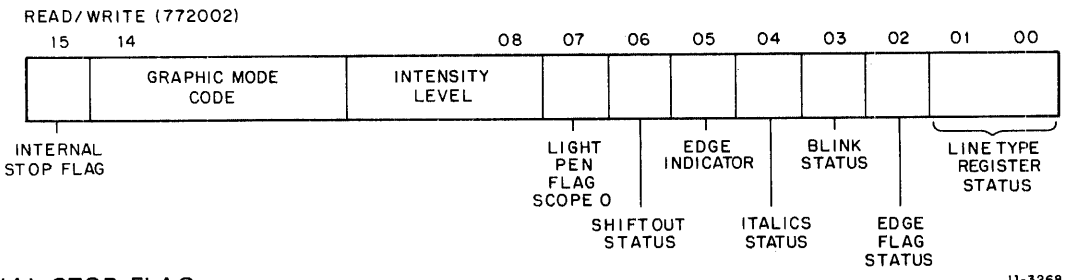
Write Considerations

Display file address is written into the DPC and an implied START is given to the VT48.

RESUME

Writing a 1 into bit 00 following an interrupt causes the VT48 to reinitiate processing (at the current DPC address) of the display file.

Figure 3-26 Display Processor Program Counter



INTERNAL STOP FLAG
0 = Flag cleared, 1 = Flag set

GRAPHIC MODE CODE
0000 = Character, 0001 = Point, XXXX etc.

INTENSITY LEVEL
000 = Dimmest, 111 = Brightest (Intensity levels are defaulted to a level 4 on initialization)

LIGHT PEN FLAG, SCOPE 0
0 = Flag cleared, 1 = Flag set

SHIFT OUT STATUS
0 = Shift in characters, 1 = Shift out characters

EDGE INDICATOR
0 = Within visible areas, 1 = Outside visible area

ITALICS STATUS
0 = Normal font selected, 1 = Italics selected

BLINK STATUS
0 = Blink disabled, 1 = Blink on

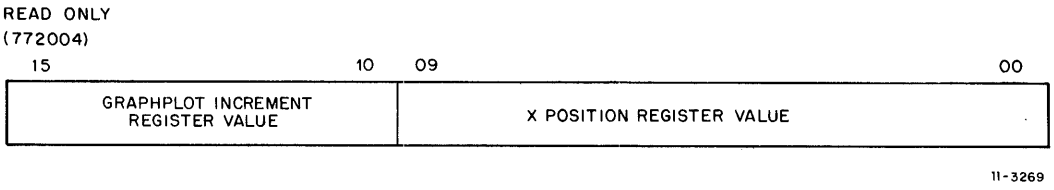
EDGE FLAG STATUS
0 = Flag cleared, 1 = Flag set

LINE TYPE REGISTER STATUS
00 = Solid line, 01 = Long dash line
10 = Short dash line, 11 = Dot dash line

READ CONSIDERATIONS
On read, parameters and status indicated above are returned

WRITE CONSIDERATIONS
The sole action produced by a write to this register is to cause the bell on the LK40 keyboard to beep

Figure 3-27 Mode Parameter Register

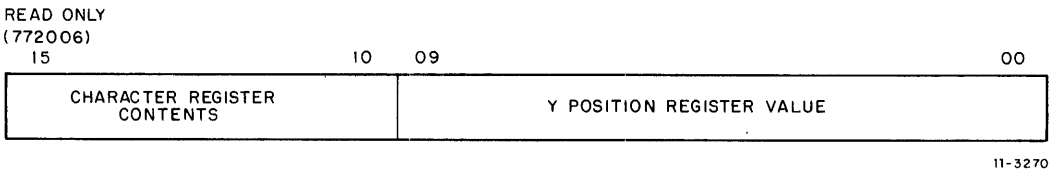


GRAPHPLOT INCREMENT REGISTER VALUE
0 to 63 unit content of graphplot Increment register

X POSITION REGISTER VALUE
X Coordinate current beam position

READ CONSIDERATIONS
On read, parameters indicated above are returned

Figure 3-28 Graphplot Increment and X Position Register

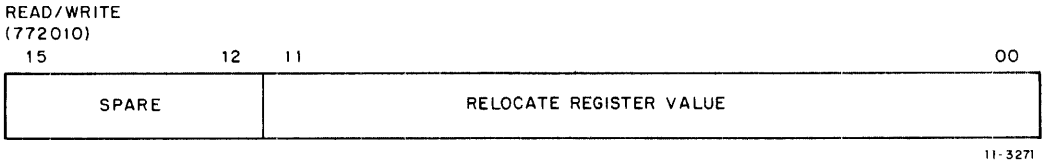


CHARACTER REGISTER CONTENTS
Current value of 6 low order bits of Character register

Y POSITION REGISTER
Y Coordinate current beam position

READ CONSIDERATIONS
On read, parameters indicated above are returned

Figure 3-29 Character Code and Y Position Register



SPARE

Not used

RELOCATE REGISTER

12-bit value added to contents of VT48 Display Program Counter to form complete DPC 17-bit address

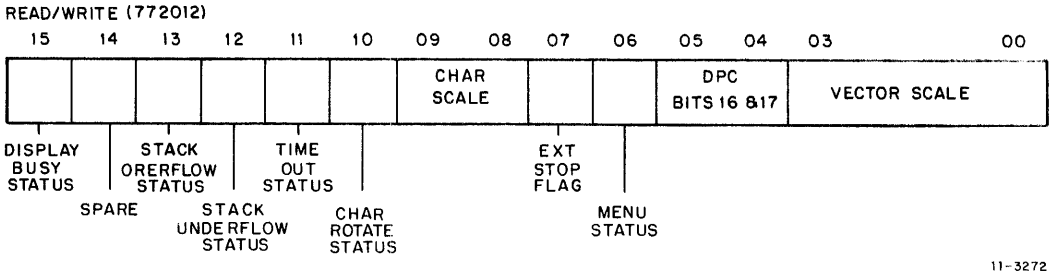
READ CONSIDERATIONS

On read, the current value of the Relocate register is returned.

WRITE CONSIDERATIONS

On write, the 12-bit relocate address is inserted into the Relocate register. The outputs from the register are in turn added to the 12 high order bits (i.e., left justified) of the display processor program counter to form a complete 17-bit DPC address.

Figure 3-30 Relocate Register



DISPLAY BUSY STATUS

0 = Display stopped by flag, 1 = VT48 was initiated or resumed

SPARE

Not used

STACK OVERFLOW STATUS

0 = No overflow, 1 = Attempt was made to "push" stack when stack was full

STACK UNDERFLOW STATUS

0 = No underflow, 1 = Attempt was made to "pop" stack when stack was at highest level

TIME OUT STATUS

0 = No time out has occurred, 1 = Time out flag set

CHAR. ROTATE STATUS

0 = Zero degrees character rotation, 1 = 90 degrees cc character rotation

CHAR. SCALE STATUS

00 = Smallest, 11 = Largest

EXTERNAL STOP FLAG

0 = No flag, 1 = External stop flag set

MENU STATUS

0 = Normal working surface, 1 = Display menu area

DPC BITS 16 AND 17

Two high order bits of VT48 Display Processor Program Counter

VECTOR SCALE

0000 = Smallest, 1111 = Largest

READ CONSIDERATIONS

On read, status parameters indicated above are returned.

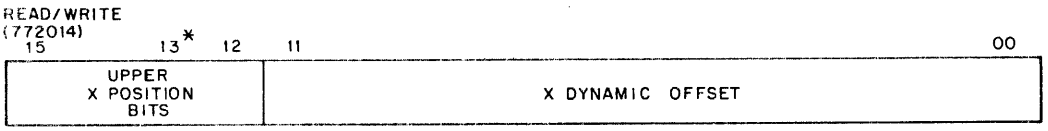
WRITE CONSIDERATIONS

Write operations affect bit 7 only. That is when written to a 1, it represents an external stop request

OTHER CONSIDERATIONS

Flags are cleared only by START, POWER CLEAR and RESUME commands

Figure 3-31 Status Parameter Register



UPPER X POSITION BITS
Conveys 4 high order bits of X Position register (Read Only)

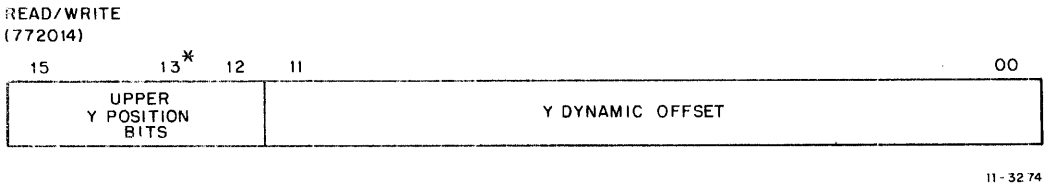
X DYNAMIC OFFSET
12-bit value added to/subtracted from X Position register depending on sign

*(BIT 13)
Conveys X offset sign value (Write Only)

READ CONSIDERATIONS
On read, the parameters above (except X sign bit) are returned

WRITE CONSIDERATIONS
On write, the 12-bit X dynamic offset value is inserted into the X Offset register and bit 13 is written as the sign bit

Figure 3-32 X Offset Register



UPPER Y POSITION BITS
Conveys 4 high order bits of Y Position register (Read Only)

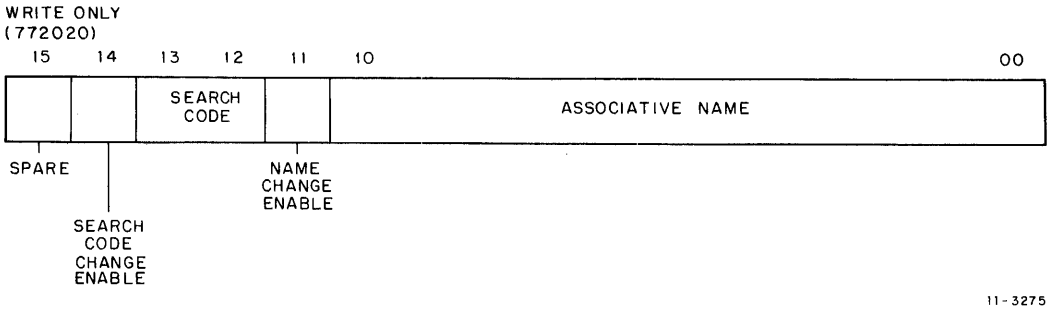
Y DYNAMIC OFFSET
12-bit value added to/subtracted from Y Position register depending on sign

*(BIT 13)
Conveys Y offset sign value (Write Only)

READ CONSIDERATIONS
On read, the parameters above (except Y sign bit) are returned

WRITE CONSIDERATIONS
On write, the 12 bit Y dynamic offset value is inserted into the Y Offset register and bit 13 is written as the sign bit

Figure 3-33 Y Offset Register



SPARE

Not used

SEARCH CODE CHANGE ENABLE

0 = No change, 1 = Enter bits 12 and 13 into Search register

SEARCH CODE

Two bit code entered into search register:

00 = No search, No interrupt
01 = Interrupt on 11 bit compare
10 = Interrupt on high order 8-bit compare
11 = Interrupt on high order 4-bit compare

NAME CHANGE ENABLE

0 = No change, 1 = Enters bits 00 through 10 into Associative Name register

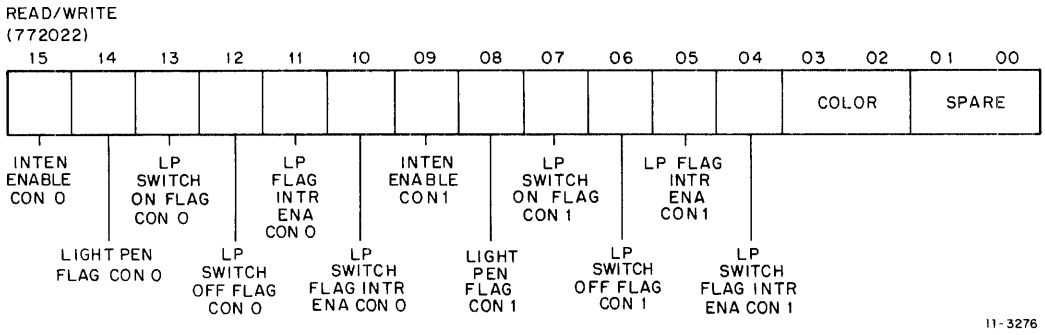
ASSOCIATIVE NAME

11, 8, or 4-bit compare value

WRITE CONSIDERATIONS

Written with 11, 8, or 4-bit compare value to search for graphic entities previously labeled (named) via the name instruction. When the value entered into the Associative Name register compares with the label in the Name register, an interrupt flag is asserted. In effect then, the Associative Name register provides a convenient means of searching the display file for previously labeled (named) graphic entities

Figure 3-34 Associative Name Register



INTEN ENABLE CON. 0

Read Value	Write Value
0 = Console 0 intensity off	Not writeable
1 = Console 0 intensity on	

LIGHT PEN FLAG CON. 0

Read Value	Write Value
0 = Console 0 light pen flag off	1 = Turn LP flag on*
1 = Console 0 light pen flag on	

LP SWITCH ON FLAG CON. 0

Read Value	Write Value
0 = No flag	1 = Turn switch flag off*
1 = Switch on flag set	

LP SWITCH OFF FLAG CON. 0

Read Value	Write Value
0 = No flag	1 = Turn switch flag on*
1 = Switch off flag set	

LP FLAG INTR ENA, CON. 0

Read Value	Write Value
0 = LP flag disabled	Not writeable
1 = LP flag enabled	

LP SWITCH FLAG INTR ENA CON. 0

Read Value	Write Value
0 = LP switch flag disabled	Not writeable
1 = LP switch flag enabled	

INTEN ENABLE CON. 1

Read Value	Write Value
0 = Console 1 intensity off	Not writeable
1 = Console 1 intensity on	

*Maintenance Switch 3 must be turned on to write these bits.

Figure 3-35 Slave Console/Color Register (Sheet 1 of 2)

LIGHT PEN FLAG CON. 1

Read Value
0 = Console 1 light pen flag off
1 = Console 1 light pen flag on

Write Value
1 = Turn LP flag on*

LP SWITCH ON FLAG CON. 1

Read Value
0 = No flag
1 = Switch on flag set

Write Value
1 = Turn switch flag on*

LP SWITCH OFF FLAG CON. 1

Read Value
0 = No flag
1 = Switch off flag set

Write Value
1 = Turn switch flag off*

LP FLAG INTR ENA, CON. 1

Read Value
0 = LP flag disabled
1 = LP flag enabled

Write Value
Not writeable

LP SWITCH FLAG INTR ENA CON. 1

Read Value
0 = LP switch flag disabled
1 = LP switch flag enabled

Write Value
Not writeable

COLOR

Read Value
00 = Green, 01 = Yellow
10 = Orange, 11 = Red

Write Value
Not writeable

SPARE

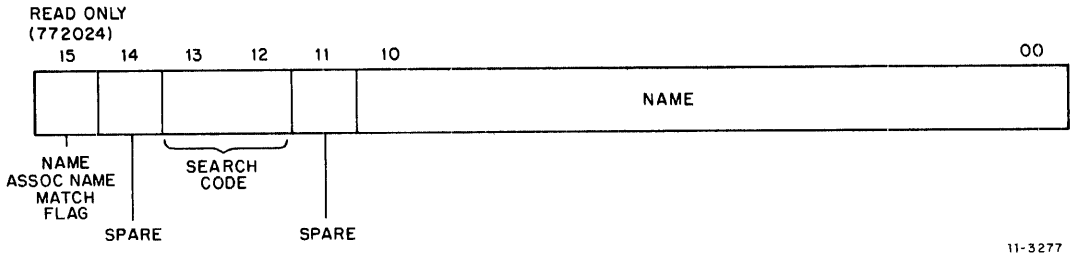
Not Used

*Maintenance Switch 3 must be turned on to write these bits.

NOTE

The intensity enable bit for console 1 is cleared at power clear and start times; the intensity enable bit for console 0 is set at these times.

Figure 3-35 Slave Console/Color Register (Sheet 2 of 2)



NAME/ASSOC NAME MATCH FLAG

0 = No flag

1 = Name match occurred, flag set

SPARE

Not used

SEARCH CODE

00 = No search, 01 = Full 11 bit compare

10 = 8 high order bit compare, 11 = 4 high order bit compare

SPARE

Not used

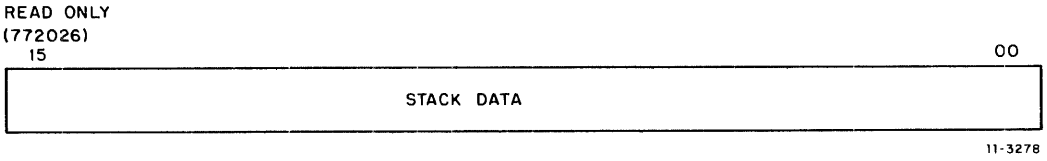
NAME

11-bit name (label) tag from Name register

READ CONSIDERATIONS

When a name match interrupt occurs, the CPU can read the name value that matched the contents of the Associative Name register. Bits 12 and 13 convey the search code written at the same time the Associative Name register was written.

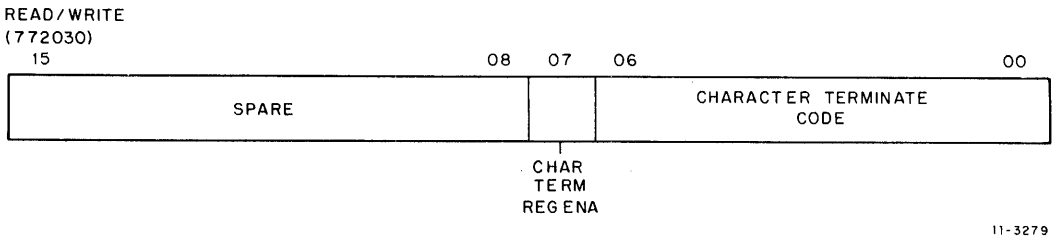
Figure 3-36 Name Register



STACK DATA

16 bits of VT48 stack data as addressed by Stack Address/Maintenance register

Figure 3-37 Stack Data Register



SPARE

Not used

CHAR TERM REG ENA

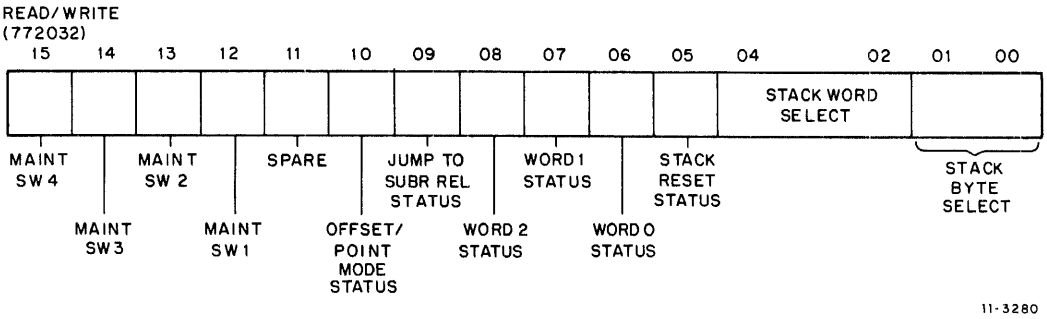
0 = Inhibit write of Character String Terminate register

1 = Enable write of Character String Terminate register

CHARACTER TERMINATE CODE

7-bit code entered into Character String Terminate register to terminate character string processing on match with character being executed.

Figure 3-38 Character String Terminate Register



MAINT SW. 4

Read Value
0 = Switch off
1 = Switch on

Write Value
0 = Turn maint. switch off
1 = Turn maint. switch on
(used to single step the graphics calculation logic and for troubleshooting)

MAINT SW. 3

Read Value
0 = Switch off
1 = Switch on

Write Value
0 = Turn maint. switch off
1 = Turn maint. switch on
(used with diagnostic Test IV)

MAINT SW. 2

Read Value
0 = Switch off
1 = Switch on

Write Value
0 = Turn maint. switch off
1 = Turn maint. switch on
(used with diagnostic Test I, II & III)

MAINT SW. 1

Read Value
0 = Switch off
1 = Switch on

Write Value
0 = Turn maint. switch off
1 = Turn maint. switch on
(used with diagnostic Test I, II & III)

OFFSET MODE STATUS

Read Value
1 = Last data executed was to load the Offset registers

Write Value
Not writeable

SPARE

Not used

JUMP TO SUBR REL STATUS

Read Value
0 = No relative jump to subroutine executed
1 = Jump to subroutine occurred on last control command executed

Write Value
Not writeable

Figure 3-39 Stack Address/Maintenance Register (Sheet 1 of 2)

WORD 2 STATUS
0 = Word not in process
1 = Word in process

WORD 1 STATUS
0 = Word not in process
1 = Word in process

WORD 0 STATUS
0 = Word not in process
1 = Word in process

STACK RESET STATUS

Read Value
0 = No meaning
1 = Stack pointer reset

Write Value
1 = Reset stack pointer

STACK WORD SELECT

Read Value
Word 0, 1, 2, 3, 4, 5, 6, or 7 selected

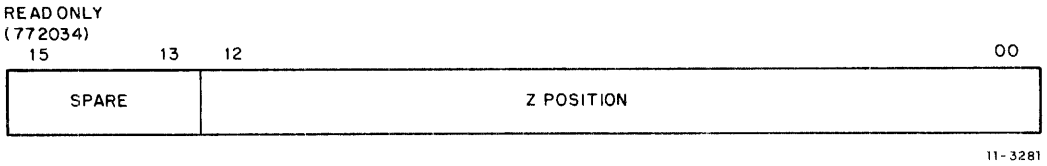
Write Value
111 = Select word 0
110 = Select word 1
101 = Select word 2
100 = Select word 3
011 = Select word 4
010 = Select word 5
001 = Select word 6
000 = Select word 7

STACK BYTE SELECT

Read Value
Byte 0, 1, 2, or 3 selected

Write Value
00 = Select byte 0
01 = Select byte 1
10 = Select byte 2
11 = Select byte 3

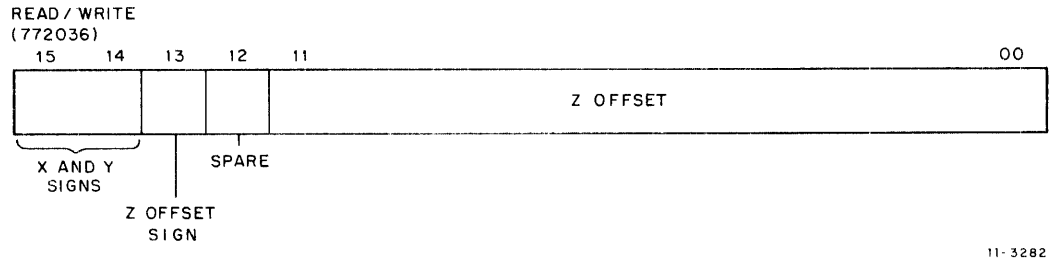
Figure 3-39 Stack Address/Maintenance Regiter (Sheet 2 of 2)



SPARE
Not used

Z POSITION
Conveys bits 2 through 13 of Z Position register

Figure 3-40 Z Position Register, Depth Queue Option



X AND Y SIGNS

Read Value	Write Value
Bit 14 = X position sign	Not writeable
Bit 15 = Y position sign	

Z OFFSET SIGN
Sign bit for Z dynamic offset

Z OFFSET
12-bit Z dynamic offset value

Figure 3-41 Z Offset Register, Depth Queue Option

3.9 IMAGE GENERATION AND MANIPULATION

3.9.1 Display File Make-up and Refresh Considerations

The display file can be defined as the sum total of all instructions (control and graphic entity) processed through the VT48 Display Processor Unit in a single refresh frame. When the 40 frames per second refresh rate* is selected, the entire display file is processed through the display processor unit at this rate. Selection of the 30 frame per second refresh rate* proportionately decreases the number of times the display file is processed. With neither of the above selected, the refresh rate is synchronized on the display file itself. That is, the jump instruction (that conveys the display file starting address) is executed without any intervening delays at the end of the display file. Under these conditions, the refresh rate is a function of display file length.

*Selected by load status A instruction.

In structuring a display file, the usual or prevalent sequence of instructions is:

1. Load Status Instructions – Such instructions are used to set up parameters regarding the images to be generated.

2. Set Graphic Mode Instruction – This establishes the type of graphic entity to be drawn. That is, point, character, long vector, etc.

3. Graphic Entity Instructions – These are image-generating instructions that produce the graphic symbols refreshed at the CRT monitor.

4. Branch Instructions – These are used to branch to different areas in the display file and may or may not be executed as required by a given application.

The instructions listed in 1 through 3 above can be used at the start of the display file to draw the first graphic symbol (or set of symbols) on the VR48 Display Monitor. Next, assuming no change in status, the set graphic mode and graphic entity instructions could be used to generate a second symbol/group of symbols. Further, a third set of instructions can be used to display a third set of symbols, etc. Should a branch instruction be inserted to switch to a subroutine location, the subroutine would normally execute load status, and set graphic mode and graphic entity instructions (in this order) to draw the symbols specified by the subroutine. In this way then, an entire display file can be structured.

3.9.2 Visible and Virtual Display File Entities

At any given time, the VT48 may be processing graphic entity instructions that are contained in the display file but are not displayed on the VR48 Display Monitor. In other words, at times only a fraction of the total number of graphic entities in the display file are refreshed at the VR48 Display Monitor. This condition stems from the fact that the point instruction allows for a 12-bit by 12-bit graphic world (virtual display area) while the long vector instructions allows for only a 10-bit "window" into the graphic world. That is, the VT48 Vector Generator only draws those graphic entities that affect a 10-bit by 10-bit "window."

NOTE

Absolute point instructions and consecutive processed long vector instructions can also be used as a means of specifying coordinate data in the virtual display area. However, for purposes of explanation, the point instruction is useful since its 12-bit limits are the same as that of the virtual display area.

If the VT48 X/Y position storage registers could only accommodate 10 bits of X/Y data rather than 12-bits, then the virtual display area and the display refresh area would be one and the same. That is, everything would fall within the shaded area on Figure 3-42. Also, if this were the case, everything in the display file would be drawn on the VR48 Display Monitor. However, since the point instruction is actually 12 bits in length (X, Y plus sign bits), it can be used to locate oneself anywhere within the 64 blocks shown on Figure 3-42. The entire virtual display area is shown in this illustration. (If only positive sign values could be used, the virtual display area would be limited to the 16 squares at the upper right of Figure 3-42.) Techniques for moving any portion of the virtual display area beneath the "window" are described in the subsequent paragraphs.

3.9.3 Windowing Through Use of Point and Offset Capabilities

Locating oneself within the virtual display area (i.e., the process known as "windowing") can be readily accomplished through use of the VT48 point and offset features. The point instruction allows one to select any 10-bit by 10-bit square in the virtual display area. The offset registers can then be

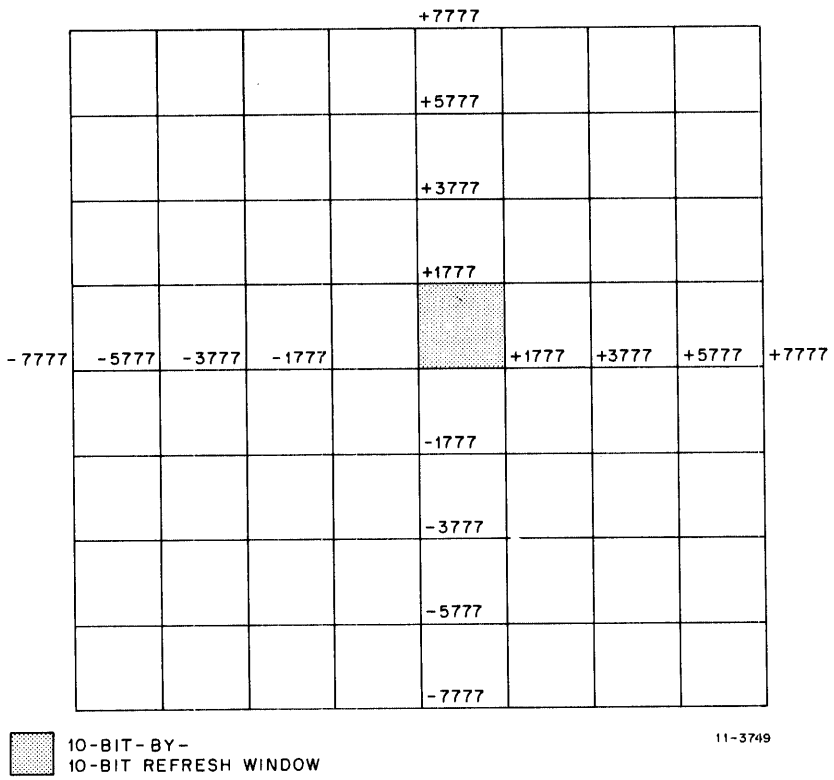


Figure 3-42 VT48 Virtual Display Area Showing Octal Point Notations

loaded with counteracting or offsetting values that have the effect of positioning the selected square in the 10-bit "window" viewing area.

Bear in mind the earlier statement about how the virtual and actual display areas would be one and the same if the X/Y position registers were limited to 10-bits in length. This is significant because the VT48 graphics calculation logic looks at each set of coordinates to determine whether the point falls within the 10-bit by 10-bit "window" area. Based on this determination, the VT48 graphics calculation logic takes one of two courses of action.

1. If the point falls within the window area, the VT48 graphics calculation logic commands the vector generator to move the beam to the addressed point within the "window."

2. If the point falls outside the window area, the graphics calculation logic disregards it and goes on to the next graphic mode instruction.

NOTE

For relative vectors, the graphics calculation logic must look at both start points and end points to determine if any portion of a vector might fall within the window area. For this discussion however, it is assumed that all points and vector segments fall outside the 10-bit by 10-bit window.

If a means existed for deceiving the graphics calculation logic so that it recognizes points, vectors, etc., as falling within the "window" area (even though they actually were defined as being elsewhere in the

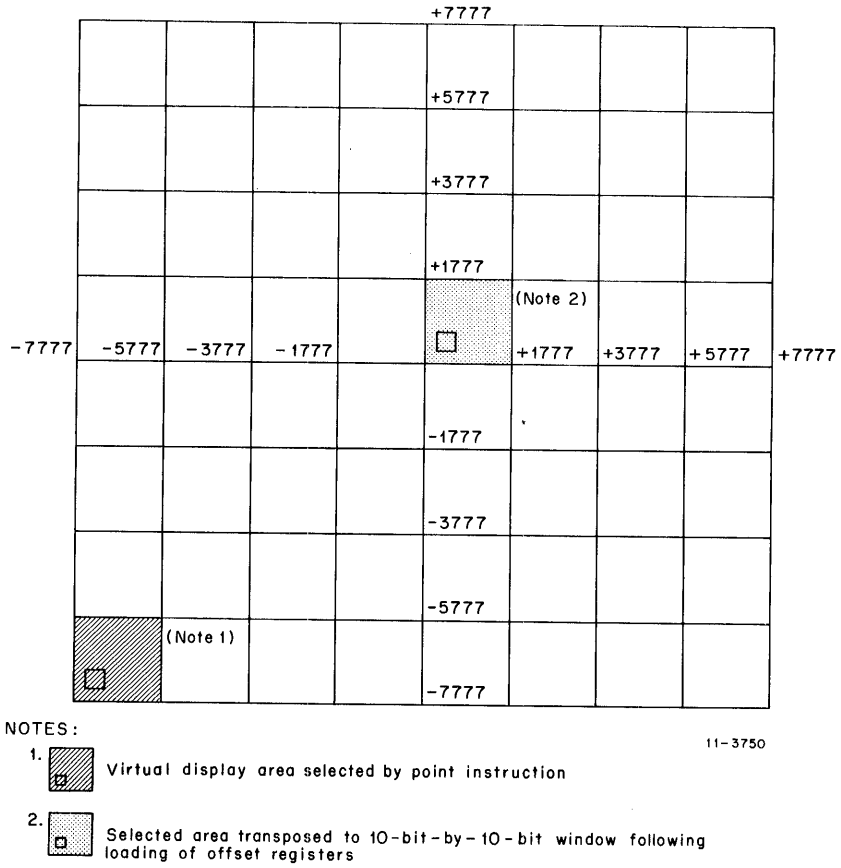


Figure 3-43 Selection of Lower Left Corner of Virtual Display Area for Insertion in Refresh Window

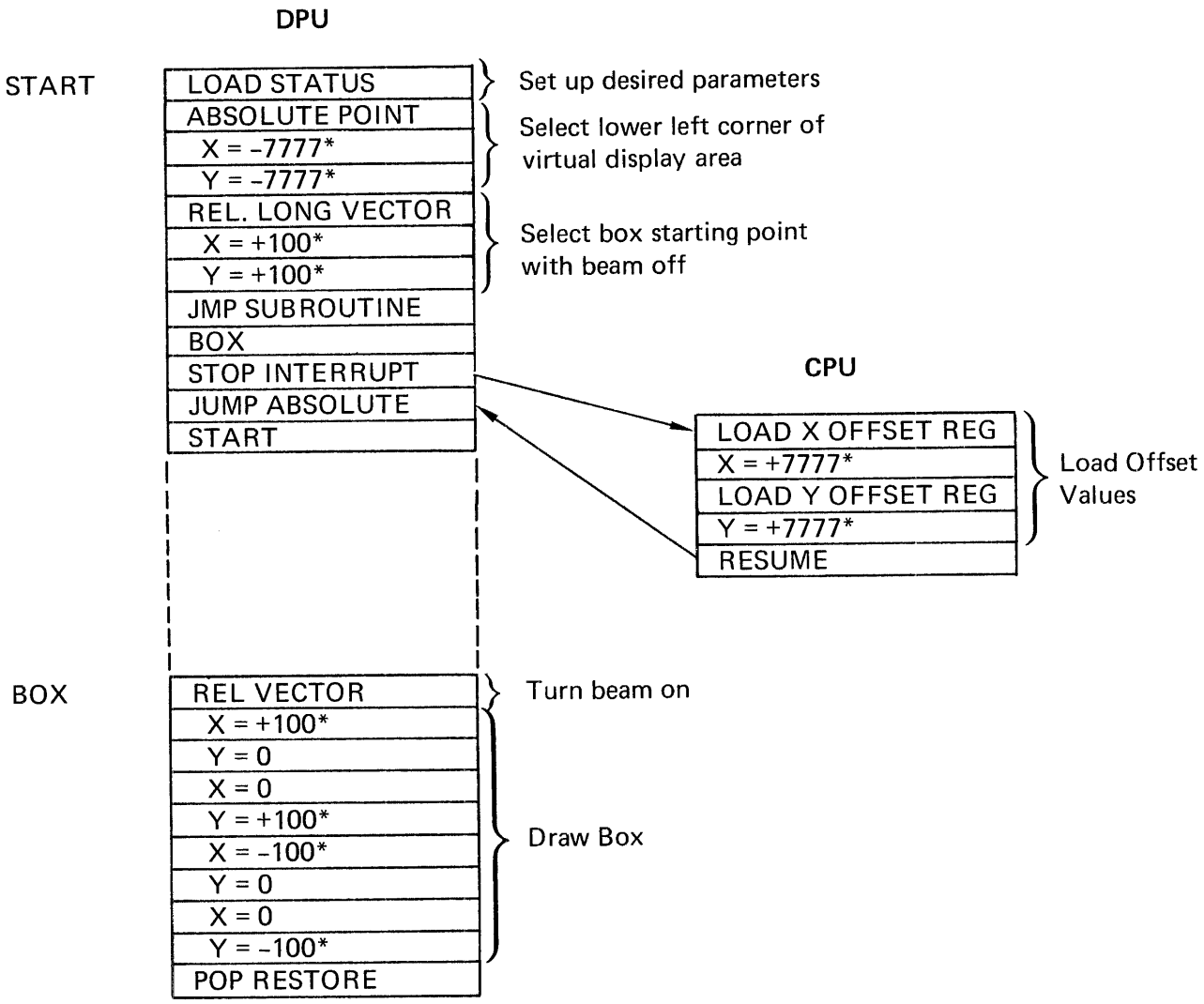
virtual display area), then the graphics calculation logic would command the vector generator to draw them on the VR48 Monitor. The X, Y offset registers within the graphics calculation logic provide this means.

The following "windowing" example assumes a set of circumstances where it is desired to select the lower left hand corner of the virtual display area for transposition to the 10-bit by 10-bit window area. Assume further that a box is to be drawn in the selected area (Figure 3-43). Following this, the offset registers are set to a value that places the selected area within the 10-bit by 10-bit window. Coding to effect this result is as shown in Figure 3-44.

If means are provided (a directory in the CRT Display Menu area) to dynamically alter the content of the X/Y offset registers, then some other square in the virtual display area can be moved beneath the refresh "window."

NOTE

The selected square in the virtual display area need not have the boundaries shown in Figure 3-43. By changing the content of the low order bits of both point instruction and offset registers, a square overlapping those outlined may be selected.



*Octal Notation

Figure 3-44 Coding to Select Lower Left Corner of Virtual Display Area for Insertion in Refresh Window

When the VR48 Console operator alters the content of the X, Y offset registers, the box shown in Figure 3-43 is not deleted from the display file. It remains part of the stream of instructions processed through the VT48. However, it is not displayed because it no longer falls within refresh "window."

3.9.4 Advantages of "Windowing"

The advantages of "windowing" might best be understood in terms of an example. Assume an application where the display system is to be used to optimize heavy equipment layout in a 100 × 200 foot building. If the entire length (200 feet) were defined as being within the refresh "window" (physically 12-by-12 inches on the CRT Monitor) then one-inch of viewing area would represent 16.6 feet of linear floor space. Obviously, this is much too congested to allow any clear manipulation of passageways, access, and egress areas to select the most efficient equipment layout.

By taking advantage of the VT48 "windowing" feature, more detailed viewing and analysis of the building floor space can be obtained. This may be done simply by defining the building perimeter as falling within the eight square foot area shown in Figure 3-45. Now one inch of the 12 inch by 12 inch viewing area (regardless of what square is moved beneath the "window") represents only 4.17 feet. Further, if the building perimeter were defined as falling within the 32 square foot area (Figure 3-45), each inch of viewing area represents 2.08 feet. By moving each square of floor space beneath the refresh window, the designer can now observe and analyze with greater clarity when determining the most efficient equipment layout.

NOTE

Ideally, the designer will have a directory in the CRT menu area which he can use to select readily any area of floor space for display in the refresh "window." The directory could be set up so that he/she could use the light pen to alter dynamically the contents of the offset registers and thereby move new areas of floor space beneath the refresh "window."

The "windowing" feature, coupled with the scaling capability (discussed later) allows the designer to see both overviews and detailed pictures of the layout problem.

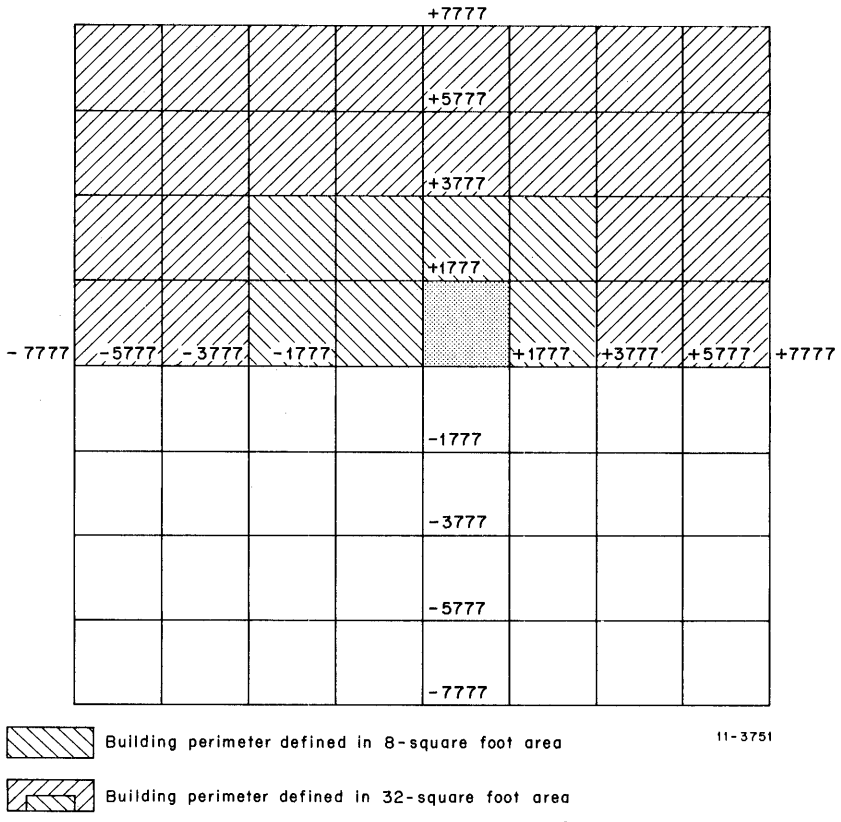


Figure 3-45 Possible Definitions of a 200 by 100-Foot Building Within Virtual Display Area

3.9.5 Image Scaling and Scissoring

The VT48 Display Processor Unit offers a wide range of scale values to be used in enlarging or reducing the size of the image(s) refreshed on the CRT monitor. This process, sometimes referred to as "zooming" is effected simply by changing the scale factor contained in the VT48 scale register. Selectable scale factors[1] range from a minimum of 1/4 normal size[2] to 3-3/4 normal size.

Effects achievable through scaling are indicated on the illustrations given on the subsequent pages. Figure 3-46 shows display file coding used to generate the image shown in Figure 3-47, Part A. The upper part of the latter illustration shows the image being generated at a 1:1 scale factor. That is, the scale factor initially selected by the load status C instruction.

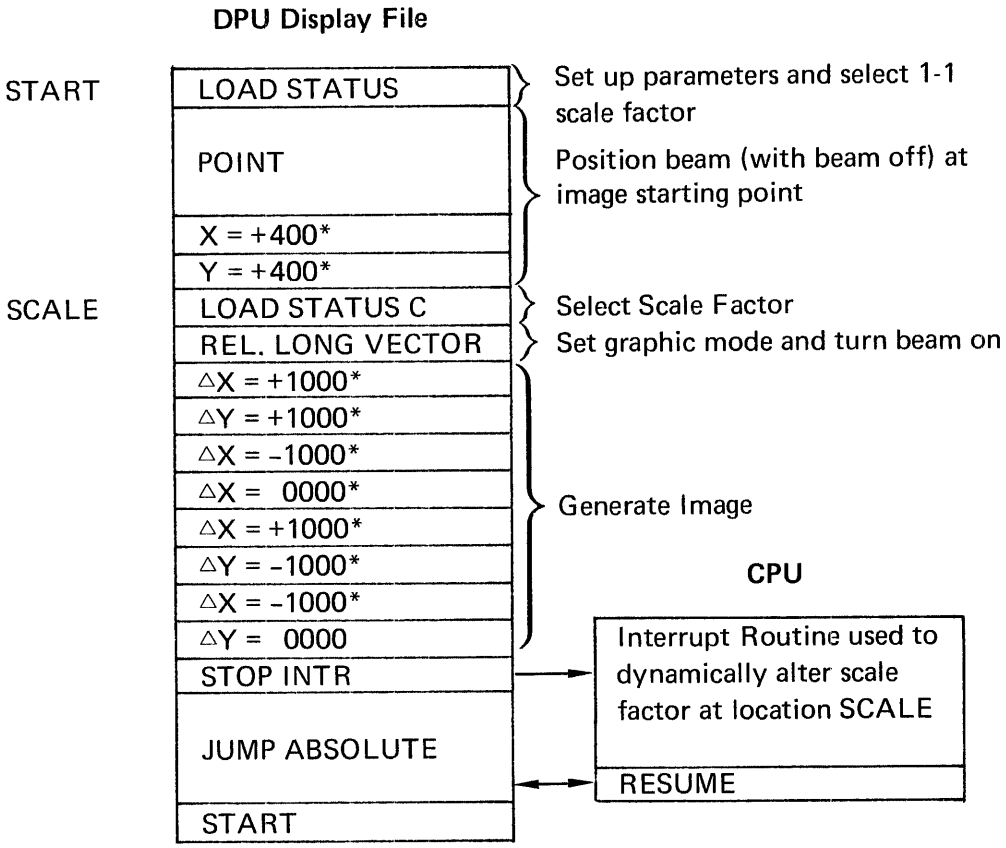


Figure 3-46 Display File Coding Used to Generate Typical Image at Desired Scale Factor

---

[1]See load status C instruction.

[2]Normal size is defined as the 1:1 scale factor, i.e., the scale factor normally used when initially generating an image.

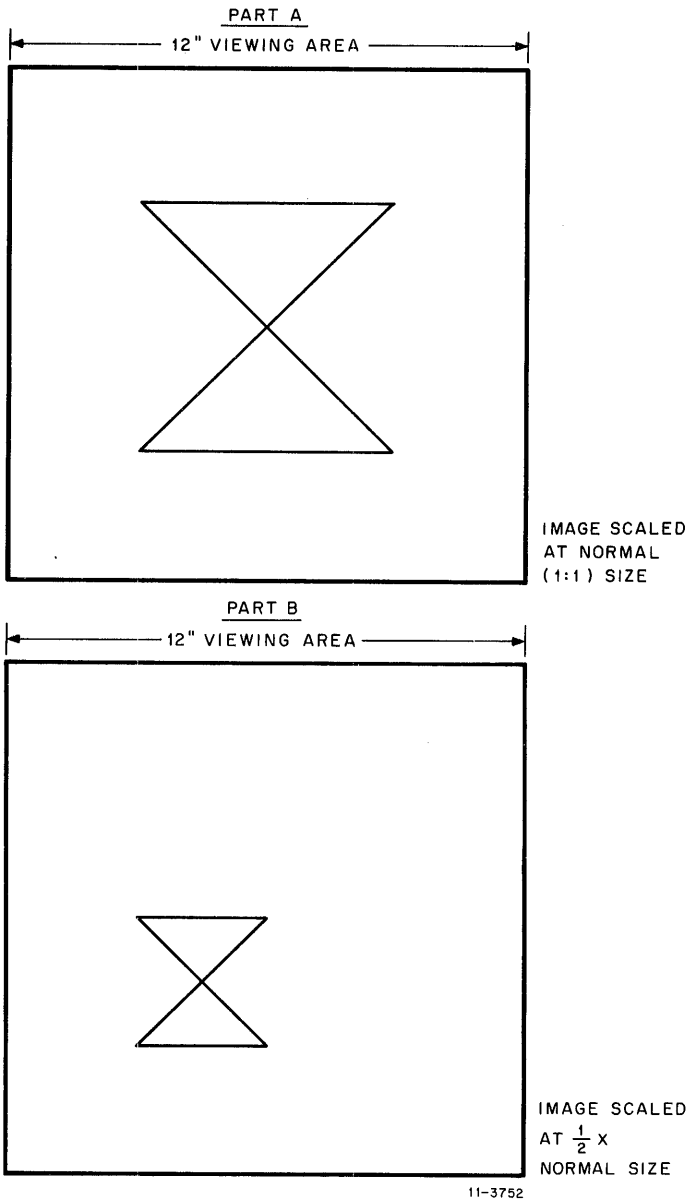


Figure 3-47 Typical Image Scaled at Normal and 1/2 Normal Sizes

Part B of Figure 3-47 shows the same image being refreshed at 1/2 scale. Notice that the image starting point has not changed. This is because the changed scale factor (load status C instruction) is so positioned that it affects the relative long vector instructions but not the point instruction. In other words, the absolute starting point is maintained.

NOTE

For a true zoom effect when changing scale factors, the content of the X/Y offset registers must also be changed. For example, if the X/Y offset registers were loaded with $+0200_8$ prior to changing the scale factor, then the reduced image would appear centered in the viewing area; that is, the same as the image is centered in Part A.

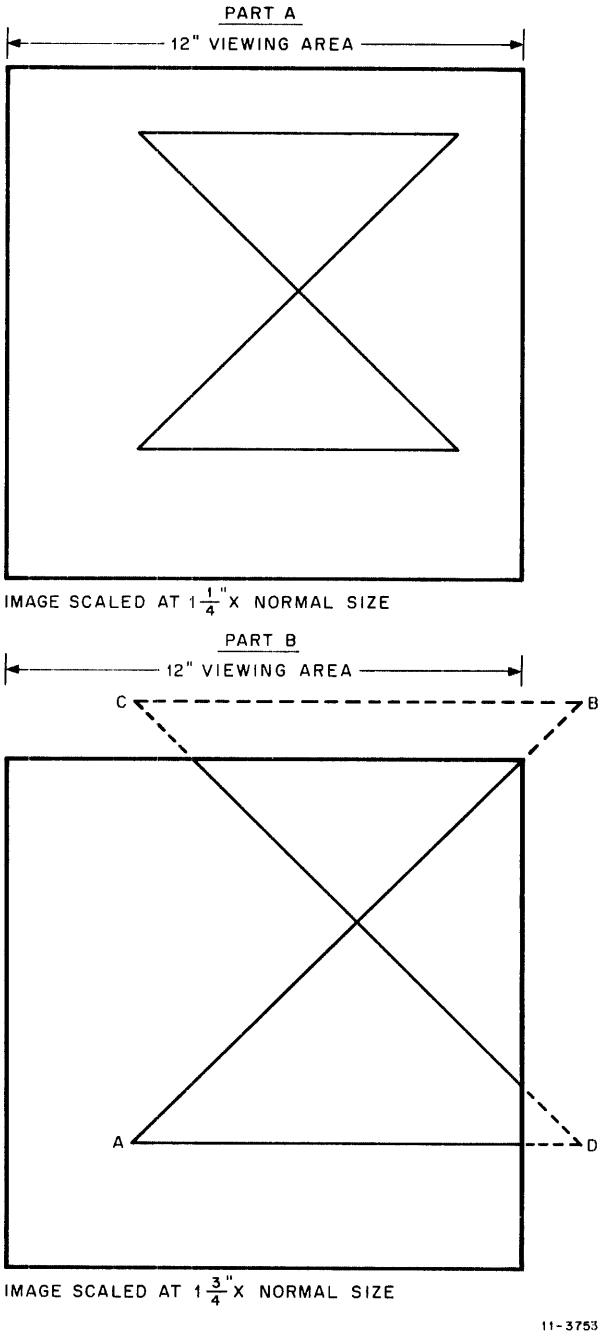


Figure 3-48 Typical Image Scaled at 1-1/4 and 1-3/4 Normal Size

The graphics calculation logic in the display processor unit always multiplies X/Y values by the values contained in the scale register. (The scale register is loaded on execution of the load status C instruction.) Consequently, the relative long vector values shown in Figure 3-46 are multiplied by one-half. As a result, the vector generator is commanded to draw smaller vectors and the image appears to be halved.

Figure 3-48 shows the same image being generated at 1-1/4 times (Part A) and 1-3/4 times (Part B). In Part A, the scale factor is changed to 1-1/4 and, as in all cases, affects the relative long vector instructions only. The size of the 10-bit by 10-bit window is sufficient to accommodate a change of this magnitude and the entire image is refreshed.

When the scale factor is set to 1-3/4 times normal image, the scissoring (sometimes called clipping) capabilities of the graphics calculation logic come into play. Only the solid lines on Part B are displayed. The dashed lines of the image are rejected as being outside the "window" by the graphics calculation logic.

On processing all lesser scaled images (i.e., smaller than 1-3/4 scale), the graphics calculation logic determined that all portions of all vectors fell within the window area. Such vectors are considered as on-to-on vectors (on-screen to on-screen) since the start and end points are within the viewing area. For each vector then, the graphics calculation logic commands the vector generator to draw the entire vector. When an image is scaled to a point where portions fall outside the viewing area, the graphics calculation logic must decide whether to draw all, some portion, or none of the vector. The four vectors shown on Figure 3-48, Part B represent all types of vectors (except on-to-on) processed through the VT48. Vector types and related courses of action taken by the graphics calculation logic are as follows:

1. Vector AB, on-to-off (on-screen to off-screen) – For vectors of this type, the Vector Generator is commanded to draw the viewable portion, i.e., from starting point to the point where the vector exits the viewing area. At the end of the draw period, the X/Y position registers are updated to contain the addressed end point (Point B). Since both coordinate values are now off-screen, bit 10 of the X/Y position registers is set.

2. Vector BC, off-to-off (off-screen to off-screen) with no viewable segment – Upon determining that no segment of this vector falls inside the viewing area, the graphics calculation logic simply updates the X/Y position registers (Point C) and then accepts the next pair of X/Y delta values for processing.

3. Vector CD, off-to-off (off-screen to off-screen) with viewable segment – In this case, the graphics calculation logic determines the entry and exit points of the viewable segment and then commands the vector generator to draw the viewable segment as a normal vector. After this, the X/Y position registers are updated to the vector end point (Point D).

4. Vector DA, off-to-on (off-screen to on-screen) – For this type, the graphics calculation logic determines where the vector enters the display window and then commands the vector generator to draw the viewable segment. After this, the X/Y position registers are updated to Point A.

Figures 3-49 and 3-50 present examples of how the VT48 scaling feature might be used in an architectural application. Figure 3-49 shows the compartment and furniture layout of an entire office building floor. In Figure 3-50, part of the office area has been scaled up for more detailed viewing. When the scale factor is increased, the desired compartments are enlarged and others are "scissored" out of the display picture. In Figure 3-50, the content of the X/Y offset registers is also changed to center the "window" on the desired area.

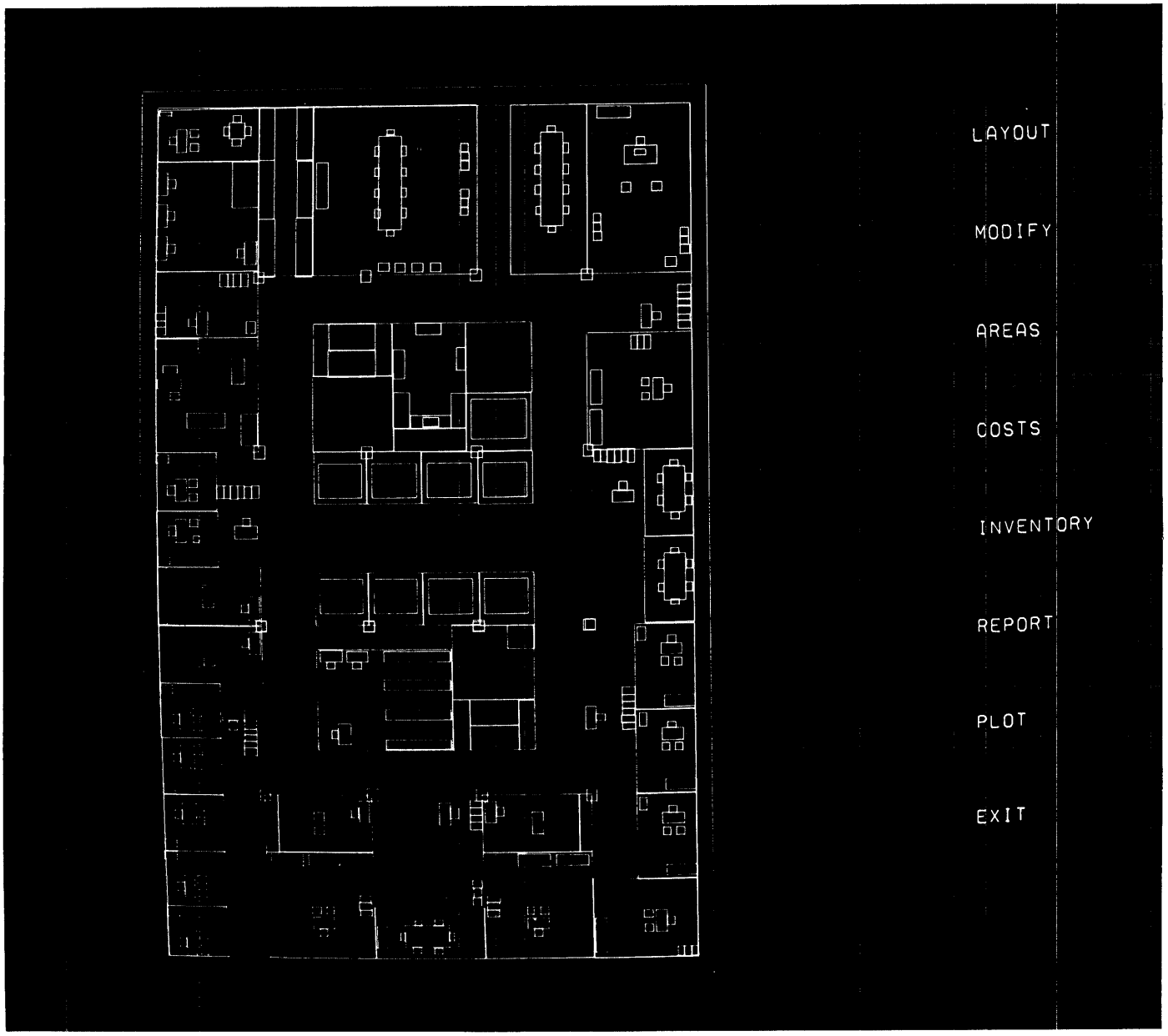


Figure 3-49 Example of Floor Plan Layout

3.9.6 Subpictures and Methods of Generation

A subpicture can be defined as any discrete portion of the total refreshed image. For example, the desks, tables and chairs shown in Figure 3-49 can be considered subpictures. So too, can the various rooms and compartments comprised in the overall image. There are two basic methods used to generate subpictures:

1. A block of in-line coding within the display file.

2. A subroutine that is branched to from the display file.

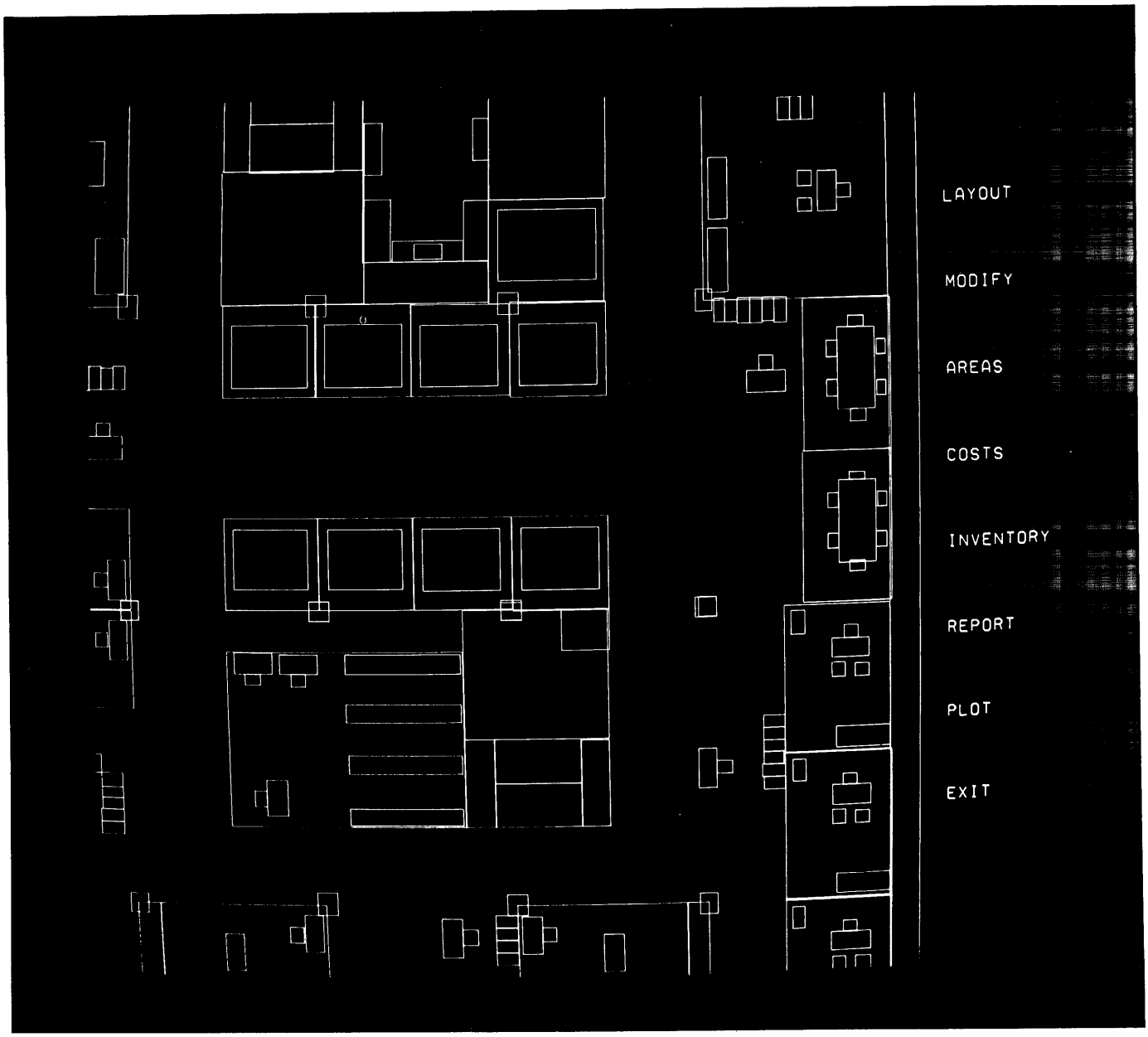


Figure 3-50 Example of Floor Plan Layout Enlarged Double Size and "Windowed" to the Lower Right Section

To generate unique subpictures, that is one-of-a-kind patterns, on the display, the first method should be used since no savings in core space can be achieved through use of subroutines. To generate repetitive images however, use of subroutines is by far the most practical and economical. Again, referring to Figure 3-49, the subpicture, or outline, indicating the presence of a chair occurs in excess of 50 times over the entire image. With a minimum of four instructions (short vector) required to generate the image, it can be seen that over 200 core locations would be consumed if each subpicture was generated independently within the display file. This is in contrast to the 6 or 7 locations consumed by a subroutine adequate to generate this type of image. In general then, repetitive, or macro type images are to be generated by taking advantage of the display processor unit JSR and POP instructions.

3.9.7 Subroutine Nesting Through use of Stack Memory

The stack memory within the display processor unit allows the user to nest subroutines. This feature is similar to the PDP-11 with the exception that a full 64 bits of information is "pushed" onto one of eight levels of the Stack memory each time a JSR (jump to subroutine) instruction is executed. The range of information stored or saved within the 64 bits for each JSR consists of:

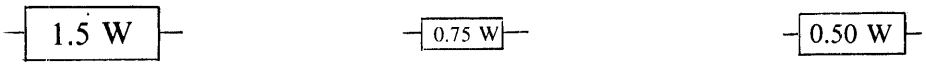
1. Display Program Counter plus one (DPC +1) – This represents the return address in the display file (i.e., the address returned to after executing the subroutine).

2. Name – This is a 4-, 8-, or 11-bit descriptor used to identify the subpicture (image) generated by the subroutine.

3. All status/parameter data such as line type, intensity, scale factor, etc.

Subroutine nesting is a useful tool when it is desired to produce repetitive images with different designators as part of a design problem.

For example, assume a printed circuit board application where three different size resistors with three different wattage ratings are involved. Each resistor appears rectangular in form; however the size differs as does the message defining the wattage rating. In such a case, the three subpictures shown below may be produced at numerous different locations on the CRT.



Assume further that eight 1.5 W, twelve 0.75 W, and twenty 0.50 W resistors are incorporated on the printed circuit board. When the display file instruction sequence points to the location of a given resistor, two subroutines (the second nested) can be used to generate the resistor image and wattage designator. The first subroutine draws the rectangular form at the scale factor appropriate to its physical size. After this, a jump is made to a second subroutine which enters the character mode and generates the applicable wattage label. The coding sequence to display a single resistor (say 1.5 W) and related wattage rating would be as shown in Figure 3-51.

If a group of resistors could be categorized further (by resistance tolerance values of 1, 5, and 10 percent) and it was desired to display the distinguishing values, then another subroutine could be nested to generate the additional part of the descriptor.

Figure 3-52 shows a coding sequence that can be used to test the storage and POPping capabilities of all eight stack memory levels. The coding indicates eight nested subroutines that are used to generate the messages LEVEL 0 through LEVEL 7 on eight successive lines of the CRT. The POP not restore instruction at the end of the LEVEL 7 subroutine returns to the POP instruction at the end of the LEVEL 6 subroutines, etc., so that all eight stack memory levels are POPped in sequence.

3.9.8 Stack Status Byte Map

The stack memory stores 64 bits of information every time a display jump to subroutine (JSR) instruction is executed. This information is stored in one of eight stack levels. That is, the first 64 bit word is stored on level 0 when the first JSR instruction is executed. If no intervening POP instruction occurs, the second 64-bit word is stored on level 1 on execution of the second JSR etc. The complete 64-bit word is divided into four 16-bit bytes, each of which is addressable and accessible via the Stack Data register. (The word level is also addressable, see Figure 3-39.)

DISPLAY FILE

```
                POINT           ;Point to a screen coordinate
                 177            ;X = 177₈
                1077            ;Y = 1077₈
                NAME            ;Load unique label for resistor type in Name reg.
                JSR             ;Execute jump to subroutine
                RECT 1          ;Address of subroutine
                N.I.            ;Next instruction

RECT 1          SHORT VECTOR    ;Enter vector mode
                .               ;Generate rectangular form of resistor
                .
                .
                .
                NAME            ;Load unique name for 1.5 W label in Name register
                JSR             ;Execute jump to subroutine
                WATT 1.5        ;Address of subroutine
                POP             ;Return to calling routine

WATT 1.5        CHAR*           ;Enter character mode
                .               ;Generate character message for 1.5 W label
                .
                .
                .
                POP RESTORE*    ;Return to calling routine
```

*It is assumed that the intensity level is changed to prevent blooming when generating character message (1.5 W). Therefore, POP restore instruction must be used to return to former intensity level when "POPping" stack.

Figure 3-51 Typical Coding Sequence for Nested Subroutines

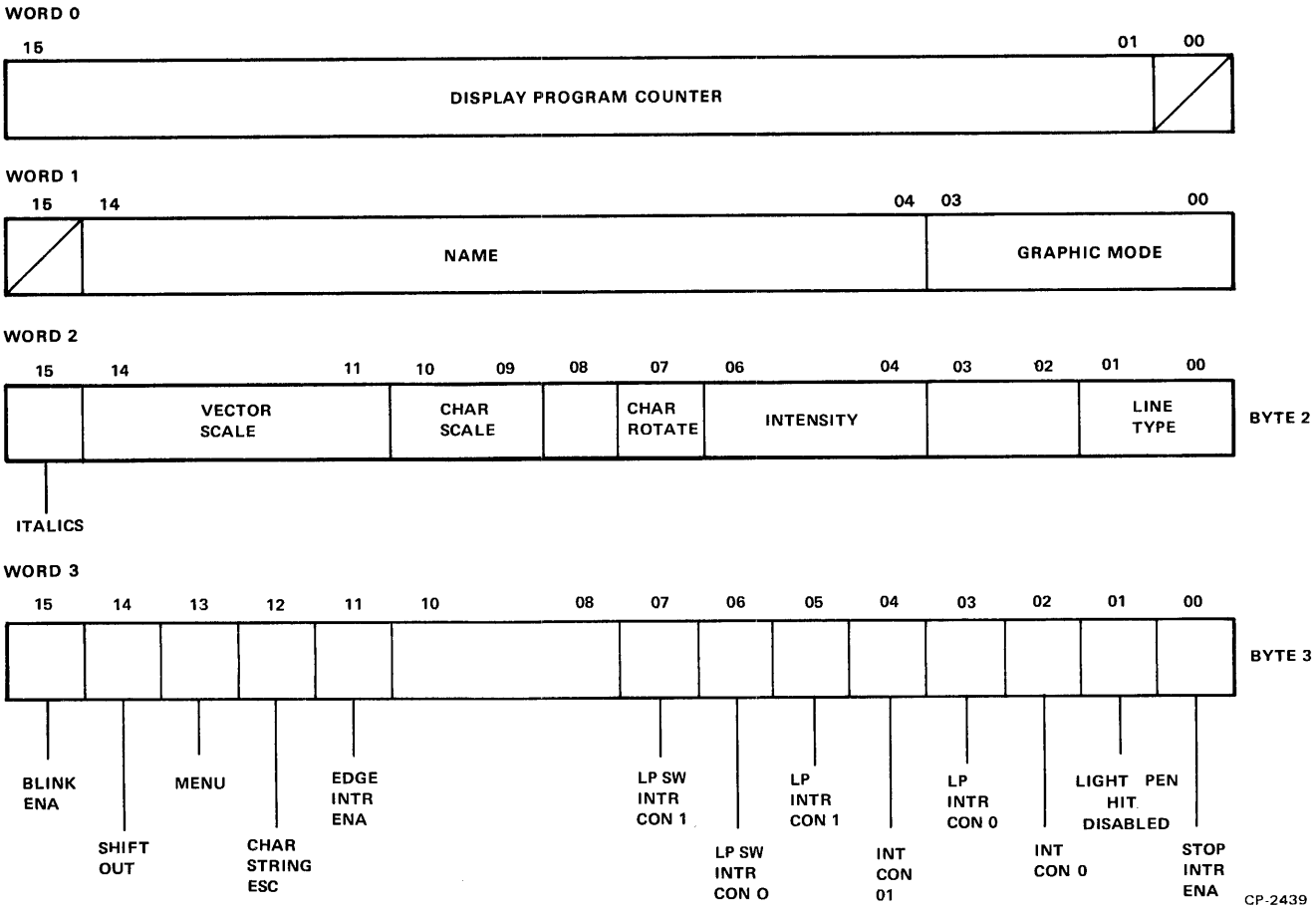
```
                DISPLAY FILE
LEVEL TEST      POINT           ;Point to 2 screen coordinate
                 400            ;X = 400₈
                1600            ;Y = 1600₈
                CHAR            ;Select character mode
                •               ;Generates text defining type or name of test
                •
                •
                •

                DNAME 0         ;Load No. 0 in Name register
                JSR             ;Execute jump to subroutine
                LEVEL 0         ;Address of subroutine
                N.I.            ;Next instruction
                •
                •
                •
                •
LEVEL 0         CHAR            ;Select character mode
                •               ;Generates control and text to write "LEVEL 0"
                •
                •
                •
                DNAME 1         ;Load No. 1 in Name register
                JSR             ;Execute jump to subroutine
                LEVEL 1         ;Address of subroutine
                POP             ;Return to calling subroutine
LEVEL 1         CHAR            ;Select character mode
                •               ;Generates control* and text to write "LEVEL 1"
                •
                •
                •
                DNAME 2         ;Load No. 2 in Name register
                JSR             ;Execute jump to subroutine
                LEVEL 2         ;Address of subroutine
                POP             ;Return to calling routine
                •
                •
LEVEL 7         CHAR            ;Select character mode
                •               ;Generates control* and text to write "LEVEL 7"
                •
                •
                •
                POP             ;Return to calling routine
```

*Line feed or carriage return control characters must be executed to write messages on successive lines.

Figure 3-52 Example of Coding Sequence for Use in Testing Stack Memory Levels

The following formats indicate the byte address and content for all four 16-bit bytes.



3.9.9 Display File Searching Through use of Name Matching Techniques

The Name and Associative Name registers are incorporated in the display processor unit to provide a convenient means of searching the display file. The search is carried out in a manner analogous to a lookup table in central processor based software. An 11-bit name value (conveyed by name instruction) is introduced in the display file just before each graphic entity, string of graphic entities, or subroutine used to generate a subpicture. Though 11 bits in length, the name or label can be divided into three sub fields which allows the display file search to be made on the basis of all 11 bits, the 8 high order bits, or the four high order bits. This feature enables grouping of graphic entities into subcatagories.

To understand the technique of name match searching, the reader must make a number of assumptions about the way graphic data is structured for a particular application. The application example pursued by this explanation expands on the sample situation described in Paragraph 3.9.7 and illustrated in Figure 3-51. This example cited a printed circuit application where three different size resistors (1.5 W, 0.75 W, and 0.50 W) are used. Quantities by resistor type are:

1. Eight 1.5 W
2. Twelve 0.75 W
3. Twenty 0.50 W

Each of these groups can be assigned names in accordance with the subcategory bit patterns shown on Figure 3-53. Now, assume that half of the last group of resistors (twenty 0.50 W) have a resistance tolerance of ±5% and the other half ±1%. Under such conditions, the name values assigned to each set of resistors would be as shown in Figure 3-54.

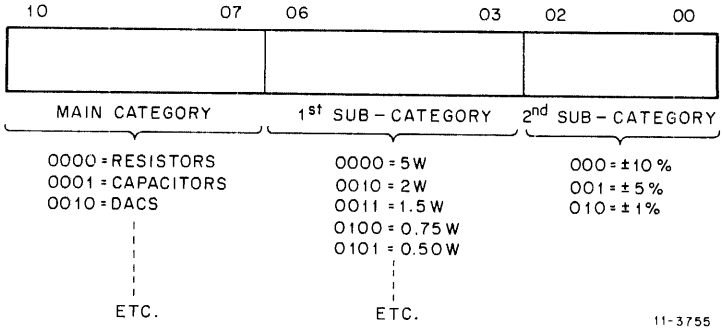


Figure 3-53 Example of Name Value Sub-Categories

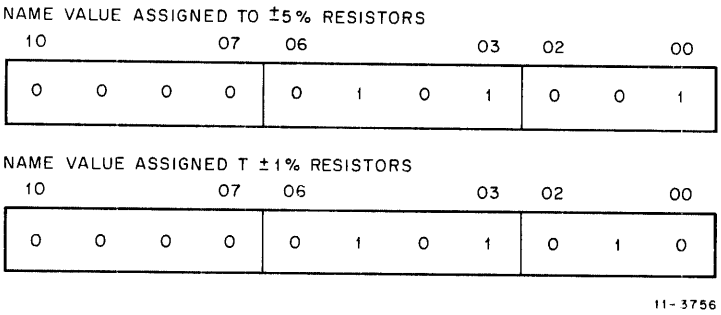


Figure 3-54 Name Bit Patterns Used to Distinguish Between Two Sub-Categories of Resistors

Now if it is further assumed that the circuit designer wishes to make all 0.50 W resistors ±1% to improve circuit performance, how can he/she readily search the display file to locate the resistors he/she wishes to change? The answer is by loading the Associative Name register with the same bit pattern (name) assigned to the ±5% (0.50 W) resistors. That is, the Associative Name register is loaded with 00000101001_2. When this is done, a name match (which causes an interrupt) occurs every time the subroutine that generates the applicable resistor is called.

If the designer wished to search the display file for all twenty 0.50 W resistors (i.e., eliminating the three bit subcategory), then a search of the high order eight bits is sufficient. That is, loading the binary value 00000101_2 into the eight high order bits, and commanding an 8-bit search code (Figure 3-36) will generate an interrupt upon detecting every 0.50 W resistor. Further, a search of all resistor-generating subroutines in the display file can be made by loading 0000_2 into the four high order bits of the Associative Name register.

The technique for loading the Associative Name register with the desired label (from keyboard or menu area using light pen) is a function that must be defined by the application software. This is also true for the particular action taken in response to each name match interrupt.

3.10 INTERRUPT HANDLING

3.10.1 General

The display processor unit is capable of generating numerous different interrupts to the host processor. All interrupts can be grouped within four general categories: operator interrupts, processing violation interrupts, display file manipulation interrupts, and external stop interrupts. An overview of the types of interrupts comprising each of these categories follows.

1. Operator or Designer Interrupts – These include interrupts generated by the console operator when using the light pen at the VR48 Display Console. It also includes edge interrupts (traversing of working surface edge by a graphic entity) occurring when changing scale factor since the operator may wish to know when any portion of a complex display pattern falls outside the "window."
2. Processing Violation – Interrupts in this category occur when the display processor detects an invalid operation. These include a timeout (occurring when the host processor fails to respond to a nonprocessor request (NPR), as well as Stack Memory overflow and underflow conditions.
3. Display File Manipulation – This includes interrupts occurring when the display processor unit recognizes an internal stop command or when a name match is detected in searching the display file.
4. External Stop – This interrupt is generated when the host processor executes a command to stop the display processor unit from further processing.

All interrupts fall within one of six priority levels internal to the VT48 Display Processor. Table 3-1 provides a breakdown of all interrupts, their meanings, priority levels, and read status locations.

If a higher order interrupt occurs, it overrides a lower priority interrupt until the higher priority interrupt is serviced. Thus, a name match interrupt at level 6 is overridden by a light pen hit at level 3. When the light pen interrupt is serviced, the name match flag is re-asserted, etc.

Table 3-1 VT48 Priority Interrupts

Interrupt and Purpose	Vector Address	Internal Priority Level	Read Status Location
Internal stop. Generated when load status A instruction is executed to stop the display processor from processing the display file.	320	4	Bit 15 of status address 772002.
External stop. Generated when PDP-11 writes a 1 into bit 7 of register 772012 to stop further processing of the display file.	320	5	Bit 7 of status address 772012.
Light pen switch on. Generated when console operator activates light pen switch at VR48 Display Monitor.	324	2	Bit 13 (console 00) of status address 772022/bit 07 (console 01) of status address 772022.
Light pen switch off. Generated when operator releases light pen switch at the VR48 Display Monitor.	324	2	Bit 12 (console 00) of status address 772022/bit 6 (console 01) of status address 772022.

Table 3-1 VT48 Priority Interrupts (Cont)

Interrupt and Purpose	Vector Address	Internal Priority Level	Read Status Location
Light pen hit flag. Generated when light pen detects a hit (light) on the VR48 Display Monitor.	324	3	Bit 7 of status address 772022. Also bit 14 (console 00) of status address 772022 and bit 8 (console 01) of same address in two console systems.
Edge transition. Generated when one or more graphic entities exceeds the confines of 12-inch × 12-inch working surface.	324	1	Bit 2 of status location 772002.
Shift out. Generated when shift out control character is detected. This interrupt is incorporated for compatibility with the GT40 system only.	330	6	Bit 6 of status location 772002.
Stack memory overflow. Indicates that the display processor has exceeded eight sequential JSR instructions (i.e., with no intervening POP instructions).	330	6	Bit 13 of status location 772012.
Stack memory underflow. Indicates that the number of POP instructions has exceeded the number of prior executed JSR instructions. For example, four sequential POP instructions follow three sequential JSR instructions.	330	6	Bit 12 of status location 772012.
Time out. Indicates that PDP-11 has failed to honor a display processor NPR request within the required time frame.	330	6	Bit 11 of status location 772012.
Name match. Indicates that a match has occurred between an 11 bit value loaded into an Associative Name register (address 772020) and 11 bit value(s) (name tags) within the display file.	334	6	Bit 15 of status location 772024.

3.10.2 Reinitiating Display File Processing Following Interrupt Handling Routines

When it is desired to reinitiate display file sequencing following the processing of an interrupt, it is not necessary to return to the start of the display file. A resume command is provided that restarts processing at the display file address where the interrupt occurred. The resume command consists of writing the display program counter (DPC) with a "1" in the bit 00 position. See the description of the display program counter.

3.11 SPECIAL FEATURES

3.11.1 Dual Console Operation

Operation of two different consoles from two different display files is a built-in design feature of the VT48. Switch-over from one display console to another is achieved through use of the load scope selection instruction. Figure 3-55 presents an example of how switch-over between display files for two different consoles might be achieved. This example assumes a dedicated PDP-11 where core locations 020000_8 through 037777_8 are assigned to the console 0 display file and locations 040000_8 through 057777_8 are assigned to a console 1 display file. The console 0 display file begins by executing a load scope selection instruction. This selects console 0 for the image generation process and also unblanks the scope. Once initiated, sequencing continues through the console 0 display file (and any called subroutines) until the jump instruction at the end of the display file is encountered.

In single console systems, the jump instruction returns processing to the start of the display file. However, with a second console and related display file involved, the jump instruction branches to the start of the second display file. The first instruction of the console 1 display file is also a load scope selection type. It is used to select console 1 for the image generation process and unblank the VR48 Display Monitor. That is, all vectors and characters processed through the VT48 affect only console 1.

At the end of the console 1 display file, another jump instruction is executed. In this case, the branch address is the start of the console 0 display file so that sequencing through this display file is reinitiated.

3.11.2 Use of Relocate Register in Systems Exceeding 32K of Memory

The display processor unit contains a writable Relocate register for use in systems exceeding 32K of core memory. The Relocate register is 12 bits in length. However, its outputs are so mated with the display program counter that they straddle bit positions 6 through 17 of the DPC (Figure 3-56). Hence, any change in bit positions 10 and 11 of the Relocate register on loading (or carry developed due to adding lower order bits to bits 6 through 15 of the DPC) effects a change in bit positions 16 and 17 of the DPC. Consequently, relocating the display file address anywhere in 128K of memory can be achieved.

3.11.3 Character String Escape

The character string escape feature eliminates the need for inserting a POP Restore instruction at the end of a character message subroutine. This capability might be used in an architectural application where EXIT signs for each doorway are generated by a subroutine.

If the Character Terminate register is loaded with the character code for the letter T, then the stack memory is automatically POPped upon completion of the EXIT sign subroutine. POPping occurs because the VT48 detects a matchup between the contents of the Character Terminate register and the letter T being processed within the display file.

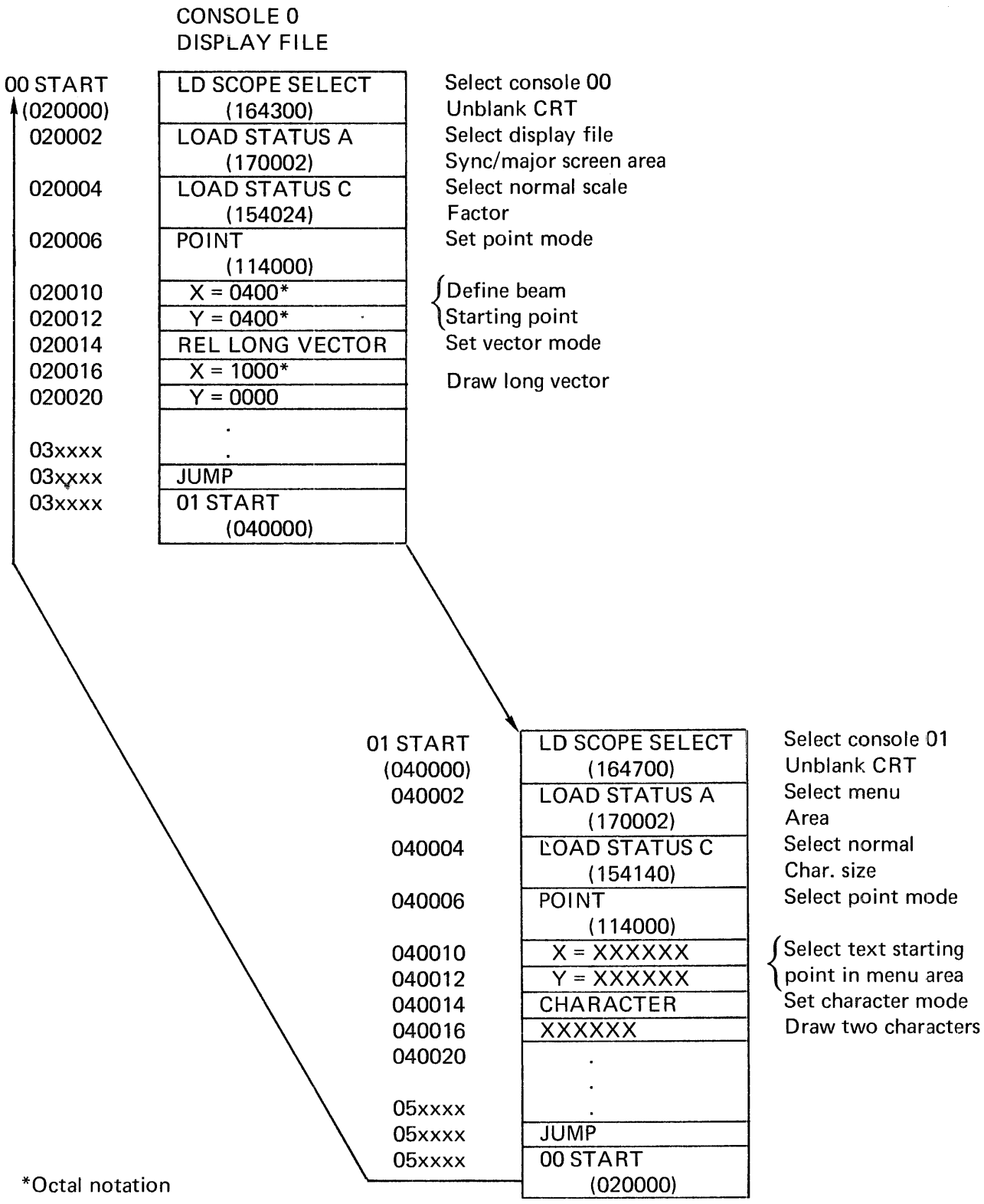


Figure 3-55 Coding Used to Switch Between Display Files When Driving Two Display Consoles

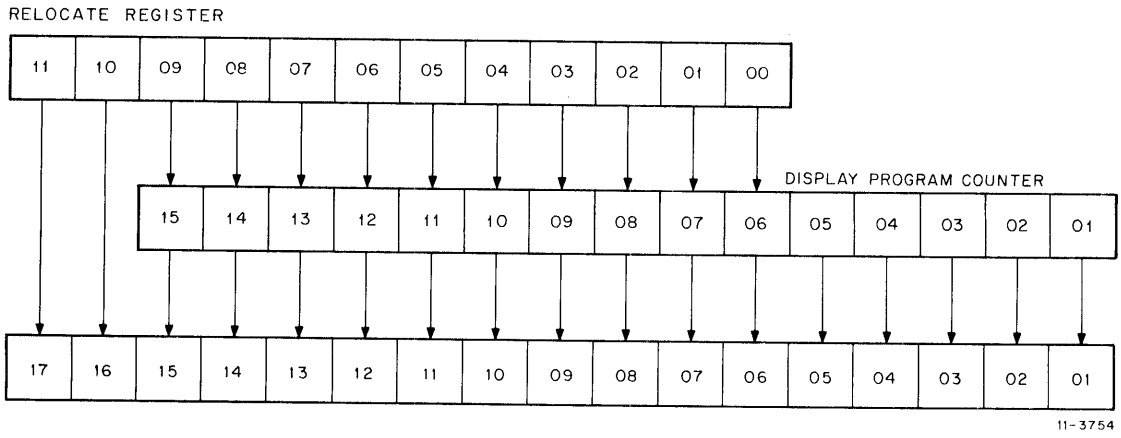


Figure 3-56 Relationship of Relocate Register to Display Program Counter

CHAPTER 4
THEORY OF OPERATION

4.1 VT48 OVERALL BLOCK DIAGRAM

4.1.1 General

An overall block diagram of the VT48 Graphics Display Processor is given in Figure 4-1. This drawing defines the principal functional areas within the VT48, and shows the routing of data by instruction type. The VT48 can process up to four sequential words at any given time, and the major control signals used to sequence data words through the VT48 are also shown in this figure.

4.1.2 VT48 Startup Processing

Startup of the VT48 occurs when the PDP-11 addresses the VT48 and supplies the display file starting address over the Unibus data lines for entry into the display program counter. Addressing the VT48 is detected by the Unibus Control logic, which in turn initiates two internal functions:

1. Loads the Buffered Data Bit (BDB) register with the 16-bit display file starting address. This is done by assertion of the UC2 DATA STR H signal which is asserted for every word transferred to the VT48.

2. Causes the display instruction control logic to assert signal DIC6 LD PC H for loading the display file starting address into the display program counter. (When the Unibus Control logic asserts UC1 ADDR ENA L, the display instruction control logic responds by asserting signal DIC6 LD PC H.)

At this time, the display file starting address is in the display program counter and presented to the PDP-11 via the Unibus address transceivers.

Coincident with loading the display program counter, the display instruction control logic initiates a nonprocessor request (NPR) cycle. This is done by asserting signal DIC3 NPR 1 H, which in turn forces issuance of signal BUS NPR L from the Unibus control. The first word in the display file is now fetched and entered into the BDB buffer.

4.1.3 Processing Sequences per Instruction Category

In the normal course of things, load status instructions are executed at the start of the display file. This is done to set up the status/parameter values attendant to the first string of graphic entities to be processed and displayed.

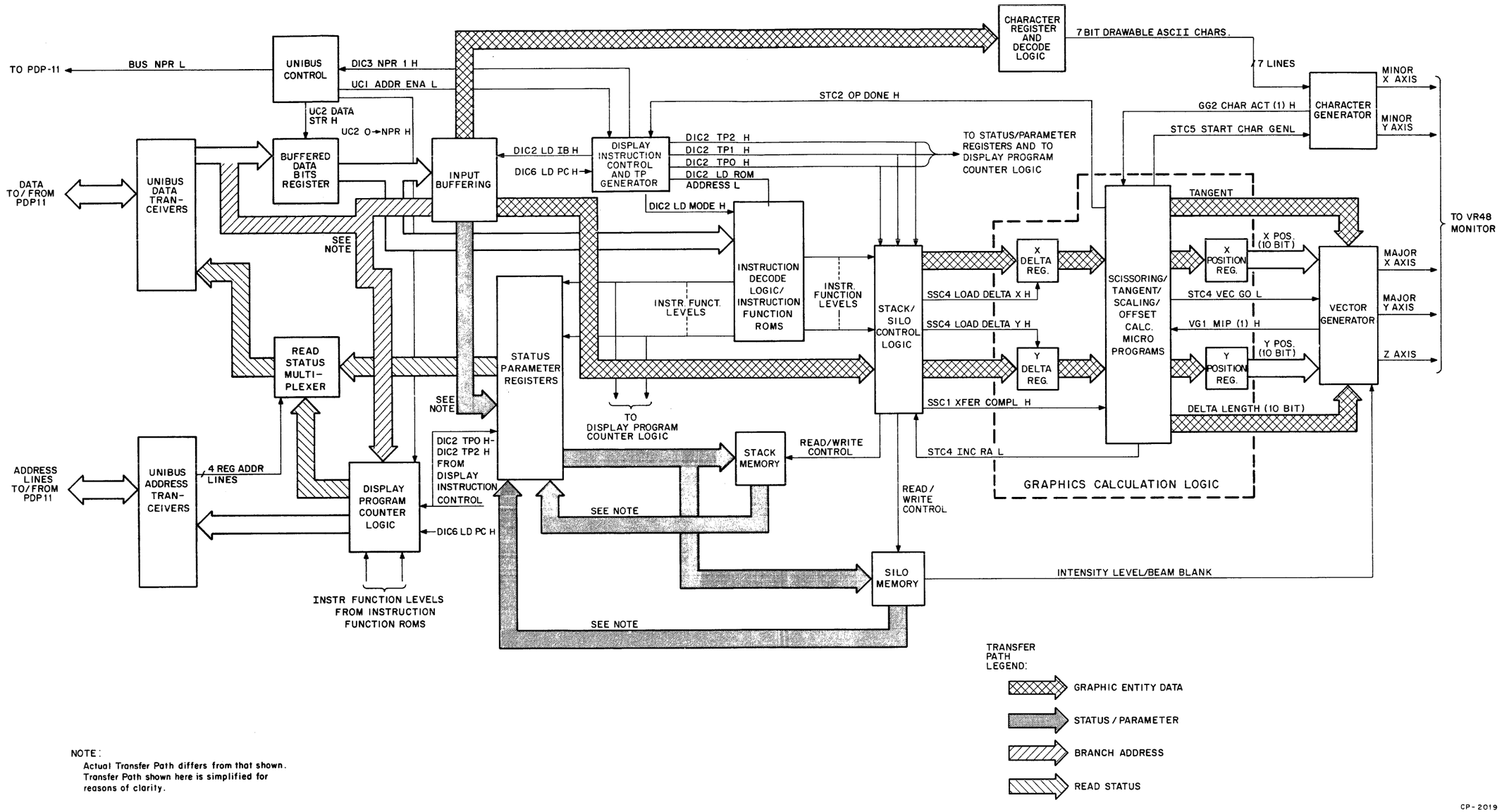


Figure 4-1 VT48 Overall Block Diagram

The subsequent paragraphs describe VT48 processing sequences by instruction type and indicate the major circuit areas involved in each type of processing. The sequence of presentation is:

1. Load status instructions (i.e., status and parameters)
2. Set graphic mode instruction
3. Graphic entity instructions
4. Branch instructions

NOTE

The load status, set graphic mode, and branch instructions all fall in the general category of control instructions. That is, bit 15 is set to a 1. The above sequence has been chosen, however, based on the typical order of instruction use within the display file.

It should also be understood at this time that the display instruction control logic initiates an NPR cycle (to fetch the next word in the display file) at the end of each instruction. This is true regardless of instruction type.

4.1.3.1 Load Status Instruction – In addition to the Unibus Control logic, processing of load status instruction involves the following VT48 circuit areas:

1. BDB (Buffered Data Bits) register
2. Input buffering
3. Display instruction control and time pulse generator
4. Instruction decode logic
5. Status/parameter registers
6. Display program counter

The first load status word is entered into the BDB register by the UC2 DATA STR H signal from the Unibus control logic. At this time, the VT48 initiates the following four control operations:

1. Asserts signal UC2 0→ NPR H to increment the display program counter by 2. This is done in anticipation of the next NPR cycle that is initiated at the end of the current word processing sequence.

2. Asserts signal DIC2 LD IB H to enter the load status word into the input buffer.

3. Asserts signal DIC2 LD MODE H signal. This loads the OP code bits from the BDB buffer into a register in the instruction decode logic. This latter circuitry now converts the OP code into an 8-bit ROM address which is later used to select a location in the instruction function ROMs. The location selected will in turn assert all enabling (function) levels to implement processing of the particular load status instruction being executed.

4. Initiates the time pulse generator timing cycle. This cycle consists of four sequential timing pulses that are generated in the following sequence:

   a. DIC2 LD ROM ADDRESS L
   b. DIC2 TP0 H
   c. DIC2 TP1 H
   d. DIC2 TP2 H

Timing signal DIC2 LD ROM ADDRESS L loads the 8-bit ROM address into an address register within the instruction decode logic. This being done, the enabling function levels (for the particular load status instruction being executed) are now asserted. The instruction function levels are applied to the status/parameter registers and they remain asserted until either of the following occurs:

1. A new 8-bit ROM address (for a subsequent instruction) is loaded into the address register in the instruction decode logic.

2. The ROM address is incremented as part of the instruction processing sequence.

The pattern of instruction function levels applied to the status/parameter registers varies depending on the instruction being executed. That is, the pattern of enabling levels applied for a load status C instruction differs from that used during execution of a load status A instruction.

With the appropriate status/parameter registers currently enabled, the status/parameter values contained in the input buffer can now be entered. The second timing pulse from the time pulse generator (DIC2 TP0 H) is used to strobe the status/parameter values into the enabled registers.

The remaining two timing pulses are used to initiate the next NPR cycle and to shut down the time pulse generator itself.

When the next NPR cycle is complete (meaning that the BDB buffer is now loaded with the second word), processing of the next instruction begins. That is, all processing activities described for the first load status word are repeated.

After all status/parameter conditions have been set up, the program will normally execute the set graphic mode instruction to define the type of graphic entity instructions to follow.

4.1.3.2 Set Graphic Mode Instruction – In some respects processing of the set graphic mode instruction is similar to processing load status instructions. All of the same circuit groups are involved and certain parameters are conveyed (i.e., intensity level, light pen interrupt enable, blink enable, and line type). These status/parameter bits are entered into the respective status/parameter registers in exactly the same way as those parameters conveyed by the load status instructions. The distinguishing feature of the set graphic mode instruction is that it readies the instruction decode logic for the graphic entity instructions to follow. Bits 11 through 14 of the OP code field are entered into a Graphic Data Mode register in the instruction decode logic. However, as long as bit 15 is a 1, the Graphic Data Mode register has no affect on the 8-bit ROM address used to select the enabling levels from the instruction function ROMs. Enabling of the outputs from the Graphic Data Mode register occurs when the graphic entity instruction (following the set graphic mode instruction) is processed. By definition, all control instructions have bit 15 set to a 1 while all graphic entity instructions have this bit set to 0. When bit 15 switches to 0 following execution of the set graphic mode instruction, it enables the output of the Graphic Data Mode register and initiates processing of the graphic entity instruction.

4.1.3.3 Graphic Entity Instructions – Processing the ten graphic entity instructions involves all blocks of Figure 1 except stack memory and the read status multiplexer. As many as four sequential graphic entity words can be in process within the VT48 at a given time. For this reason, processing of graphic entity instructions must be looked at from the following standpoints:

1. Processing from BDB register through the stack/silo control logic.

2. Processing by the graphics calculation logic.

3. Processing by the vector generator/character generator. Either one or the other of these two circuit areas is involved in drawing all graphic entities. They are never operational simultaneously, however.

A graphic entity can be in any one of the preceding phases of processing at any given time. In addition, the BDB register may be loaded with a fourth graphic entity (or part thereof in cases of multi-word instructions) before the preceding three are sequenced along. Consequently, the VT48 may be processing up to four graphic instructions at any given time.

This feature of handling sequential words simultaneously necessitates use of the silo memory to store the status/parameter data attendant to each successive graphic word. Silo memory operation is discussed more fully later in this paragraph.

4.1.3.4 Processing from BDB Register Through the Stack/Silo Control Logic – Except for characters, all graphic entity instructions convey X/Y data in one or two words. The task of the front end of the VT48 in processing graphic entity instructions can be summarized as follows:

1. Decode the instruction type to determine whether the X/Y data is conveyed in one or two words.

2. Load the Delta X/Delta Y registers in the graphics calculation logic consistent with the manner in which the data is conveyed. That is, in a short vector instruction, the X and Y data are loaded simultaneously since they are conveyed in a single word. For long vectors, the Delta X register is loaded during the first word while the Delta Y register is loaded during the second.

3. Write the status/parameter data (entered prior to the set graphic mode instruction) into silo memory. Entry of such data into silo memory serves two purposes:

   a. Provides a means of storing display surface related parameters so that they are available to control beam/light pen status at such time as the graphic entity is being drawn. Beam control parameters are intensity, blink, and line type which must be available to the vector generator throughout the graphic drawing period.

      Light pen status consists of the enabled/disabled condition, allowing or inhibiting light pen strike interrupts on the graphic entity being drawn.

   b. Provides a means of accessing the status/parameter data (for the graphic entity being drawn) on occurrence of display surface related interrupts, i.e., light pen strike, light pen switch, or display surface edge transition. Such access is necessary to re-enter the status/parameter data into the appropriate registers thereby making it available to PDP-11 software via the read status multiplexer.

      NOTE
      Though not shown on Figure 4-1, the silo memory also stores and makes available display program counter address data.

4. Informs the graphics calculation logic that it has now been loaded with delta X/Y data and can begin the calculations pertinent to the type of graphic entity being processed. processed.

The graphic word is strobed from the BDB register to the input buffer by DIC2 LD IB H. The content of bit 15 (0) is loaded from the BDB buffer into the display instruction decode logic by DIC2 LD MODE H. This enables the outputs of the Graphic Data Mode register to address the instruction function ROMs as described in Paragraph 4.1.3.2. At this time then, the instruction function levels applicable to the type of graphic entity being processed are applied to data steering logic on the Stack/Silo control module. Also, the X/Y coordinate data (X coordinate only if it is a two word

instruction) is applied from the input buffer to the stack/silo control logic. As such, the X/Y coordinate data is now ready for transfer to the graphics calculation logic.

The strobe signals used to transfer the delta X/Y data to the graphics calculation logic are SSC4 LOAD DELTA X H and SSC4 LOAD DELTA Y H, respectively. Generation of these signals is under control of the instruction function levels supplied from the instruction function ROMs. When the word in the input buffer contains both delta X and delta Y values, the stack/silo control logic issues both strobe signals simultaneously. If the delta X/delta Y values are conveyed in two sequential words (e.g., point, long vector, etc.), then the delta X strobe only is issued during the first word. That is, the enabling levels supplied from the instruction function ROMs allow generation of only a single strobe. When the second word is fetched and contained in the input buffer, the enabling level pattern (supplied from the instruction function ROMs) changes and the delta Y strobe is generated.

NOTE

The stack/silo control logic contains multiplexers that select character scale and graph increment data as well as delta X/Y data from the input buffer. Multiplexer input selection is also under control of the enabling levels supplied from the instruction function ROMs.

After the delta X/Y values have been strobed into the graphics calculation logic, the stack/silo control logic must write the attendant status/parameter values into silo memory. The silo memory has four addressable locations for writing/reading status and parameter data. Writing is under control of the stack/silo control logic and each write access occurs just prior to informing the graphics calculation data that transfer of the delta X/Y values is complete. Reading silo memory is basically under control of the graphics calculation logic and each read access is made just prior to commanding the vector generator/character generator to begin drawing the graphic entity. Due to the VT48 word sequencing arrangement, the silo memory write address always leads the read address.

Immediately after writing the status/parameter data into silo memory, the stack/silo control logic informs the graphics calculation logic that it has received the delta X/delta Y data, and therefore can begin its calculation routines. Assertion of signal SSC1 XFER COMPL H is made to indicate that transfer of the delta X/delta Y data is complete.

4.1.3.5 Processing by the Graphics Calculation Logic – The graphics calculation logic consists of a microprocessor and an arithmetic unit that performs all calculations necessary to display graphic entities on the VR48 Display Monitor. Operation of the microprocessor is such that it sequences through a different routine (microprogram) for each of the different types of graphic entity instructions. The types of calculations performed by this logic are as follows:

1. Scissoring – This is performed on all vector instructions. The arithmetic unit calculates whether all, any part, or none of the vector falls within the VR48 working surface area. (Note: This is the 12-inch × 12-inch area of the VR48 Display Monitor.) If any part of the vector falls within the display surface, the arithmetic unit determines where the vector intersects with the display surface edge and then treats the viewable segment as a drawable vector. If all of the vector is within the display surface, then the calculations described below are carried out.

2. Tangent Calculation – The vector generator requires two sets of inputs for every drawable vector (i.e., vector or segment thereof falling within the display frame). These are:

   a. Delta Length – This is the greater of the delta X/Y values conveyed by the instruction.

b. Tangent – This is the ratio of minor axis (delta length) to major axis and it is calculated by the graphics calculation arithmetic unit.

Given the preceding two values plus sign bits, the vector generator can draw vectors anywhere within the display working surface.

3. Scaling – The arithmetic unit multiplies all delta X/Y values by the currently assigned scale factor. This is done prior to scissoring operations.

4. Offset Calculations – The VT48 incorporates offset registers which can be used for "windowing" within the virtual display file. The offset value adds to (or subtracts from) all vector/character start and end points. By incrementing (decrementing) the Offset register contents, say every ten display frames, the displayed graphic entities appear to move. The arithmetic unit calculates new start/end points for all graphic entities every time the Offset register is updated.

5. Light Pen Calculations – Whenever a light pen strike interrupt on a vector is being serviced, the arithmetic unit must calculate the exact point (along the vector length) where the light pen strike occurred.

When processing a graphic entity, the particular microprogram entered is a function of the graphic data mode code supplied from the instruction decode logic. (This is not shown on Figure 4-1, but it is loaded into the graphics calculation logic at the same time the SSC1 XFER COMPL H signal is issued.)

One initial action in all microprogram routines is to take the data from the delta X/delta Y registers and load it into internal registers within the arithmetic unit. This frees the front end of the VT48 (display instruction control and stack/silo control) to begin processing the next instruction. The graphics calculation logic informs the display instruction control that the delta X/Y registers are free and to begin processing the next instruction by asserting signal STC2 OP DONE H.

Upon completing the calculations necessary for the type of graphic entity being displayed, the graphics calculation logic asserts one of two signals:

1. STC4 VEC GO L – This signal is asserted to command the vector generator to start drawing the vector. It is asserted by those microprograms involved in vector calculation processes after the tangent and delta length values have been loaded into the vector generator.

2. STC5 START CHAR GEN L – This signal is asserted to command the character generator to draw a character. It is asserted after the graphics calculation logic has positioned the beam at the character starting point.

NOTE

Just prior to asserting either of the above two signals, the graphics calculation logic updates the silo memory address. This is done so that the beam control parameters (intensity, blink, etc.) attendant to the graphic entity about to be drawn are presented to the vector generator. This action also makes all status/parameter data available for re-entry into the status/parameter registers in the event of a light pen interrupt. The microprogram asserts signal STC4 INC RA L to update the silo memory address.

During the period that the vector/character generators are busy drawing their related graphic entities, each asserts a signal back to the graphics calculation logic. These signals and their purposes are:

1. VG1 M1P (1) H – As long as it is asserted, this signal informs the graphics calculation logic that the vector is being drawn (move in process). Consequently, if the graphics calculation logic has completed calculations for the next vector, it must wait until this signal is negated before it can load tangent and delta length and then assert STC4 VEC GO L.

2. GG2 CHAR ACT (1) H – This is asserted throughout the period that the character is being drawn. Again, it informs the graphics calculation logic to wait until the current character is finished before issuing another draw command.

At the conclusion of each vector draw period, the graphics calculation logic sends the contents of the X/Y position registers to the vector generator. Here they are loaded into X/Y position D/A converters so that the beam is held at the addressed vector end point. (That is, provided that the addressed end point is within the display surface area.) When drawing continuous connected vectors, this allows the addressed end point of one vector to be the starting point of the next.

When processing points that fall within the display frame, the graphics calculation logic simply updates the contents of the X/Y position registers for every point plotted.

4.1.3.6 Processing by the Vector Generator/Character Generator – The vector and character generators form the vectors and characters displayed on the VR48 Display Monitor. The vector generator is discussed first.

To draw a vector, the vector generator develops a pair of voltage ramps that are applied to the X/Y deflection amplifiers in the VR48 Display Monitor. For 45-degree vectors, the ramp voltages developed are of equal magnitude. For zero-degree or 90-degree lines, only a single ramp is developed. For all other vectors the larger ramp is supplied to the major axis deflection amplifier while the smaller ramp is applied to the minor axis deflection amplifier.

The major axis is determined by the larger delta X/delta Y value supplied to the graphics calculation logic. The latter logic supplies the larger delta value to the vector generator as the 10-bit delta length. As stated earlier, the graphics calculation logic also supplies a 10-bit tangent value (ratio of minor to major axis).

The vector generator uses the 10-bit delta length to develop the primary ramp voltage. The magnitude of this ramp is directly proportional to the 10-bit delta length value. It is this ramp that is fed to the major axis deflction amplifier in the VR48.

The minor axis ramp is developed by attenuating the major ramp by an amount equal to the tangent value. That is, if the tangent value is half the delta length, then the major ramp is attenuated 50 percent thereby producing a minor axis ramp equal to one-half the major axis ramp. The net result is a vector drawn at a 22.5-degree angle.

The vector generator also contains two D/A converter circuits – one for X and one for Y. These circuits are updated with the outputs of the X/Y position registers (in the graphics calculation logic) at the close of the vector drawing process. In this way, the beam is held at the addressed vector end point. The D/A converters are also used during point instruction processing.

The character generator is activated only when processing characters. Two 7-bit ASCII characters (per each instruction word) are taken from the input buffering and entered into the Character register. Here, the character decode logic distinguishes between control characters (e.g., carriage return) and drawable characters. The character generator is only enabled for drawable characters.

The character generator itself consists of a microprocessor that sequences through a series of short strokes to form each character. The X/Y voltages for each stroke are sent to the VR48 over the minor X/Y axis lines.

The character generator uses the 7-bit ASCII code (supplied from the Character register) to address a starting location in an internal ROM. This location contains the first stroke value in the series of strokes needed to form the character. Following this, an internal clock circuit selects each successive location in the ROM so that all strokes forming the character are executed.

4.1.3.7 Branch Instructions – Execution of branch instructions involves the following circuit groups: display program counter, stack memory, stack/silo control logic, instruction decode logic, display instruction control, input buffering, and BDB buffer. The branch instructions fall into two categories, each requiring a different processing procedure. These categories are:

1. JUMP/JSR Instructions – Upon execution of the JUMP/JSR instructions, all current status/parameter data (contained in the status/parameter registers) and the current display file address in the display program counter (DPC) are written into stack memory. Following this, the new display file address conveyed by the JUMP/JSR instruction is entered into the DPC and branching occurs.

2. POP Restore/POP Not Restore Instructions – Execution of the first of these instructions requires that the stack memory be POPped one location (address) and the status/parameter data be restored to the status/parameter registers. Also, the DPC address (stored during the preceding JSR instruction), is re-entered into the display program counter. The POP not restore instruction POPs the stack memory by one location, but does not re-enter status/parameter data and the DPC address into the related registers.

Processing of all branch instructions through the display instruction control and instruction decode logic is nearly identical. The exception lies in the fact that the D1C2 LD IB H signal is asserted only for JUMP relative and JSR relative instructions. Both instructions convey the relative address within the nine low order bits of the word. This must be stored while the stack memory is being written so that later the relative address can be written into the DPC as the branch address.

In the case of the JUMP absolute and JSR absolute instruction, the branch address is conveyed in the second word of the instruction. Consequently, it is entered into the DPC directly from the Unibus transceivers.

Loading of the DPC branch address is effected by the display instruction control, which asserts signal D1C2 LD PC H at the proper time. Assertion always occurs after the stack memory has been written for a JSR instruction.

As with other type instructions, the instruction function ROMs provide the necessary enabling levels to implement branch instruction processing. The enabling levels applied to the stack/silo control logic allow for PUSHing/POPping the stack at the proper times. The function levels applied to the display program counter control logic allow for selecting the proper source of the branch address; i.e., input buffer for JUMP/JSR relative instruction and Unibus transceivers for JUMP/JSR absolute instructions.

The stack memory itself is eight locations deep allowing for the nesting of this many subroutines.

4.1.4 Read Status Multiplexer and Register Addressing

The content of the status/parameter registers and the display program counter are applied to a read status multiplexer for sampling by the PDP-11 program. The read status multiplexer has 16 addressable input sources with all but one assigned to the various status/parameter registers. Selection of a

particular status/parameter register for reading (via the Unibus data transceivers) is effected by asserting the proper address code.

The VT48 also contains address decoding logic for writing certain registers (see discussion on addressable registers under VT48 Operation and Programming). This logic, though not shown on Figure 4-1, simply decodes the content of the four register address lines and asserts a particular enabling level for writing the addressed register.

4.2 DETAILED DESCRIPTIONS

4.2.1 Vector Data Flow (Transfer Path)

A block diagram showing the VT48 transfer path for refreshing vector graphic data is given in Figure 4-2. All vector (and point) data travels over the same path. However, since some instructions require only a single word while others require two words, the technique used to implement the transfer differs.

The primary path for vector data on entry in the buffered data buffer is to the input buffer, buffered input buffer and then to multiplexers in the stack and silo control logic. There are two identical multiplexers, one handling X coordinate data and the other receiving Y coordinate data. Each multiplexer has four sets of inputs. The particular set selected for transfer to the related delta input multiplexer depends on the type of instruction being executed as follows:

1. Short Vector, Basic Short Vector, and Relative Point Instructions – These are all single word instructions, meaning that X/Y coordinate data is conveyed in a single word; Y coordinate data is conveyed in bits 00 through 05 and X coordinate data in bits 07 through 12. For these instructions, X/Y coordinate data is transferred through the 4:1 multiplexer simultaneously.

2. Long Vector, Absolute Vector, and Point/Offset Instructions – These are two-word instructions where the X component is sent in the first word and the Y component in the second word. Hence 12/10 bits are passed through the X 4:1 multiplexer on receipt of the first word. Similarly 12/10 bits are passed through the Y 4:1 multiplexer on processing the second word.

3. Graphplot X/Graphplot Y Instructions – The load status B instruction is used to load the VT48 Increment register prior to executing graphplot X/graphplot Y instructions. The increment is multiplexed through the appropriate 4:1 multiplexer at the same time the coordinate data (supplied by the graphplot instruction) is sent through the other 4:1 multiplexer.

4. Character Instruction – The amount that the beam must be stepped between each byte of the character instruction varies with respect to the selected character size. Consequently, the character step value is sent through the multiplexer during character processing.

From the 4:1 multiplexers, vector data is taken to associated 2:1 multiplexers. The latter circuits allow direct loading of X/Y offset registers (in the graphics calculation logic) from the Unibus transceivers.

The delta X/Y registers are used to store the delta X/Y values prior to the scissoring and tangent calculation process. The graphics calculation logic involves three hex PCBs and has its own block diagram coverage later in this section. At this point, it can be said, however, that for any given vector, this logic issues four major outputs as follows:

1. Delta Length – This is the length of the major axis; that is, the larger of the delta X/delta Y values.

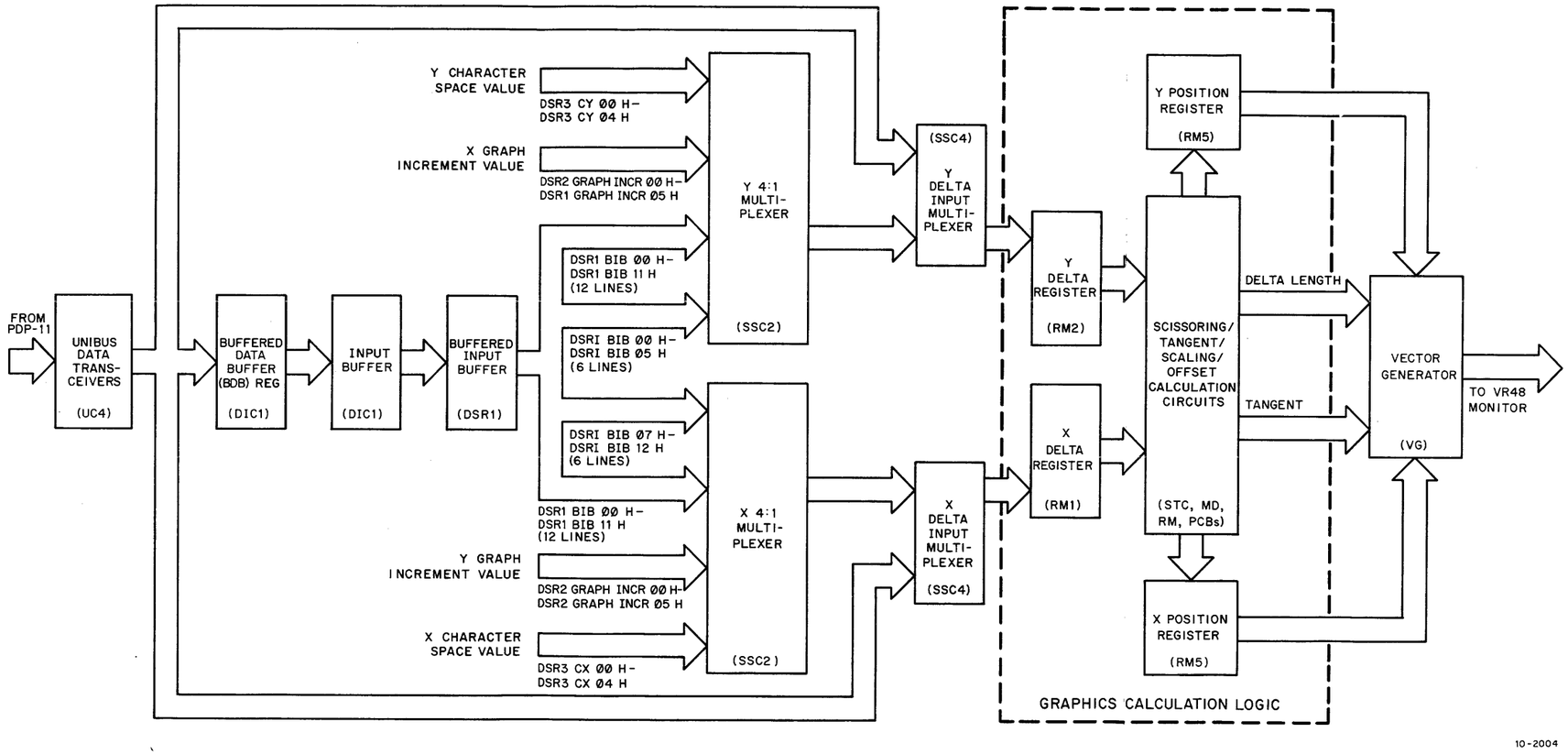


Figure 4-2 Vector Data Flow, Block Diagram

2. Tangent – This is the ratio of minor axis length to major axis length. The tangent and delta length (along with sign values) are used by the vector generator to physically draw the vector on the VR48 working surface.

3. X Position – This conveys the vector start and end points in the X coordinate.

4. Y Position – This conveys the vector start and end points in the Y coordinate.

The X and Y position values are applied to the vector generator at the conclusion of the vector drawing process. They are used by the vector generator to maintain the beam at the addressed vector end point in anticipation of the next graphic word. When one on-screen vector is followed immediately by another, the end point of the first vector is also the addressed starting point of the second vector.

4.2.2 Character Data Flow (Transfer Path)

A block diagram showing the transfer path used to refresh characters at the VR48 is shown in Figure 4-3. Each character instruction conveys two 7-bit ASCII characters over the Unibus. The low-order character is conveyed in bits 00 through 06, while the upper or high-order character is presented via bits 08 through 14.

On entry into the VT48, both characters are strobed into the BDB register and input buffer in sequence. From the input buffer, both characters are entered into the Character register. However, the entire low-order character is taken through the buffered input buffer as are the four low-order bits of the high-order character. The three high-order bits of the upper character are channeled directly from input buffer to Character register.

The Character register stores both characters so that they can be processed in sequence. Characters are loaded into the register only when the VT48 is being operated in the character graphic mode. That is, signal DIC8 CHARACTER H must be asserted at the Character register load logic for loading to occur.

Due to the fact that each character instruction contains two words, the display instruction decode logic treats the character instruction as a two-word instruction. Consequently, only one character at a time is allowed to transfer through the character select multiplexer for processing. During word zero (signal DIC8 WORD 0 H asserted), the low-order character is processed. During word one (DIC8 WORD 1 H asserted), the upper character is processed.

All characters exiting the Character register fall into one of two categories as follows:

1. Drawable (printable) Characters – For these characters, the character generator must be enabled to physically draw (refresh) the character on the face of the VR48 Display Monitor.

2. Control (nonprintable) Characters – These characters (carriage return, line feed, superscript, etc.) require that the character generator be inhibited. Some control characters have unique spacing requirements (e.g., subscript and superscript) that require a special change in beam position prior to drawing the next printable character. These requirements are handled by the graphics calculation logic.

When the control character decode logic determines that the character is *not* a control character (meaning that the character must fall in the printable category), it negates signal DSR6 CHAR INHIBIT LEVEL H. As a result, the graphics calculation logic asserts signal STC5 START CHAR GEN L. This in turn causes the character generator to accept the character from the character select multiplexer and draw it on the VR48 Display Monitor. (A block diagram of the character generator itself is described in a later paragraph.)

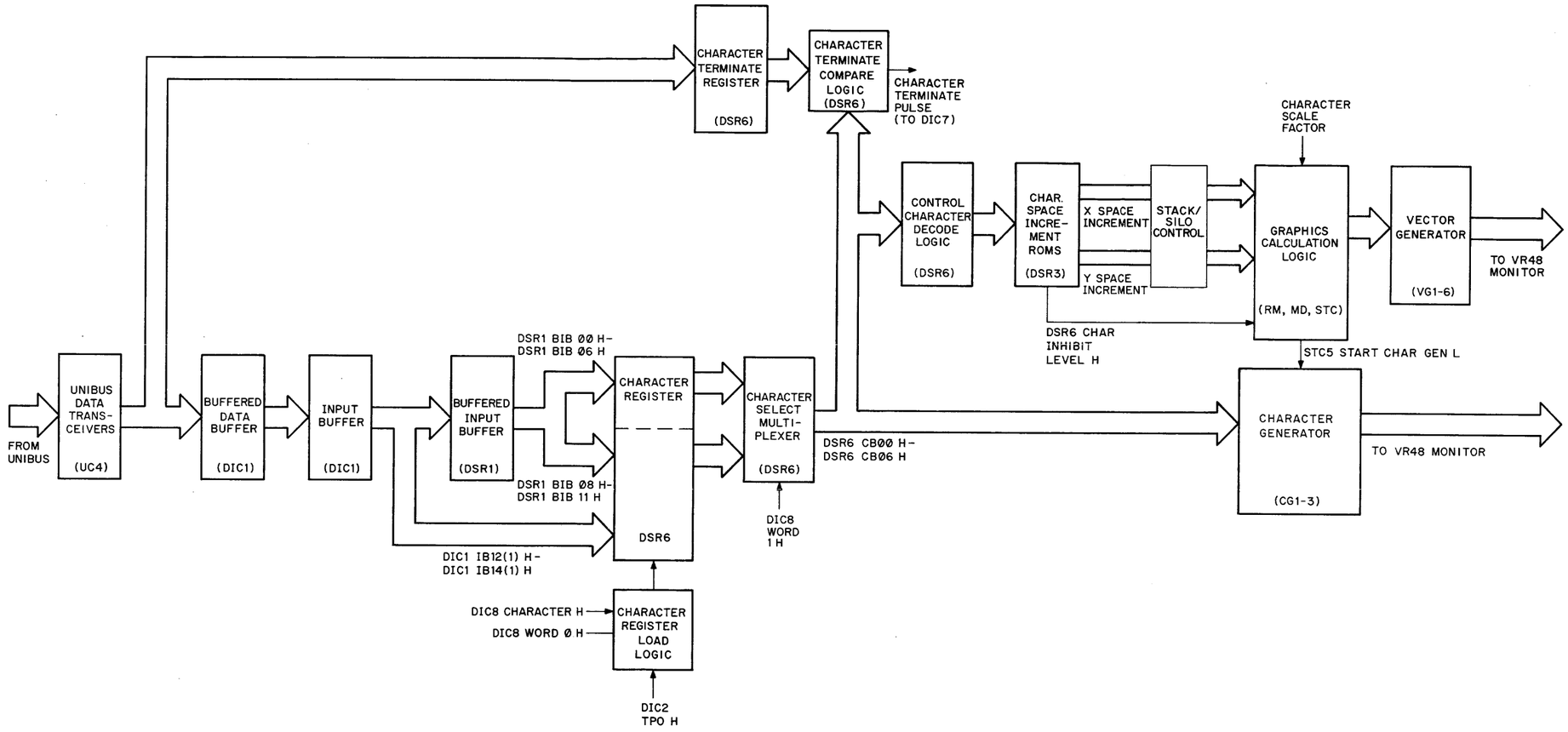


Figure 4-3 Character Transfer Path, Block Diagram

When the control character decode logic detects that the character exiting the Character register is a control character, it asserts signal DSP6 CHAR INHIBIT H. The character generator is now inhibited until such time as another drawable character is detected.

Even though the character generator is shut down, there may be a beam movement requirement because of the execution of a special control character. When printable characters are being executed, the space between characters (depending on scale factor) is a constant. However, if a subscript control character is detected in a character stream, it means that the beam must now be moved down and to the right prior to drawing the next printable character. The character space increment ROMs convey a coded value indicating the amount of X/Y beam movement required by each control character. The X/Y value issued from these ROMs is always a constant. However, the character scale factor may affect the X/Y position value sent to the vector generator. Of course, the VR48 beam is always blanked when moving the beam during control character execution.

An additional feature of the character decoding logic is the character terminate compare circuit. The VT48 is capable of forcing execution of a POP restore instruction under certain character matching conditions. As a prerequisite, the Character Terminate register must be loaded beforehand with the desired character code. That is, the character code, which, when matched by the character in process, will cause escape from the character processing sequence.

The character from the Character Terminate register is compared with every character exiting the Character register. When a match occurs, the character terminate compare logic issues the character terminate pulse to force execution of the POP restore instruction by the instruction decode logic.

4.2.3 Display Instruction Control and Time Pulse Generator Block Diagram Discussion

Figure 4-4 shows a block diagram of the circuits used to control instruction sequencing from the time the NPR cycle is generated until the instruction function levels are asserted to the other VT48 control circuit areas.

Except for startup conditions (start of the display file), the display instruction control sequence is triggered in one of two ways:

1. Triggering on Processing Control Instructions – Control instructions do not involve the graphics calculation logic, vector generator or character generator as part of their execution sequence. Therefore, the display instruction control initiates the next instruction processing cycle upon completion of the current cycle.

2. Triggering on Processing Graphic Entity Instructions – When graphic entity instructions are processed, the graphics calculation logic determines the rate at which instructions can be sequenced through the VT48. This logic not only controls the time at which the vector/character generators are triggered, but also controls the times at which it is ready to accept additional X/Y data from the input buffer. Therefore, the display instruction control must wait until the graphics calculation logic commands it to begin the next instruction processing cycle.

4.2.3.1 Basic Display Instruction Control Processing Cycle – The display instruction control processing cycle begins when the instruction sequence startup logic issues three outputs.

NOTE

Triggering of this logic can occur only after a previous instruction has been completed. Also, triggering cannot occur unless the next word has been fetched from the PDP-11 and is setting in the buffered data bit (BDB) register. This is discussed later.

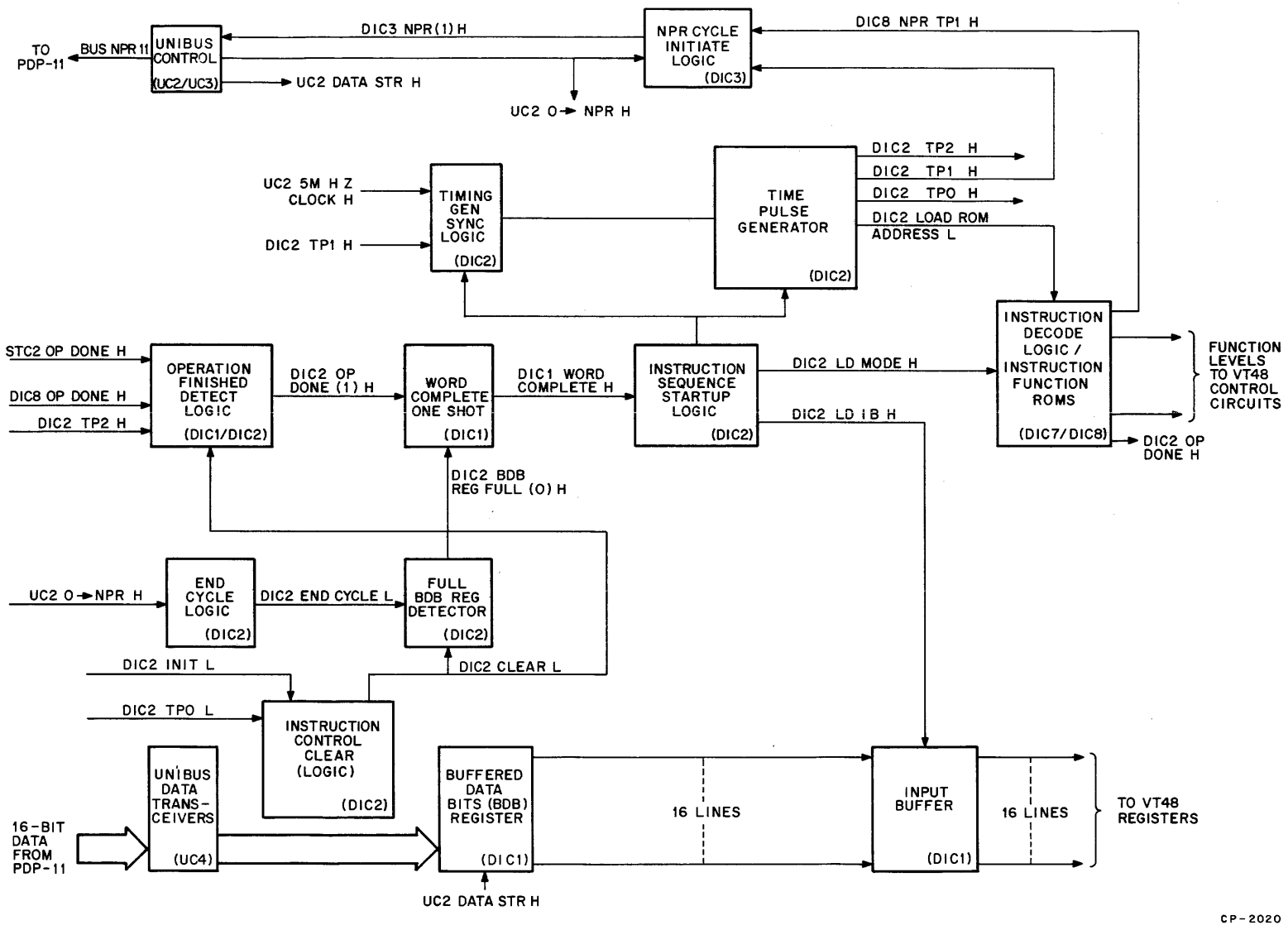


Figure 4-4 Display Instruction Control and Time Pulse Generator, Block Diagram

The three outputs from the instruction sequence startup logic and their purposes are:

1. DIC2 LD IB H – This strobes the word from the BDB register to the input buffer, meaning that the control/graphic data is presented to the appropriate VT48 registers.

2. DIC2 LD MODE H – This loads the OP code field of control instruction into the Control Mode register in the instruction decode logic.

3. The third output (undesignated) readies the time pulse generator to begin the word processing cycle.

When enabled, the time pulse generator produces four 200 ns timing pulses in sequence. The pulses are designated DIC2 LOAD ROM ADDRESS L, DIC2 TP0 H, DIC2 TP1 H, and DIC2 TP2 H. The first of these loads the ROM address in the instruction decode logic, which results in assertion of the function levels that enable the particular processing circuits that implement execution of the instructions. With these levels asserted throughout the instruction processing period, the remaining timing signals can now be used for strobing and clocking functions unique to the particular instruction.

The time pulse generator is not allowed to free run for two reasons:

1. Loading the BDB register during the NPR cycle is asynchronous. Consequently, there is no way of precisely anticipating the loading of a new 16-bit word from the PDP-11

2. The amount of time required by the graphic calculation logic to process graphic instructions varies. Hence, the display instruction control logic must wait until commanded before triggering a new instruction cycle.

For both of these reasons, the time pulse generator must be halted at the end of one instruction and then restarted (resynchronized) at the start of the next. This requirement is taken care of by the timing generator sync logic. When the undesignated signal from the instruction sequence startup logic asserts, the timing generator sync logic is enabled in such a way that subsequent clock pulses (UC2 5 MHZ CLOCK H) are allowed to pass through this logic and clock the time pulse generator itself.

However, only four pulses are allowed to pass through the timing generator sync logic. Note that timing pulse DIC2 TP1 H is applied to this logic. Assertion of this signal begins a disabling process that allows only one more clock pulse to pass. Hence the time pulse generator is disabled following generation of DIC TP2 H.

4.2.3.2 Implementation of NPR Cycle – Once the input buffer has been loaded and the instruction processing cycle has been entered, the display instruction control is free to initiate an NPR cycle. That is, the BDB register is now free to accept another word. One of the instruction function levels asserted from the instruction function ROMs is signed DIC8 NPR TP1 H. This signal is asserted throughout the processing cycle of most instructions. Assertion of this signal (together with generation of timing pulse DIC2 TP1 H) triggers the NPR cycle initiate logic. Signals DIC NPR (1) H and BUS NPR L are now produced (in sequence) to begin the NPR cycle.

At the conclusion of the NPR cycle (when the 16-bit word is on the Unibus data lines) the Unibus control logic asserts UC2 DATA STR H to enter the word into the BDB register. Also at this time, signal UC2 0 → NPR H is asserted by the Unibus control logic. Besides incrementing the DPC, the latter signal is used for the following purposes:

1. Clears the NPR cycle initiate flip-flop to ready it for the next NPR cycle.

2. Causes the end cycle logic to assert the DIC2 END CYCLE signal. Application of this signal to the full BDB register detect circuit causes assertion of signal DIC2 BDB REG FULL (0) H. This action, in turn, enables triggering of the word complete one shot multivibrator.

From the preceding, it can be seen that the word complete one shot cannot be triggered (to initiate the instruction sequence startup logic) until such time as the BDB register has been loaded with an instruction word from the PDP-11.

NOTE

The full BDB register detect logic is cleared every processing cycle at TP0 time. This results from the application of signal DIC TP0 L to the instruction control clear logic.

4.2.3.3 Operation Finished Detect Logic – As stated earlier in this paragraph, the instruction processing sequence is triggered in one manner for control instructions and another for graphic entity instructions. A prerequisite for triggering the instruction processing sequence is recognition that the preceding processing sequence has been completed. Such recognition is effected by the operation finished detect logic. Detection occurs in one of two ways:

1. Control Instructions – Since all processing functions attendant to control instructions are completed within two time pulse generator cycles, signal DIC8 OP DONE H (ANDed with DIC2 TP2 H) is used to enable the operation finished detect logic. Signal DIC8 OP DONE H is asserted throughout all control instruction words as a means of initiating the next word processing cycle.

NOTE

Signal DIC8 OP DONE H is also asserted for every word except the last word of multiword graphic instructions. It is not asserted during single word or the last word of multiword graphic instructions because the display instruction control must wait until the graphics calculation logic has accepted the X/Y data for processing before triggering the next cycle.

2. Graphic Entity Instructions – For these instructions, signal STC2 OP DONE H, issued from the graphics calculation logic, informs the display instruction control when it may initiate the next word processing cycle. Assertion of this signal indicates that the graphics calculation logic has accepted the previously loaded X/Y data (loaded during preceding graphic entity instruction) and is therefore ready to receive additional X/Y data.

Both of the preceding conditions cause assertion of the DIC2 OP DONE (1) H signal. This in turn ripples through the word complete one shot (provided that the BDB register is full) to trigger the instruction sequence startup logic.

The operation finished detect logic is cleared during the word processing cycle in the same way as the full BDB register detector.

4.2.4 Instruction Decoding Circuits, Block Diagram Discussion

Figure 4-5 shows a block diagram of the circuits used to decode VT48 instructions. Regardless of instruction type, the technique for decoding and initiating instruction processing is fundamentally the same and can be considered to occur in three phases:

1. Storing of the OP code/graphic mode field for decoding.

2. Interpretation and conversion of the OP code into an 8-bit instruction select ROM (read only memory) address.

3. Assertion of instruction function and instruction subfunction levels that are used (together with the timing signals) to carry out instruction processing. Particular levels asserted for any given instruction are a function of the ROM address developed in phase 2 above.

As a further aid in understanding the instruction decoding process, Table 4-1 is presented. This table summarizes instruction OP codes, related 8-bit ROM addresses, and the function/subfunction levels asserted for each instruction type. Familiarity with the instruction fields other than the OP code (VT48 operation and programming) will also help to understand the design intent of the instruction decode logic.

Two registers are provided for storing the OP code/graphic mode field of the instruction being decoded. The reason for the two registers stems from the fact that the sequence for initiating graphic entity instructions differs from that of control mode instructions. That is, a graphic entity instruction must always be preceded by the set graphic mode instruction (itself a control instruction), which defines the type of graphic entity instruction to follow. Consequently, decoding graphic entity instructions can be considered as a two-step process:

1. Decoding of the Set Graphic Mode Instruction – This not only defines the type of graphic data instruction to follow, but also establishes certain parameters (intensity, line type, etc.) for the graphic entity to be processed.

2. Decoding and processing of the graphic entity (point, character, vector, etc.) instruction itself.

In contrast, control instructions are decoded by looking at the entire OP code of the single control instruction and then simply converting it to an 8-bit ROM address to assert the required instruction processing levels.

4.2.4.1 Control Instruction Decoding – The subsequent paragraphs describe the decoding process for all control instructions other than the set graphic mode instruction.

An analysis of the control mode instruction OP code fields shows that they can vary from five to seven bits in length. All OP code bits are brought into the instruction decode logic via the buffered data bits (BDB) register. The high order bit (bit 15) is applied to the control mode/graphic mode select flip-flop. Since OP code bit 15 is always a "1" for control instructions, this flip-flop is always set throughout decoding and execution of control instructions.

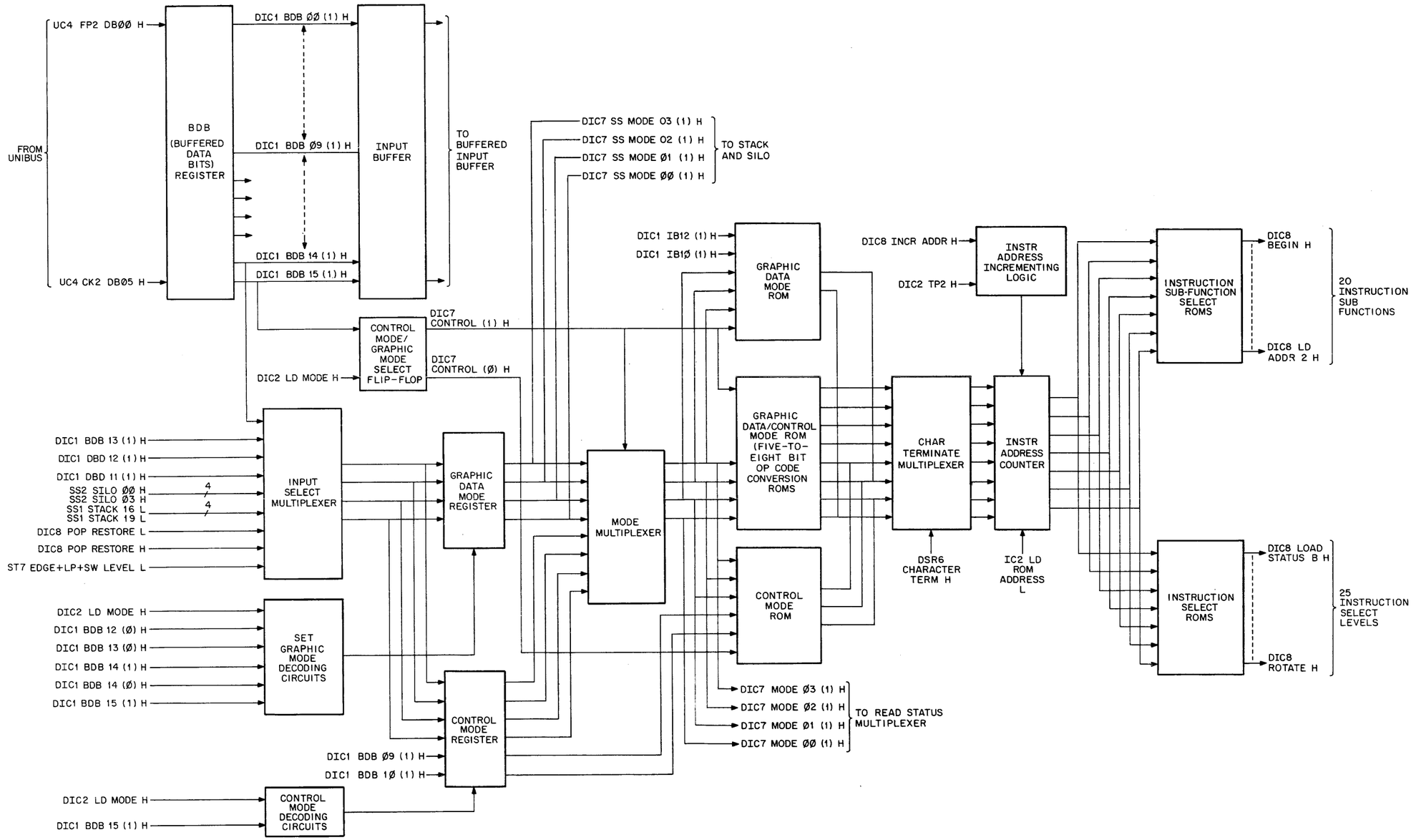


Figure 4-5 Instruction Decode Logic, Block Diagram

Table 4-1 Instruction Decode Functions/Subfunctions per Instruction Type

	OP CODE GRAPHIC MODE 15	14	13	12	11	10	09	DIC7 ROM ADDRESS	SUB FUNCTION SELECT: DIC8 RESTORE Q H	DIC8 OP DONE H	DIC8 A WORD Q H	DIC8 BEGIN H	DIC8 REG Q H	DIC8 LOAD PC H	DIC8 INCR ADDR H	DIC8 PC ADD TO H	DIC8 PC ADD TO L H	DIC8 NPR-TP H	DIC8 NPR-TP2 H	DIC8 HOLD H	DIC8 A WORD 1 H	DIC8 A WORD 2 H	DIC8 ZINC ADDR H	DIC8 ZOP DONE H	DIC8 PC DONE H	DIC8 LD REL REG H	DIC8 RESTORE 1 H	DIC8 LD ADDR Z H	FUNCTION
ROTATE	0	0	0	1	0	–	–	11 000 011	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 ROTATE H
	0	0	0	1	0	–	–	11 000 100	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	DIC8 ROTATE H
	0	0	0	1	0	–	–	11 000 101	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	1 DIC8 ROTATE H
								11 000 110	SPARE																		
LONG VECTOR[a]	0	0	0	1	0	–	–	11 000 111	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 VECTOR H
	0	0	0	1	0	–	–	11 001 000	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	DIC8 VECTOR H
	0	0	0	1	0	–	–	11 001 001	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1 DIC8 VECTOR H
SET GRAPHIC MODE[b]	1	X	X	X	X	–	–	11 001 010	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 DATA MODE H
								11 001 011	SPARE																		
NAME	1	1	0	1	0	–	–	11 001 100	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 NAME H
LOAD STATUS C	1	1	0	1	1	–	–	11 001 101	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 LOAD STATUS C H
								11 001 110	SPARE																		
								11 001 111	SPARE																		
ABSOLUTE VECTOR	0	1	0	0	1	–	–	11 010 000	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 ABSOLUTE VECTOR H
	0	1	0	0	1	–	–	11 010 001	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	DIC8 ABSOLUTE VECTOR H
	0	1	0	0	1	–	–	11 010 010	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1 DIC8 ABSOLUTE VECTOR H
RELATIVE POINT	0	0	1	1	0	–	–	11 010 011	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0 DIC8 RELATIVE POINT H
	0	0	1	1	0	–	–	11 010 100	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	1 DIC8 RELATIVE POINT H
CIRCLE/ARC[c]	0	1	0	0	0	–	–	11 010 101	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 CONIC H
	0	1	0	0	0	–	–	11 010 110	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	DIC8 CONIC H
	0	1	0	0	0	–	–	11 010 111	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1 DIC8 CONIC H
OFFSET[d]	0	0	0	1	1	–	–	11 011 000	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 OFFSET H
	0	0	0	1	1	–	–	11 011 001	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	DIC8 OFFSET H
	0	0	0	1	1	–	–	11 011 010	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	1 DIC8 OFFSET H
								11 011 011	SPARE																		
POINT[d]	0	0	0	0	1	–	–	11 011 100	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 POINT H
	0	0	0	0	1	–	–	11 011 101	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	DIC8 POINT H
	0	0	0	0	1	–	–	11 011 110	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1 DIC8 POINT H
LOAD STATUS A	1	1	1	1	0	–	–	11 011 111	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	DIC8 LOAD STATUS A H
BASIC SHORT VECTOR	0	0	1	1	1	–	–	11 100 000	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DIC8 BASIC SHORT VECTOR H
	0	0	1	1	1	–	–	11 100 001	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	DIC8 BASIC SHORT VECTOR H
								11 100 010	SPARE																		
								11 100 011	SPARE																		

a. Bit 12 of the Rotate instruction is used to select Rotate rather than basic long vector.

b. Content of Bits 11 through 14 defines type of graphic data instruction that follows.

c. Instruction used with Depth Queuing Option.

d. Bit 12 of graphic data instruction is used to select between point and offset instructions.

Table 4-1 Instruction Decode Functions/Subfunctions per Instruction Type (Cont)

	OP CODE GRAPHIC MODE 15	14	13	12	11	10	09	DIC7 ROM ADDRESS	SUB FUNCTION SELECT: DIC8 RESTORE O H	DIC8 OP DONE H	DIC8 A WORD O H	DIC8 REGIN H	DIC8 REGO H	DIC8 DADDR H	DIC8 INCR ADDR H	DIC8 PC ADD 10H	DIC8 PC ADD 01H	DIC8 XPRTH H	DIC8 XPRTL H	DIC8 HOLD H	DIC8 A WORD 1 H	DIC8 A WORD 2 H	DIC8 ZINC ADDR H	DIC8 ZOP DONE H	DIC8 DC DOWN H	DIC8 D REL REG H	DIC8 RESTORE 1 H	DIC8 LD ADDR Z H	FUNCTION
BASIC VECTOR[e]	0	0	1	0	1	X	–	11 100 100	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	DIC8 BASIC VECTOR H
GRAPHPLOT Y[e]	0	0	1	0	1	X	–	11 100 101	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 GRAPH Y H
JUMP RELATIVE	1	1	1	0	0	0	1	11 100 110	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	DIC8 JUMP REL H
								11 100 111	SPARE																		
BASIC VECTOR[e]	0	0	1	0	0	X	–	11 101 000	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	DIC8 BASIC VECTOR H
GRAPHPLOT X[e]	0	0	1	0	0	X	–	11 101 001	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	DIC8 GRAPH X H
JUMP TO SUB-ROUTINE ABSOLUTE	1	1	1	0	0	1	0	11 101 010	0	1	1	0	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	DIC8 JSR H
	1	1	1	0	0	1	0	11 101 011	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	DIC8 JSR H
JUMP TO SUB-ROUTINE RELATIVE	1	1	1	0	0	1	1	11 101 100	0	0	1	0	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	DIC8 JSR REL H
	1	1	1	0	0	1	1	11 101 101	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0	DIC8 JSR REL H
JUMP ABSOLUTE	1	1	1	0	0	0	0	11 101 110	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	DIC8 JUMP H
	1	1	1	0	0	0	0	11 101 111	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	DIC8 JUMP H
								11 110 000	SPARE																		
SHORT VECTOR[f]	0	0	0	0	1	–	–	11 110 001	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	DIC8 SHORT VECTOR H
	0	0	0	0	1	–	–	11 110 010	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	1	DIC8 SHORT VECTOR H
								11 110 011	SPARE																		
								11 110 100	SPARE																		
LOAD STATUS B/ LOAD STATUS BB[g]	1	1	1	1	1	X	–	11 110 101	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	DIC8 LOAD STATUS B H
POP, NOT RESTORE/ LOAD SCOPE SELECTION[h]	1	1	1	0	1	0	1	11 110 110	1	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	DIC8 POP H
CHARACTER[f]	0	0	0	0	0	–	–	11 110 111	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DIC8 CHARACTER H
	0	0	0	0	0	–	–	11 111 000	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	DIC8 CHARACTER H
								11 111 001	SPARE																		
POP RESTORE	1	1	1	0	1	1	0	11 111 010	1	0	1	0	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	DIC8 POP RESTORE H
	1	1	1	0	1	1	0	11 111 011	0	0	0	1	0	0	1	1	0	0	0	0	1	0	0	0	0	1	0	DIC8 POP RESTORE H
NOP/LOAD SCOPE SELECTION[h]	1	1	1	0	1	0	0	11 111 100	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	DIC8 NOP H
								11 111 101	SPARE																		
								11 111 110	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
								11 111 111	SPARE																		

e. Bit 10 of graphic data instruction is used to select between Basic Vector and Graphplot instructions.

f. Display instruction control treats these as two word instructions since they contain two sets of graphic data that require processing.

g. Bit 10 of load status BB instruction indicates load status BB rather than load status B.

h. OP code for load scope selection instruction is contained within NOP and POP OP codes.

The next four bits of the OP code (bits 11–14) are taken through a multiplexer for application to two registers: the Graphic Data (Entity) Mode register and the Control Mode register.

NOTE

These four bits are multiplexed because they define the graphic mode when executing graphic data instructions. Under certain VT48 processing conditions, these four bits must be accessed from the stack/silo memory and inserted into the Graphic Data Mode register. This is done to restore the graphic mode field to the Graphic Data Mode register under certain interrupt and POP restore conditions.

Under normal conditions (no POP restore or edge, light pen, light pen switch interrupts in process), signal DIC8 POP RESTORE L is a high. This selects bits 11 through 14 from the BDB register for application to the Graphic Data Mode and Control Mode registers.

With a control instruction being processed, the Control Mode register (rather than Graphic Data Mode register) must be loaded with the OP code. This is accomplished by the control mode decoding circuit. This circuit samples the content of bit 15 (always a 1 for control instructions) and loads the Control Mode register coincident with the generation of the DIC2 LD MODE H signal. (This signal, which also sets the control mode/graphic mode select flip-flop is issued by the display instruction control logic after the incoming word has been loaded into the BDB register.)

Bits 9 and 10 of the OP code field are also applied to and loaded into the Control Mode register at this time. Note, however, that they are not applied to the mode multiplexer as are bits 11 through 14. Instead, they are applied directly to the control mode ROM.

Once the Control Mode register is loaded, conditions are set up for converting the instruction OP code into its related 8-bit ROM address code. With the control mode/graphic mode select flip-flop set, the mode multiplexer selects the output of the Control Mode register for application to the graphic data/control mode ROM. Three additional effects produced at this time by the control mode/graphic mode select flip-flop are:

1. Signal DIC7 CONTROL (1) H high, serves to inhibit the BUT graphic data mode ROM. That is, no outputs can issue from this ROM as long as the control mode/graphic mode select flip-flop is set.

2. Signal DIC7 control (1) H is applied to the graphic data/control mode ROM along with the content of bits 11 through 14 from the Control register. Thus the five high-order bits of the OP code are applied to this ROM.

3. Signal DIC7 CONTROL (0) H being low enables the BUT control mode ROM.

The conversion from five bit OP code to eight bit ROM address is accomplished by the graphic data/control mode ROM. Note that there are eight lines exiting this ROM. When the OP code of the control instruction is 5 bits in length, the 8-bit value issued from the ROM (and passed through the character terminate multiplexer) is determined entirely by the content of bits 11 through 15 as applied to the graphic data/control mode ROM.

Control instructions whose OP codes exceed 5 bits in length employ the BUT control mode ROM. For such instructions the content of bits 9 and 10 are used together with bits 11 through 14 to effect branch on micro test (BUT) multiplexing of the eight bit ROM address. The outputs from the BUT control

mode ROM are wire ORed to three outputs of the graphic data/control mode ROM. A 0 output over any of the three BUT control mode ROM lines forces the corresponding line of the 8-bit ROM address to ground. Consequently, the input to the instruction address counter is correspondingly altered.

The character terminate multiplexer is used solely when the POP-on character terminate capability is enabled. When a character match occurs, under such conditions, the VT48 must force execution of the POP restore instruction. When signal DSR6 CHARACTER TERM H is asserted, the character terminate multiplexer applies the 8-bit address of the POP restore command to the instruction address counter.

After the 8-bit ROM address is fully settled on the address lines, signal DIC2 LD ROM ADDRESS L is issued to load the address into the instruction address counter. A counter is required here because some VT48 instructions require a second and possibly a third word for their execution. After the first word of an instruction has been processed, the instruction address incrementing logic issues an output to increment the 8-bit instruction address. This can be seen by looking at the long vector instruction on Table 4-1. Note that the DIC8 INCR ADDR H subfunction level is asserted during each of the first two words of the instruction. It is this signal that allows incrementing the instruction address for processing the second and third words of the instruction.

The instruction address counter issues a 7-bit output for application to the instruction function select ROMs and instruction subfunction select ROMs. It is these latter circuits that assert the signal levels necessary to implement processing each word of an instruction. There are 25 instruction select levels and 20 instruction subfunction levels. All levels and the times of their assertion are indicated on Table 4-1.

NOTE

Since the high-order bit of all 8-bit ROM addresses is always a 1, it is sufficient to apply seven bits of the 8-bit code to select the various levels from the instruction select and instruction subfunction select ROMs.

4.2.4.2 Set Graphic Mode Instruction Decoding – The set graphic mode is a control instruction since, like the others, bit 15 is a 1. However, it is handled differently within the instruction decode logic for two reasons.

1. The graphic mode field (bits 11 through 14) must be loaded into Graphic Data Mode register as well as into the Control Mode register.

2. Any one of ten possible bit patterns can exist in the mode field (bits 11 through 14).

Each of these bit patterns selects the same 8-bit ROM address upon executing the set graphic mode instruction. This means that the same instruction level and the same instruction subfunction levels are selected regardless of the graphic mode being selected.

Loading the Graphic Data Mode register with bits 11 through 14 is accomplished by the set graphic mode decoding circuits. This logic samples the content of bits 12 through 15 (bit 15 being set to a 1). Analysis of the ten possible graphic data mode codes shows that all but two have bit 14 at the 0 level. For these instructions it is only necessary to sample bits 15 (1) and 14 (0) to load the Graphic Data Mode register. For the remaining two graphic mode patterns, it is necessary to sample from bits (12 through 15) to issue the load signal. Signal DIC2 LD MODE H strobes these circuits to load the Graphic Data Mode register at the same time that the Control Mode register is being loaded.

Even though the Graphic Data Mode register is now loaded with the graphic mode field, it is not allowed to present its outputs to the graphic data mode/control mode ROM at this time. Since the graphic mode/control mode select flip-flop is set (bit 15 = 1), the outputs of the Control Mode register are selected by the mode multiplexer.

Since there are ten different graphic data modes, any one of ten possible bit patterns may now be issued from the Control Mode register. However, the graphic data/control mode ROM is so programmed that all graphic mode codes force the same 8-bit address pattern at the output. Once the 8-bit instruction address has been loaded into the instruction address counter, the set graphic mode instruction is handled in the same way as all other instructions; that is, it asserts the necessary instruction level and instruction subfunction levels to process the instruction.

4.2.4.3 Graphic Data Instruction Decoding – As mentioned at the start of this paragraph, decoding of graphic data instructions is actually a two-step process. The first step is execution of the set graphic mode instruction which sets up the conditions for processing a particular graphic entity instruction. The preceding paragraph described how the decoding process for the set graphic mode field places the graphic mode field in the Graphic Data Mode register as well as the Control Mode register. It is this action that sets up the instruction decode logic to decode the instruction that follows.

All graphic mode instructions have bit 15 set to a 0. This is significant because it is this feature that distinguishes graphic data mode instructions from control instructions. When the set graphic mode instruction is executed prior to the graphic mode instruction, it leaves the control mode/graphic mode select flip-flop in the set state. Now, when the graphic data mode instruction is executed, this flip-flop is cleared. This has a threefold effect:

1. Switches signal DIC7 CONTROL (1) H low so that now the output of the Graphic Data Mode register is selected for application to the graphic data/control mode ROM.

2. Inhibits the BUT control mode ROM and enables the BUT graphic data mode ROM.

3. Swiches the bit 15 input to the graphic data/control mode ROM to 0; this effectively changes the input address so that now each of the ten graphic modes can be decoded.

NOTE

During execution of graphic data mode instructions, the set graphic mode decoding logic and control mode decoding circuits are inhibited (bit 15 = 0). Consequently, there can be no change in either related register when graphic data mode instructions are executed.

Each graphic mode code selects a unique 8-bit address at the output graphic data/control mode ROM; that is, the same as the case for the control instructions. The 8-bit address may be modified by the output of the BUT graphic data mode ROM for certain instructions (offset, basic vector). Once the 8-bit ROM address is loaded into the instruction address counter, the graphic data instruction is handled in the same way as described for the others.

4.2.5 VT48 Status/Parameter Data Routing

The presence of stack and silo memories within the VT48 makes the routing of the status and parameter data (associated with each graphic entity) somewhat complex. Figure 4-6 is presented here as an aid to understand the types of status/parameter data to be transferred within the VT48 and the system conditions requiring internal routing.

In general, there are three system conditions that call for routing of status/parameter data:

1. Initial Status/Parameter Setup - Such routing occurs when setting up the parameters attendant to a given graphic data entity that is to follow. In this case, load status, name, and set graphic mode instructions are executed to load the parameters into the related registers. This means that the parameter/status data is available for entry into the silo memory when the subsequent graphic data instruction is executed. It also means that the status parameter data is available for entry into the stack memory should a JSR instruction be executed.

2. Generation of one of the three display surface related interrupts, i.e., light pen, light pen switch or display surface area edge transition by a point/vector. Under these conditions, all status/parameter data at the silo memory read address (i.e., attendant to the graphic entity being processed and therefore directly associated with the generated interrupt), must be re-entered into the related status/parameter registers. This is done so that all status/parameter data is presented to the read status multiplexer for access by the central processor.

3. Execution of JSR and POP Restore Instructions - When a JSR instruction is executed, the current status/parameter and display program counter information is entered into the stack memory. When a POP restore instruction is executed at the end of a subroutine, the status/parameter and display program counter information is restored to the related registers. This feature eliminates the requirement of status/parameter setup when it is desired to return to a given area in the display file following execution of a subroutine.

A POP restore instruction is also executed on a character terminate match at times when the character string escape feature is in use.

4.2.5.1 Routing of Status/Parameter Data During Initial/Setup Operations - When setting up status/parameters prior to executing a graphic entity instruction (or string of graphic instructions) two instruction types are used:

1. Load status instructions
2. Set graphic mode instruction

For load status instructions, the blocks designated Status/Parameter registers and Name register (Figure 4-6) are to be loaded. That is, when the NAME instruction is executed, the 11-bit coded name value is entered into the Name register. In this case, the transfer path involves the BDB register, input buffer, buffered input buffer, stack/silo/input buffer multiplexer, and then the Name register itself. For the other load status instructions (load status A, load status B, load status BB, load status C, and load scope selection), the transfer path is the same as above except that on exiting the stack/silo/input buffer multiplexer, status data is entered into the related status/parameter registers.

The set graphic mode instruction is handled differently from all others. That is because it conveys the graphic mode (type of graphic entity instruction to follow) as well as status/parameter data (intensity, blink, line type, and light pen interrupt enable). Graphic mode code routing is from the BDB register to the Graphic Data Mode register. Status/parameter data routing is via the same path as load status A, etc.

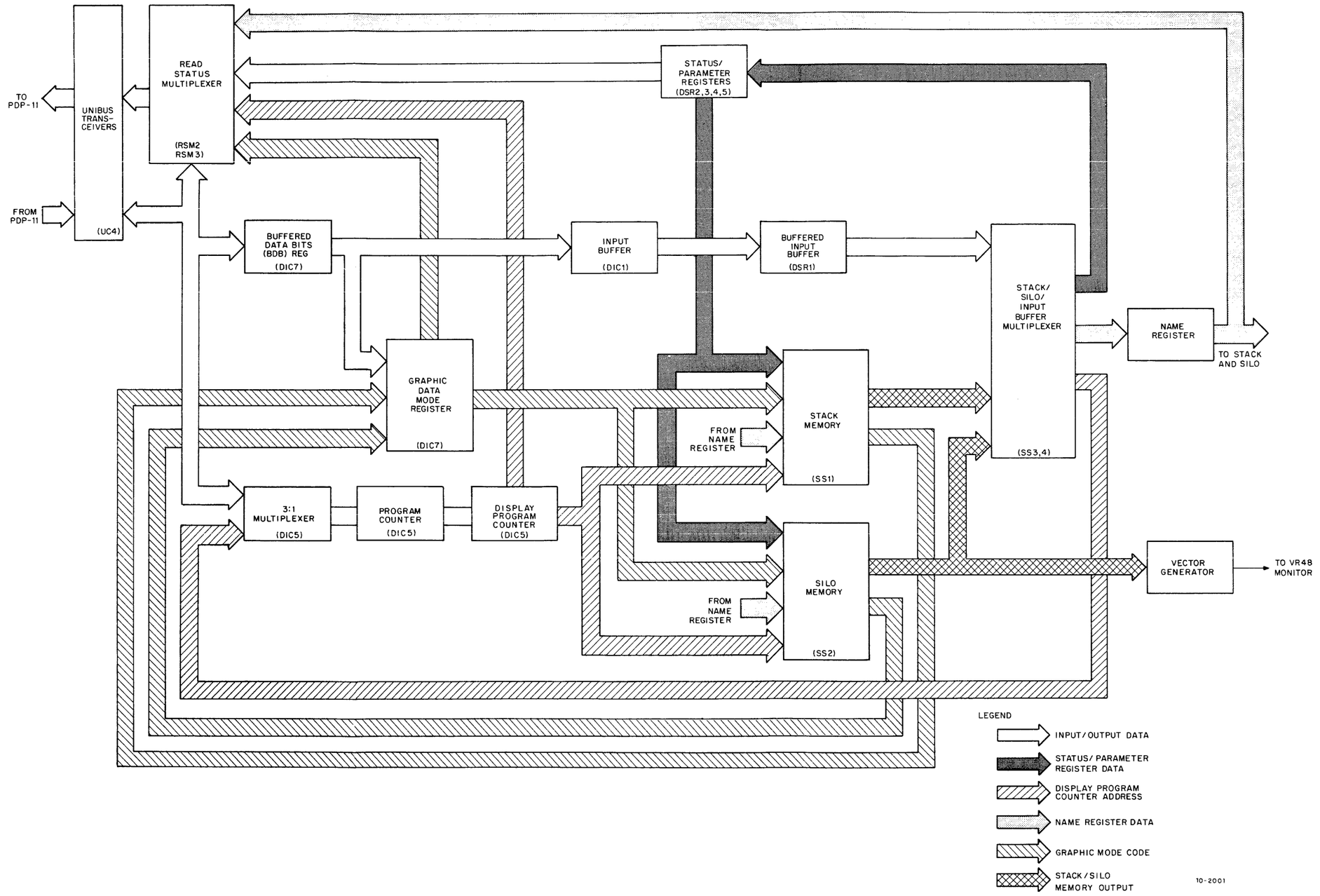


Figure 4-6 Status/Parameter Data Routing, Block Diagram

With all pertinent status/parameter (and mode) data now loaded, conditions are set up for entering such data into the silo memory on execution of the graphic entity instruction that follows the set graphic mode instruction.

NOTE

Status data is entered into silo memory (and the silo address is updated) only on execution of graphic data instructions.

Status/parameter data is entered into the silo memory directly from the related registers as shown on Figure 4-6. The content of the display program counter (DPC) is also entered into the silo memory.

NOTE

The display file address within the DPC is always updated by 2 during each word processing cycle. Thus when processing a string of graphic entity words, with no attendant changes in status/parameters or mode, the only changing value entered into silo memory is the DPC address. Changes in the status and mode configurations occur only when intervening load status or set graphic mode instructions are executed.

The silo memory addressing and updating technique is such that all status and parameter data is present at the proper silo read address when the related graphic data entity is being processed by the vector/character generator.

This is necessary for two reasons:

1. The related intensity, line type, and blink parameters must be available to the vector generator at the time the vector/character is being processed. (This is shown by the lower solid black line on Figure 4-6).

2. The status/parameter data must be available for re-entry into the related registers should a display surface related interrupt occur.

4.2.5.2 Routing of Status/Parameter Data on Generation of Display Surface Related Interrupts – There are three interrupt conditions within the VT48 that necessitate transfer of the status/parameter data from the silo memory to the related status/parameter registers. These interrupt conditions are:

1. Light Pen Hit

2. Light Pen Switch Transition

3. Edge transition interrupt (part of drawn vector is transitioning edge of major display working surface)

Processing of each of these types of interrupts by the central processor requires accessing of status/parameter data attendant to the particular graphic entity that initiated the interrupt. Since the status/parameter data for the graphic entity being processed is at the current silo read address, it can be readily strobed back into the related registers. And from the registers, it can be accessed by the central processor via the read status multiplexer.

The routing path from silo to the status/parameter registers is via the stack/silo/input buffer multiplexer for all data except the graphic mode code. The latter is taken directly back to the Graphic Data Mode register for presentation to the read status multiplexer.

By accessing the right registers, the CPU can now determine the following regarding the graphic data entity initiating the interrupt:

1. Graphic Mode – This indicates the type of graphic entity involved in the interrupt.

2. Display Program Counter Address – Indicates the display file address of the graphic entity.

3. Status/Parameter Register Data – Conveys the status and parameter conditions set up prior to processing the string of graphic words containing the particular graphic data entity initiating the interrupt.

4. Name Register Data.

4.2.5.3 Routing of Status/Parameter Data on Execution of JSR and POP Restore Instructions – The stack memory is used only on execution of JSR and POP restore instructions. Loading of the stack memory occurs on the former, while re-entry of status/parameter data to the appropriate registers occurs on the latter.

The display program counter (DPC) address entered into the stack memory differs depending on whether a jump to subroutine (JSR) absolute or jump to subroutine relative instruction is being executed. The difference is as follows:

1. The JSR absolute instruction is a two-word instruction. Therefore, the address entered into the stack is DPC + 4.

2. The JSR relative is a one-word instruction and DPC + 2 is entered into the stack.

Entry of status/parameter data into the stack on either JSR instruction is direct from the related registers to the stack.

The POP restore instruction is generally used at the end of subroutine to return to that area in the display file where a preceding JSR instruction was executed. By taking the pertinent status/parameter data from the stack and entering it into the related registers, the normal needs of loading status and setting graphic mode (via execution of their related instructions) are eliminated.

Routing of status/parameter data from stack memory to the related registers is as follows:

1. Display program counter (DPC), Name register contents, and Status/Parameter register data are all re-inserted via the stack/silo/input buffer multiplexer.

2. The graphic mode code is entered directly into the Graphic Data Mode register.

With the data restored to the registers, the proper status/parameters will be inserted into the silo memory on execution of the next graphic entity instruction accessed by the DPC.

4.2.6 Display Program Counter Input/Output Flow

A block diagram showing the routing of display file address data between the various registers, adders, and special memories within the VT48 is given in Figure 4-7. With the exception of the load DPC control logic, all blocks shown on this illustration convey address data to/from the display program counter (DPC). In general, the routing path followed is a function of the particular VT48 instruction being executed at any given time. A summary of the system conditions that can affect the loading of the DPC and possibly change the source/destination of the stored address data is given below:

1. Start – This involves loading the starting display file address from the PDP-11 into the DPC.

2. Normal Update – This involves simply incrementing the DPC contents by 2 during graphic data instructions and many of the control instructions.

3. Display Surface Related Interrupts – Certain interrupts require routing of address data between silo memory and the DPC.

4. Execution of Jump Relative and Jump to Subroutine (JSR) Relative Instructions.

5. Execution of Jump Absolute and Jump to Subroutine Absolute Instructions.

6. Execution of POP Restore Instruction – This requires routing of address data between stack memory and the DPC.

7. Loading of the Relocate Register.

Routing of DPC address data for each of the preceding conditions is described in the subsequent paragraphs.

4.2.6.1 Start DPC Address Routing – During VT48 processing startup, the display file starting address must be loaded from the PDP-11 into the DPC. The starting address is received from the Unibus and taken directly to the 3:1 multiplexer circuit. This multiplexer accepts data from the Unibus transceivers under conditions where neither signal DIC6 PC ADD 10 L nor signal DIC PC ADD 01 L is asserted.

From the 3:1 multiplexer, the high-order 11 bits are taken through the program counter to the DPC. The four low-order bits are taken directly to the DPC.

Loading the DPC is effected by signal DIC4 START L, which is asserted during the VT48 startup sequence only. With the DPC now loaded, the starting address for the display file is presented to the PDP-11 over the Unibus. Consequently, the first NPR cycle fetches the first word of the display file.

4.2.6.2 DPC Normal Update – Once the display file starting address has been loaded, the DPC is simply updated by 2 to fetch each sequential word in the display file. There are some additional loading requirements for certain instructions such as jump and JSR instructions as described later.

4.2.6.3 Routing During Display Surface Related Interrupts – For every graphic data entity processed, the contents of the DPC are entered into the silo memory as well as being used for fetching the next word in the display file. When a display surface related interrupt (light pen hit, light pen switch transition, or working surface edge transition) is generated, the application program needs to know the display file address of the exact graphic entity involved in the interrupt. The DPC address for that graphic entity is at the current silo memory read address when the related interrupt pulse is issued.

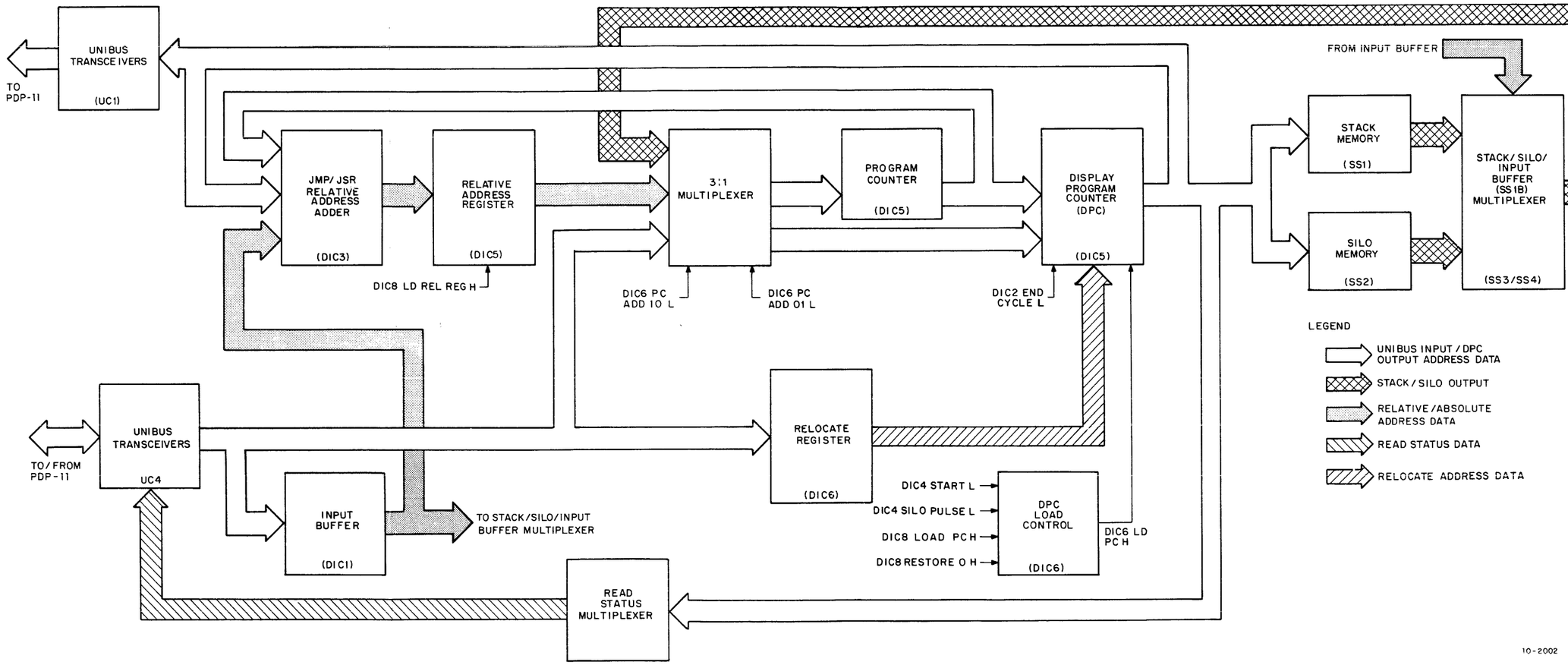


Figure 4-7 Display Program Counter, Input/Output Flow Block Diagram

When the interrupt pulse is generated, the DPC address is taken from the silo memory, through the 3:1 multiplexer and entered into the DPC.

All three types of interrupts cause assertion of the DIC4 SILO PULSE L signal. This in turn generates DIC6 LD PC H as a high to load the DPC.

The DPC contents from the silo memory are passed through the 3:1 multiplexer due to the assertion of signal DIC6 PC ADD 10 L. This signal is asserted for all three types of display surface related interrupts.

The DPC address is available to the PDP-11 via the read status multiplexer. (Refer also to the block diagram discussion on status/parameter routing.)

4.2.6.4 Routing During Execution of Jump Relative and Jump to Subroutine Relative Instructions – Both the jump relative and jump to subroutine relative instructions provide 8-bit relative address information on execution of the instruction. The 8-bit address value is taken through the input buffer and applied to the JUMP/JSR relative address adder. Note that this adder also receives the content of the DPC. (The four low-order bits come from the DPC while the 11 high-order bits are supplied from the program counter.) The output, then, equals the current DPC address plus/minus the relative address. The relative address is now entered into the Relative Address register by signal DIC8 LD REL REG H. This signal is asserted for loading the register only during execution of the jump relative and JSR relative instructions. Also asserted for these instructions only is the DIC6 PC ADD 01 L signal. This causes the 3:1 multiplexer to select the output of the Relative Address register for entry into the DPC. Loading of the DPC itself is effected by asserting the DIC8 LOAD PC H signal applied to the DPC load control logic.

With the new address now in the DPC (current DPC plus relative address), fetching is re-initiated in the new area of the display file.

The JSR relative instruction has an additional requirement. Prior to entering the new address into the DPC, the old contents of the DPC (updated by 2) is inserted into the stack memory. This is done because the stack memory stores all pertinent status/parameter data (i.e., for the graphic entity instruction executed prior to the JSR) as a prerequisite for POP restore instructions contained at the end of subroutines.

4.2.6.5 Routing During Execution of Jump Absolute and JSR Absolute Instructions – Jump (JMP) absolute and jump to subroutine (JSR) absolute are both two-word instructions where the second word contains the new address within this display file (i.e., where fetching is to resume). The second word is entered into the input buffer immediately after being transferred from the Unibus to the BDB buffer. Signal DIC6 ADD 10 L is asserted during TP2 of the second word processing period. This enables the output of the stack/silo/input buffer multiplexer into the 3:1 multiplexer. After this, signal DIC6 LD PC L is asserted to strobe the branch address into the DPC. (Loading of the DPC occurs because signal DIC8 LOAD PC H is asserted on the second word of both the JMP and JSR instructions.) For the JSR absolute instruction, the original contents of the DPC (updated by 4) are entered into the stack memory. This occurs prior to loading of the absolute address from the 3:1 multiplexer.

4.2.6.6 Routing on Execution of POP Restore Instructions – For POP restore instructions, the DPC address value (entered into the stack memory during JSR instructions) must be re-inserted into the DPC. This is done to return to the area in the display file where fetching occurred prior to execution of the JSR instruction.

During execution of the POP restore instruction, signal DIC6 PC ADD 10 L is asserted to select the output of the stack memory (via the stack/silo/input buffer multiplexer) as the source of the address data to be strobed into the DPC. Actual loading of the DPC occurs due to assertion of the DIC8 RESTORE 0 H signal.

4.2.6.7 Routing on Loading of Relocate Register – The VT48 contains a writable Relocate register for use in those systems exceeding 32K of core memory. The Relocate register is 12 bits in length. However, its outputs are so mated with the DPC that they straddle bit positions 6 through 17 of the DPC. Hence, any change in bit positions 10 or 11 of the Relocate register on loading (or carry developed due to adding lower order bits to bits 6 through 15 of the DPC) effects a change in the contents of bit positions 16 and 17 of the DPC. Consequently, relocating of the display file address anywhere in 128K of memory can be accomplished.

The relocate address is taken from the Unibus to the Relocate register. From there it is applied directly to the DPC.

4.2.7 Graphics Calculation Logic

The graphics calculation logic consists of a microprocessor that initiates and carries out the necessary calculation sequences for display of all graphic entities regardless of type. This logic is contained on three hex PCBs and can be considered to be divided into two parts:

1. Control Logic – This controls the sequencing of the microprogram and provides asynchronous control signals to other major VT48 circuit groups. That is, it informs the vector/character generators when to begin processing. It also informs the display instruction control logic when it may load new delta X/delta Y data and it informs the stack/silo control logic when to update silo memory.

2. Arithmetic Unit (Multiplier/Register Logic) – This logic accepts delta X/delta Y information from the silo/stack control logic and performs all calculations commanded by the microprogram to prepare the graphic entity for display on the VR48 Display Monitor. For vectors, the logic scales, scissors, and calculates the tangent – all preparatory to informing the vector generator to draw the vector. For characters, this logic updates the beam position for each character prior to commanding the character generator to draw the character. An additional function of this logic is to perform all necessary calculations on the occurrence of light pen and edge interrupts.

4.2.8 Control Logic

A block diagram of the graphics calculation control logic is shown in Figure 4-8. A basic breakdown of the processing activities carried out by this logic for a given graphic entity is as follows:

1. Microprogram sequencing startup.
2. Internal microprogram sequencing and branch-on microtest addressing
3. Microprogram sequencing shutdown

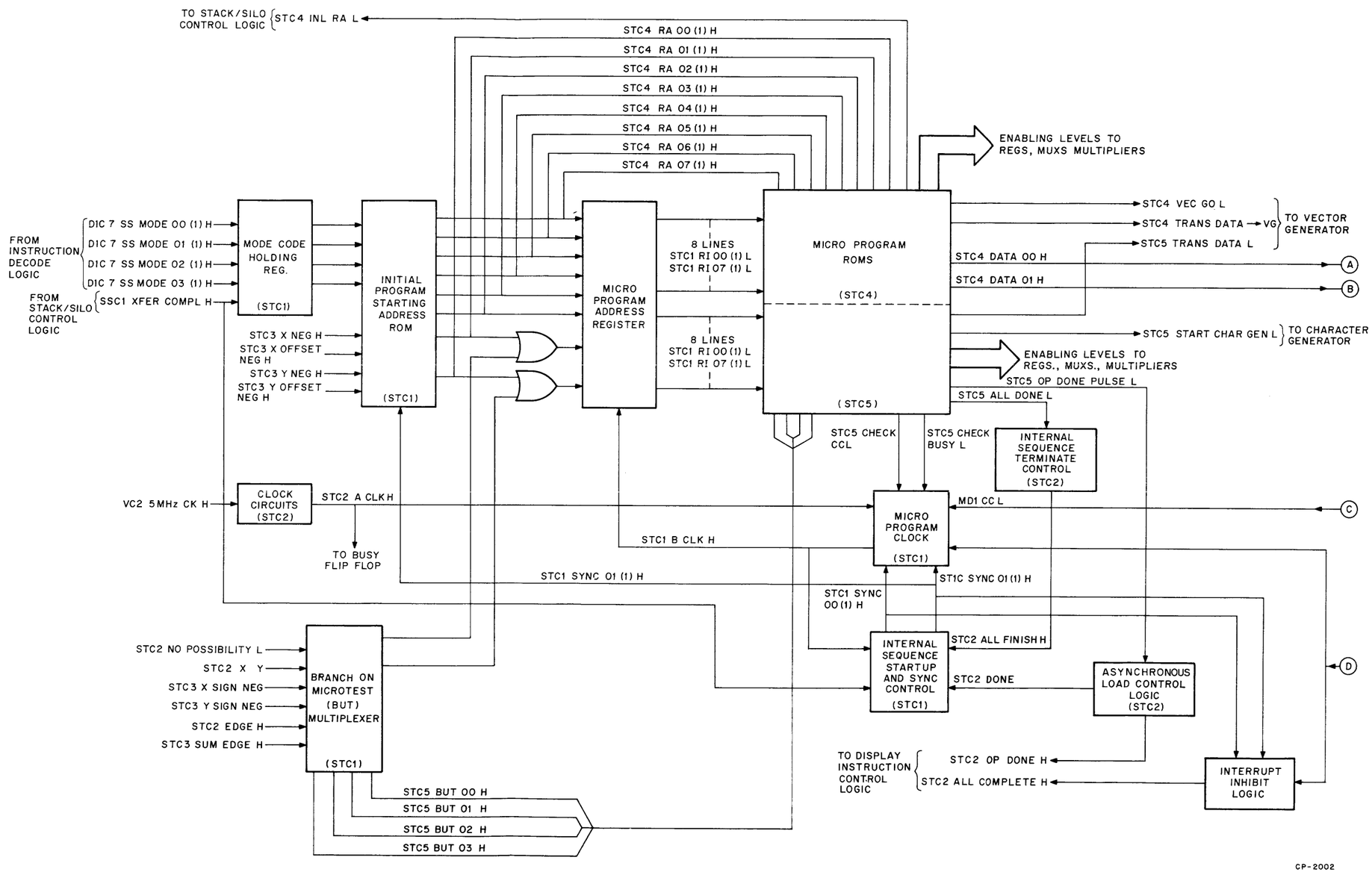


Figure 4-8 Graphics Calculation Control Logic, Block Diagram (Sheet 1 of 2)

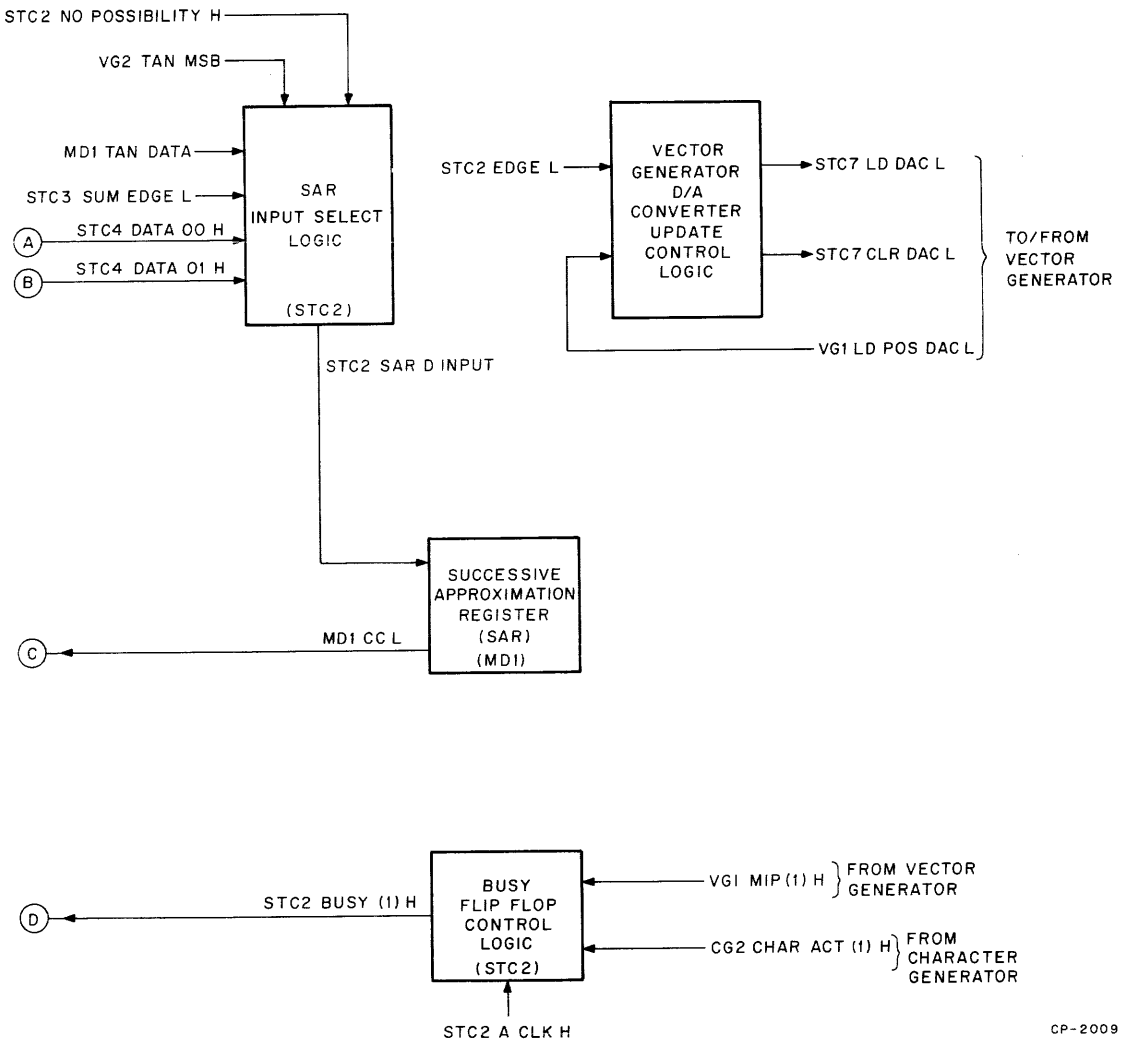


Figure 4-8 Graphics Calculation Control Logic, Block Diagram (Sheet 2 of 2)

4.2.8.1 Microprogram Sequencing Startup – Processing of a graphic entity begins with receipt of the SSC1 XFER COMPL H signal from the stack/silo control logic. This signal is received only after the Delta X/Delta Y registers (Figure 4-1) have been loaded. That is, regardless of the number of words (one or two) making up the graphic instruction. Signal SSC1 XFER COMPL H is applied to two circuit areas as follows:

1. Mode Code Holding Register – Application here is to load the graphic mode code supplied from the instruction decode logic. This code determines the initial starting address where processing is to begin within the microprogram ROMs. That is, the graphic mode code for an absolute point selects one starting address, the code for a character selects another starting address, and the code for a relative long vector selects still another, etc.

2. Internal Sequence Startup and Sync Control Logic – This logic consists of two sequentially arranged flip-flops that are used to control assertion of the initial program starting address ROM output signals and also to govern generation of the clock signals used to sequence through the microprogram. An additional function is to enable the interrupt inhibit logic throughout the graphic entity processing period. That is, by negating signal STC2 ALL COMPLETE H, interrupts (other than edge/light pen related interrupts) are inhibited until processing of the graphic entity is complete.

Application of signal SSC1 XFER COMPL H to the internal sequence startup and sync control logic sets the first of the two sequentially-arranged flip-flops. At this time, signal STC1 SYNC 00 (1) H is asserted while signal STC1 SYNC 01 (1) H remains negated. Continued negation of the latter signal is important, since as long as it is negated, the eight outputs from the initial program starting address ROM are allowed to convey the starting address supplied by the graphic mode code stored in the holding register.

Application of the asserted STC1 SYNC 00 (1) H signal to the microprogram clock allows generation of a single STC1 B CLK H pulse. (This signal is derived from STC2 A CLK H that is produced continuously by the clock circuits.) The leading edge of the STC1 B CLK H signal is used to clock the starting address (specified by the graphic mode code) into the microprogram Address register.

NOTE

The starting address for a given graphic mode code can vary slightly depending on the sign bits of the delta X/delta Y data. The sign bits of the X/Y offset values can similarly alter the starting address. Slight variations in starting addresses are required since it may be necessary to complement one or both delta values (depending on vector direction) prior to adding them to the preceding X/Y position values. Application of the delta sign bits and offset sign bits of the initial program starting address ROMs serves to vary the starting address.

With the starting address now loaded, it becomes necessary to disable the outputs of the initial program starting address ROM. When this is done, sequencing through the microprogram is effected by the RA lines supplied from the microprogram ROMs themselves. The trailing edge of the STC1 B CLK H signal sets the second flip-flop (STC1 SYNC 01 (1) H asserted) in the internal sequence starting and sync control logic. Assertion of the STC1 SYNC 01 (1) H signal inhibits (negates) all outputs from the initial program starting address ROM. This signal remains asserted for the entire period that the graphic entity is being processed through the graphics calculation logic. In this way the control logic switches from the startup to internal sequencing condition.

4.2.8.2 Internal Microprogram Sequencing and Branch-on Microtest Addressing – The microprogram ROMs have a total of 67 outputs. Of these, 12 are used to effect internal sequencing through the microprogram, (i.e., to carry out the necessary operations for processing the graphic entity). The remaining 55 constitute levels that are asserted as part of a calculation process or to command another VT48 circuit area to initiate some processing activity. For example, the microprogram asserts signal STC5 START CHAR GEN L at the end of the character calculation sequence. That is, after the microprogram has scaled the space between the characters (large-scaled characters require greater space than smaller-scaled characters) and updated the X/Y position registers to the starting point of the character about to be drawn by the character generator.

NOTE

Due to space limitations, only some of the major levels that interact with other principle VT48 circuit groups are shown on Figure 4-8. Also shown are those levels that can temporarily halt internal sequencing and those that can shut down sequencing after a graphic entity has been processed.

The 12 microprogram ROM outputs used to effect internal sequencing are divided into two groups as follows:

1. Eight ROM address lines, STC4 RA 00 (1) H through STC4 RA 07 (1) H – With the outputs of the initial program starting address ROM now disabled, these eight lines represent the major address lines feeding the microprogram Address register. Thus, with no other factors intervening, the next address clocked into the microprogram Address register (i.e., by the next STC1 B CLK H signal) is that specified by the ROM address lines and the microprogram now branches to that address.

NOTE

At the second address in the microprogram, a second pattern of enabling levels will be asserted from the ROMs to carry out certain additional activities required by the graphic entity processing sequence. Also, a new set of ROM address levels are asserted to sequence to the next microprogram location.

2. Four branch-on microtest (BUT) lines – These lines are used to address specific conditioning inputs applied to the branch-on microtest (BUT) multiplexer circuits. When a conditioning input is asserted, it modifies the two low-order bits of the RA lines. This means that the BUT lines can be used to sample or detect certain system conditions, and as a result, modify the microbranching address by up to four locations. This feature provides for greater flexibility when different system conditions call for different processing sequences within the microprogram.

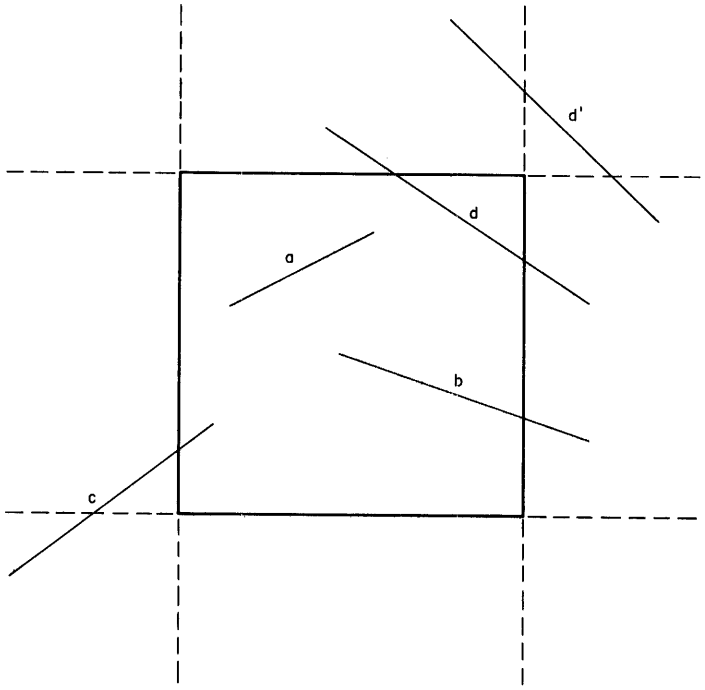
Only a few of the system conditions that can affect the outputs of the branch-on microtest multiplexer are shown on Figure 4-8. The conditions that can call for scissoring a vector are indicated by signals STC3 SUM EDGE H and STC2 EDGE H. Consequently, these signals are sampled by the microprogram to determine in which category the vector falls. These categories are as follows:

1. On-to-on Vector – Start and end points of vector are within display frame. No scissoring required.

2. On-to-off Vector – Start point is within display frame, but end point falls outside display frame. Scissoring required.

3. Off-to-on Vector – Start point is outside display frame, but end point is within display frame. Scissoring required.

4. Off-to-off Vector – Both start and end points are outside display frame. Under these conditions, a segment of the vector may or may not transect the display frame. If it does, scissoring is required; otherwise, no scissoring is necessary. Figure 4-9 summarizes the types of possible vectors.



DESIGNATION	TYPE	SCISSOR	DRAW
a.	ON-TO-ON	NO	YES
b.	ON-TO-OFF	YES	YES
c.	OFF-TO-ON	YES	YES
d.	OFF-TO-OFF	YES	YES
d'.	OFF-TO-OFF	NO	NO

11-4173

Figure 4-9 Possible Drawable and Non-Drawable Vector Types

Signal STC2 EDGE H is asserted whenever the starting point of a vector (about to be processed) is off the display surface. Signal STC3 SUM EDGE H is asserted whenever the end point of the vector is off the display surface. Consequently, these two signals can be used to branch to any one of four possible locations depending on vector type.

The outputs from the branch-on microtest multiplexer affect only the two low-order bits of the 8-bit ROM address. As a result, branching on microtest is limited to four ROM locations.

Every time signal STC1 B CLK H asserts, the next ROM address (as modified by the branch-on microtest outputs) is loaded into the microprogram Address register. In this way, sequencing through the entire microprogram is achieved.

4.2.8.3 Asynchronous Loading of the Next Graphic Entity – Early in each microprogram sequence (i.e., for each graphic entity type), signal STC5 OP DONE PULSE L is asserted. Assertion by the microprogram occurs only after the data (required for the calculation) has been taken from the delta X/delta Y registers and stored in pre-position/holding registers within the arithmetic logic. This means that the delta X/delta Y registers (and also the mode code Holding register) are now free to accept additional data from the stack/silo control and the display instruction control. Signal STC5 OP DONE PULSE L is applied to the asynchronous load control logic which in turn raises the STC2 OP DONE H signal. The latter signal is supplied to the display instruction control logic informing it to process another instruction. Should the next instruction be a long vector, for example, the display instruction control and stack/silo control will proceed to load the delta X/delta Y registers with no adverse effect on graphics calculation processing.

At the same time signal STC2 OP DONE H is asserted to the display instruction control logic, signal STC2 DONE is supplied to the internal sequence startup and sync control logic. The latter signal is used to clear the first of the two sync flip-flops so that signal STC1 SYNC 00 (1) H is no longer asserted. This does not affect generation of clock signals to the microprogram Address register since signal STC1 SYNC 01 (1) H remains asserted.

When the stack/silo control logic has loaded the delta X/delta Y registers, it will issue the transfer complete signal (SSC1 XFER COMPL H) again. Under conditions where the graphic mode code has changed, this has a dual result:

1. A new graphic mode code is loaded into the mode code Holding register. This has no effect, however, since the outputs of the initial starting address program ROM are disabled. (Signal STC1 SYNC 01 (1) H remains asserted).

2. Clocks the first of the two sync flip-flops (in the internal sequence startings and sync control logic) set.

Neither of the preceding actions affect the present (in-process) calculations. They do, however, ready the control logic to re-initiate processing immediately following the last step in the current calculation sequence.

4.2.8.4 Temporary Halts in Microprogram Sequencing – There are two conditions (one internal and one external) that can cause a temporary halt in microprogram sequencing:

1. The Successive Approximation register (SAR) has not finished calculating the intersection between vector and display frame (scissoring), or the vector tangent, or other calculations. Internal to the calculation logic is a Successive Approximation register (SAR) that is used in scissoring, and tangent, as well as other calculations. This circuit module is clocked a fixed number of times before coming up with the answer. Throughout the period that the SAR is being clocked, signal STC5 CHECK CC L is asserted to temporarily inhibit the microprogram clock from loading the next ROM address. When the SAR is finished with its calculation, it issues signal MD1 CC L. This is also applied to the microprogram clock where it is used to re-initiate ROM address loading.

2. At times, the calculation logic may finish all necessary computations before the vector/character generator is finished drawing the preceding graphic entity. Under such conditions, microprogram sequencing must cease until the vector/character generator is no longer busy. Figure 4-8 shows that the busy flip-flop control logic receives two external signals as follows:

   a. Vector generator move in process signal (VG1 MIP (1) H) - When asserted, this indicates that the vector generator is in the process of drawing a vector, and as such, it causes the busy flip-flop to be set.

   b. Character generator character active signal (CG2 CHAR ACT (1) H) - This indicates that the character generator is drawing a character. It, too, sets the busy flip-flop

NOTE

There are also other conditions that set the busy flip-flop as discussed in the paragraphs covering the graphics calculation arithmetic unit.

To determine whether sequencing must be temporarily halted, the microprogram asserts signal STC5 CHECK BUSY L. If the graphics calculation logic has finished processing a vector, the microprogram samples the status of the busy flip-flop to determine if the vector generator is still engaged in drawing the previous vector. If it is, the control logic must now wait before updating silo memory and commanding the vector generator to draw the next vector.

NOTE

The silo memory controls the intensity and blink status for the current graphic entity being drawn by the vector/character generator. At the conclusion of the drawing process, the silo memory is updated to present beam parameters for the next graphic entity.

As long as the busy flip-flop is set, clocking of the next microprogram address into the microprogram Address register is inhibited. When the vector/character generator is finished drawing the vector/character, it lowers its related input signal allowing the busy flip-flop to be clocked reset. This allows the microprogram to resume sequencing.

4.2.8.5 Microprogram Sequencing Shut Down -The microprogram sequence for each graphic type concludes with assertion of the STC5 ALL DONE L signal. Assertion occurs at the last microprogram address in each graphic entity processing routine. That is, after the silo memory has been updated, (assertion of STC4 INC RA L to stack/silo control logic) and when the vector/character generator is being commanded to draw the graphic entity. Application of the STC5 ALL DONE L signal is made to the internal sequence terminate control logic. This logic in turn asserts STC2 ALL FINISH H for application to the internal sequence startup and sync control logic. When the next STC1 B CLK H signal occurs, the second sync flip-flop in this logic is cleared to negate the STC1 SYNC 01 (1) H signal. This readies the control logic to begin immediate processing of the next graphic entity. That is, it enables the outputs of the initial program starting address ROMs. If the next graphic entity has already been loaded, signal STC1 SYNC 00 (1) H is in an asserted state to allow clocking of the microprogram Address register (see discussion on asynchronous loading of the next graphic entity).

4.2.8.6 Successive Approximation Register Input Control – Many of the calculations required to process a graphic entity involves the Successive Approximation register (SAR). Depending on which type of calculation it wishes to initiate, the microprogram must select the proper input source. The microprogram uses signals STC4 DATA 00 H and STC4 DATA 01 H to select any one of four input sources. These input signals and the type of calculations they involve are as follows:

1. MD1 TAN DATA – Selected by the microprogram when it wishes to calculate the tangent of a vector. The calculated result is supplied to the vector generator.

2. STC2 NO POSSIBILITY H – Used in the first part of a scissoring calculation for an OFF to OFF vector that traverses the display frame.

3. STC3 SUM EDGE L – Used in the scissoring calculation for all vectors including that for the second part of an OFF to OFF vector.

4. VG2 TAN MSB – Used in light pen strike calculations.

The single output signal supplied from the SAR input select logic for transfer to the SAR is STC2 SAR D INPUT.

4.2.8.7 Vector Generator D/A Converter Update Control Logic – When the vector generator is finished drawing a vector, the D/A converter used to store the current beam position must be updated in one of two ways:

1. If the end point of the vector is within the display frame, the D/A converters are updated by loading the contents of the X/Y position registers into the vector generator D/A converters.

2. If the end point of the vector is outside the display frame, the D/A converters are cleared so that the beam (in the blanked state) is held at the center of the display frame.

NOTE

Whenever a vector end point falls outside the display frame, the D/A converters are always cleared in anticipation of generating the next drawable vector.

When the vector generator completes drawing a vector, signal VG1 LP POS DAC L switches low. At this time, the D/A converter update control logic samples for the edge condition; i.e., the end point of the vector is outside the display frame. If it is within the display frame, signal STC7 LD DAC L is issued to load the X/Y position values. If the end point is outside the display frame, signal STC7 CLR DAC L is issued to clear the D/A converters.

Both of the preceding signals are also issued during point/character processing (under microprogram control) as is discussed in the paragraphs covering arithmetic unit operation.

4.2.8.8 Control Logic Asynchronous Interactive Control Signals – This paragraph summarizes the asynchronous signals transferred between the control logic and the other major circuit areas of the VT48. Table 4-2 summarizes all asynchronous signals and defines their uses.

Table 4-2 Summary of Graphics Calculation Logic Asynchronous Interactive Control Signals

Signal Name	Purpose
SSC1 XFER COMPL H	Supplied from stack/silo control logic to indicate that another graphic entity has been loaded. This means that new delta X/delta Y values have been loaded into the delta X/delta Y registers. Also, if a set graphic mode instruction was processed (prior to the graphic instruction conveying new delta X/delta Y data), this signal loads a new graphic mode code into the mode code holding register.
STC2 OP DONE H	Informs the display instruction control logic that the delta X/delta Y values have been accepted for processing. Therefore, the display instruction control is free to process the next instruction and have the stack/silo control logic load the next set of delta X/delta Y values into the delta X/delta Y registers.
STC4 INC RA L	Informs the stack/silo control logic to update the silo memory address. This is done to provide the necessary parameter data (blink/intensity level) for the graphic entity about to be drawn by the vector/generator.
STC2 ALL COMPLETE H	Informs the display instruction control logic that the graphics calculation logic, vector generator, and character generator are all in a non-busy status. Therefore, the display instruction control is free to process an interrupt (i.e., an interrupt other than a light pen or display frame edge interrupt).
STC4 TRANS DATA VG	Loads delta length and tangent values into the vector generator prior to commanding it to draw the vector.
STC5 TRANS DATA	Loads parameter data (intensity value/blank status/major and minor axis sign values) into the vector generator prior to commanding the vector generator to draw the vector.
STC4 VEC GO L	Commands the vector generator to start drawing the vector.
STC5 START CHAR GEN L	Commands character generator to start drawing character.
VG1 MIP (1) H	Informs the graphics calculation control logic that the vector generator is busy drawing a vector.
CG2 CHAR ACT (1) H	Informs the graphics calculation control logic that the character generator is busy drawing a character.
VG1 LD POS DAC L	Indicates that the vector generator has completed drawing a vector.
STC7 LD DAC L	Loads the vector generator D/A converters with X/Y position data after completing drawing a vector and also when updating X/Y positions for displayable points/characters.
STC7 CLR DAC L	Clears the vector generator D/A converters when vector end points and points fall outside the display frame.

4.2.9 Graphics Calculation Arithmetic Unit

The paragraphs that follow describe the various calculations carried out by the arithmetic logic.

4.2.9.1 Overview of Arithmetic Operations and Related Flow Drawings – To understand the objectives of the various graphics calculation routines, a knowledge of the relationship of virtual to visable display areas is essential. Also, an understanding of the use of the offset registers is required. Therefore, a reading of the descriptions given in the Operation and Programming section (i.e., the paragraphs following the Image Generation and Manipulation paragraph) is in order before continuing on to the subsequent paragraphs.

All calculation sequences executed by the arithmetic circuits (PCBs M7051 and M7052, Figure 4-10) are illustrated in the 12 sheets making up the 7053 program flow diagrams of the print set. All descriptions given here on the arithmetic circuits follow the same sequence as that depicted in the individual flow diagrams.

The flow diagrams have their own set of abbreviations dictated by the large amount of information appearing on each sheet. In some cases, the abbreviations are similar to those appearing on the logic diagrams. For instances, OP DONE appearing on the flow diagrams indicates assertion of the STC5 OP DONE PULSE L signal on logic diagram STC5. In other instances there is no similarity, as in the case where the flow diagram indicates CLR DIRECTION (meaning to clear the X/Y negative sign flip-flops, STC3) while the signal asserted by the ROMs to carry out the function is STC4 CLR DATA H. Table 4-3 provides a cross reference and meanings of abbreviations appearing on the flow diagrams and the related mnemonics appearing on the logic prints.

Individual signals asserted at each ROM address are indicated on the M7053 Data ROM pattern drawings of the print set. Reference to these drawings is necessary to determine how the data is being routed and steered through the various registers, multiplexers, and adders for each graphics calculation sequence.

The range of control signals used to alter branch addresses within the various microprogram sequences are shown in Figure 4-11. This illustration is presented as a convenient reference when analyzing the various microprogram sequences within the print set flowcharts. Signals asserted at integrated circuit BUT MUX 1 affect the LSB of the microprogram address. Similarly, signals asserted at integrated circuit BUT MUX 2 affect the content of the second LSB of the microprogram address. (See drawing STC1 in the print set.)

A comparison of branching addresses for the four types of relative long vectors indicates how the branch on microtest multiplexing logic (BUT MUX) selects one branch address versus another. After loading the pre-position registers, etc., (at the start of the relative long vector processing sequence), the microprogram flow is directed to go to 067_8 BUT 00. This means that the next address in the microprogram is 067_8 as modified by the signals applied to the 00 inputs on Figure 4-11. (That is, signals STC3 SUM EDGE H and STC2 EDGE H.) A comparison of how these signals alter the selected branch address is shown below:

Rel Vector Type	STC3 SUM EDGE H	STC2 EDGE H	Microprogram Start Address
ON-to-ON	L	L	67_8
ON-to-OFF	H	L	66_8
ON-to-ON	L	H	65_8
OFF-to-OFF	H	H	64_8

When the beam is currently on screen and the processed delta X/Y values indicate that the beam will remain on screen, then neither signal STC3 SUM EDGE H nor STC2 EDGE H is asserted. Therefore,

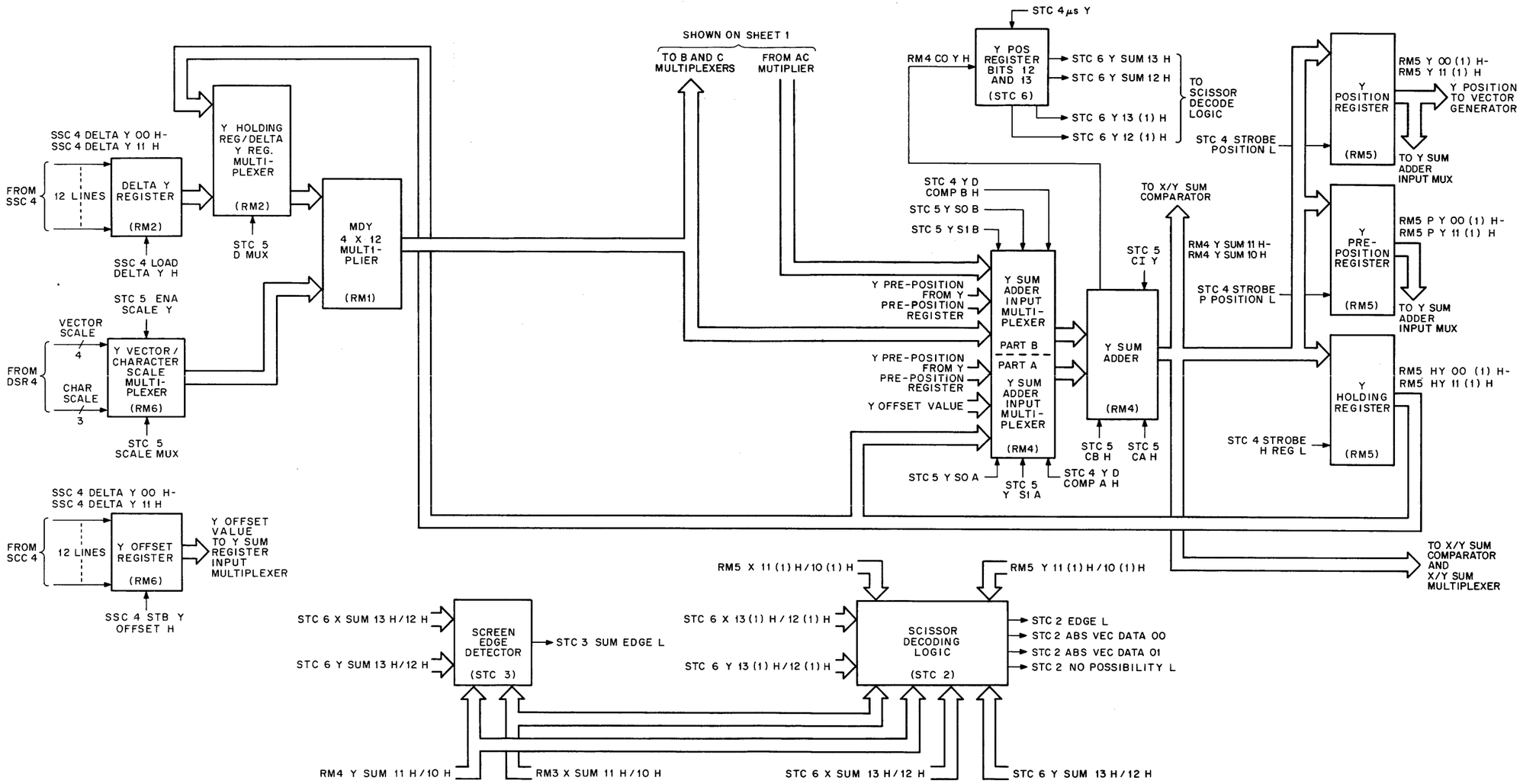


Figure 4-10 Graphics Calculation Logic Arithmetic Unit (Sheet 1 of 2)

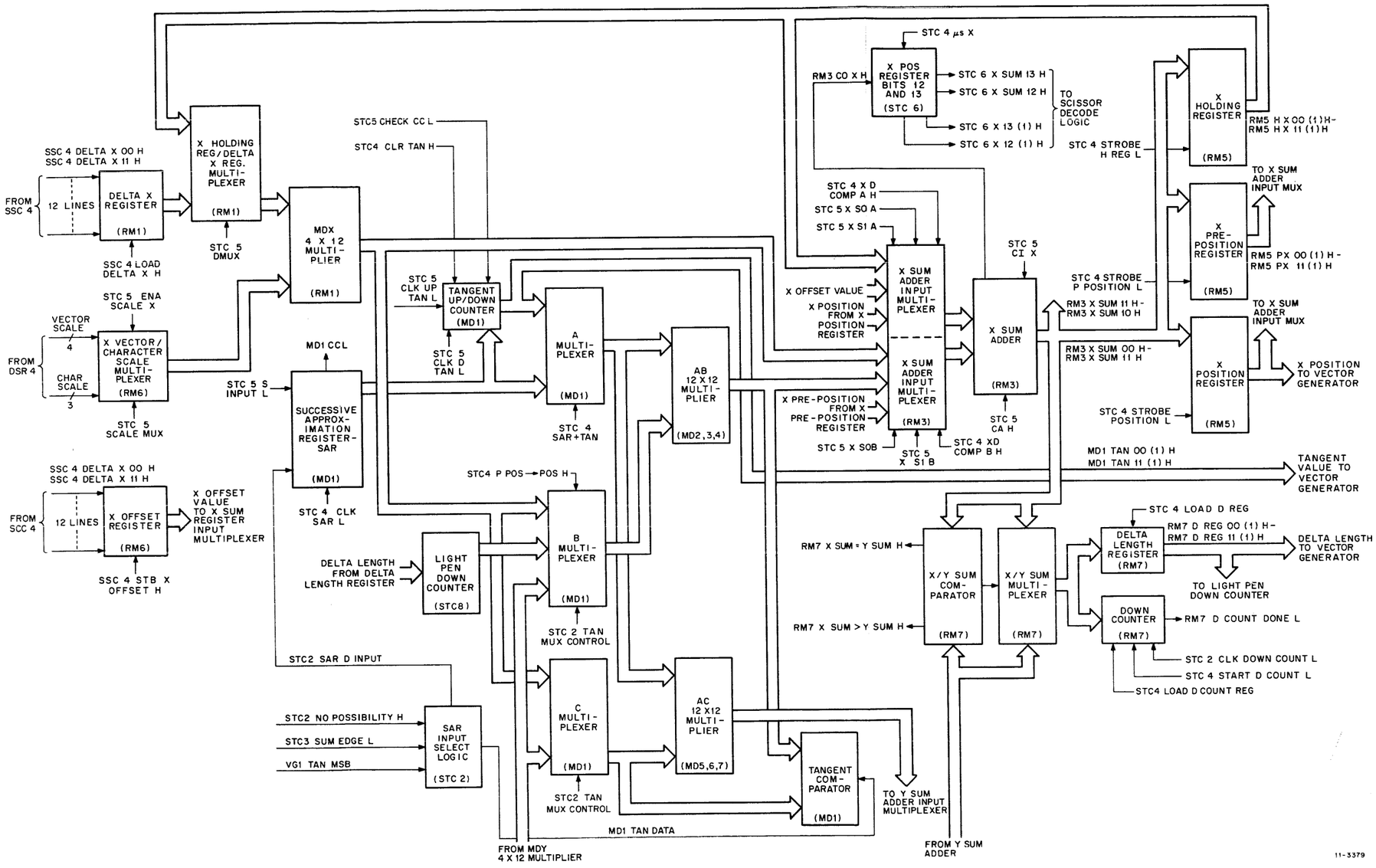


Figure 4-10 Graphics Calculation Logic Arithmetic Unit (Sheet 2 of 2)

Table 4-3 Arithmetic Unit Flowchart Legend

Abbreviation/Mnemonic (Logic Print Mnemonic)	Meaning
X (RMS X 00 (1) H - RM5 X 11 (1) H)	X Position register contents
Y (RM5 Y 00 (1) H - YH (1) H)	Y Position registers contents
ΔX (SSC4 DELTA X 00 H - SSC4 DELTA X 11 H)	X Value entered into Delta X register
ΔY (SSC4 DELTA Y 00 H - SSC4 DELTA Y 11 H)	Y Value entered into Delta Y register
SCALE (DSR4 VEC SCALE 00 (1) H - DSR4 VEC SCALE 03 (1) H/ DSR6 CHAR SCALE 00 (1) H - DSR6 CHAR SCALE 03 (1) H)	Vector scale or character scale Factor applied to X and Y vector/ character scale multiplexers
PX (RM5 PX 00 (1) H - RM5 PX 11 (1) H)	X Pre-position register contents
PY (RM5 PY 00 (1) H - RM5 PY 01 (1) H)	Y Pre-position register contents
SAR	Successive Approximation register
TAN (MD1 TAN 00 (1) H - MD1 TAN 11 (1) H)	Tangent UP/DOWN counter
HX (RM5 HX 00 (1) H - RM5 HX 11 (1) H)	X Holding register contents
HY (RM5 HY 00 (1) H - RM5 HY 11 (1) H	Y Holding register contents
LPDC (STC5 LPDC 00 (1) H - STC5 LPDC 11 (1) H)	Light pen down counter
INC RA (STC4 INC RA L)	Increment silo memory read address
TRANS DATA (STC5 TRANS DATA L/ STC4 TRANS DATA VGL)	Load delta length and tangent, etc., values into vector generator
START VEC (STC4 VEC GO L)	Commands vector generator to draw vector
OP DONE (STC5 OP DONE PULSE L)	Informs display instruction control to process next instruction

Table 4-3 Arithmetic Unit Flowchart Legend (Cont)

Abbreviation/Mnemonic (Logic Print Mnemonic)	Meaning
ALL DONE (STC5 ALL DONE L)	Indicates ROM sequence complete.
LOAD D COUNT (STC4 LOAD D COUNT REG L)	Load down counter
FIND TAN	Calculate tangent process using SAR
FIND EDGE	Calculate edge transition point
CHECK CC (STC5 CHECK CC L)	Check conversion complete (SAR is finished calculating factor or tangent)
CLK-D FAC (STC5 CLK D TAN L)	Clock down factor
FAC	SAR multiplication factor (fraction)
MDX	Multiplied delta X
MDY	Multiplied delta Y
LOAD EDGE IND	Load edge indicator flip-flop to indicate beam has traversed edge
CLR DIRECTION	Clear X and Y sign negative flip-flops (STC3)
BUT	Branch on microtest
CONTROL (DSR6 CHAR INHIBIT LEVEL L)	Control character
CR (DSR6 CR H)	Carriage return
CHAR IN PROG (CG2 CHAR ACT (1) H)	Character in progress
SILO RESYNC	Synchronize silo memory at read/write starting addresses following interrupt processing
MA	Major axis
MI	Minor axis
EDGE FLAG (STC5 EDGE FLAG H)	Sent to read status multiplexer to indicate beam edge transition

Table 4-3 Arithmetic Unit Flowchart Legend (Cont)

Abbreviation/Mnemonic (Logic Print Mnemonic)	Meaning
INTER (STC4 INTER PULSE L)	Set interrupt flag
START D COUNT (STC4 START D COUNT L)	Start down count within down counter
CLR TAN (STC5 CLR TAN H)	Clear tangent UP/DOWN counter
CLR SAR (STC5 INPUT L)	Clear successive approximation register
CMP X TO Y	Compare outputs of X/Y sum adders
TAN → ANALOG	Reload tangent into vector generator during light pen hit processing
LPD COUNT	Light pen down counter
LOAD DAC (STC4 LOAD DAC L)	Load vector generator D/A converter
1770 → D COUNT	Load 1770_8 into down counter
CONTROL & CR	Control character and carriage return
ROTATE	Rotate character on CRT
CHECK BUSY (STC5 CHECK BUSY L)	Check status of busy flip-flop
LOAD Δ REG (STC4 LOAD D REG L)	Load Delta Length register
Δ REG → LPDC (STC5 LOAD LPD COUNT L)	Transfer contents of Delta Length Register to light pen down counter
1000 → D COUNT	Load 1000_8 into down counter
XSUM	X SUM adder
YSUM	Y SUM adder
SUM EDGE (STC3 SUM EDGE H is asserted)	Addressed point/end point is off screen

Table 4-3 Arithmetic Unit Flowchart Legend (Cont)

Abbreviation/Mnemonic (Logic Print Mnemonic)	Meaning
SUM EDGE (STC3 SUM EDGE H is negated)	Addressed point/end point is on screen

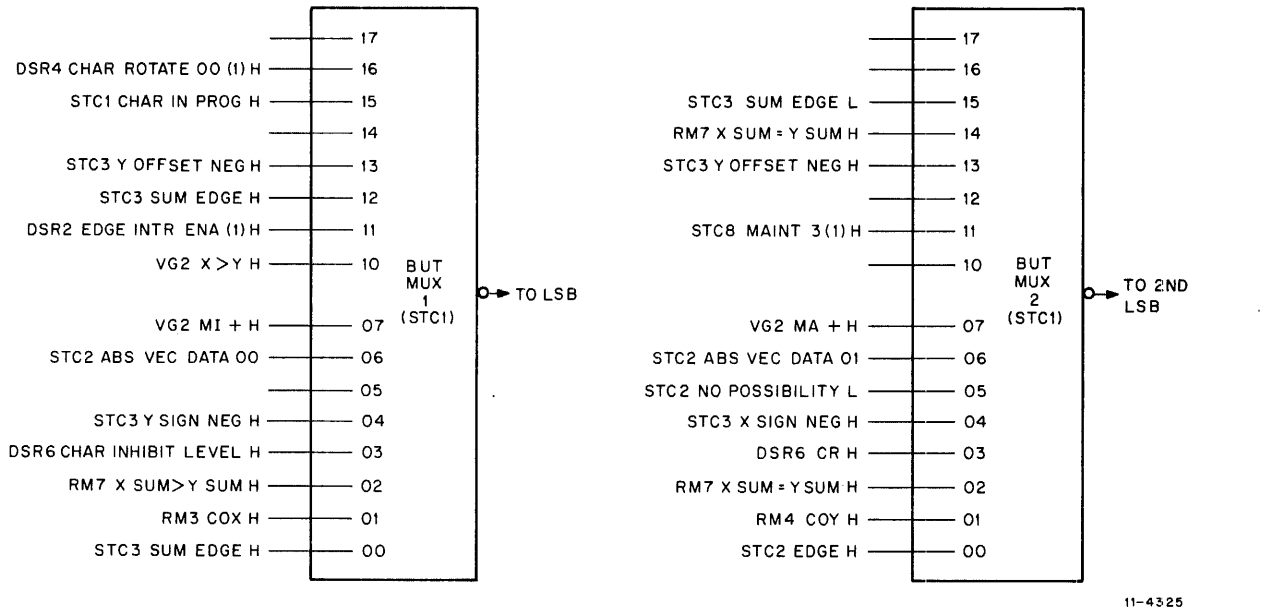


Figure 4-11 Branch on Microtest Steering Signals

both BUT MUX circuits on Figure 4-11 issue 1 or high outputs and consequently the 067_8 starting address ($110\ 111_2$) is selected.

When signal STC3 SUM EDGE H is asserted while signal STC2 EDGE H is negated, it means that the vector starting point is on screen, but the ending point is off screen. Consequently, the branch address is altered to select the microprogram address of the ON-to-OFF vector. That is, address 066_8 or $110\ 110_2$.

A third set of circumstances exists when signal STC2 EDGE H is asserted while signal STC3 SUM EDGE is not. This means that the vector starting point (as stored by the X and Y position registers) is off screen but the ending point is on screen. Hence, the OFF-to-ON vector starting address (065_8 or $110\ 101_2$) is selected.

The final set of conditions occurs when the vector starting point and vector ending point are both off screen. Here, both input signals to the BUT MUX logic are asserted. Therefore, the OFF-to-OFF branch address (064_8) is selected.

Other signals applied to other input pins on the branch on microtest multiplexing circuits produce similar effects. That is, they alter the branch address consistent with the conditions prevalent within the graphics calculation logic.

4.2.10 Absolute Point Processing

The microprogram sequence for processing absolute point instructions is given on sheet 1 of the M7053 Program Flow Diagram (see print set). The sequence can begin at any one of 16 (20_8) starting addresses as governed by the sign bits of the delta X/Y values and the X/Y offset registers. With all sign bits positive, the sequence is initiated at 001_8. If the delta X/Y sign bits are positive, and both offset register signs are negative, the starting address is steered to location 004_8. Further, if the delta X/Y signs are both negative and the offset register sign bits are both negative, then a starting address of 015_8 is selected.

The processing activities carried out at each of the 16 starting addresses is basically the same. First, the addressed end point coordinate values are entered into their respective pre-position registers (Figure 4-10). The processing sequence for the X coordinate is now described.

The delta X value is multiplied by the scale factor and then added to/subtracted from the X offset value. The result is entered into the X Pre-Position register. The value now contained in this register represents the intended point location or beam destination in the X coordinate. All steps in this processing activity are represented by the expression (ΔX) (SCALE) $\pm$ X OFF $\rightarrow$ PX (sheet 1 of the flow diagram). Signals asserted by the microprogram to effect this arithmetic operation are shown at address 001_8 on the M7053 Data ROM Pattern Prints.

Multiplication of the delta X value by the scale factor occurs in the MDX 4 $\times$ 12 multiplier. Signals STC5 ENA SCALE L and STC5 SCALE MUX are both asserted to select the vector scale factor. Also, signal STC5 D MUX is asserted to select the delta X value rather than the output of the X Holding register. The product of the delta X and scale factor is now applied to the lower half of the X sum adder input multiplexer. Similarly, the content of the X Offset register is applied to the upper half of this same multiplexer. The microprogram selects these two sets of inputs for transfer through the multiplexer to the X sum adder. (Signal STC5 X S0A is negated and signal STC5 X S1A is asserted to select the output of the X Offset register. Selection of the multiplied delta X, MDX, is effected by asserting signal STC5 X S0B and negating signal STC5 X S1B.) The addition/subtraction of the two values (MDX and X offset) now takes place in the X sum adder. The product output is then strobed into the X Pre-Position register through assertion of signal STC4 STROBE P POSITION L. Processing of the Y delta value times scale factor etc., is the same as that described for the X coordinate.

With the pre-position registers, now loaded (actually, loading occurs at the trailing edge of the STC4 STROBE P POSITION L signal), the OP DONE signal (STC2 OP DONE H) is asserted to inform the display instruction control logic that it can now initiate loading the next set of delta X/Y values into the delta X/Y registers. Also, at this time, the next address in the microprogram sequence is asserted.

In most cases, the microprogram address is steered to ROM address 047_8 BUT 00. Exceptions occur when the arithmetic logic must add two negative values. For example, from ROM address 013_8, the microprogram branches to address 042_8 because both the delta X and X offset values are negative. At this latter location, the sole processing activity is to add 1 to the contents of the X Pre-Position register. The reason for this can be seen in terms of an example where the beam address in the X coordinate is -0377_8 and the negative offset value is 0100_8. In such a case, the resultant value eventually entered into the X Position register should be 7301_8 (i.e., 0000_8 minus 0477_8). However, the arithmetic operation

performed at ROM address 013_8 results in the product being in error by one LSB as shown in the following example.

ABS PT ΔX	000 011 111 111
COMPL ABS PT ΔX	111 100 000 000
X OFFSET	000 001 000 000
COMPL X OFFSET	111 110 111 111
COMPL ABS PT ΔX	111 100 000 000
+COMPL X OFFSET	111 110 111 111
+Two's COMPL (STC5 CIX ASSERTED)	1
ANS	111 011 000 000

(The carry out (RMS C0X H) occurring at the X sum adder MSB also sets the two high order bits (12 and 13) of the X Position register.)

The assertion of signal STC5 CIX (carry in X) provides the two's complement of only one of the two numbers to be added. As a result, the product, 7300_8, is off by one LSB. The operation carried out at ROM address 042_8 adds one to the contents of the X Pre-Position register in the following way:

X PRE-POSITION	111 011 000 000
ASSERT STC5 CIX	1
ANS.	111 011 000 001

Following this operation, the value contained in the X Pre-Position register is the correct X coordinate value.

When the microprogram branches to location 047_8 BUT 00, the determination is made whether the point is on or off screen (Paragraph 4.2.8.2). If the last processed point was on screen and the in-process point is on screen then the operations at address 047_8 are carried out. Similarly, if the last processed point were off screen and the in-process point is on screen, the branch address is altered to location 045_8 and the processing functions at this ROM address are carried out.

4.2.10.1 ON-to-ON Point Processing – When processing points in the ON-to-ON category, the down counter (Figure 4-10) is used for a time-out function. The objective when loading the down counter is to enter a value equivalent to the time required to deflect the beam to its new addressed position. During the period that the down counter is counting, the busy flip-flop (STC2 BUSY (1) H) is set. At the conclusion of the down count period, the busy flip-flop is cleared and the point intensity flip-flop is set (STC2 POINT INTENSITY H).

A problem arises in selecting which coordinate value (X or Y) is to be loaded into the down counter. The desired value is the larger of the two since this represents the greater distance over which the beam must be deflected. The X/Y sum comparator logic is used together with detection of carry out bits at the sum adder MSBs to ensure that the correct value is loaded into the down counter.

The first processing activity at ROM location 047_8 is to subtract the contents of the X register from the X Pre-Position register. (Similar activity occurs for the respective Y registers.) If the value contained in the X Pre-Position register is greater than in the X Position register, then a carry results. (Signal RM3 COX H is asserted.)

NOTE

Whenever a smaller number is subtracted from one larger, a carry results. Conversely, whenever a larger number is subtracted from one smaller, no carry results.

After the subtraction process takes place, the busy flip-flop status is checked. If set, it means that the down counter is busy timing out a previous point/vector. Consequently, the microprogram must halt temporarily until the flip-flop is cleared. Once the busy flip-flop is no longer set (RM7 D COUNT DONE L asserts to toggle graphic-operation-in-progress flip-flop, STC2, and clear busy), the down counter is loaded with the larger (X/Y) value. This occurs via the X/Y sum comparator, and X/Y sum multiplexer.

After loading the down counter, the microprogram branches to location 057_8 BUT 01. Note that the input signals being sampled at the BUT 01 locations are the carry out signals from the X/Y sum adders. If carries occurred at the MSB of both adders, it means that the values contained in both X/Y pre-position registers exceeded those contained in the X/Y position registers. It also means that the beam is to be moved in a positive direction in both coordinates.

Under these conditions, it can further be said that the correct value is now contained in the down counter. That is, the larger value produced at ROM location 047_8 and detected by the X/Y sum comparator logic. Hence, when both carry out signals are asserted (RM3 COX H and RM4 COY H), the microprogram branches to location 054_8 and no further subtractions are necessary.

When the value in either pre-position register (or both) is smaller than that contained in the corresponding Position register, then an additional calculation is necessary. The reason for this can be seen in terms of an example where the beam is to be moved 0400_8 units to the left (X coordinate) i.e., from 1577_8 to 1177_8. Assume further that the movement in Y is negligible and that, therefore, the desired

value to be entered into the down counter is 0400_8. In such a case, the subtraction process occurring at ROM location 047_8 produces the following results.

X PRE POS (1177_8)	001 001 111 111
X POS (1577_8)	001 101 111 111
2's COMPL X POS	110 010 000 001
X PRE POS	001 001 111 111
2's COMPL X POS	110 010 000 001
ANS	111 100 000 000

The result of this arithmetic operation produces a product of 7400_8, a value far greater than that required to time out the deflection in the X axis. Note also that no carry results from the operation. At ROM location 055_8, the arithmetic operation is now reversed so that the contents of the X Pre-Position register are subtracted from the contents of the X position registers. This subtraction produces the following results.

X POS (1577_8)	001 101 111 111
X PRE POS (1177_8)	001 001 111 111
2's COMPL X PRE POS	110 110 000 001
X POS	001 101 111 111
2's COMPL X PRE POS	110 110 000 001
ANS	000 100 000 000

The value obtained by the subtraction occurring at ROM location 055_8 is the correct value (0400_8) since it represents the actual amount that the beam must be deflected in the X axis. Hence, at this location, the down counter is loaded again, that is, with the correct value superseding the erroneous value loaded at location 047_8.

From ROM location 055_8, the microprogram now branches to location 054_8. (Similar branching occurs from locations 056_8 and 057_8 at times when their respective processing conditions prevail.) At this location, the X and Y position registers are loaded with the X/Y coordinate values of the point about to be processed. The microprogram now checks the busy flip-flop. This is necessary prior to incrementing the silo memory read address (INC RA, STC4 INC RA L). In other words, if the vector generator still indicates move in process (VGI MIP (1) H is asserted), the microprogram must wait

until the vector generator is finished to update the silo. Otherwise, the intensity level might be changed in the middle of processing a preceding vector/character.

Once the busy flip-flop is cleared, the silo memory read address is incremented and the microprogram branches to ROM address 326_8. Here, two things occur:

1. The D/A converters in the vector generator are loaded with the X/Y Position register values (LOAD DAC, STC7 LD DAC L). At this time then, the D/A converters begin resolving the analog output voltage necessary to deflect the beam to the addressed position.

2. The TRANS DATA signal (STC5 TRANS DATA L) is asserted to convey intensity level and menu area select parameters to the vector generator.

The microprogram now branches to ROM address 142_8 where the down counter is started (STC4 START D COUNT L asserts) to time out the VR48 beam deflection period. Starting the down counter also sets the busy flip-flop. The microprocessor now branches to 104_8 where the ALL DONE (STC5 ALL DONE L) signal is asserted (Paragraph 4.2.8.5). The microprogram now branches to location 374_8 in anticipation of the next graphic entity (Paragraph 4.2.8.5) or a light pen strike.

4.2.10.2 ON-to-OFF Point Processing – The ON-to-OFF point processing sequence is entered at location 046_8. This occurs because the STC3 SUM EDGE H signal asserts to alter the ROM branch address from 047_8 to 046_8. At the latter address, the contents of the X and Y pre-position registers are entered into the respective position registers.

NOTE

The X/Y coordinate values of all off screen points are always entered into the X/Y position registers. This is necessary because at some future time in the display file, a point or vector will be processed to return the beam to an on screen status. Hence, storing off screen points, if necessary, to determine from what direction the beam is being returned on screen.

Following the loading of the position registers, the busy flip-flop is checked to determine if the preceding graphic entity has finished its processing cycle. If so, the silo memory address is updated (STC4 INC RA L) and the edge indicator flip-flop (STC3 EDGE INDICATOR (1) H) is set.

The microprogram now branches to location 060_8 where the following occurs:

1. The D/A converters in the vector generator are cleared (STC4 CLR DAC L asserts). This is done to position the beam (in a blanked status) at the center of the screen. That is, at an X/Y address of $1000_8/1000_8$. By doing this, the beam is now in optimum position to be deflected to the next on screen point/vector in the display file.

2. The microprogram also asserts the TRANS DATA and ALL DONE signals to complete the processing cycle. The microprogram now branches to location 374_8 in anticipation of the next graphic entity (Paragraph 4.2.8.5) or light pen strike.

4.2.10.3 OFF-to-ON Point Processing – The microprogram sequence for off screen-to-on screen points is undertaken at ROM address 045_8. A significant thing to remember about processing points in this category is that the vector generator D/A converters are in a cleared state (see preceding paragraph). Consequently, when the beam is returned to an on-screen location, deflection of the beam to the addressed point is from center screen (X = 1000_8, Y = 1000_8). And it is the difference between

1000_8 and the coordinate point of the greater axis that must be loaded into the down counter to time out the deflection period.

At this point in the microprogram (045_8) it is not known where on the screen the beam is being returned. Therefore, the initial processing activity is to subtract 1000_8 from the contents of both pre-position registers. This in effect tests the contents of the pre-position registers to see if the addressed point falls in the upper right quadrant of the screen, Figure 4-12. Note that in this quadrant, all X, Y address values exceed 1000_8. And since subtraction of a smaller number from a larger number always produces a carry, the generation of carries from both adders indicates that the beam address is in the upper right quadrant. Similarly, if no carries result, the beam address must be in the lower left quadrant. That is, at a location where both values are less than 1000_8.

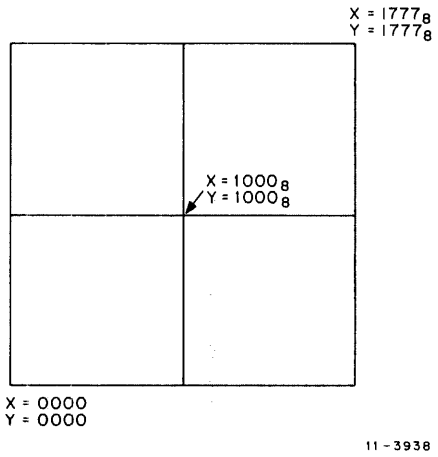


Figure 4-12 VR48 Monitor Beam Address Values

After subtracting 1000_8 from the contents of both pre-position registers, the microprogram checks the busy flip-flop to ensure that the down counter is clear and ready to accept a new time out value. Once the busy flip-flop is cleared, the down counter is loaded with the larger of the two coordinate values; that is as detected by the X/Y sum comparator.

NOTE

The value now entered in the down counter remains there only if carries are detected at the outputs of both adders during the ROM location 045_8 subtraction operation.

The microprogram now branches to location 053_8 BUT 01. The 01_8 location of the BUT multiplexer samples for carry conditions and alters the ROM address accordingly. When the beam address is in the upper right quadrant, both carries occur and the sequence branches directly to location 050_8. If either or both carries fail to occur, it means that the beam is in one of the other three quadrants as follows:

1. Carry in X but none in Y – Beam is in lower right quadrant.
2. Carry in Y but none in X – Beam is in lower left quadrant.
3. Neither carry – Beam is in lower left quadrant.

For case 3, the activities at ROM address 053_8 must be carried out. These are:

1. The contents of the pre-position registers are subtracted from 1000_8 to find the larger distance (X or Y) over which the beam must be deflected.

2. The larger value (detected by the X/Y sum comparator) is loaded into the down counter. The microprogram then branches to location 050_8 unconditionally.

Analyses of the processing activities at ROM locations 052_8 and 051_8 show subtraction processes applicable to the particular quadrant wherein the on-screen beam position falls. That is, at location 052_8 (lower right quadrant), the contents of the Y Pre-Position register are subtracted from 1000_8. At location 051_8, the contents of the X Pre-Position register are subtracted from 1000_8. In both cases (after loading the larger value into the down counter) an unconditional branch to 050_8 is made to continue ON-to-OFF point processing.

At the latter location, the on screen beam address values are entered into the X/Y position registers provided that the busy flip-flop (STC2 BUS (1) H) is cleared. (When cleared it indicates that the vector/character generator is not currently engaged in drawing a graphic entity). Also, with the busy flip-flop cleared, the microprogram can now update the silo memory (INC RA). The remaining task at location 050_8 is to load the edge indicator flip-flop to indicate that the beam has traversed the edge of the working surface and is now on screen. The remaining processing functions (ROM location 326_8, etc.) have already been described in the ON-to-ON point processing paragraph.

4.2.10.4 OFF-to-OFF Point Processing – When an absolute point off screen is followed by another off screen point, processing activities are minimal. These activities are effected at ROM location 044_8 and consist of:

1. Loading the X/Y position registers with the new off screen point values.

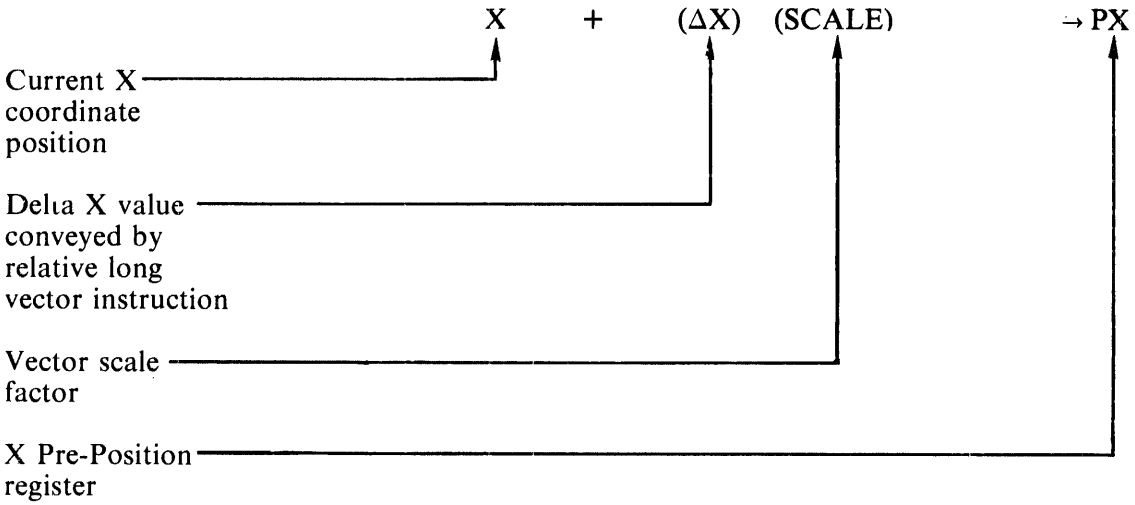
2. Checking the busy flip-flop to ensure that the vector/character generator is free to accept new silo memory data (intensity/menu).

3. Incrementing the silo memory read address (INC RA).

4.2.11 Relative Vector Processing

The microprogram sequences for all types of relative vectors are shown on sheets 2 through 5 of the M7053 Program Flow Diagram. At the top of each sheet, the processing activities occurring on entry into each ROM starting address are repeated. Each of the four possible starting addresses (031_8 through 034_8) is selected on the basis of the sign bits of the delta X/Y values. That is, if both delta values are positive, then 031_8 is the starting address. If delta X is positive and delta Y is negative, then microprogram sequencing is begun at ROM address 032_8, etc. (Paragraph 4.2.8.1).

At all four starting addresses, the objective is the same, that is, to load the addressed end points of the relative long vector into the X/Y pre-position registers. This can be seen from the expressions appearing in the block of location 031_8. The expression for the X coordinate can be explained as shown below:



To accomplish this arithmetic operation, the current X position (stored in the X Position register) is routed to the X sum input multiplexer, part A. The delta X value is multiplied by the scale factor in the MDX 4 × 12 multiplier. The product is then routed to the X sum input multiplexer, part B. The multiplexer input selection signals asserted at ROM location 031_8 are as shown below:

SELECTION SIGNALS

STC5XS1A	STC5XS0A	SELECTED SOURCE
0	1	X POSITION REG
STC5XS1B	**STC5XS0B**	**SELECTED SOURCE**
0	1	MULTIPLIED DELTA X (MDX)

The product now appearing at the output of the X sum adder represents the addressed end point in the X coordinate. Signal STC4 STROBE P POSITION L is also asserted at ROM location 031_8. The trailing edge of this signal (occurring when the microprogram sequences to the next location) strobes the addressed end point into the X Pre-Position register.

NOTE

The pre-position registers provide convenient temporary storage. Pre-position registers are necessary because the vector generator may currently be drawing a previously calculated vector. In such a case, the X/Y position registers currently contain the addressed end point of that vector. In a sense then, it can be said that the X/Y position registers are currently busy because the vector generator D/A converters are not updated (i.e., to the addressed end point) until the vector generator is through drawing the vector.

The Y coordinate of the addressed end point is entered into the Y Pre-Position register in a way similar to that just described. Other activities occurring at location 031_8 are housekeeping operations necessary for subsequent calculations in processing all vector types. These activities are:

1. Clearing the Successive Approximation register (SAR)
2. Clearing the tangent up/down counter.

When the sign of one or both delta values is negative, it is necessary to subtract to arrive at the addressed vector end point. For example, if delta Y is a negative value (while delta X is positive) the operation for the Y coordinate at ROM location 032_8 is

$$Y - (\Delta Y)\,(SCALE) \rightarrow PY$$

In such a case, the arithmetic operation is basically the same as that just described for X. The exception lies in the fact that the output of the Y Position register is complemented (STC4 Y D COMP A H is asserted) before the addition operation takes place.

Once the pre-position registers have been loaded and the registers cleared, the microprogram branches to location 067_8 BUT 00_8. It is here that the determination is made as to vector type by sampling the status of the STC3 SUM EDGE H and STC2 EDGE H signals (Paragraph 4.2.8.2). Branching by vector type is to the following locations.

1. ON-to-ON, 067_8
2. ON-to-OFF, 066_8
3. OFF-to-ON, 065_8
4. OFF-to-OFF, 064_8

The flow sequence now continues on separate sheets of the flow diagram as described in the subsequent paragraphs.

4.2.11.1 ON-to-ON Vector Processing – When neither signal STC3 SUM EDGE H nor signal STC2 EDGE H is asserted at the relative long vector starting address, it indicates that both the vector start point and end point are on screen. Therefore, the vector is an ON-to-ON vector. The microprogram sequence for this vector is shown on sheet 2 of the 7053 program flow diagram.

Two principal calculations are carried out as part of the ON-to-ON vector processing sequence. These are:

1. Determine delta length.
2. Calculate tangent.

Both these values are required by the vector generator to draw the vector (see discussion on vector generator).

At ROM location 067_8, the dela X/Y values are multiplied and then placed in the respective holding registers. At this time, then, the holding registers contain the delta X/Y magnitudes. These are to be used later in the tangent calculation. Loading of the Delta Length register also occurs at ROM location 067_8. The delta length is the larger of the delta X/Y values exiting the X/Y sum adders. The larger value is detected by the X/Y sum comparator, which, in turn, enables the proper set of input lines to the X/Y sum multiplexer. From the latter circuit, the delta length is strobed into the Delta Length register (signal STC4 LOAD D REG asserts). At this time then, one of the two calculations necessary for ON-to-ON vector processing is completed.

Signal STC5 OP DONE PULSE L is also asserted at this ROM location to inform the display instruction control to begin processing the next instruction.

The microprogram now branches to one of three locations to begin the tangent calculation process.

4.2.11.2 Tangent Calculation – The tangent calculation process can begin at any of three locations because there are three basic possibilities regarding the vector to be drawn:

1. The delta X/Y lengths are equal (RM7 × SUM = Y SUM H asserts). This is the case when the vector is to be drawn at a 45-degree angle (i.e., 45-degree, 135-degree, etc.). In such a case, the tangent up/down counter is loaded with all 1s. That is, a value equal to the tangent of a 45-degree angle.

2. The delta X value exceeds delta Y. (Signal RM7 × SUM < Y SUM H is asserted.)

3. The delta Y value is greater than delta X. (Signal RM7 × SUM > Y SUM H is negated.)

The miroprogram samples the input signals applied to locations BUT 02_8 to determine which branch location (063_8, 062_8, or 061_8) is appropriate to the vector to be drawn. If the vector is to be drawn at a 45-degree angle, (ROM location 061_8) the tangent up/down counter which was cleared at the relative long vector starting address is now clocked down by 1. This has the effect of loading it to all 1s because decrementing an up/down counter already in a cleared state forces an all 1s output. Hence all 1s are sent to the vector generator just prior to asserting the STC4 VEC GO L signal.

NOTE

At location 061_8, the microprogram also checks to make sure that the delta lengths are not equal to zero – a situation that could occur if a relative long vector instruction were executed with the delta X/Y fields equal to zero. The check is made by bringing the contents of the Y Holding register through the Y sum adder while at the same time bringing all zeros through the X sum adder. If signal RM7 × SUM = Y SUM H (from the X/Y sum comparator) asserts, it means the delta X/Y values are equal to zero. In such a case, the microprogram sequence is terminated at ROM location 071_8. If RM7 × SUM = Y SUM H remains negated, the microprogram branches to 073_8 and then continues.

For vectors where one delta length exceeds the other, the microprogram branches to either 062_8 or 063_8 and then goes to 072_8. Assume that the delta X value is greater than delta Y and that the vector shown in Figure 4-13 is being processed. This vector is to be drawn at an 11.25-degree angle and has the following delta magnitudes:

Delta X = 0777_8
Delta Y = 0177_8

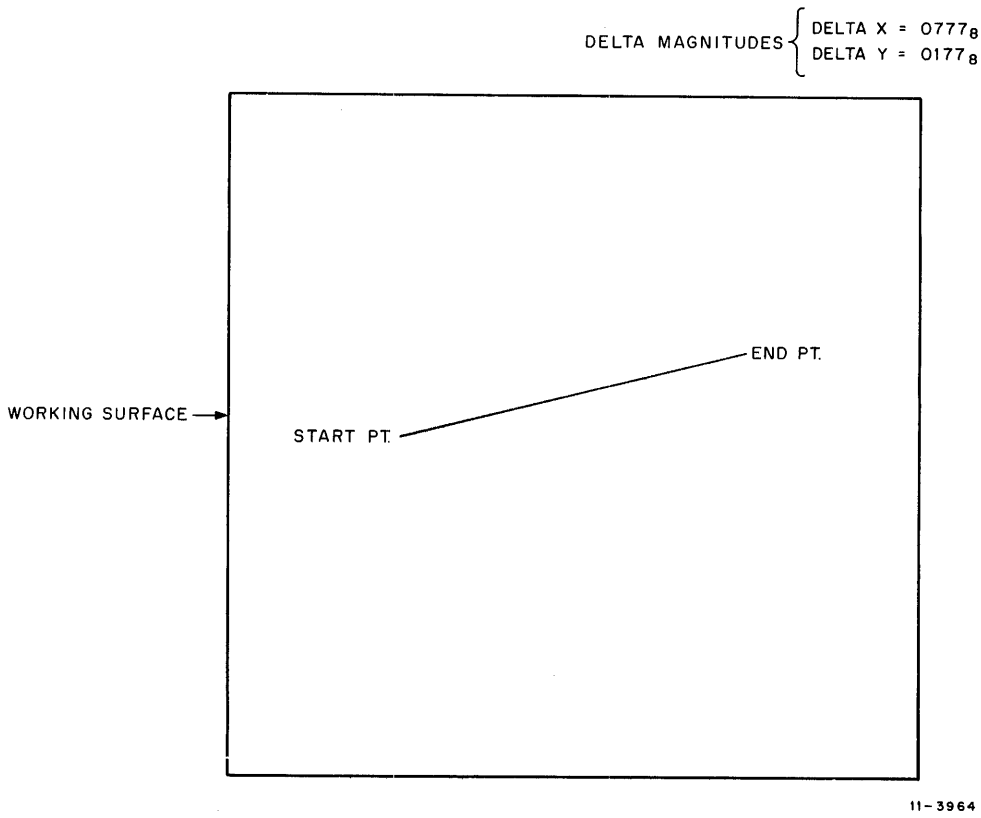


Figure 4-13 Typical ON-to-ON Vector Requiring Tangent Calculation

Since the delta Y value is one-fourth that of delta X, the tangent value that must be sent to the vector generator is 0.2_8 or 0.010000000000. The tangent is calculated by multiplying the larger delta value (in this case, delta X) by the content of the SAR and then comparing it to the smaller delta value. Initially, that is when the SAR is cleared at the start of the relative long vector sequence, the SAR contains a value of one-half (0.4_8 or 0.100000000000_2). The first multiplication of the delta X value by this fraction produces the answer shown below:

```
      000 111 111 111
     .100 000 000 000
    ------------------
      000000000000
     000000000000
    000000000000
   000000000000
  000000000000
 000000000000
000000000000
000000000000
000000000000
000111111111
-------------------------
0000111111111.100000000000
```

This product, 0377_8, is still greater than that of the delta Y value. Both the product and the delta Y value are applied to the tangent comparator. Because the product is greater than delta Y, signal MD1 TAN DATA asserts and a zero is clocked into the SAR to change the fraction from one-half to one-quarter (0.010000000000_2). Following each clocking of the SAR another multiplication takes place. With the fraction now changed the product produced on the second multiplication is:

```
      000 111 111 111
     .100 000 000 000
    ------------------
      000000000000
     000000000000
    000000000000
   000000000000
  000000000000
 000000000000
000000000000
000111111111
000000000000
-------------------------
0000011111111.100000000000
```

This product 0177_8 now matches the delta Y value and for all practical purposes, the tangent is already contained in the SAR. The SAR is clocked a full 12 times for each tangent calculation. However, the fraction contained therein undergoes no further changes, since signal MD1 TAN DATA only asserts when the value exiting the AB Multiplier exceeds that conveyed from the C multiplexer.

After the twelfth clocking of the SAR, signal MD1 CCL asserts to indicate conversion complete. At this time, the tangent value is strobed from the SAR into the tangent up/down counter. Hence, it is now positioned for transfer to the vector generator when it is ready to accept the next set of parameters.

At ROM locations 062_8 and 063_8, a check is made to see if the smaller delta value is equal to zero. For example, at ROM location 062_8, HY (delta Y) value is compared to zero. If it is equal to zero it means that the vector is a horizontal line and therefore no tangent need be calculated. In such a case, signal RM7 × SUM = Y SUM H asserts and the microprogram branches to location 070_8. That is, it skips the tangent calculation activity at location 072_8.

4.2.11.3 ON-to-ON Vector Processing Following Tangent Calculation – After the tangent calculation, the microprogram branches to 070_8. Note that the processing activities here are the same as those for a 45-degree vector at location 073_8. The functions carried out at location 070_8 can be summarized as follows:

1. The busy flip-flop is checked to make sure that the vector/character generator is not currently engaged in drawing a vector/character. Also, if the vector generator is in a busy status, the light pen down counter (LPDC) is currently engaged in the countdown process.

2. When the busy flip-flop output negates, the silo memory address is updated to supply the next set of outputs (intensity/menu) to the vector generator.

3. The LPDC is now loaded with the delta length (0777_8 for the sample vector shown in Figure 4-13).

4. Signal STC4 TRANS DATA → VG asserts to transfer the delta length and tangent values to the vector generator.

The microprogram now branches to 323_8 BUT 11_8. The sampling at the BUT 11_8 location is only meaningful if the maintenance mode 3 flip-flop is set. That is, the fact that the edge interrupt flip-flop may be set has no effect because the processing activities at ROM locations 323_8 and 322_8 are identical.

At both of the above locations, signal STC5 TRANS DATA L is asserted to transfer the remaining parameters (major/minor axis signs, etc.) to the vector generator. Also, the X/Y position registers are updated so that they contain the end point of the ON-to-ON vector.

The microprogram now branches to location 325 where the STC4 VEC GO L signal is asserted to command the vector generator to draw the vector. After this, the sequence shuts down as described in Paragraph 4.2.8.5.

4.2.11.4 ON-to-OFF Vector Processing – In processing ON-to-OFF vectors, there are three principal calculations that are made prior to commanding the vector generator to draw the unscissored portion of the vector. These calculations are:

1. Search for edge (scissor vector) – Since the vector starting point is on screen and it is known that the beam is going off screen, the point at which the vector traverses the working surface area must be found. Once the vector is scissored, the delta length and tangent values can be determined in the same way as for an ON-to-ON vector.

2. Determine delta length – This is done by the X/Y sum comparator which selects the larger of the two values (i.e., exiting the X/Y sum adders) for entry into the Delta Length register.

3. Calculate tangent – This is accomplished in a manner similar to that for ON-to-ON vectors.

The microprogram sequence for ON-to-OFF vectors begins on sheet 3 of the M7053 Program Flow Diagram. At the first ROM location, 066_8, the vector delta values are multiplied by the scale factor and placed in their respective holding registers. This action stores the delta magnitudes in each axis preparatory to scissoring the vector at the next stage in the microprogram sequence.

After loading the holding registers, the OP DONE (STC5 OP DONE L) signal is asserted to inform the display instruction control that it can begin processing the next instruction.

Also, the Delta Length register is loaded with the larger of the two values exiting the X/Y sum adders. This is done in anticipation of a light pen hit on the unscissored portion of the vector; that is, at such time as the unscissored segment is being drawn by the vector generator. Later in the sequence, the larger delta value is taken from the Delta Length register and loaded into the light pen down counter. The reason for doing so is explained in the paragraph covering light pen hit processing.

From ROM location 066_8, the program branches to one of four locations (depending on sign bits) where the scissoring action occurs. An explanation of scissoring action is best implemented when using a sample ON-to-OFF vector. Assume the vector about to be scissored is the one shown in Figure 4-14. This vector has a starting point of X = 1000_8 and Y = 0400_8. Its delta length magnitudes are X = 2777_8 and Y = 1377_8. The latter values represent the contents of the respective holding registers at the time that the scissoring operations are initiated.

For purposes of drawing the visible segment of the vector, the delta lengths must be converted to the following:

1. X = 777_8 – The value represents the difference between the X coordinate starting point and the edge of the visible area ($1000_8 + 0777_8 = 1777_8$).

2. Y = 0377_8 – This value represents the difference between the Y coordinate start point and the Y coordinate point where the vector exits the visible area.

Since both the delta X and delta Y values are positive, the microprogram branches from 066_8 to 103_8 to implement the scissoring calculations for the sample vector. To find the edge in the X coordinate, the arithmetic operation carried out is:

X + (HX) (SAR)

where:

X = vector starting point - 1000_8

Hx = delta X magnitude - 2777_8

SAR = 0.4_8 - This is the initial value placed on the Successive Approximation register just prior to initiating the scissoring calculation.

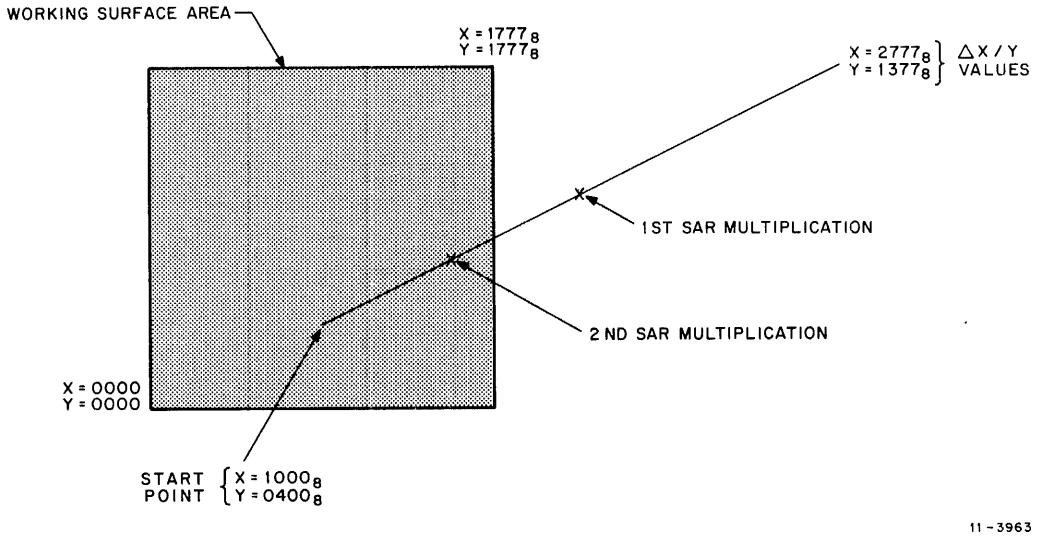


Figure 4-14 Example of Scissored On-to-OFF Vector

During scissoring calculations, the STC3 SUM EDGE L signal is gated through the SAR input select logic for application to the Successive Approximation register. The SAR is clocked 12 times to carry out the scissoring function. The objective of this action is to arrive at a fraction which, when multiplied (by the content of the X Holding register, 2777_8), produces a product of 0777_8. That is, a product which when added to the X coordinate starting address provides a total of 1777_8.

With each clocking of the SAR, (and subsequent multiplication by HX), the fraction therein changes so that it becomes closer and closer to the exact value needed to produce the precise product. Figure 4-15 presents a pictorial example of how repetitive multiplications of the changing SAR fraction move the calculated point closer and closer to the working surface edge (1777_8). The arithmetic operations that both change the location of the calculated end point and the fraction within the SAR are shown in Figure 4-16. Six of the twelve computations for the scissoring calculation are shown in this latter illustration.

The product of each multiplication (HX) (SAR) is added to the contents of the X Position register in the X sum adder. The first addition produces an end point answer (2377_8) in which bit 10 is set to a 1. This causes assertion of the STC3 SUM EDGE signal, which in turn changes the fraction in the SAR

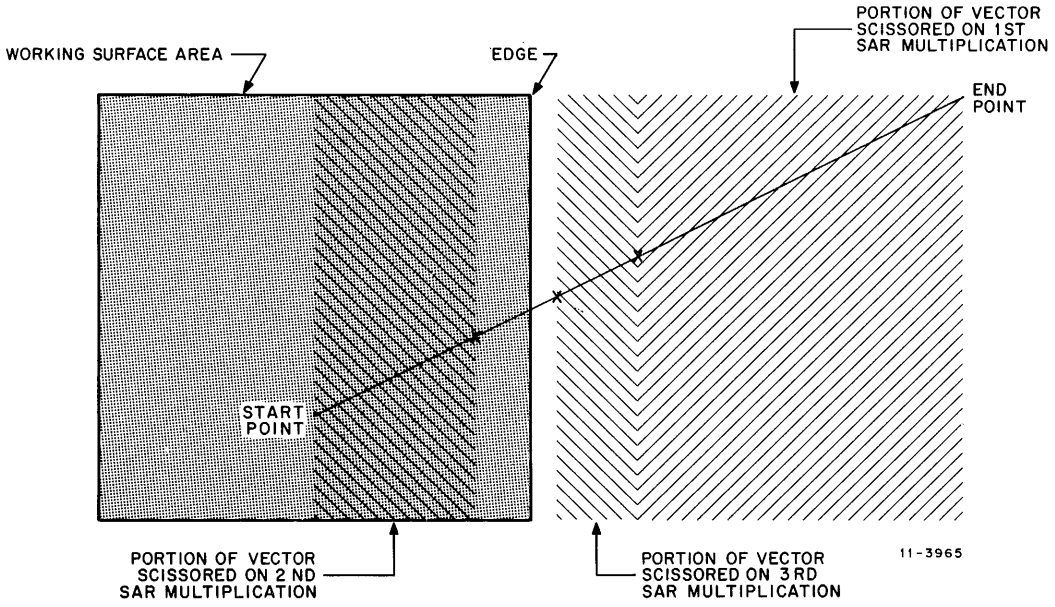


Figure 4-15 Example of Edge Approximation Calculations Used in Vector Scissoring

to that shown for the second arithmetic operation (i.e., 0.2_8). Note that the first end point answer is half the vector length shown on Figures 4-14 and 4-15. Also, the product of the first (HX) (SAR) multiplication (1377_8) is too great because the product, when added to the X Position register, produces an end point which is off screen.

The second arithmetic operation in the scissoring calculation produces an end point answer which is on screen, but less than the screen edge value of 1777_8. Note, however, that the calculated value is closer to the screen edge (by an amount of 0200_8) than that of the first arithmetic operation. Another significant point regarding the second multiplication of the HX and the SAR is that the product (0577_8) is now one-fourth, rather than one-half of the X delta value. The addition operation from the X sum adder has bit 10 cleared to a 0. Therefore, signal STC3 SUM EDGE does not assert. The 0 now clocked into the SAR produces the binary fraction shown for the third arithmetic operation.

The third step in the scissoring sequences moves the calculated end point still closer to the 1777_8 value. The SAR fraction is now of a value that produces an answer (1077_8) that is within 100_8 of the desired 0777_8 product. The calculated end point following the third arithmetic operation is also shown pictorially on Figure 4-14. Note that this point again is half as close as the point calculated in the preceding operation.

The (HX) (SAR) products and end point produced during the first six scissoring calculations can be summarized as follows:

	(HX)(SAR)	Calculated End Point		(HX)(SAR)	Calculated End Point
1.	1377_8	2377_8	4.	0737_8	1737_8
2.	0577_8	1577_8	5.	1017_8	2017_8
3.	1077_8	2077_8	6.	0767_8	1767_8

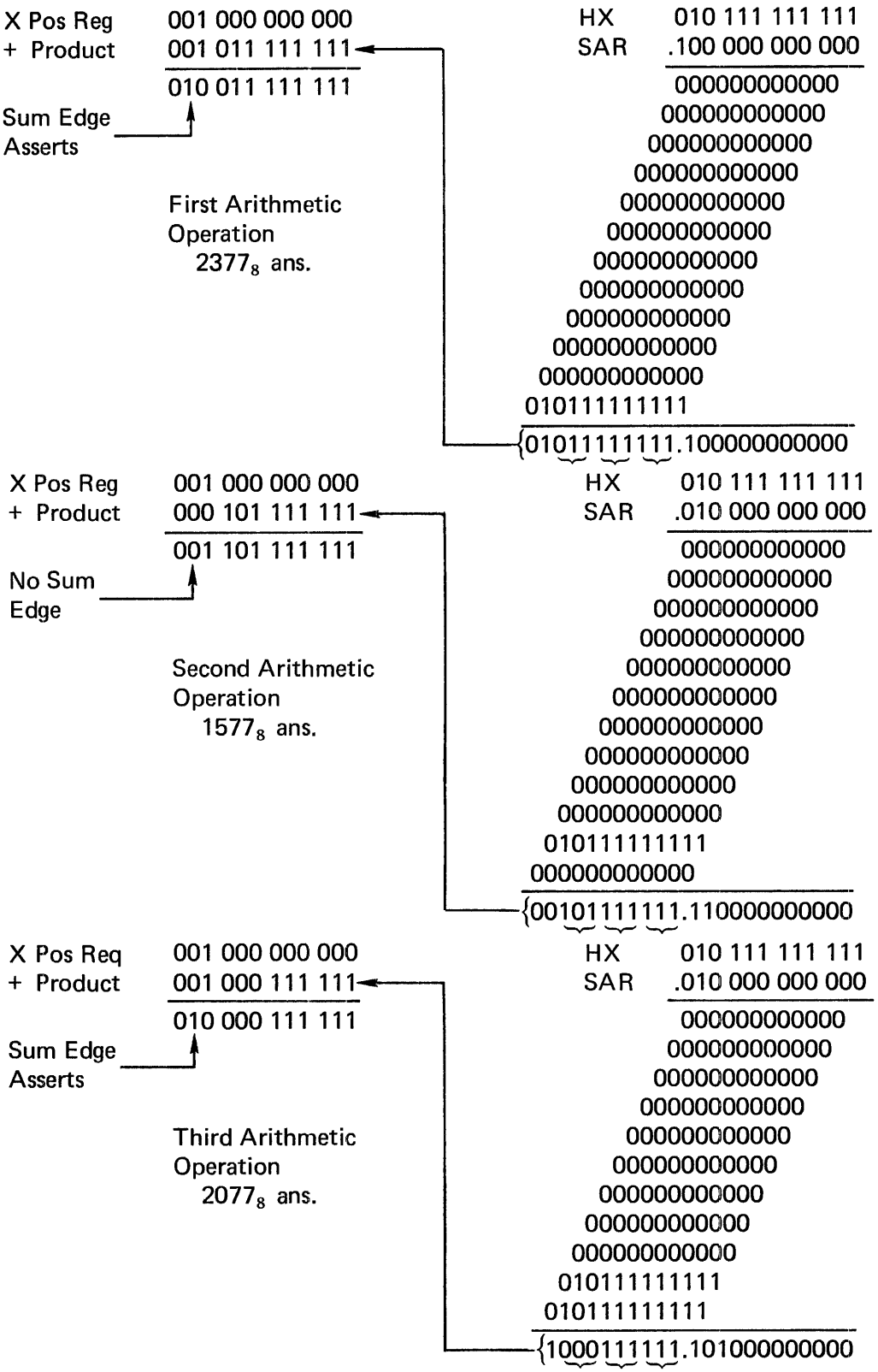


Figure 4-16 First Six of Twelve Computations Required for Scissoring Calculations, Part 1

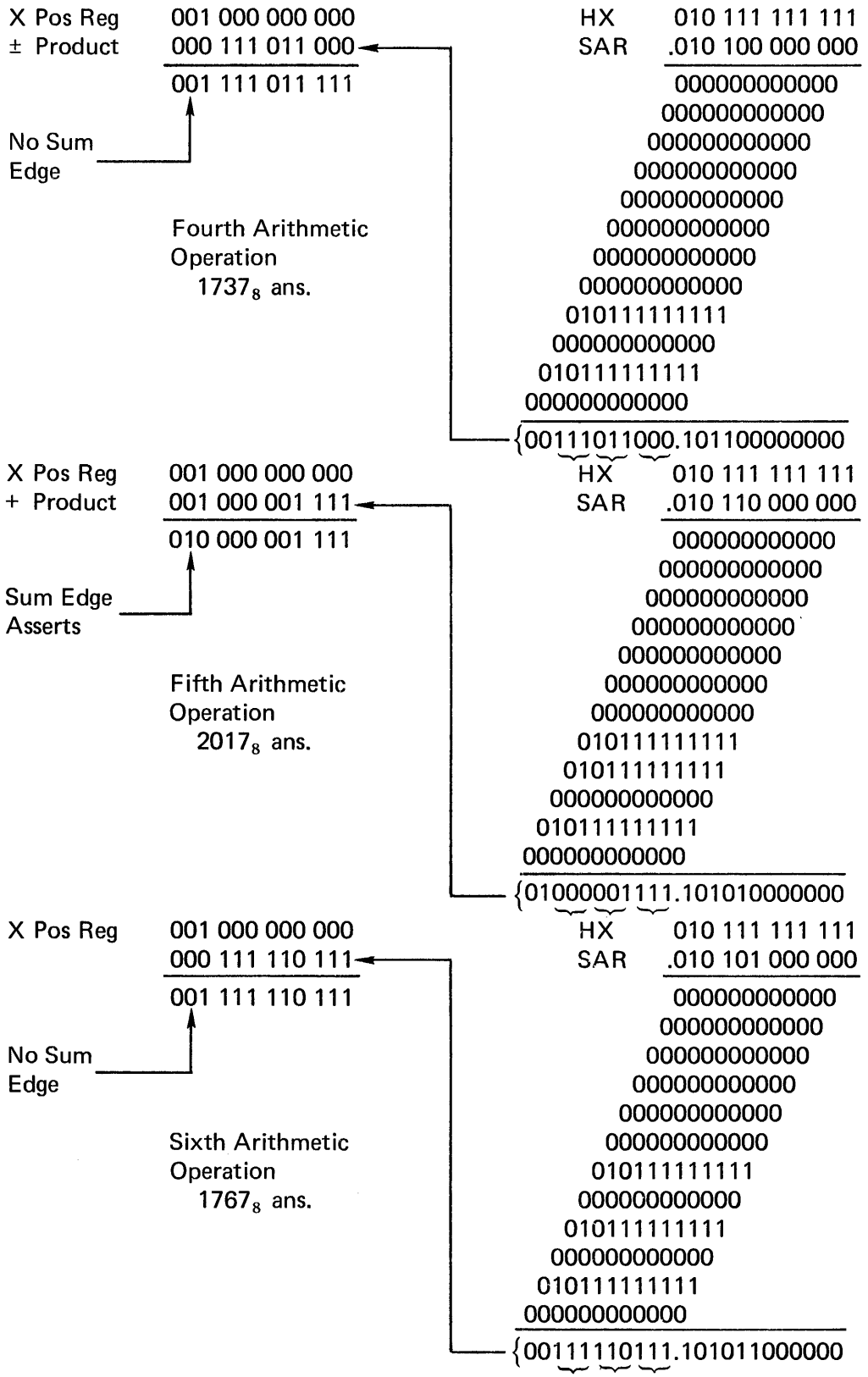


Figure 4-16 First Six of Twelve Computations Required for Scissoring Calculations, Part 2

From the foregoing, it can be seen that after 6 of the 12 arithmetic operations used to scissor the vector, the end point is within 10_8 of the desired point. By the time all 12 clockings of the SAR occur (coupled with attendant (HX)(SAR) multiplications, etc) the fraction in the SAR is within one LSB of the correct value.

The contents of the Y Holding register are also multiplied by the content of the SAR on each arithmetic operation. Again, when the fraction is fully derived, it also produces the correct coordinate answer in the Y axis.

The end of the scissoring calculation processes is indicated by asserting the conversion complete (MD1 CC L) signal from the SAR. This signal allows clocking of the Microprogram Address register to resume, provided that the busy flip-flop is clear.

The microprogram now branches to location 106_8 or 107_8 depending on the status of the STC3 SUM EDGE H signal. If this signal is currently clear, it means that the output of the X sum adder is exactly at the edge of the visible area. However, if STC3 SUM EDGE H is asserted, it means that the output of the X sum adder is off by one LSB (i.e., 2000_8 rather than 1777_8). Therefore, the factor must be clocked down by one so that the sum edge signal is now negated.

NOTE

At ROM location 103_8, the trailing edge of the STC5 CHECK CC L signal strobes the factor from the SAR to the tangent up/down counter. At ROM location 106_8, signal STC5 CLK D TAN L is asserted to decrement the factor in the tangent up/down counter by 1 LSB.

At ROM location 107_8, the objectives are twofold:

1. Load Contents of the Delta Length Register into the Light Pen Down Counter (LPDC) – This is done in anticipation of a light pen hit on the visible portion of the vector at the time when it is being drawn by the vector generator. See discussion on light pen hit processing.

2. Load Delta Length into Delta Length Register – In the example used in this discussion, an X delta length of 0777_8 (taken from X sum adder) is loaded, since its magnitude exceeds that of the Y coordinate. Again, detection of the larger value is effected in the X/Y sum comparator.

From ROM location 107_8, the microprogram branches to one of four locations depending on the status of the edge interrupt enable and maintenance mode 3 flip-flops. With neither flip-flop set, the microprogram branches to location 217_8 (sheet 6) where the tangent calculation for the scissored vector is carried out. If the edge interrupt flip-flop is set, a branch is made to location 305 (sheet 11), where an interrupt is generated and the edge transition points are recalculated for sampling by the PDP-11. Under edge interrupt conditions, the visible portion of the vector is not drawn by the vector generator.

4.2.11.5 ON-to-OFF (Scissored) Vector Tangent Calculation – The microprogram sequence for calculating the tangent of scissored vectors is shown on sheet 6 of the 7053 program flow diagram. The primary difference between the tangent calculation process for scissored versus ON-to-ON vectors is that when processing scissored vectors, the LPDC is already loaded at the time the sequence is entered. Also, the ON-to-ON sequence checks the status of the edge interrupt and maintenance mode 3 flip-flops after, rather than before, the tangent calculation operation. The discussion covering ON-to-ON tangent calculation (Paragraph 4.2.11.2) sufficiently explains the processing functions shown on sheet 6 of the flow diagram.

4.2.11.6 OFF-to-ON Vector Processing – The OFF-to-ON vector processing sequence is entered at ROM location 065_8 (sheet 4 of the M7053 Program Flow Diagram). The objectives to be carried out here can be considered as fivefold:

1. Calculate the edge coordinates where vector re-enters display surface. This arithmetic operation also produces the delta length of the visible vector segment.

2. Load vector generator D/A converters with edge coordinate values. This is done to position the beam (in blanked state) at the starting point of the visible vector segment. A time out period calculated by the down counter accompanies the beam positioning action.

3. Previously calculated delta lengths for visible vector segment are re-routed through X/Y sum adders to determine the larger of two which is then loaded into Delta Length register. This suffices for the delta length calculation.

4. The delta length value is loaded into light pen down counter in anticipation of light pen hit on visible segment.

5. The tangent value for visible vector segment is calculated. This is the final calculation prior to loading the vector generator and commanding it to draw the vector.

Objectives 1 through 3 are depicted on sheet 4 of the flow diagrams. The latter are shown on sheets 3 and 6 respectively.

Figure 4-17 shows an example of an OFF-to-ON vector and indicates the various parameters involved. As for all relative long vector cases, the vector end point is calculated and stored in the pre-position registers before the particular routine appropriate to the vector type is entered. Hence, the microprogram enters ROM location 065_8 (sheet 4) with the on screen end point of the vector already stored. That is, the pre-position registers contain:

$PX = 0777_8$
$PY = 0377_8$

At location 065_8, the delta lengths are calculated and stored in the holding registers (e.g., (ΔX) (SCALE) $\rightarrow$ HX). This is done because the delta lengths (multiplied by the SAR fraction) are eventually added to the vector end point coordinates to find the screen edge point. After storing the delta lengths, the STC5 OP DONE PULSE L is asserted to inform the display instruction control to begin processing the next instruction.

At ROM location 077_8, the coordinates for the on screen end point are entered into the X/Y position registers. This positions the coordinates for the arithmetic operation performed at the next location. The busy flip-flop is checked to ensure that all circuits are freed up for the find edge calculation undertaken at the next location.

At this time, all parameters are properly positioned for finding the set of edge coordinates. Branching is now a function of the sign bits of the delta magnitudes. In the sample vector given in Figure 4-17, both sign bits are negative. In such a case, the microprogram branches to location 114_8. Here, the product of the delta magnitude times the SAR fraction is added to the end point coordinates. The manner in which the product of the delta magnitudes times the fraction (in the SAR) is developed, is discussed in the paragraph covering ON-to-OFF vector edge calculation.

For the example given in Figure 4-17, the X-output of the AB 12 $\times$ 12 multiplier is 1000_8, or one-third the delta X magnitude. This value, added to an X coordinate end point of 0777_8, produces an X sum adder output of $X = 1777_8$. That is, the X coordinate of the point where the vector re-enters the screen

edge. A similar calculation is made for the Y coordinate values to produce a Y edge coordinate of 0777_8 at the output of the Y sum adder.

As in all edge calculations, the fraction developed in the SAR may be one that produces a product that is off by one LSB; e.g., 2000_8 rather than 1777_8. The factor from the SAR is strobed into the tangent up/down counter on generation of the conversion complete signal (MD1 CC L at ROM location 114_8). If the fraction is off by one LSB, signal STC3 SUM EDGE L will be asserted. In such a case, the microprogram branches to location 126_8 where the value in the tangent up/down counter is decremented by 1. The output from the X sum adder now changes from 2000_8 to 1777_8, a value which places the beam precisely on the screen edge.

The microprogram now proceeds to location 127_8 to load the edge point coordinates into the X/Y position registers. This is necessary, since at the next step (ROM location 152_8) the beam is moved to the edge point preparatory to drawing the visible segment. Deflection of the beam to the edge point is accomplished by loading the vector generator D/A converters with the edge point coordinates (LOAD DAC).

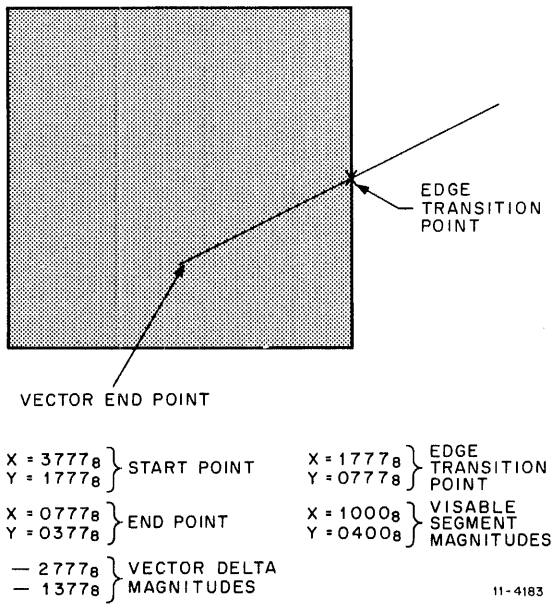


Figure 4-17 Typical OFF-to-ON Vector

Before loading the D/A converters, the beam is at center screen having been placed there by some prior processed ON-to-OFF vector. This means that regardless of where the edge coordinates are, the beam must be deflected one-half the working surface area to arrive at the edge point. For this reason, 1000_8 is loaded into the down counter to time out the deflection process.

At the next location (156_8), the delta length of the visible segment is calculated by multiplying the delta magnitudes (in the holding registers) by the factor in the tangent up/down counter; (HX)(FAC) and (HY)(FAC). The larger of the two values exiting the X/Y sum adders is detected by the X/Y sum comparator. The larger value is then strobed into the Delta Length register (LOAD Δ REG) via the X/Y sum multiplexer.

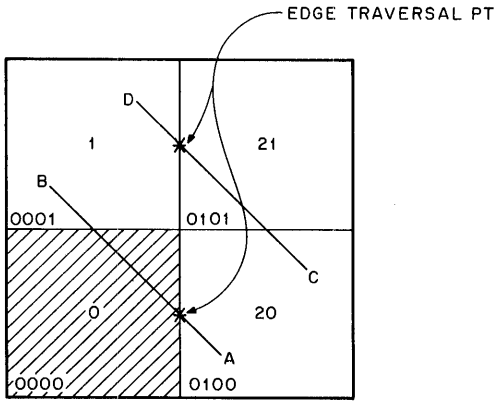
Also, the clock to the down counter is now enabled to initiate the time out period for the beam deflection. Next, the busy flip-flop is checked to detect the end of the time out period. When it occurs, the microprogram branches to ROM location 107_8 (sheet 3).

At this point, all calculations strictly pertaining to an OFF-to-ON vector are completed. Remaining operations such as loading the light pen down counter and calculating the tangent for the visible segment are covered in the discussion of ON-to-OFF vector processing.

4.2.11.7 OFF-to-OFF Vector Processing – The scissoring decode logic plays a primary role in processing OFF-to-OFF vectors. This logic consists of three pre-programmed ROMs (STC2) that collectively decode the locations of vector start and end points within the virtual display area. All addressed start and end points for OFF-to-OFF vectors must fall within the sectors depicted on sheet 12 of the M7053 Program Flow Diagram. The binary notations by the blocks along the base line and left edge of this illustration represent the four high order bits (10–13 MSBs) of the X/Y coordinates. Design of the scissoring decoding logic is such that it samples the stored four high order bits (in both X and Y) to determine the vector starting point. Simultaneously, the logic also samples the four high order bits (again for both X and Y) exiting the X/Y sum adders to determine the vector end point. The input ROMs are so programmed, that by looking at all 16 input bits simultaneously, they determine whether the vector start and end points fall within sectors that could produce a visible segment.

Figure 4-18 represents a section of the complete virtual display area depicted on sheet 12 of the flow diagram. This illustration shows two vectors whose start and end points fall within sectors capable of producing a visible segment. However, only one of the two vectors passes through the display surface and consequently has a visible portion. The OFF-to-OFF processing sequence must not only distinguish between the two types, but must also calculate the delta lengths and tangents of any visible segments.

The flow sequence for OFF-to-OFF vector processing is given on sheet 5 of the flow diagram.



VECTOR AB (VISABLE SEGMENT)

START PT:	$X = 2377_8, Y = 0377_8$
EDGE TRAVERSAL PT:	$X = 1777_8, Y = 0777_8$
END PT:	$X = 0377_8, Y = 2377_8$

VECTOR CD (NO VISABLE SEGMENT)

START PT:	$X = 3377_8, Y = 1377_8$
EDGE TRAVERSAL PT:	$X = 1777_8, Y = 2777_8$
END PT:	$X = 1333_8, Y = 3377_8$

11-4182

Figure 4-18 Two Typical OFF-to-OFF Vectors

A summary of the microprogram processing activities indicated here is as follows:

1. Route the vector end point coordinates through the X/Y sum adders to determine if the sectors involved in the vector start and end addresses can produce a visible segment.

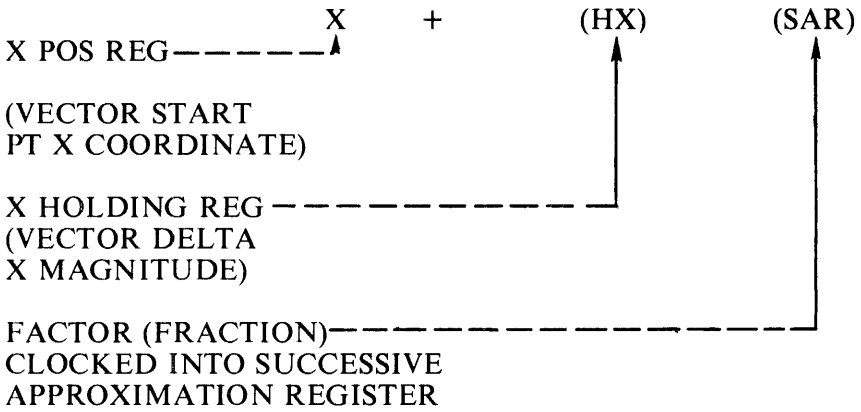
2. Branching now occurs depending on the possibility of a visible segment.

   a. If no possibility of a visible segment exists, the microprogram sequence terminates.

   b. If there is a possibility of a visible segment, the X/Y delta magnitudes are loaded into the X/Y holding registers preparatory to calculating the edge traversal point; that is the point where the vector first intersects the display area.

3. Sign bits are sampled to detect vector direction and thereby steer the microprogram to the appropriate calculation sequence.

4. Find the edge point; that is the point where the vector first intersects the display surface from the vector start point. The edge point coordinate for X is calculated by:



5. On finding the edge, the factor (fraction) from the SAR is strobed into the tangent up/down counter; then a check for the SUM EDGE condition is made. If the vector is one having a visible segment, the SUM EDGE signal may be asserted under conditions where the factor in the tangent up/down counter is off by one LSB. Therefore, the factor is incremented by one with the objective of placing the beam on the display surface.

6. A second check of SUM EDGE is made now to determine if the vector has a visible segment. This is done in the following way.

   a. If the SUM EDGE signal is still asserted, it means that one of the four high order bits (i.e., for one of the coordinates, Y coordinate in the case of vector CD) is still asserted. In other words, one coordinate remains outside the 0000_8 to 1777_8 range. In the case of vector CD on Figure 4-18, the X coordinate is within display surface limits but the Y coordinate is at 2777_8 following edge traversal point calculation. Hence there can be no visible segment of the vector and microprogram sequencing is terminated.

   b. If the SUM EDGE signal is negated, it means that both coordinates are on screen; that is, both fall within the 0000_8 to 1777_8 range. Therefore, there is a visible segment and since the start point of the segment is now known, the vector can be processed and generated in a manner similar to an ON-to-OFF vector.

The sequence appearing on sheet 5 of the flow diagram is now described. At ROM location 064_8, the vector end point coordinates are routed through the X/Y sum adders. The end point coordinates are then sampled by the scissoring decode logic along with the previously stored vector start point coordinates. If the start and end point sectors are such that they cannot produce a visible segment (e.g., sector 77 to sector 23) then signal STC2 NO POSSIBILITY L remains low and the microprogram branches to ROM location 162_8. Here the vector end points are simply stored (PX → X and PY → Y) after which the graphics calculation logic terminates the sequence and accesses the next graphic entity.

If the start and end points fall in sectors that can produce a visible segment, then signal STC2 NO POSSIBILITY L switches high and the microprogram branches to location 162_8. Here, the delta X/Y values are multiplied by the scale factor and the resultant magnitudes are placed in the X/Y holding registers. The STC5 OP DONE PULSE L signal is then asserted since the delta X/Y registers are free to receive new values.

The microprogram now branches to one of four locations depending on the sign bits of the delta X/Y magnitudes. In the example given in Figure 4-18, vector direction is such that X is negative going while Y is positive.

In such a case, the microprogram branches to ROM location 135_8 to begin the visible segment calculation process. The objective at this location is to find the edge traversal point, and to do so, a factor (fraction) must be clocked into the SAR. The factor inserted here (when multiplied by the delta magnitudes and added to the start point coordinates) produces outputs from the X/Y sum adders equivalent to the edge traversal point.

For this calculation, the signal selected as the input source to the SAR is STC2 NO POSSIBILITY H. This signal is a high whenever one of the following two conditions exists during the SAR clocking process:

1. The calculated point (on a single clocking of the SAR) is one that falls in another sector but that sector is one that cannot possibly produce a visible segment. For example, vector CD (Figure 4-18) is divided in half on the clocking of the SAR. The calculated point then is halfway along the segment in sector 21. And a vector starting in sector 20 and ending in sector 21 cannot possibly produce a visible segment.

2. The calculated point falls in the same sector as the start point, but again, that sector cannot produce a visible sector. Such would be the case for a vector falling entirely within sector 20; Figure 4-18.

Signal STC2 NO POSSIBILITY H is a low whenever the calculated point falls within the display area or could potentially fall within sector 0. All calculated points within sector 0 have X/Y coordinate values of less than 1777_8. However, it is possible for a calculated point to have one of the two coordinate values less than 1777_8. For example, assume that vector CD extended further into sector 1. In such a case, the calculated traversal point (on the second clocking of the SAR) would fall inside sector 1 meaning that the X coordinate value is less than 1777_8. Hence signal STC2 NO POSSIBILITY H switches low.

Clocking of the SAR continues (for 12 clockings) until the value contained therein generates a product equal to the edge traversal point. This product appears at the output of the X/Y sum adders. At this time, the conversion complete signal (MD1 CC L) signal is produced, and in turn, STC5 CHECK CC L is asserted to enter the factor into the tangent up/down counter.

The microprogram now branches to ROM location 151_8, BUT 12_8. At BUT multiplexer location 12_8, the status of the STC3 SUM EDGE signal is sampled. This is done because even if the OFF-to-OFF vector has a visible sector, it is possible that the factor now in the tangent up/down counter could be

off by one LSB. That is, it could generate a product that is off screen by one unit. Therefore if signal STC3 SUM EDGE H is asserted, the factor is clocked up by one LSB to place the beam on screen.

NOTE

Clocking up of the factor is in contrast to decrementing the factor as is done in the ON-OFF and OFF-to-ON scissoring calculation. This is because in the two latter cases, the edge traversal calculation is being made from a point on screen. Here the edge traversal point is being calculated from a point off screen.

Signal STC5 CLK UP TAN L asserts (at location 150_8) to clock up the factor. Following this, the sequence proceeds to location 153_8, BUT 15_8. At BUT multiplexer location 15_8, the status of the STC3 SUM EDGE H signal is immediately sampled again. If it is still asserted, it means that one of the two coordinates (produced at the outputs of the X/Y sum adders) is outside the 0000_8-to-1777_8 range. Therefore, the calculated edge traversal point is off screen and there can be no visible segment. This is the case for vector CD on Figure 4-18. The X coordinate value of 1777_8 is on screen but the Y coordinate value of 2777_8 is off screen. As a result, the end point coordinates are loaded into the position registers (location 152_8) and the microprogram sequence for the OFF-to-OFF vector is terminated.

If signal STC3 SUM EDGE H is no longer asserted after clocking up the factor, the traversal point must be on screen and the microprogram branches to location 151_8. Here the X/Y coordinates for the edge traversal point are loaded into the X/Y position registers. These coordinates now represent the starting point for the visible segment. They are to be loaded into the vector generator D/A converters to position the beam at the starting point. This is done at the next processing stage.

Now that the on screen edge point has been found, the task of calculating the visible segment is basically the same as processing an ON-to-OFF vector. The sequence at ROM location 151_8 continues by:

1. Clearing the SAR since it will be used later in the ON-to-OFF vector scissoring calculations.

2. Checking the busy flip-flop status to determine that the vector/character generator has completed drawing the previous graphic entity. If so, the sequence proceeds to location 161_8.

Next in the sequence, the vector generator D/A converters are loaded with the content of the X/Y position registers to deflect the beam to the edge point. That is, the starting point of the visible segment. Also, a time out period (1000 → D COUNT) is initiated, since the beam must be moved from center screen (having been placed there by a prior processed ON-to-OFF vector) to edge point. This distance amounts to half screen deflection or a value of 1000_8. The sequence now advances to ROM location 163_8 where the time out period is initiated (START D COUNT). Also accomplished here is the calculation of the delta magnitudes for the visible segment plus off screen overlap. This is necessary to come up with a value to enter into the light pen down counter (LPDC). The delta lengths are calculated by multiplying the vector delta length (HX) times the factor (TAN, in tangent up/down counter) and subtracting the product from the delta length (HX). In the AB vector shown on Figure 4-18, the calculated values for both X and Y are 1777_8. This happens to be a 45-degree angle and both delta magnitudes are the same. For other angles, the larger delta magnitude exiting the X/Y sum adders is strobed into the delta length register (LOAD Δ REG). This is done preparatory to loading the LPDC at the next processing stage. This calculation is similar to that carried out for the ON-to-OFF vector at ROM location 066_8 (sheet 3).

The busy flip-flop is constantly checked at location 163_8 until the time out period for positioning the beam at the edge point is complete. On completion, the program branches to location 103_8 BUT 04

(sheet 3) where the ON-to-OFF vector processing sequence is undertaken. That is, all calculations pertaining strictly to the OFF-to-OFF vector are now complete. Refer to the ON-to-OFF vector discussion to continue the microprogram sequence.

4.2.12 Absolute Vector Processing

The absolute vector processing sequence is shown on sheet 7 of the 7053 Program Flow Diagram. The main objective of all processing activities shown on this sheet is to determine X/Y delta magnitudes (and signs) prior to branching to the microprogram location for the appropriate vector type (i.e., ON-to-ON, ON-to-OFF, etc.).

The operations performed at ROM addresses 223_8 through 242_8 are the same as those described for absolute points (Paragraph 4.2.10). Also, when the arithmetic logic must add two negative numbers, the activities of ROM locations 243_8 through 245_8 come into play. These activities are described for ROM location 042_8 under absolute point processing.

Once the proper vector and point values have been loaded into the pre-position registers, the miroprogram branches to ROM location 253_8, BUT 00_8. This means that signals STC3 SUM EDGE H and STC2 EDGE H are sampled to determine in which vector category (ON-to-ON, ON-to-OFF, etc.) the upcoming absolute vector falls. If both the start and end points of the absolute vector are on screen, the microprogram initiates ON-to-ON vector processing at ROM location 253_8. If the starting point is off screen and the end point is on screen, the microprogram branches to location 251_8 to initiate OFF-to-ON vector processing, etc.

4.2.12.1 ON-to-ON Absolute Vector Processing – At location 253_8, the absolute vector end point values are routed through the X/Y sum adders to detect carry bits. (Also, the direction or sign flip-flops on drawing STC3 are cleared). Detection of carry bits is necessary because it enables determination of vector direction for the processing activities carried out at ROM locations 254_8 through 257_8. That is, if no carries result, it means that the vector is positive going in both coordinate: Hence, the sequence branches to location 254_8, where the delta magnitudes are loaded into the holding registers and the delta length (larger of the X/Y values) is loaded into the Delta Length register.

If carries result at the output of both the X and Y sum adders it means that the vector is negative going in both coordinates. Now, the microprogram branches to location 257_8. Here, both the holding registers and the Delta Length register are loaded as before, but in addition, the sign flip-flops for both coordinates are set so that the vector generator will draw the vector in a negative going direction. The discussion in Paragraph 4.2.10.1 describes in greater detail the purpose of carry bit detection.

Once the delta magnitudes have been loaded and the sign bits set (in accordance with the vector direction), the microprogram branches to the routine for ON-to-ON vector processing (sheet 2).

4.2.12.2 ON-to-OFF, OFF-to-ON, and OFF-to-OFF Absolute Vector Processing – All absolute vectors other than ON-to-ON go through a special sampling process to determine vector direction. The sampling process employs signals STC2 ABS VEC DATA 00 and STC2 ABS VEC DATA 01 from the scissor decoding logic to determine where the vector start and end points lie. Figure 4-19 shows those sectors (sheet 12 of the flow diagram) lying immediately to the right and above the screen viewing area (sector 0). Also shown here is an ON-to-OFF vector that has a positive going X delta magnitude and negative going Y delta magnitude. Since it is an ON-to-OFF vector, the sequence is entered at ROM

location 252_8 where the sign flip-flops are cleared (CLR DIRECTION). The microprogram now branches to location 263_8 BUT 06_8. These two signals are now sampled to determine vector direction as follows:

STC2 ABS VEC DATA 00	STC2 ABS VEC DATA 01	ON-to-OFF Vector Direction
1	1	Sector $0 \rightarrow 4_2$ or 7_2
1	0	Sector $0 \rightarrow 5_2$
0	1	Sector $0 \rightarrow 1_2$ or 3_1
0	0	No possibility

This sampling determines the sector where the absolute vector end point falls. In the example in Figure 4-19 the end point falls in location 7 (0111_1). Therefore, the microprogram branches to 260_8. Two things are now known about the vector to be processed: the vector type is ON-to-OFF, and since the end point falls in vector 7, the X delta magnitude must be positive going. What remains to be determined is the direction of the Y delta magnitude (negative going on Figure 4-19). To find the direction

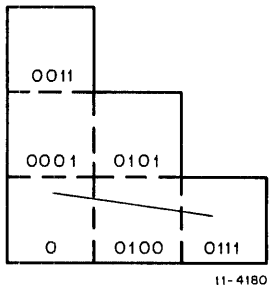


FIG. ABSVEC

Figure 4-19 Example of Typical Absolute Vector

of the Y coordinate, the start point coordinate (Y) is subtracted from the end point coordinate (PY). Since the Y value is greater than that of PY, no carry results ($\overline{COY}$). Hence the Y component must be negative going and a branch is made to ROM location 267_8. Here the delta magnitudes are calculated as follows:

1. The X start point (X) is subtracted from the end point (PX-X → HX) to obtain the X delta magnitude.

2. The Y end point (PY) is subtracted from the start point (Y-PY → HY) to obtain the Y delta magnitude.

Also, at ROM location 267_8 the Y sign flip-flop is set to indicate to the vector generator that the Y component of the vector is negative. The Delta Length register is then loaded and the routine branches to the ON-to-OFF vector sequence (sheet 3).

The other absolute vector types are treated similarly to that just described. That is, the start and end point locations are sampled to determine direction and then a branch is made to the location having the appropriate calculation sequence. Once delta magnitudes and direction have been calculated, a branch is made to the routine appropriate to the vector type.

4.2.13 Relative Point/Graphplot Processing

The initial starting address activities for both graphplot instructions and the relative point instruction are shown on sheet 1 of the M7053 Program Flow Diagram. In all cases, the prime function performed at the starting address is to calculate the addressed end point coordinates and load them into the pre-position registers. Following this, the microprogram simply branches to location 047_8 to continue the sequence explained for absolute point processing (Paragraph 4.2.10).

4.2.14 Character Processing

The microprogram sequence for processing characters is shown on sheet 8 of the 7053 Program Flow Diagram. Although the character generator draws the character independent of the graphics calculation logic, it is the vector generator that must move the beam for the spacing between characters (and also for control characters such as line feed, back space and carriage return). Consequently, the graphics calculation logic must update the X and Y position registers for each character about to be drawn.

At ROM locations 035_8 through 040_8 the X, Y values for the updated position are initially placed in the X/Y pre-position registers. The delta X/Y values times the character scale are added to the X/Y coordinate values (X, Y) for the previous characters.

NOTE

The delta X/Y values are variables since the beam deflection requirements for subscript and superscript characters differ from that of straight line text.

Once the coordinates for the updated beam position are contained in the pre-position registers, a branch is made to location 113_8 BUT 03_8. The sampling occurring at this BUT multiplexer location detects the following:

1. If the character is not a control character or carriage return character (itself a control character), then it means that the character is drawable. Hence, if the character is on screen, a command to draw the character must be sent to the character generator. This occurs later in the microprogram sequence.

2. If the character is a control character but not a carriage return (signal DSR6 CHAR INHIBIT LEVEL H asserted and signal DSR6 CRH negated), then the beam status/direction must be determined and appropriate action taken. That is, where is the beam currently positioned and where is it going, ON-to-ON, ON-to-OFF, etc.?

3. If the character is a control character and also a carriage return (signals DSR6 CHAR INHIBIT LEVEL H and DSR6 CRH both asserted) the microprogram branches to location 110_8. For carriage returns, the values in the offset registers must be taken into account because carriage returns are treated like absolute points. All other characters are treated as relative points.

4.2.14.1 Processing Drawable Characters – At ROM location 113_8 the beam status is sampled to determine its current location and destination (signals STC3 SUM EDGE H and STC2 EDGE H are sampled by the BUT multiplexer). The processing functions implemented for the individual beam conditions are:

1. OFF-to-OFF (ROM location 130_8) – Since the beam is off screen and going off screen, the X/Y position registers are updated to the new off screen address. A check is made of the busy flip-flop and once cleared (previous graphic entity finished being drawn), the silo memory is updated (INC RA) and the display instruction control is informed to process the next instruction (OP DONE).

2. OFF-to-ON (ROM location 131_8) – Since the beam is returning on screen, 1000_8 must be loaded into the down counter. This is performed to time out the period necessary to deflect the beam from its current center screen position to the edge point where the beam is returning on screen. The microprogram then branches to 050_8 (sheet 1) where it picks up the processing sequence for an off screen-to-on screen point.

3. ON-to-OFF (ROM location 132_8) – Here the X/Y position registers are updated to the new off screen coordinate values and the edge indicator is set. Then a check of the busy flip-flop is made and when cleared, the silo address is updated (INC RA) and the display instruction control is informed to process the next instruction (OP DONE). The microprogram then branches to location 060_8 (sheet 1) to pick up the sequence for an ON-to-OFF screen point.

4. ON-to-ON (ROM location 133_8) – At this location, the sequence for processing a drawable character is initiated. The first thing done here is to update the X/Y position registers to the start point for the next character. Though these registers are updated at this point, the vector generator D/A converters are not updated until the end of the sequence (ROM location 076_8). This is because the character to be drawn may be at the end of a vector as is character B in Figure 4-20. In such a case, the vector end point is the start point for the character rather than the updated position entered into the X/Y position registers.

Since a prior graphic entity (such as the vector in Figure 4-20) may still be in the process of being drawn, the busy flip-flop is checked. Once cleared, the silo memory is updated to provide the parameters (intensity, menu-select) for the character to be drawn. The microprogram then proceeds to location 111_8 where the TRANS DATA signal asserts. This effects actual transfer of the new silo parameters to the vector generator.

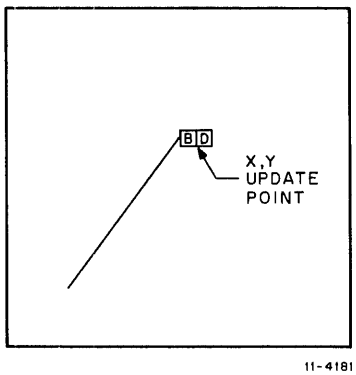


Figure 4-20 Example of Character Drawn at End of Vector

At ROM location 074_8, another check of the busy flip-flop is made and if cleared, the command to draw the character is issued to the character generator (signal STC5 START CHAR GEN L asserts).

At the next location, the busy flip-flop is checked to determine when the character generator is finished drawing the character. Once this flip-flop is cleared, the microprogram moves to 076_8 where the vector generator D/A converters are updated to the new X/Y coordinate position. The sequence is then terminated in the normal way by asserting the OP DONE and ALL DONE signals.

4.2.14.2 Processing Control Characters Other than Carriage Return – The control characters that affect beam position (i.e., other than carriage return) are line feed, back space and space bar. Each of these requires that the beam be deflected in the blanked state; however, the beam may be on screen and remaining on, on screen and going off, etc. Therefore, there are four different processing locations for the different beam conditions. For the OFF-to-OFF, OFF-to-ON and ON-to-OFF conditions, the processing activities are identical to those described in the preceding paragraph. For the ON-to-ON condition, the X/Y position registers are updated after which (busy flip-flop cleared) a branch is made to the last two stages of the character draw sequence.

4.2.14.3 Carriage Return Processing – The carriage return processing sequence varies depending on whether the character rotate status is in effect. For normal straight line and horizontal text, the carriage return command places the beam at the left edge of the working surface. If the character rotate status is in effect, the carriage return command places the beam at the bottom edge of the working surface. The processing activities carried out for both cases are similar except that the X-axis is affected in one case and the Y-axis in the other.

For the non-rotated condition, a value of zero is entered into the X Pre-Position register, since this (if not affected by the X Offset register contents) will left justify the beam to the left edge of the screen. This occurs at ROM location 201_8 where the Y delta magnitude is also entered in the Y Pre-Position register.

Next, the X offset value is added to/subtracted from the content of the X Pre-Position register. For example, if the X Offset register contains $+0100_8$, the value in the Pre-Position register is changed from all zeros to 0100_8. If the offset value is negative, the complement of the offset value (plus one LSB) is entered into the X Pre-Position register. Again, if the content of the X Offset register were -0100_8, then the complemented value would be $111\ 110\ 111\ 111_2$. However, this would be off by one LSB as explained for absolute point processing. Hence one LSB is added to produce a product of 7700_8 rather than 7677_8.

After adding the offset value, the program branches to ROM location 333_8 BUT 00_8. If the carriage return is on screen, the deflection value for the larger coordinate (always X in horizontal text) must be loaded into the down counter. This is necessary to time out the beam deflection period when, for example, the beam must be returned from full right edge to the left edge position. To calculate the correct value for entry into the down counter, the contents of the X/Y position registers are subtracted from the contents of the respective pre-position registers. The larger value exiting the X/Y sum adders is then entered into the down counter. From ROM location 333_8, the routine branches to 057_8 BUT 01 to enter the sequence for ON-to-ON point processing.

At ROM locations 330_8, 331_8, and 332_8, the processing activities are the same as those described for a drawable character.

4.2.15 Light Pen Hit Processing

4.2.15.1 Processing Objectives – When processing light pen hits at the VR48 Display Monitor, the objectives are threefold:

1. To generate an interrupt to the central processor to indicate the occurrence of a light pen hit.

2. To calculate the X/Y coordinate values of the light pen hit and then load these values into the X/Y position registers. From here, the central processor is able to sample (read status) the X/Y coordinate values to identify the position of the light pen hit.

3. To continue graphic entity processing on receipt of a Resume command from the central processor. Resumption can take two forms:

   a. Light pen hit occurs during character string being processed – the graphics calculation logic resumes operation by commanding the character generator to draw the next character in sequence.

   b. Light pen hit occurs upon vector being drawn – graphics calculation logic resumes drawing remainder of vector; that is, from point of light pen hit to addressed end point.

> NOTE
> **In processing light pen hits on vectors, the graphics calculation logic draws the segment of the vector from starting point up to the light pen hit, then generates the interrupt, etc., and finally draws the remaining segment following receipt of the Resume command.**

4.2.15.2 ROM Starting Address Setup – The microprogram sequence for processing light pen hits is shown on sheet 9 of the 7053 Program Flow Diagram (see print set). Entry into the starting address (377_8) is unique. That is, it is unlike any of the instructions where the starting address is defined by the contents of the mode code Holding register.

A light pen hit from either console asserts signals STC3 L.P. HIT L and STC3 L.P. HIT H. These are used on the STC1 drawing to select the desired starting location.

In a static state, that is when no operations are going on, the output of the graphics calculation ROMs is 374_8. This is so because all instruction sequences (for all graphic entities) branch to this address on conclusion of processing. (Signal STC5 ALL DONE asserts to produce signal STC2 ALL FINISH H, which in turn shuts down microprogram sequencing, Paragraph 4.2.8.5.) If no new graphic entity code has been loaded by the display instruction control logic, the graphics calculation logic sits at this address until one of two things occurs:

1. A new mode code is entered into the mode code Holding register (Paragraph 4.2.8.1).

2. A light pen strike (that is, on the previous processed graphic entity which is now being drawn by the vector/character generator) occurs. It is this action that asserts the STC3 L.P. HIT L and STC3 L.P. HIT H signals.

The second of these signals is applied to the initial program starting address ROMs (STC1) where it forces an all 1s condition output (377_8). Signal STC3 L.P. HIT L is applied to the pulse sync logic (STC1), and on occurrence of the next clock pulse, signal STC1 L.P. SYNC H asserts (E68-11). This signal is used to effect the following:

1. Clear the microprogram Address register (STC1) to produce an all 0s output. This action in turn forces an all 1s output from the microprogram address ROMs themselves. Hence, signals STC4 RA 00 H through STC4 RA 07 H are all high to present an address of 377_8 to the microprogram Address register.
2. Jam presets/clears the two sync flip-flops to place them in the initial startup condition (Paragraph 4.2.8.1). With signal STC1 SYNC 01 H now negated, the all 1s output from the initial program starting address ROMs is also presented to the microprogram Address register. This further ensures that the address on the register input lines is 377_8.

The next clock pulse clocks the address into the microprogram Address register and also sets the second sync flip-flop. At this time, then, the light pen hit starting address is in the Address register and light pen hit processing begins.

4.2.15.3 Example of Light Pen Hit Calculations – The light pen hit processing sequence is best explained in terms of an example where all conditions are known. Such an approach enables the reader to understand the reasons for the individual processing activities at each stage in the overall sequence. The example in this case involves a positive going vector as shown in Figure 4-21. The complete range of processing parameters associated with this example are:

1. Vector Type ON → OFF.
2. Starting and ending coordinates.

START	END
$x = 0000_8$	$x = 3777_8$
$y = 0000_8$	$y = 0777_8$

3. Value entered into light pen down counter (LPDC) before issuing VEC GO (STC4 VEC GO L) to the vector generator – 3777_8.
4. Calculated tangent value entered into vector generator Tangent register.

   2^{-2} ($010\ 000\ 000\ 000_2/0.2_8$)

5. Other parameters set up in vector generator prior to issuing STC4 VEC GO L.

   a. VG2 X > Y H – This signal is high since the X component of the vector is greater than that of Y.

   b. VG2 MA + H – This signal is asserted because the major axis (X) is positive.

   c. VG2 MI + H – This signal is asserted since the minor axis (Y) is also positive.

6. Location of light pen hit:

   $x = 1000_8$
   $y = 0200_8$

7. Value in light pen down counter (LPDC) on L.P. hit – 2777_8.

The light pen down counter plays a key role in computing the X and Y coordinate values of the light pen hit, i.e., the values that must eventually be entered into the X/Y position registers for sampling by the PDP-11. Bear in mind that this counter is always loaded with the delta length (length of major axis) prior to commanding the vector generator to draw the vector. Furthermore, once the vector is started, the light pen begins counting down and continues (i.e., under no L.P. hit conditions) counting until a vector done condition (signal VG1 G (1) H negates) is detected from the vector generator. An all zeros condition in the LPDC only occurs during ON-to-ON vectors and coincides with the end point of a vector.

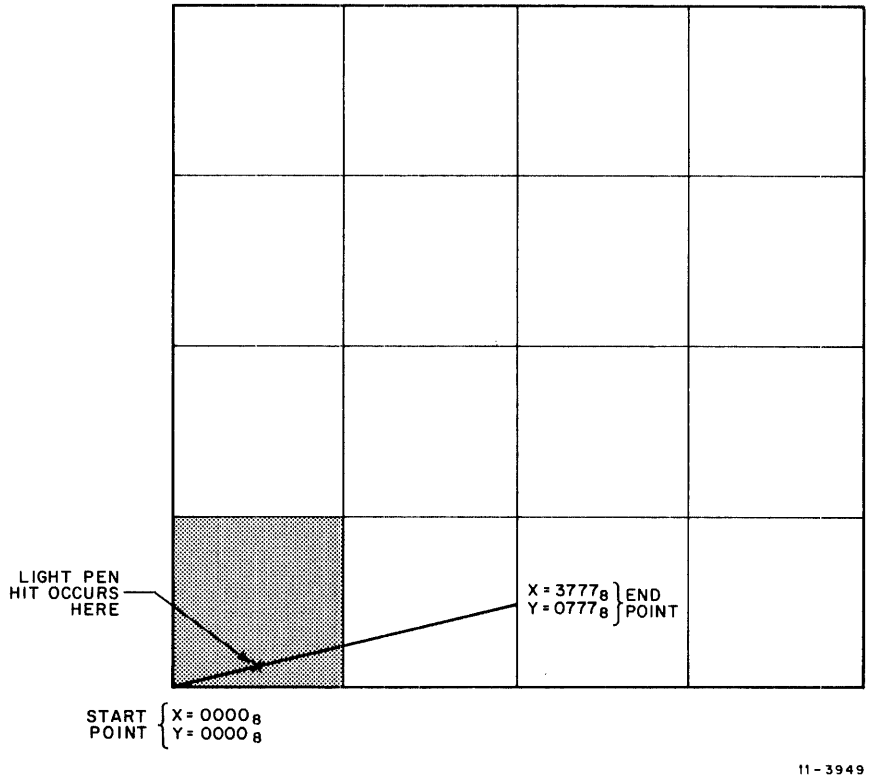


Figure 4-21 Example of Light Pen Hit on Typical Vector

The down counting action of the LPDC in effect keeps track of the major axis beam position as the vector travels along the path from start point to end point. Since all light pen hits stop the down counter at the time of the hit, computation of the L.P. hit coordinate (i.e., major axis) is relatively simple. That is, the addressed end point minus the content of the LPDC supplies the light pen hit coordinate in the major axis. Referring to the example given on Figure 4-21, the major axis L.P. hit coordinate can be calculated as follows:

3777_8 (M.A. end pt.) – 2777_8 (LPDC) = 1000_8

The count in the light pen down counter also comes into play in calculating the light pen hit coordinate in the minor axis. The formula for calculating this value is:

Y L.P. Hit = Y – (LPDC count) (Tan.)

In the vector example shown on Figure 4-21, the magnitude of the Y coordinate is one quarter that of the X coordinate, yielding a tangent value of 0.2_8 (0.25_{10}). Multiplication of the LPDC (2777_8) by the

tangent (0.2_8) takes place in the AC multiplier. (The tangent is retrieved from the vector generator Tangent register as described later.) Multiplication of these two values produces the point shown below:

```
(LPDC)      010 111 111 111
(TAN.)     .010 000 000 000
           ------------------
            000000000000
           000000000000
          000000000000
         000000000000
        000000000000
       000000000000
      000000000000
     000000000000
    000000000000
   010111111111
  000000000000
  ------------------------
  00101111111 . 110000000000
```

Only the bits to the left of the point are taken from the AC multiplier to the Y sum adder input multiplexer. That is, the 12 low order bits of the product are dropped. The resultant value (577_8) is now subtracted from the Y coordinate value to produce the following answer:

Y L.P. Hit = $777_8 - 577_8$

Y L.P. Hit = 200_8 ans.

When both L.P. hit coordinate values have been calculated, they are entered into their respective position registers for status sampling by the PDP-11. The preceding example is given to indicate the types of calculations necessary to arrive at the L.P. hit coordinates. The actual implementation of these calculations is now discussed in the description of the flow sequence that follows.

4.2.15.4 Light Pen Hit Flow Sequence – This paragraph assumes that the parameters described for Figure 4-21 remain in effect. The initial action on entry at ROM location 377_8 (sheet 9 of the flow diagram) is to clear two registers that may contain values pertinent to a current graphics calculation process. That is, values for some graphic entity (another vector) following the current vector being drawn by the vector generator. Clearing of both registers (tangent up/down counter and Successive Approximation register, SAR) is necessary since both are to be used in subsequent stages of L.P. hit processing.

From 377_8, the microprogram branches to 375_8 BUT 15_8. The branch on microtest (BUT) sampling is made here to determine if a character is in progress, meaning that the character generator is currently drawing a character. Since the example being used in this description is a vector, it can be assumed that the STC1 CHAR IN PROG H signal is not asserted. Therefore, the microprogram branches to ROM location 375_8 rather than 374_8. At 375_8, the contents of the X/Y position registers are routed to the

X/Y pre-position registers. This is done to save the addressed end points of the vector now being drawn by the vector generator.

NOTE

When the L.P. hit coordinates are calculated, they are written over the addressed end point values now in the position registers. Hence, the contents of the X/Y position registers must be saved.

Also occurring at 375_8 is the clocking down of the tangent up/down counter. Whereas this counter was cleared at the previous ROM location, a single clocking now changes the contents from all 0s to all 1s. This is necessary because at the next ROM address (376_8) the contents of the light pen down counter (LPDC) are to be loaded into the X Holding register. The route for this transfer involves the B multiplexer and the AB 12 × 12 multiplier. By making the contents of the tangent up/down counter all 1s, it ensures that the output of the A multiplexer is also all 1s.

As a result, the contents of the LPDC (2777_8) is multiplied by all 1s and the value exiting the AB 12 × 12 multiplier is the same as that of the LPDC itself. In this way, 2777_8 is entered into the X Holding register at ROM location 276_8. Also at this ROM address, all zeros are entered into the Y Holding register. This is done by simply strobing the X Holding register inputs while holding the Y sum adder input multiplexer outputs inoperative.

At the next stage (373_8) in the microprogram sequence, the contents of the X Holding register (LPDC count, 2777_8) is entered into both the X and Y holding registers. Though already contained in the X Holding register, it is necessary to enter it in the Y Holding register because the light pen down count is involved in calculating both coordinates of the L.P. hit. And to multiply the light pen down count by the tangent value, it is necessary first to enter it into the Y Holding register.

The route for transferring the contents of the X Holding register to the Y Holding register is unique. The transfer path involves: the X Holding register/Delta X register multiplexer; the MDX 4 × 12 multiplier; and the C multiplexer. From the latter circuit, the light pen down count is multiplied by all 1s as supplied from the output of the A multiplexer. (This is because the tangent up/down counter still contains all 1s.) The output of the AC 12 × 12 multiplier is then taken through the Y sum adder input multiplexer, then through the Y sum adder for entry into the Y Holding register.

The 2777_8 LPDC value is also taken through the X sum adder input multiplexer and X sum adder for entry into the X Holding register. At this time, then, both holding registers contain the 2777_8 LPDC value.

The microprogram now sequences to 372_8 where the tangent value is retrieved from the vector generator. This is done by gating the VG2 TAN MSB signal through the SAR input select logic for entry into the SAR itself. The transfer is serial since the graphics calculation logic supplies serial shift pulses (STC4 CLK SAR L) to the vector generator Tangent register until such time as the SAR asserts the conversion complete (MD1 CC L) signal. The latter occurs after 12 clock pulses to indicate that the tangent value is now contained in the SAR. (Assertion of MDI CCL also halts further clock pulses to the vector generator Tangent register.)

Following transfer of the tangent, the sequence branches from location 372_8 to 371_8 BUT 10_8. The BUT sampling here determines which coordinate value is larger (X or Y) and then modifies the ROM address accordingly. Bear in mind that in calculating the smaller L.P. hit coordinate, the light pen down counter value (2777_8) is multiplied by the tangent (0.2_8). In the example given in Figure 4-21, the Y coordinate is the smaller value. Therefore, X is greater than Y (signal VG2 X>Y is asserted) to cause the microprogram to branch to ROM address 370_8. If the converse were true (Y>X), the microprogram branches to location 371_8.

At location 370_8, the contents of the X Holding register are transferred back to the X Holding register via the X sum adder input multiplexer and X sum adder. The 2777_8 value from the Y Holding register is now multiplied by the tangent (0.2_8) contained in the SAR. Routing for this multiplication process is as follows:

1. Y Holding register through Y Holding register/Delta Y register multiplexer through MDY 4 × 12 multiplier through C multiplexer to input of AC 12 × 12 multiplier.

2. Content of SAR (0.2_8) is taken through A multiplexer to input of AC 12 × 12 multiplier.

Multiplication of the two values now takes place and the product (0577_8) is sent to the Y Holding register via the Y sum adder input multiplexer and Y sum adder. At this time then, the X Holding register contains 2777_8 and the Y Holding register contains 0577_8.

Additional activities carried out at location 370_8 are:

1. The tangent value is reshipped to the vector generator (TAN → ANALOG). This is necessary since the tangent value was clocked out of the vector generator Tangent register at ROM location 372_8. Since the final segment of the vector (i.e., following the L.P. hit) must be drawn on receipt of the Resume command, the tangent value must be resupplied to the vector generator.

2. The larger value going through the X/Y sum adders (2777_8) is entered into the Delta Length register. That is, via the X/Y sum comparator and X/Y sum multiplexer. This is done in anticipation of calculating the new delta length (i.e., for second segment of vector) as is discussed later.

The microprogram now branches to 367_8 BUT 07. At this time, the BUT multiplexer (STC1) samples the signs of the major and minor axes to determine vector direction. In the example on Figure 4-21, the signs of both axes are positive. In such a case, the microprogram branches to 364_8 where the following arithmetic operations occur:

1. The contents of the X Holding register (2777_8) are subtracted from the X Position register (3777_8) and then entered into the X Position register. Now, the X Position register contains the major axis coordinate (1000_8) for the L.P. hit.

2. The contents of the Y Holding register (0577_8) are subtracted from the contents of the Y Position register (0777_8) and then entered into the Y Position register. The Y Position register now contains the minor axis L.P. hit coordinate (0200_8).

Because both vector coordinates were positive, a subtraction process is used to calculate both L.P. hit coordinates. Assume however, that the same length vector is being drawn but in a negative direction. In such a case, the addressed end point in the Y axis would be $(-)1000_8$ rather than $+0777_8$. If it is further assumed that the L.P. hit occurs in the respective position, then the L.P. hit coordinates would be X = 1000_8, Y = 1577_8 (i.e., 0000_8 minus 0200_8). The arithmetic calculations shown at ROM location 365_8 (minor axis negative) would produce this set of coordinates if the Y component were negative rather than positive going. The range of calculations implemented by ROM location 367_8 through 365_1 are capable of accommodating all vector direction possibilities.

The Successive Approximation register (SAR) and tangent up/down counter are also cleared at each ROM location. This is done in anticipation of the search for edge procedure undertaken at the next processing stage.

At location 357_8, two major functions are implemented:

1. Load the vector generator D/A converters (LOAD DAC) with the L.P. hit X/Y coordinates – This positions the DACs at the start point of the second segment of the vector.

2. Search for edge – This action involves those boxes on sheet 3 of the flow diagrams that define the find edge operation (i.e., 103_8, 106_8, and 107_8) for a positive going vector. Searching for the edge is necessary here since the calculated value (0777_8) at the output of the X sum adder represents the delta length for the remaining segment of the vector; this value must be provided to the vector generator to draw the remaining segment, i.e., just as it is necessary to draw any normal vector.

Following the search for edge operation, the microprogram branches to 353_8 where the L.P. interrupt flag is set (STC4 INTER PULSE L is asserted to generate STC7 EDGE + LP + SW PULSE as a high). Also, at this time, the Delta Length register is loaded with 0777_8 preparatory to transferring the delta length (for the remaining segment) to the vector generator. A third operation here is to load the LPDC with the newly calculated delta length. This is done so that should a second light pen hit occur (on the second segment of the vector following the Resume command), the graphics calculation logic simply recycles through the L.P. hit microprogram with no ill effect.

At the next ROM location, 352_8, two housekeeping operations (clear tangent up/down counter and SAR) are performed and the down counter is loaded with 1770_8. This is a large time out value that requires approximately 12 μs before an all zeros condition is reached. Since the down counter sets the busy flip-flop (which is also set by the interrupt flag), it ensures that the vector generator D/A converters have sufficient time to stabilize at the L.P. hit coordinates before the Resume command is received and the process of drawing the second segment is initiated.

NOTE

It is unlikely that a Resume command would be received within a 12 μs time span. However, should it happen, the down counter holds the busy flip-flop set until the time out period is complete.

At ROM location 347_8, clocking of the down counter begins and signal STC4 TRANS DATA → VG L is asserted. The latter action transfers the delta length for the second segment of the sector to the vector generator. Conditions are now set up to draw the second segment of the vector on receipt of the Resume command.

At ROM location 346_8, the busy flip-flop is continuously checked until the cleared status is detected. When cleared, it indicates that the Resume command has been received allowing the microprogram to initiate the following:

1. Assert signal STC4 VEC GO L (START VEC) to command the vector generator to begin drawing the second segment of the vector.

2. Load the X and Y position registers with the addressed end point of the original vector. These values were saved at the start of the L.P. hit sequence (375_8). Since the Resume command has now been received, meaning that the PDP-11 has already sampled the L.P. hit coordinates, the original coordinates can now be loaded back into the X/Y position registers.

At the final location in the L.P. hit microprogram, signal STC5 ALL DONE L is asserted to initiate sequencing shutdown (Paragraph 4.2.8.5).

4.2.16 Edge Interrupt Processing

When an ON-to-OFF vector is being processed and the edge interrupt flip-flop is set, the edge interrupt flow is entered. This sequence is shown on sheet 11 of the M7053 Program Flow Diagram. The principal objective of this routine is to return the X/Y edge coordinates to the X/Y position registers for sampling by the PDP-11. This holds true for all vector types traversing the screen edge; that is, ON-to-OFF, OFF-to-ON and OFF-to OFF.

The edge interrupt routine is always entered from ROM location 312_8 on sheet 3 of the flow diagram. At this location, the busy flip-flop is checked to determine if the vector/character generator is still in the process of drawing a graphic entity. In other words, drawing of the previously processed graphic entity must be completed before the edge interrupt routine is entered.

Two things occur on entry into the microprogram at ROM location 205_8 (sheet 11):

1. The interrupt signal is asserted to inform the PDP-11 of an interrupt condition. Also at this time, signal STC7 EDGE + LP + SW LEVEL L asserts to set the busy flip-flop (STC2 BUSY (1) H).

2. The silo memory address is incremented so that the parameters (intensity level and menu) appropriate to the graphic entity traversing the edge are available to the read status multiplexer; i.e., for status sampling purposes.

The microprogram next moves to ROM address 204_8 where the contents of the pre-position registers are routed through the X/Y sum adders. This is to distinguish between vector types. The values in the pre-position registers represent the addressed end point of a vector and can fall in one of two categories:

1. The X/Y coordinates define an off screen address and signal STC3 SUM EDGE L asserts on routing the contents of the pre-position registers through the X/Y sum adders. Since this signal asserts only when processing ON-to-OFF vectors, it means that the edge transition occurred during an ON-to-OFF vector.

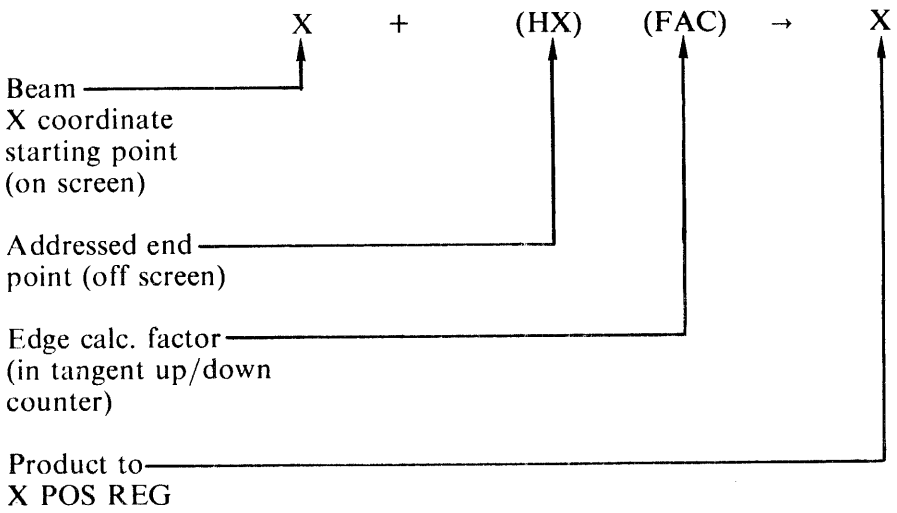
2. The X/Y coordinates define an on screen address as is the case when OFF-to-ON or OFF-to-OFF (with visible segment) vectors are being processed. In these two cases, STC3 SUM EDGE L does not assert on routing the contents of the pre-position registers through the X/Y sum adders.

It should be noted at this time that in the latter two cases, the X/Y position registers currently contain the coordinates of the point where the vector re-enters the screen edge. That is, they are already available for PDP-11 status sampling on detection of the interrupt. In the case of an ON-to-OFF vector, these coordinates must now be calculated to make them available to the central processor. Hence, if signal STC3 SUM EDGE L asserts, the microprogram branches to ROM location 202_8 following receipt of the Resume command. If this signal does not assert, the path at ROM location 203_8 is entered.

Two additional activities undertaken at ROM location 204 are the assertion of the edge flag signal (STC5 EDGE FLAG H) and the strobing of the edge indicator. Signal STC5 STROBE EDGE IND L is asserted to strobe the edge indicator flip-flop (STC3 EDGE INDICATOR (1) H). However, the flip-flop sets only if signal STC3 SUM EDGE L is also asserted; that is, at those times when the graphics calculation logic is processing an ON-to-OFF vector. Consequently, the PDP-11, by sampling both the edge flag status and the edge indicator status, can determine whether the beam is off screen and going on screen or if it is on screen and going off.

The microprogram now checks the busy flip-flop status before proceeding on to the next ROM location (i.e., 202_8 or 203_8). Once the PDP-11 recognizes the interrupt and issues a Resume command, the busy flip-flop is cleared and sequencing resumes.

4.2.16.1 Edge Interrupt Processing for ON-to-OFF Screen Vectors – The first objective when processing an ON-to-OFF vector is to calculate the coordinates where the beam exits the screen area. From location 202_8, the program proceeds directly to one of four ROM locations (depending on sign bits) where the applicable arithmetic operations are carried out. For example, if both sign bits are positive, the edge coordinates are calculated at ROM location 213_8. The expression for the X coordinate indicates the following arithmetic operation.



The X register initially holds the X coordinate for the on screen starting point. This value is added to the product of the X coordinate end point times the edge calculation factor (calculated at ROM locations 103_8 and 106_8, sheet 3, and now contained in the tangent up/down counter). The value exiting the X sum adder is the X coordinate for the point where the beam exits the screen. A similar operation is executed to determine the Y coordinate. By placing the X/Y coordinates in the X/Y position registers, they are now available for sampling by the PDP-11 via the read status multiplexer. At this time then, the microprogram asserts the interrupt signal again as an indication that the edge coordinates are now available for status sampling.

The microprogram now proceeds to location 206_8 where the edge flag signal is asserted again. The status of the busy flip-flop is continuously checked to determine if the PDP-11 has sent the Resume command.

NOTE
The Resume command clears the STC7 EDGE + LP + SW LEVEL H flip-flop which, in turn, allows the busy flip-flop to be clocked clear.

On receipt of the Resume command, the vector end point coordinates (defining a point somewhere off screen in the virtual display area) are now taken from the pre-position registers and strobed into the X/Y position registers. These coordinates are now available for PDP-11 sampling if desired.

The microprogram now proceeds to location 221_8, where the following are effected:

1. The vector generator D/A converters are cleared (signal STC4 CLR DAC L asserts) – Since the graphics calculation logic recognizes the beam as going off screen, the beam (blanked) must now be positioned at center screen. This places the beam in optimum position to deflect to the next vector returning on screen.

2. A value of 1000_8 is loaded into the down counter – This is a time out value used to ensure that the beam has sufficient time to return to center screen before another graphic entity is processed by the graphics calculation logic.

NOTE

If the ON-to-OFF vector had a start point right next to the screen edge, it means that a time period equivalent to one-half screen deflection must elapse before the beam returns to center screen. The value entered into the down counter is sufficiently large to cover this deflection period.

At the final ROM location, 214_8, the silo memory is resynchronized to its starting read/write address values. This is simply a housekeeping function which does not affect the sequence of reading/writing the silo memory. Next, the process of clocking the down counter is initiated to begin the beam deflection time out period. When it is complete, and the busy flip-flop clears, the microprogram proceeds to location 104_8 to shut down the sequencing process.

4.2.16.2 Edge Interrupt Processing for OFF-to-ON and OFF-to-OFF Vectors – When signal STC3 SUM EDGE L remains negated ($\overline{\text{SUM EDGE}}$), it means the beam is going on screen. Hence, the microprogram switches from 204_8 to 203_8; that is, after the Resume command has been received to clear the busy flip-flop. It can be assumed that at the time that switching to ROM location 203_8 occurs, the PDP-11 has already retrieved the screen edge coordinates for the vector being processed. This is because the edge coordinates have already been calculated (and are in the X/Y position registers) before the edge interrupt routine is initially entered. (See flow sequence on sheets 4 and 5.) At location 203_8 then, it is only necessary to load the on screen end point coordinates into the X/Y position registers. Again, this is done to make the on screen coordinates available for PDP-11 sampling. Once the position registers are loaded, the interrupt signal is generated and the microprogram proceeds to location 205_8.

Here the vector generator D/A converters are loaded with the X/Y Position register values. Although the visible portion of the vector is not drawn under edge interrupt conditions, the beam must be deflected to its on screen address. Since the beam is currently at center screen (having been deflected there by some prior ON-to-OFF vector), the maximum distance to any one screen point is one-half the viewing area. Because of this, 1000_8 is loaded into the down counter to time out the period required to deflect the beam to its on screen address. The busy flip-flop is now checked to detect the Resume command from the PDP-11. When it occurs, the microprogram proceeds to location 214_8 where the shutdown sequence is initiated.

4.2.17 Vector Generator Simplified Block Diagram Discussion

Figure 4-22 shows a block diagram of the VT48 vector generator circuit. The fundamental purpose of this circuitry is to draw constant velocity vectors on the working surface (major area) of the VR48

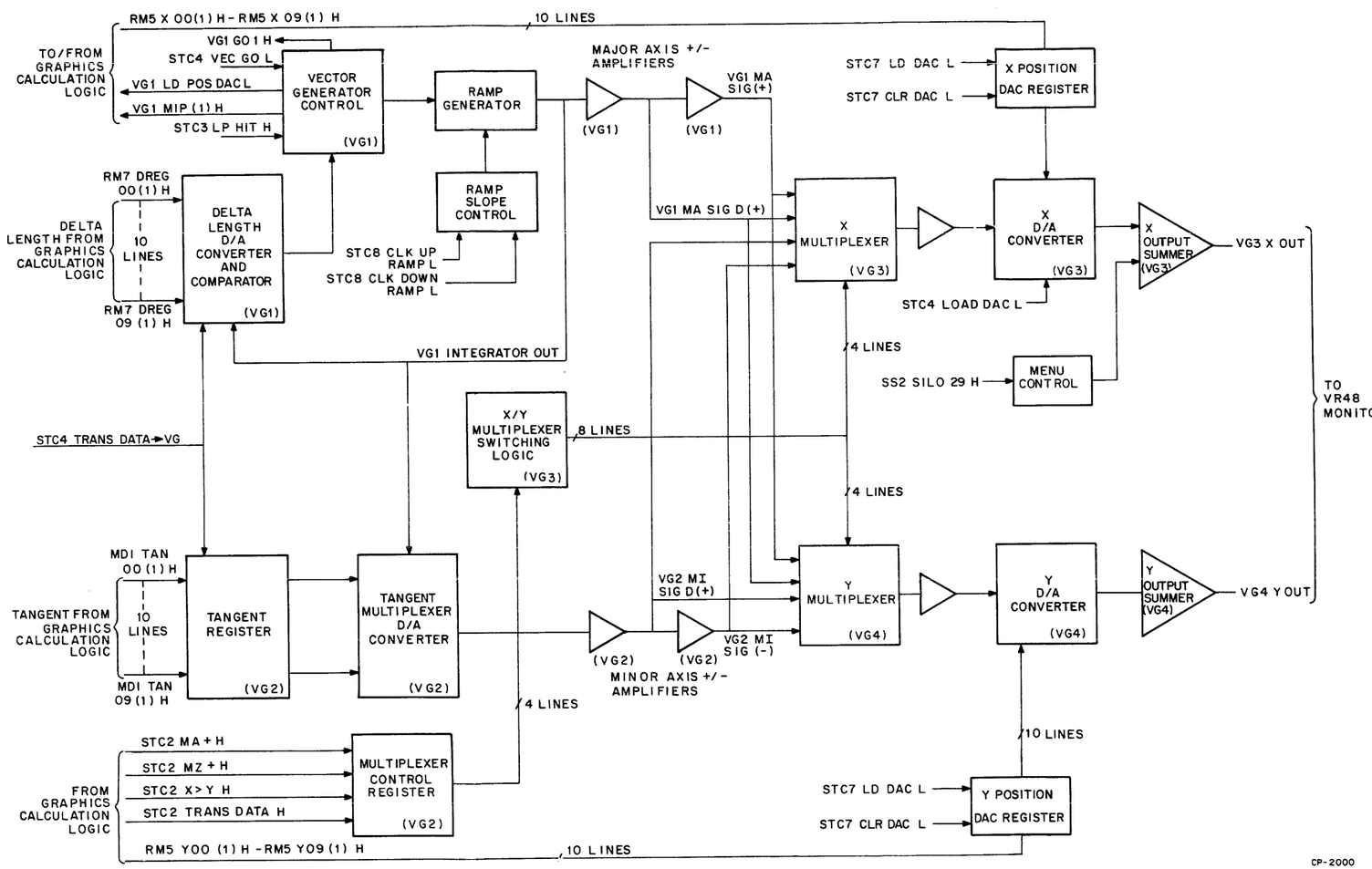


Figure 4-22 Vector Generator, Simplified Block Diagram

Display Monitor. All inputs necessary for vector generation are supplied from the graphics calculation logic. In general, the inputs supplied from this logic can be broken down into two categories:

1. X/Y beam position update signals – Before the start of the first graphic instruction in the display file, the output of the D/A converter is such that the beam (in the blank state) is positioned at the center of the screen. A point instruction can now be executed to position the beam at any desired location on the display frame. Point instructions for on screen points simply provide new 10-bit X/Y position data to update the respective X/Y position DAC registers. After the first vector is drawn (and after each drawable vector thereafter), the 10-bit X and Y positioning signals are received for updating the DAC registers to the addressed on screen end point. In this way, the end point of one vector serves as the initial position for the next.

2. Vector drawing signals – Generation of a vector requires the input of four sets of signals as follows:

    a. Delta length – This is a 10-bit input that defines the length of the major axis. The larger of the delta X/delta Y values (conveyed by the vector instruction) is the major axis.

    b. Tangent – This is a 12-bit value that is equivalent to the ratio of the minor axis to the major axis.

    c. Major axis sign (+, –).

    d. Minor axis sign (+, –).

In addition to the above, certain control and loading signals are received. These are discussed in the subsequent paragraphs.

4.2.17.1 Ramp Generation – The 10-bit major axis (delta) length is applied to the delta length D/A converter and comparator circuit. Loading is effected by STC4 TRANS DATA → VG under control of the graphics calculation microprogram. Assertion of the loading signal cannot occur at times when the vector generator is busy drawing a vector. This is because the graphics calculation logic microprogram constantly monitors the vector generator move-in-process signal (VG1 MIP (1) H). When asserted, this signal causes the microprogram to withhold loading of the delta length until the VG1 MIP (1) H signal is negated after drawing a vector.

Loading of the delta length occurs before the graphics calculation logic sends the command to begin drawing the vector. This means that the output of the delta length D/A converter has time to stabilize before drawing of the vector begins. The voltage value developed at the output of the delta length D/A converter is directly proportional to the binary input. It also represents the cutoff voltage for the ramp generator circuit.

Generation of a vector is initiated when the graphics calculation logic asserts the STC4 VEC GO L signal to the vector generator control logic. This signal is used to trigger a one-shot multivibrator which, after a microsecond delay, enables the ramp generator. The latter circuit consists of an integrator that generates a positive ramp voltage for as long as it is enabled. This ramp voltage (VG1 INTEGRATOR OUT) represents the major axis of the vector. A comparator circuit (within the delta length D/A converter and comparator block) continuously samples the magnitude of the ramp. When the ramp voltage attains the magnitude of the delta length D/A converter, the comparator circuit issues a cutoff signal to the vector generator control logic. As a result, the ramp generator is now inhibited since the vector drawing process is complete.

For a full display frame vector, the graphics calculation logic presents all 1s in the 10-bit delta length value. This in turn results in a 10 V output from the delta length D/A converter. In such cases, the ramp generator produces a ramp ranging from 0–10 V before cutoff occurs.

4.2.17.2 Ramp Slope Control – The vector generator employs a special circuit to control the speed at which the ramp is generated. Monitoring of the ramp speed is carried out in the graphics calculation logic. Here, a down counter (light pen down counter, loaded with the delta length prior to the start of each ramp generating period) is used to determine when the end of a vector should occur. Down counting of the LPDC is accurate since the counter clock is crystal controlled.

Clocking of the LPDC is enabled when signal VG1 GO 1 H is asserted at the start of the ramp generation process. Clocking is inhibited when the same signal is negated at the end of the ramp generation period. If the down counter counts down to zero before signal VG1 GO 1 H negates, it means that ramp generation is too slow. In such a case, the graphics calculation logic issues signal STC8 CLK UP RAMP L. In turn, the ramp slope control changes its output voltage to increase the rate of ramp generation. Conversely, if the LPDC contains some value greater than zero when signal VG1 GO 1 H is negated, it means that ramp generation is too fast and must be clocked down. Under these conditions, signal STC8 CLK DOWN RAMP L is asserted to alter the output voltage of the ramp slope control circuit.

4.2.17.3 Tangent Register and Multiplying D/A Converter Circuit – The second set of inputs required by the vector generator in drawing vectors is the tangent. This is loaded into the tangent register by signal STC4 TRANS DATA → VG. As stated earlier, the tangent is equal to the ratio of the minor axis to the major axis. For all 45-degree angles, the delta X and delta Y values are equal, and this means that a 10-bit tangent value of all 1s is supplied from the graphics calculation logic to the Tangent register.

The 10-bit output from the Tangent register is applied to a multiplying D/A converter that also receives the ramp signal from the ramp generator. Note that the output of the multiplying D/A converter is applied to a pair of minor axis amplifiers while the output of the ramp generator is applied through a pair of major axis amplifiers. Under conditions where the delta X/delta Y values are equal and a 45 degree angle must be drawn, the output of the multiplying D/A converter is exactly the same as that from the ramp generator itself. This means that the X/Y deflection voltage inputs to the VR48 deflection amplifiers are the same and a vector (regardless of specified length) is drawn at a 45 degree angle.

The reason why the output from the multiplying D/A converter is the same as the ramp generator input lies in the fact that the Tangent register contains all 1s. Put in another way, it means that the developing ramp voltage is being multiplied by a value of one and therefore no attenuation of the ramp voltage occurs. When delta lengths are unequal, the Tangent register is loaded with some value less than all 1s (i.e., some fraction). Consequently, some attenuation of the ramp voltage occurs. Therefore, the output from the tangent multiplying D/A converter is produced as a linear ramp that is proportionately smaller than that produced by the ramp generator.

4.2.17.4 Multiplexer Control Register and X/Y Multiplexer Swiching Logic – The graphics calculation logic supplies three signals that select a pair of major/minor axis ramps for transfer through the X/Y multiplexers to the respective X/Y D/A converters. These signals are applied to the Multiplexer Control register and consist of:

1. STC2 MA + H – This signal is asserted when the major axis of the vector is positive.
2. STC2 MI + H – This signal is asserted when the minor axis of the vector is positive.
3. STC2 X > Y H – This signal is asserted when the delta X value is greater than the delta Y value.

A fourth signal (STC5 TRANS DATA L) is also applied to this register. This signal is asserted only when the graphics calculation logic is processing vectors. It is used to load the bit pattern presented by the preceding three signals and is asserted prior to assertion of the STC VEC GO L signal; that is, before the vector generator control logic commands the ramp generator to begin the vector draw process.

One of the four output lines from the Multiplexer control register is used to enable or turn on the X/Y multiplexer switching logic. The other three outputs are used to effect X/Y multiplexer selection (switching) and have mnemonics similar to the inputs. That is, VG2 MA + H, VG2 MI + H, and VG2 X > Y H.

The X/Y multiplexer switching logic samples the latter three inputs to determine the assertion pattern. It is this pattern that determines which pair (one for X and one for Y) of eight possible outputs is asserted to the X/Y multiplexers for major axis (+, -)/ minor axis (+, -) input selection. Figure 4-23 shows three types of vectors, the input pattern to the X/Y multiplexer switching logic, and the inputs selected for the X/Y multiplexers.

In the vector shown in part a, both delta X/delta Y components are positive and the delta X magnitude exceeds delta Y. This results in the larger or major (positive) ramp being fed through the X multiplexer for transfer to the X output summer circuit. The Y delta component, being the smaller, causes selection of the minor ramp (also positive) being fed through the Y multiplexer for application to the related summer circuit.

In part b of Figure 4-23, the delta X/Y components both remain positive, but now the Y delta magnitude exceeds that of X. As a result, the inputs to the X/Y multiplexers are now switched. That is, the major axis ramp is now fed through the Y leg of the output circuit while the minor axis is fed through the X leg.

Part c of the illustration shows the conditions where one of the negative going input ramps is selected. In the part c vector, the Y delta component is negative going and X is greater than Y. As a result, the positive going major ramp is selected for transfer through the X leg while the negative going minor ramp is fed through the Y leg.

The outputs of the X/Y multiplexers are taken through respective summer circuits which, in turn, provide the ramp voltage inputs to the VR48 deflection amplifiers.

4.2.17.5 X/Y Position DAC Registers and X/Y D/A Converters - As mentioned earlier, the X/Y position DAC registers are updated at the end of each on screen vector to hold the beam at the addressed beam end point. The 10-bit X/Y position values, when loaded into the X/Y DAC registers, are converted to proportional analog voltage outputs by the respective D/A converters. The outputs from the latter circuits are in turn taken through their respective summer circuits to maintain the beam at the addressed end point.

Loading of the X/Y DAC registers is under control of the graphics calculation logic microprogram. Loading occurs when signal STC7 LD DAC L is asserted.

Two processing conditions call for loading these registers as follows:

1. On completion of drawing an on screen vector - When the vector generator finishes drawing a vector, the vector generator control logic issues signal VG1 LD POS DAC L. In response, the graphics calculation logic asserts STC7 LD DAC L to load the X/Y position values into the X/Y position DAC registers.

X/Y MULTIPLEXER SWITCHING LOGIC INPUT PATTERN	SELECTED X MUX INPUT	SELECTED Y MUX INPUT
VG2 MA + H VG2 MI + H VG2 X > Y H	VG1 MA SIG(+)	VG2 MI SIG D(H)
VG2 MA + H VG2 MI + H $\overline{\text{VG2 X > Y H}}$	VG2 MI SIG D(+)	VG1 MA SIG(+)
VG2 MA + H $\overline{\text{VG2 MI + H}}$ VG2 X > Y H	VG1 MA SIG(+)	VG2 MI SIG(−)

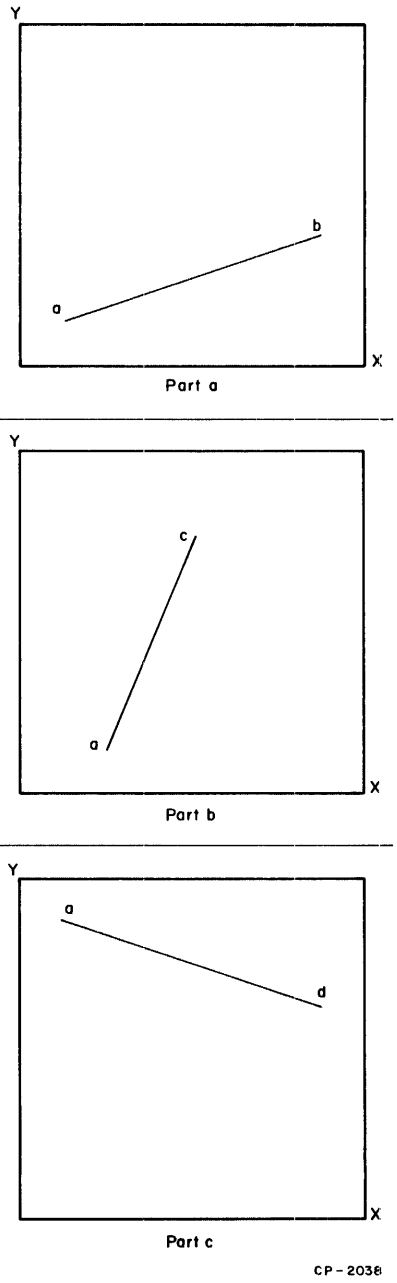


Figure 4-23 Examples of Vector Select Multiplexing

2. When processing on screen points – Under this conditions, the ramp generating circuits are held inoperative while the X/Y position DAC registers are updated for every point processed. If the points are within the working surface (displayable), signal STC7 LD DAC L is asserted to enter the X/Y position data into the respective DAC registers. If the point is outside the viewable area signal STC7 CLR DAC is issued to clear the DAC registers.

4.2.17.6 Menu Area Selection – The VT48 incorporates a feature for displaying data to the right of the working surface. This feature is program-selectable via the load status A instruction and when selected, sets bit 29 in silo memory. Hence, whenever a graphic entity (usually a character) is being processed with the menu area selected, signal SS2 SILO 29 H is asserted. This enables the menu control circuit whose output in turn alters the reference voltage output of the X output summer circuit. As a result, the graphic entity is displayed (drawn) in the menu rather than working surface area.

4.2.18 Character Generator Simplified Block Diagram Discussion

Figure 4-24 is a block diagram of the circuits comprising the VT48 character generator. Characters, in seven bit ASCII code, supplied from the character register (DSR6) are constantly applied at the input address conversion ROMs. This logic samples the incoming character code and provides a 12-bit coded output to the Address register/counter. Though every ASCII character is decoded by the address conversion ROMs, loading of the 12-bit code occurs only when the timing and control logic issues the load (CG2 LOAD L) signal. Generation of this signal is under control of the start character generator signal supplied from the graphics calculation logic. The latter logic controls the amount that the X/Y position D/A converters (within the vector generator) must be updated between characters. This is necessary because spacing between characters varies in accordance with the selected character scale. It should also be understood that signal STC5 START CHAR GEN L is asserted only for drawable characters. This precludes the possibility of unwanted control character patterns being loaded into the Address register/counter.

With the issuance of the CC2 LOAD L signal, the 12-bit pattern, (whose contents represent the starting address for the character to be drawn), is loaded into the Address register/counter. At the same time, the timing and control logic asserts the character active signal (CG2 CHAR ACT (1) B L) to the graphics calculation logic. This signal is asserted to indicate that the character is being processed and remains asserted until after the last stroke in the character has been drawn.

The 12-address line output of the Address register/counter now selects the starting or first stroke in the character to be drawn. A string of strokes is used to draw each character on the VR48 screen. The length of the string of strokes is a variable depending on character makeup. The letter X, for example, consists of two straight lines allowing the use of longer and therefore fewer strokes. The numeral 8, on the other hand, is complex and requires many short strokes for its formation.

Sequencing through the strokes is accomplished by the increment signal (CG2 INCR H) which is issued after processing each stroke until the end-of-character code is detected. This condition is detected by the interpreter ROMs which asserts signal CG2 EOC ? H on detection of the end-of-character code. Signal CG2 EOC ? H is used by the timing and control logic to shut off further stroke processing. That is, signal CG1 INCR H is negated so that incrementing of the character stroke stream ROM address is discontinued. Similarly, signal CG2 CHAR ACT (1) B L is now negated to inform the graphics calculation logic that processing of the character is now complete.

For each stroke, the character stroke stream ROMs provide a 6-bit output to the interpreter ROMs. Three outputs are for the X coordinate and three are for Y. Each group consists of a 2-bit stroke length value and an associated sign bit.

In addition to detecting the end-of-character code, the interpreter ROMs are used to change the coded output in such a way that the character is drawn 90 degrees from horizontal. Such action results only when the DSR4 CHAR ROTATE 00 (1) H signal is asserted. This is supplied from the status/parameter registers.

The X and Y buffers are inserted between the interpreter ROMs and the D/A converters to ensure zero current flow through the D/A converters between character strokes. That is, at times when the buffers are disabled by the timing and control logic after each stroke. The buffers are disabled for a

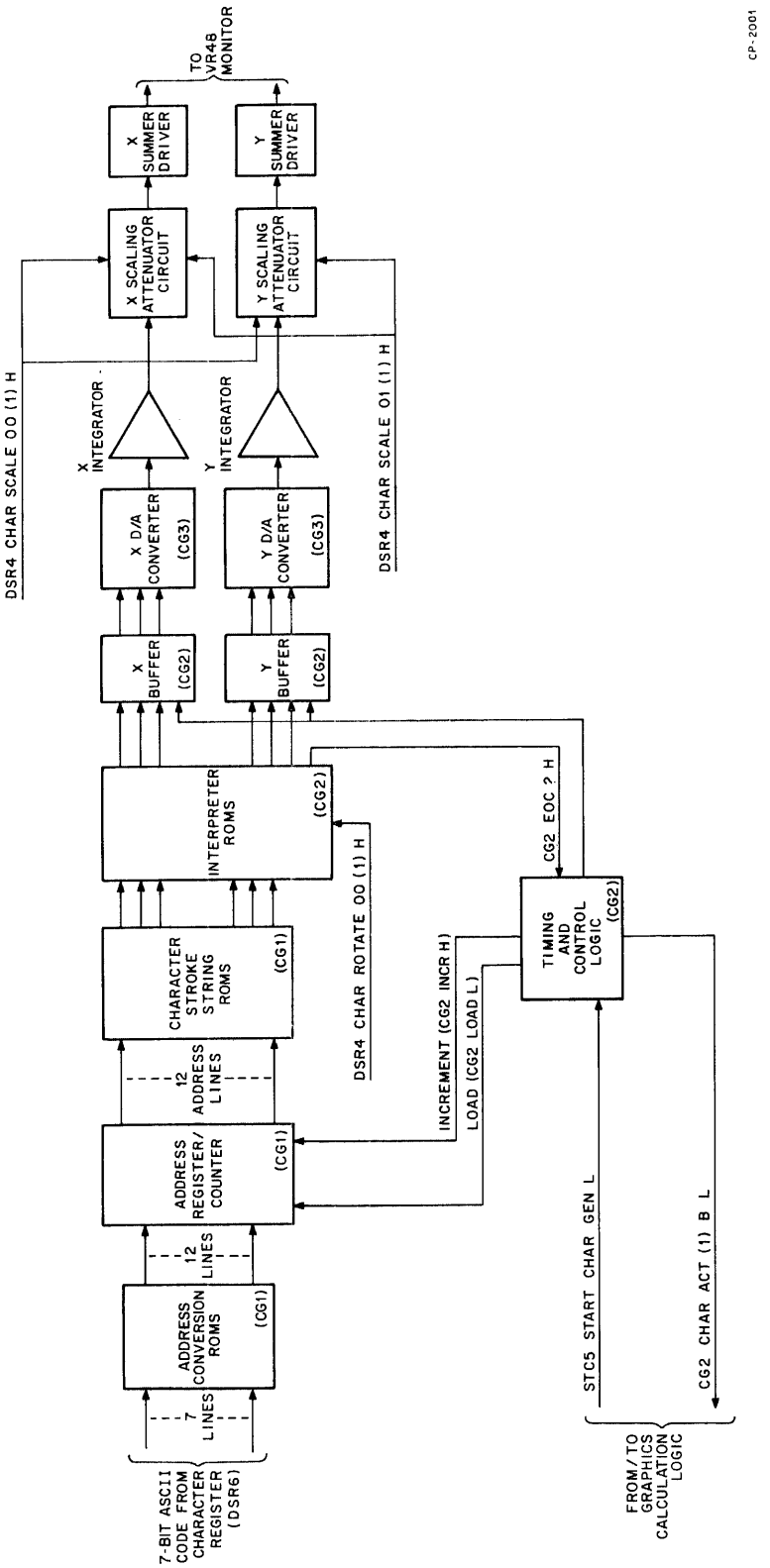


Figure 4-24 Character Generator, Simplified Block Diagram

period of 300 ns after each stroke to allow for beam settling. During this period, the output from each buffer defines the zero current flow condition. This is an important factor in VR48 beam positioning precision.

The timing and control logic determines the amount of time that the buffers are allowed to present their coded outputs to the D/A converters. This in turn means that the amount of time that the integrators are permitted to integrate is also governed from the timing and control logic. In effect then, the timing and control logic acts to scale individual stroke lengths. This feature is incorporated to minimize the number of ROM locations required for the overall character stroke string. It should not be confused with the character scale value loaded into the status/parameter registers by the load status C instruction.

The X and Y integrators present their outputs to respective scaling attenuator circuits. These circuits receive the 2-bit character scale from the status/parameter registers. The coded inputs make the scaling circuits attenuate the ramped voltage outputs from the integrators. In this way, it is possible to select four different attenuation levels and consequently four different character sizes on the VR48 screen.

The summer driver circuits take the voltages from their respective attenuator circuits and amplify them so that they can be driven over a control cable to the VR48 Display Monitor.

4.2.19 Stack/Silo Addressing and Control Logic

The subsequent paragraphs describe the logic used to address and control the stack/silo memories. The stack control logic is activated under the following system conditions:

1. Execution of JSR and JSR relative instructions
2. Execution of POP and POP Restore instructions
3. Addressing of the Stack Status register

In contrast to the above, the silo memory control logic is activated only during execution of the various graphic mode instructions. Thus the two sets of control logic are described separately.

4.2.19.1 Stack Memory Control Logic – Figure 4-25 shows a block diagram of the stack memory control logic. To understand stack memory addressing techniques, it is first necessary to understand the makeup of the stack memory itself. The stack memory is 32 bits wide by 16 bits deep – meaning that there are 16 possible addressable locations. Because the status data supplied to the stack memory contains 64 bits of information, it is necessary to update the stack memory by two (two sequential writes/reads) for each JSR/POP instruction.

Addressing of stack memory locations is effected via the stack pointer, which has a 4-bit address output. During initialization, the stack pointer is cleared to the all zeros state while the top of the stack detector, consisting of a single flip-flop, is jammed preset. This readies the addressing logic for execution of the fist JSR/JSR relative instruction in the display file.

NOTE

Should the program attempt to execute a POP/POP restore instruction at this time (i.e., with the top of stack flip-flop set), the stack violation detection would assert the SSC1 STACK UNDERFLOW (1) H signal causing an interrupt.

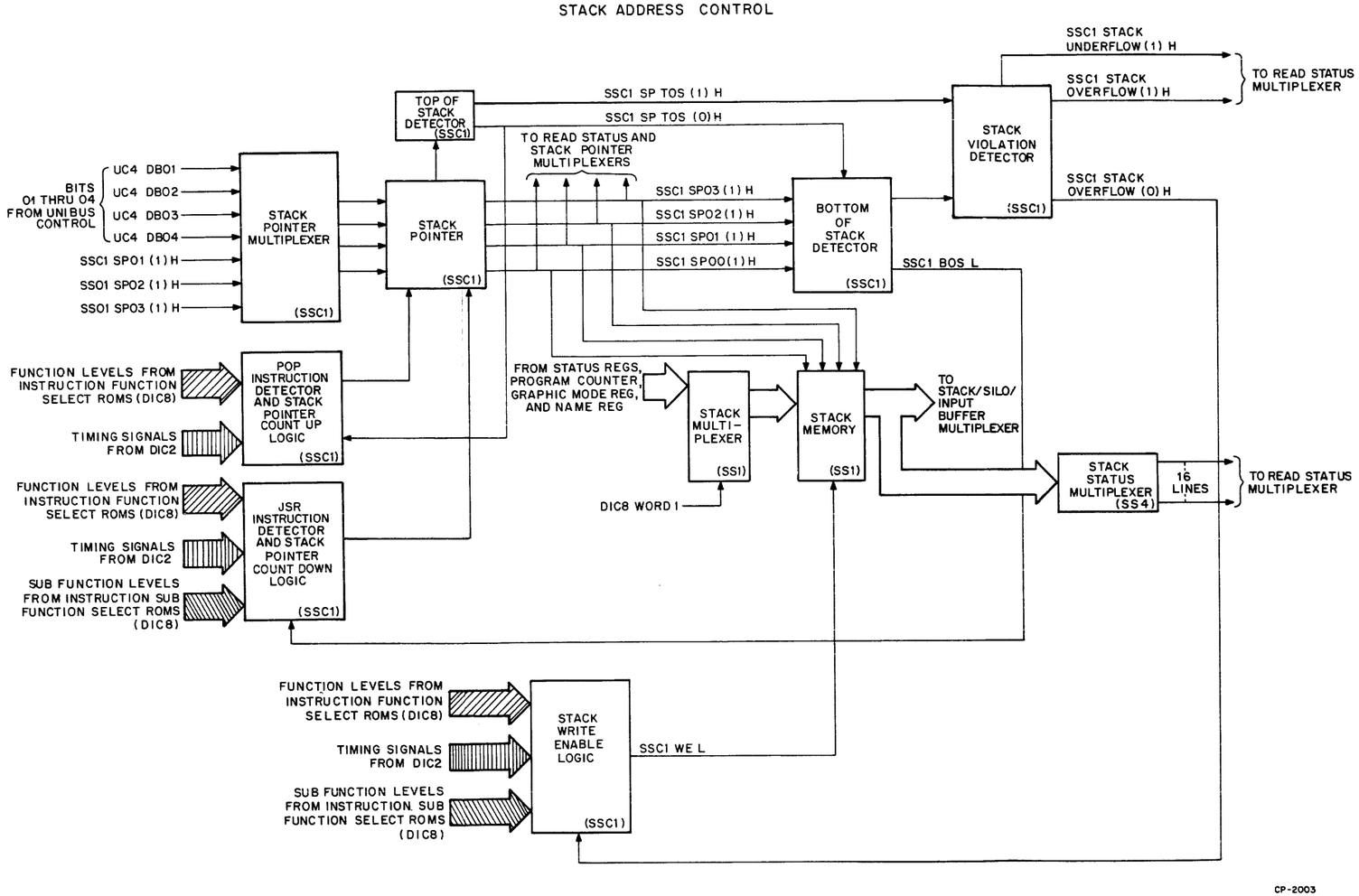


Figure 4-25 Stack Memory Addressing Logic, Block Diagram

Execution of the first JSR/JSR relative instruction clears the top of stack detector flip-flop and also effects the following:

1. Asserts the output of the JSR instruction detect and stack pointer countdown logic. This clocks the stack pointer to 1111_2 or the first stack memory write address.

2. Asserts signal SSC1 WE L at the output of the stack write enable logic. At this time, the first 32 bits of status/parameter data are written into stack memory.

NOTE

The timing signals applied to the JSR instruction detect and stack pointer countdown logic always update the stack pointer (i.e., to the proper address) before the output from stack write enable logic is asserted.

3. Asserts the output of the JSR instruction detect and stack pointer countdown logic a second time. This clocks the stack pointer to 1110_2; that is, the address for writing the second 32 bits of status/parameter dates.

4. Asserts the output of the stack write enable logic a second time to write the second 32 bits into stack memory.

NOTE

The stack multiplexer selects the low-order 32 bits when signal DIC8 WORD 1 H is negated. This multiplexer selects the high-order 32 bits when signal DIC8 WORD 1 H is asserted.

Clearing the top of the stack detector upon execution of the first JSR instruction causes signal SSC1 SP TOS (0) H to switch high. This in turn enables the POP instruction detector and stack pointer count up logic meaning that this logic is now ready to execute POP/POP restore instructions.

If the first JSR/JSR relative instruction is followed by additional JSR instructions (with no intervening POP/POP restore instruction), the stack is decremented by two for every JSR instruction executed.

That is, until a maximum of eight JSR instructions have been executed. Following the eighth JSR instruction, the stack pointer is sitting at address 0000_2 – the bottom of the stack. Since the top of the stack detector remains cleared at the time (SSC1 TOS (0) H is high), the bottom of the stack detector is enabled. Signal SSC1 BOS L now asserts to inhibit further processing of JSR/JSR relative instructions; that is, until at least one POP/POP restore instruction is executed.

Should the program attempt to execute another JSR instruction at this time, a two-fold effect would result:

1. The stack violation detector asserts signal SSC1 STACK OVERFLOW (1) H to set the related status bit and cause an interrupt.

2. Signal SSC1 STACK OVERFLOW (0) H switches low to inhibit the stack write enable logic from writing into stack memory.

Whereas, execution of JSR instruction decrements the stack pointer by 2, execution of POP instructions increments the stack pointer by 2. If the stack pointer is at the bottom of the stack, up to eight POP/POP restore instructions can be executed in sequence (no intervening JSRs). On the eighth POP

instruction, the stack pointer address is 1111_2. Should the program attempt another POP instruction at this time, a carry is developed from the stack pointer to clock set the top of the stack detector. Signal SSC1 SP TOS (0) H now switches low to inhibit the POP instruction detector and stack pointer countup logic. Also signal SSC1 SP TOS (1) H is asserted to the stack violation logic. The latter circuit in turn asserts SSC1 STACK UNDERFLOW (1) H to set the appropriate status bit and generate an interrupt.

4.2.19.2 Stack Memory Addressing from the Unibus – The VT48 also allows for addressing any location in stack memory for status sampling purposes. The particular stack address is supplied over the Unibus when writing the Stack Address/Maintenance register (772032). Any one of the 16 stack pointer address locations can be selected via Unibus lines UC4 DB01 through UC4 DB04. This selects the desired 32 bits of status information for application to the stack status multiplexer. Whereas, this multiplexer has 32 input lines, but only 16 output lines (going to the read status multiplexer), additional means are required to determine which 16 bits are to be presented to the read status multiplexer. This selection is effected by the content of the low-order bit (data bit 00) from the Unibus. When this bit conveys a 0, the low-order 16 bits are conveyed to the read status multiplexer. When the bit conveys a 1, the 16 high-order bits are sent. In this way, the entire contents of the stack memory can be accessed for status sampling.

4.2.19.3 Silo Addressing Control Logic – Figure 4-26 shows a block diagram of the silo memory control logic. The silo memory is 68 bits wide by 4 bits deep. That is, there are four possible locations to write into/read from. The times for both writing and reading are governed by certain system conditions as follows:

1. Writing – A silo memory location is addressed and written into only during execution of graphic data instructions, and at a time after the display instruction control has loaded the graphics calculation logic with the graphic data. In other words, as soon as the X/Y delta values are passed to the graphics calculation logic for processing, the related status/parameter data is entered into silo memory.

2. Reading – The silo memory read address is updated (for read access) by the graphics calculation logic just before this logic informs the vector/character generator to commence drawing the graphic entity. In this way, the parameters used to draw the graphic entity (intensity/blank) are available to the vector generator during the graphic draw period. Also, all status/parameter data is available to the proper registers in the event of a light pen or edge transition interrupt. (See discussion on status/parameter routing).

At initialization, the two flip-flops comprising the write addressing logic are jammed clear to 00_2. Conversely, the two flip-flops making up the read addressing logic are jammed preset to 11_2. This is done because the graphics calculation logic always increments the read address after completing the calculation process. Given this set of conditions, the write/read addressing sequence is as follows:

1. The status/parameter data for the first graphic instruction in the display file is written at silo address 00_2 after the graphics calculation logic has been loaded with the X/Y data. Immediately after this, the graphics calculation logic is informed to begin processing the X/Y data and the write address is updated to 01_2. The latter action is done in anticipation of processing the next graphic instruction.

2. The read address remains at 11_2 during the period that the graphics calculation logic is processing the X/Y data. When it is finished (i.e., just prior to commanding the vector/character generator to draw the graphic entity), it sends a signal to update the silo read address. The read address is, in turn, clocked from the 11_2 state to the 00_2 state. The proper silo address has now been selected for presenting the status/parameter data to the other VT48 circuits.

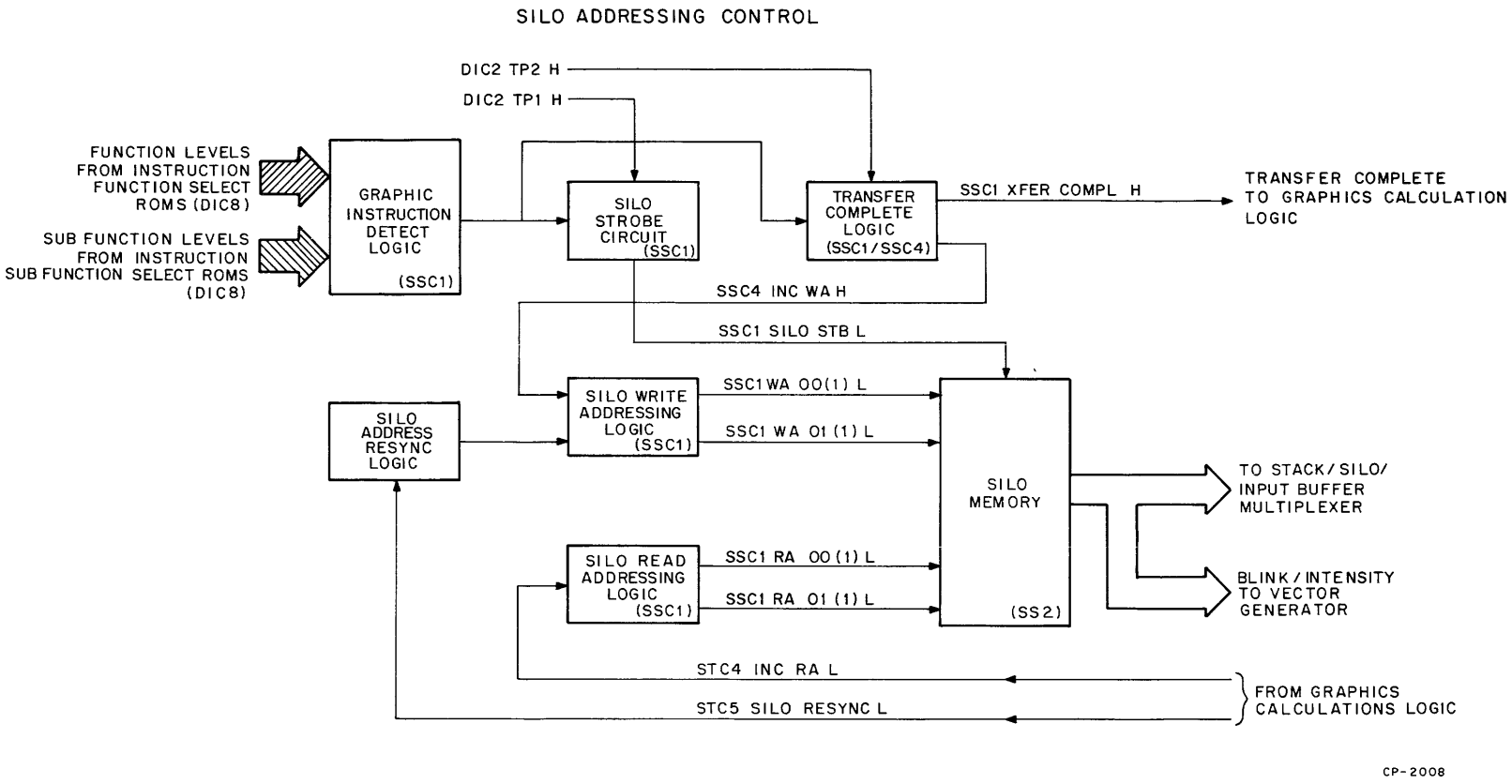


Figure 4-26 Silo Memory Addressing Logic, Block Diagram

From the foregoing, it can also be seen that the write address always leads the read address by at least one address location.

The graphic instruction detect logic (Figure 4-26) receives inputs from the instruction function select ROMs and the instruction subfunction select ROMS (Figure 4-5). These inputs have a two-fold significance as follows:

1. The function select inputs indicate that the instruction being executed is a graphic mode instruction.

2. The subfunction select input determines when the graphics calculation logic has been loaded with the graphic (X/Y) data. Some instructions present the graphic data to the VT48 in a single word, while other instructions require two (or more) words to present the X/Y/Z data. For the latter type instructions, the transfer complete signal to the graphics calculation logic must be delayed until all words conveying the graphic data have been processed.

Once all graphic data is transferred, the output from the graphic instruction detect logic is asserted. This enables both the silo strobe circuit and the transfer complete logic. At TP1 time, the silo strobe circuit issues SSCI SILO STB L to load the status/parameter data into silo memory. At TP2 time, the transfer complete logic issues two signals as follows:

1. Signal SSCI XFER COMPL H is asserted to the graphics calculation logic to indicate that the X/Y data has been loaded and is ready for processing.

2. Signal SSC4 INC WA H is asserted to update the silo write address.

As discussed earlier, the silo write address leads the read address. Normally the write address leads the read address by one, but under certain system conditions the lead may increase to two. When the graphics calculation logic is finished processing a graphic entity, it asserts signal STC4 INC RA L to update the silo read address and select the proper status/parameter data for the graphic entity about to be drawn.

The silo address resync logic also receives an input (STC5 SLO RESYNC L) from the graphics calculation logic. This signal is issued at the conclusion of processing light pen and edge interrupts. Its purpose is to reinitialize the silo write/read addresses to the $00/11_2$ states, respectively.

4.2.20 Read Status Multiplexer

The status and parameters of the various information registers within the VT48 are sampled via the read status multiplexer (see overall block diagram discussion). A block diagram showing the way in which the various registers are addressed for sampling via the read status multiplexer is given in Figure 4-27. Due to space limitations, not all registers are shown on this illustration. However, their presence and accessibility are implied under the blocks labeled miscellaneous registers.

The read status multiplexer itself is capable of accessing any one of 16 sets of inputs (each set 16 bits wide) under control of Unibus address lines UC1 A01 H through UC1 A04 H. The addressing technique is straightforward. That is, an address of 0000_2 accesses the contents of the display program counter (DPC); an address of 0001_2 accesses certain miscellaneous status registers; and so on.

Table 4-4 shows the status data conveyed for each read access address in greater detail. The source of all status bits (by source drawing set number) is also indicated.

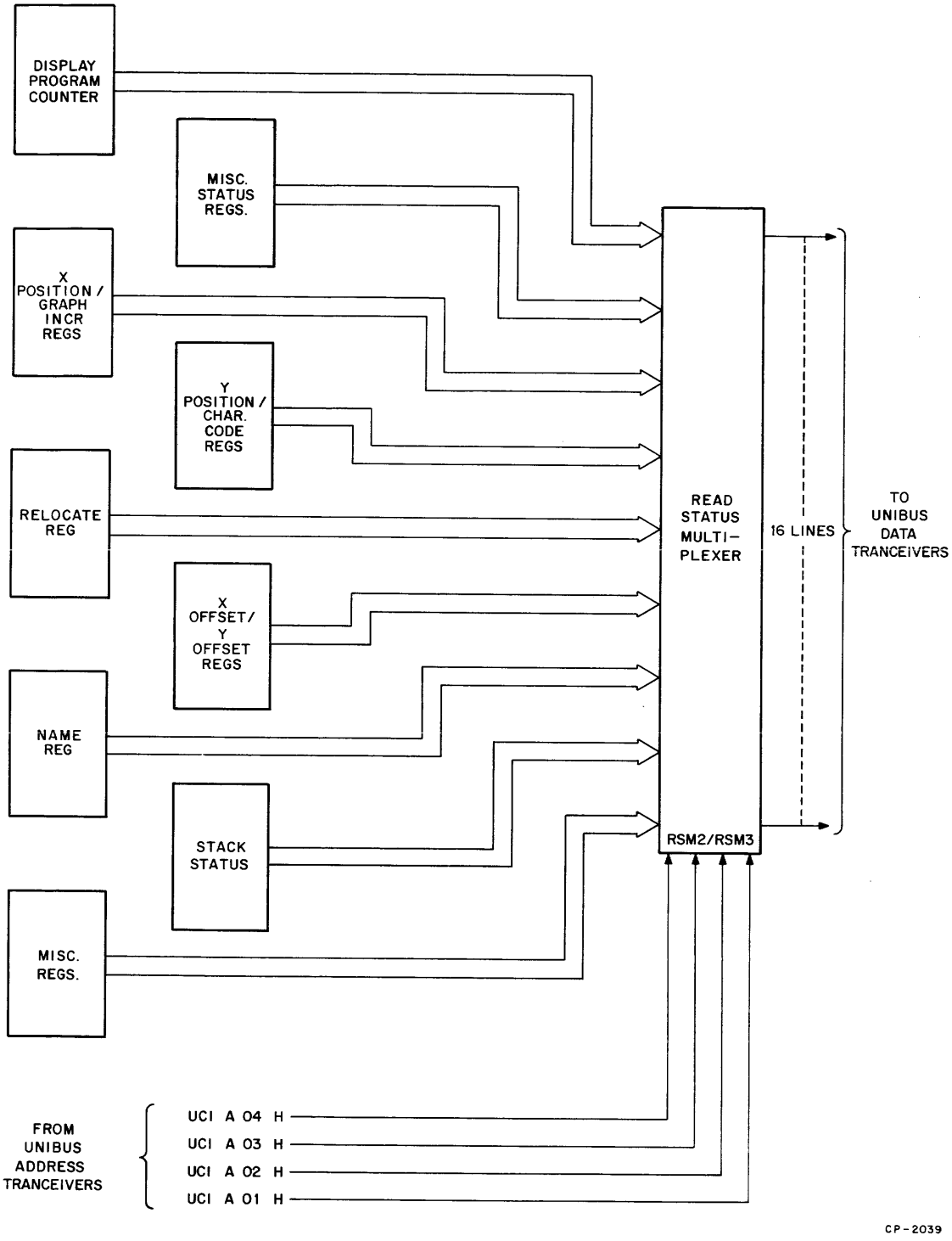


Figure 4-27 Read Status Multiplexer, Block Diagram

Table 4-4 Summary of Read Status Bits

ADDRESS	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
772000	DIC5 DPC 15 H														DIC5 DPC 01 H	GND
	← DISPLAY PROGRAM COUNTER →															
772002	RSM1 STOP H	DIC7 MODE 03 (1) H	DIC7 MODE 02 (1) H	DIC7 MODE 01 (1) H	DIC7 MODE 00 (1) H	DSR5 INTENSITY 02 (1) H	DSR5 INTENSITY 01 (1) H	DSR5 INTENSITY 00 (1) H	RSM1 LIGHT PEN FLAG 00 H	RSM1 SHIFT OUT H	STC2 EDGE H	DSR2 ITALICS (1) H	DSR5 BLINK ENA (1) H	EDGE FLAG (1) H	DSR5 LINE 01 (1) H	DSR5 LINE 00 (1) H
		← GRAPHIC MODE →				← INTENSITY →									← LINE TYPE →	
772004	DSR2 GRAPH INC 05 (1) H	DSR2 GRAPH INC 04 (1) H	DSR2 GRAPH INC 03 (1) H	DSR2 GRAPH INC 02 (1) H	DSR2 GRAPH INC 01 (!) H	DSR2 GRAPH INC 00 (1) H	RM5 X 09 (1) H	RM5 X 08 (1) H	RM5 X 07 (1) H	RM5 X 06 (1) H	RM5 X 05 (1) H	RM5 X 04 (1) H	RM5 X 03 (1) H	RM5 X 02 (1) H	RM5 X 01 (1) H	RM5 X 00 (1) H
	← GRAPHIC INCREMENT REGISTER →						← X POSITION REGISTER →									
772006	DSR6 CB 05 H	DSR6 CB 04 H	DSR6 CB 03 H	DSR6 CB 02 H	DSR6 CB 01 H	DSR6 CB 00 H	RM5 Y 09 (1) H	RM5 Y 08 (1) H	RM5 Y 07 (1) H	RM5 Y 06 (1) H	RM5 Y 05 (1) H	RM5 Y 04 (1) H	RM5 Y 03 (1) H	RM5 Y 02 (1) H	RM5 Y 01 (1) H	RM5 Y 00 (1) H
	← CHARACTER CODE →						← Y POSITION REGISTER →									
772010					DIC6 REL 11 (1) H	DIC6 REL 10 (1) H	DIC6 REL 09 (1) H	DIC6 REL 08 (1) H	DIC6 REL 07 (1) H	DIC6 REL 06 (1) H	DIC6 REL 05 (1) H	DIC6 REL 04 (1) H	DIC6 REL 03 (1) H	DIC6 REL 02 (1) H	DIC6 REL 01 (1) H	DIC6 REL 00 (1) H
	← GND →				← RELOCATE REGISTER →											
772012	DIC4 BUSY (1) H	GND	RSM1 STACK OVER-FLOW H	RSM1 STACK UNDER-FLOW H	VC2 TIME OUT (1) H	DSR4 CHAR ROTATE 00 (1) H	DSR4 CHAR SCALE 01 (1) H	DSR4 CHAR SCALE 00 (1) H	RSM1 EXT STOP FLAG H	DSR2 MENU (1) H	DIC5 DPC 17 H	DIC5 DPC 16 H	DSR4 VEC SCALE 03 (1) H	DSR4 VEC SCALE 02 (1) H	DSR4 VEC SCALE 01 (1) H	DSR4 VEC SCALE 00 (1) H
							← CHAR SCALE REG. →				← DISPLAY PROGRAM COUNTER, BITS 16/17 →		← VECTOR SCALE REGISTER →			
772014	STC6 X 13 (1) H	STC6 X 12 (1) H	RM5 X 11 (1) H	RM5 X 10 (1) H	RM6 X OFF 11 (1) H											RM6 X OFF 00 (1) H
	← X POSITION 4 HIGH ORDER BITS →				← X OFFSET REGISTER →											
772016	STC Y 13 (1) H	STC6 Y 12 (1) H	RM5 Y 11 (1) H	RM5 Y 16 (1) H	RM6 Y OFF 11 (1) H											RM6 Y OFF 00 (1) H
	← Y POSITION 4 HIGH ORDER BITS →				← Y OFFSET REGISTER →											
772020 (WRITE ONLY)	← NOT USED (GND) →															
772022	DSR5 INT SCOPE 00 (1) H	RSM1 LIGHT PEN FLAG 00 (1) H	RSM1 LPSW ON FLAG 00 H	RSM1 LPSW OFF FLAG 00 H	DSR5 LIGHT PEN INTR 00 (1) H	DSR5 LP SWITCH INTR ENA 00 (1) H	DSR5 INT SCOPE 01 (1) H	RSM1 LIGHT PEN FLAG 01 (1) H	RSM1 LPSW ON FLAG 01 H	RSM1 LPSW OFF FLAG 01 H	DSR5 LIGHT PEN INTR 01 (1) H	DSR5 LP SWITCH INTR ENA 01 (1) H	DSR COLOR 01 (1) H	DSR2 COLOR 00 (1) H		
													← COLOR REGISTER →			
772024	RSM1 NAME FLAG H	NOT USED	DSR3 SEARCH 01 (1) H	DSR3 SEARCH 00 (1) H	NOT USED	DSR4 NAME 10 (1) H										DSR4 NAME 00 (1) H
						← NAME REGISTER →										
772026	SS4 STACK STAT 15 H															SS4 STACK STA7 00 H
	← STACK MEMORY STATUS →															
772030										DSR6 CT 06 (1) H						DSR6 CT 00 (1) H
	← NOT USED (GND) →									← CHARACTER TERMINATE REGISTER →						
772032	STC8 MAINT 4 (1) H	STC8 MAINT 3 (1) H	DIC1 MAINT SW 02 (1) H	DIC1 MAINT SW 01 (1) H	NOT USED	DIC8 OFFSET H	DIC8 JSR H	DIC8 WORD 2 H	DIC8 WORD 1 H	DIC8 WORD 0 H	SSC1 TOS (1) H	SSC1 SP 03 (1) H	SSC1 SP 02 (1) H	SSC1 SP 01 (1) H	SSC1 SP 00 (1) H	SSC1 MUX A (1) H
												← STACK POINTER →				
772034					Z13 (1) H											Z 02 (1) H
	← NOT USED (GND) →				← Z REGISTER →											
772036	STC3 X OFFSET NEG H	STC3 Y OFFSET NEG H	Z OFFSET H	NOT USED	Z OFF 11 (1) H											Z OFF 00 (1) H
					← Z OFFSET REGISTER →											

4.2.21 Unibus Transfer Timing

The timing signals used to implement data transfers between the PDP-11 and the VT48 are shown on Figure 4-28.

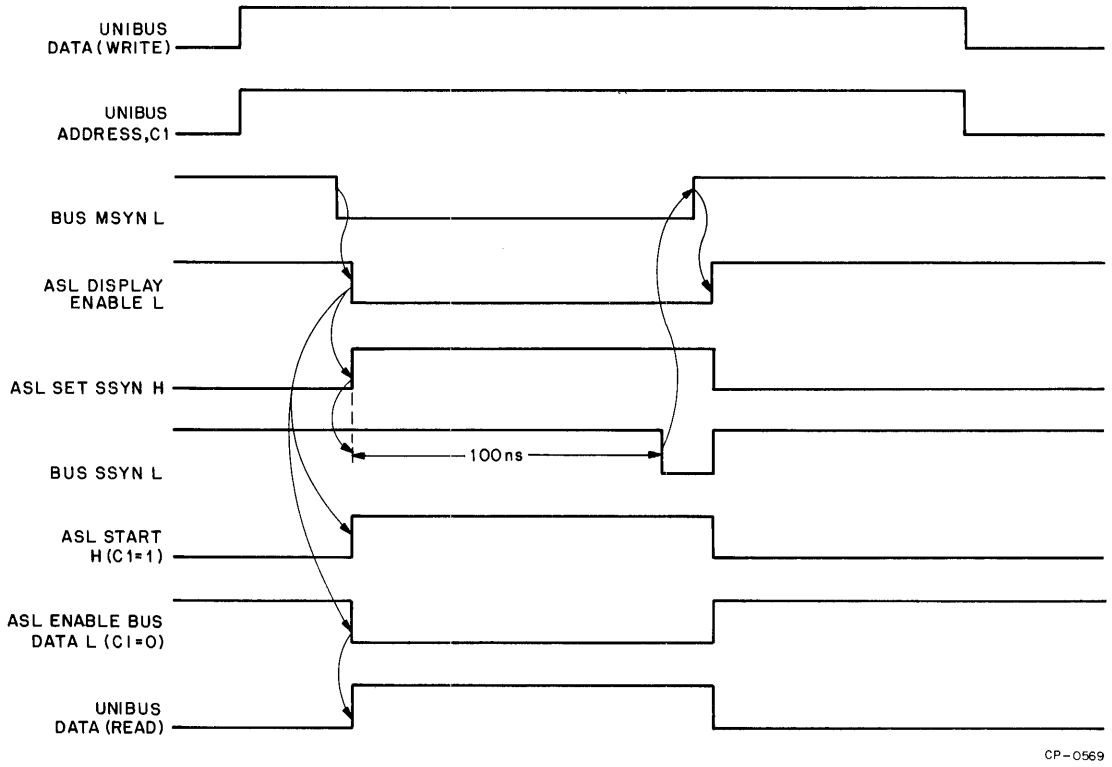


Figure 4-28 Unibus Transfer Timing

4.2.22 Control Instruction Flow Diagrams

Figures 4-29 and 4-30 show flow sequences for the set graphic mode and jump to subroutine (JSR) instructions respectively. The sequence for the set graphic mode instruction indicates the way that line type, intensity and other parameters are entered into the appropriate registers. Loading of status data during load status instructions is accomplished in a similar way. The JSR flow diagram indicates how stack data is entered into stack memory and also how the display program counter is updated to contain the branch address.

4.2.23 Vector Generation Circuits, Detailed Block Diagram Description

4.2.23.1 Types of Input Signals – The vector generator is designed to generate a constant velocity vector (line that represents a quantity having direction and magnitude) between two coordinates on a VR48 Display Monitor. To accomplish this, the following digital signals are received and then converted to appropriate X and Y analog signals which drive a monitor display:

- Initial X and Y positions (starting point of the vector).
- Tangent of the angle ratio of (minor axis length/major axis length) between the major and minor axis. The major axis is defined as the one with the greatest change in value.
- Length of the major axis.
- Signs (+ or –) of major and minor axis.

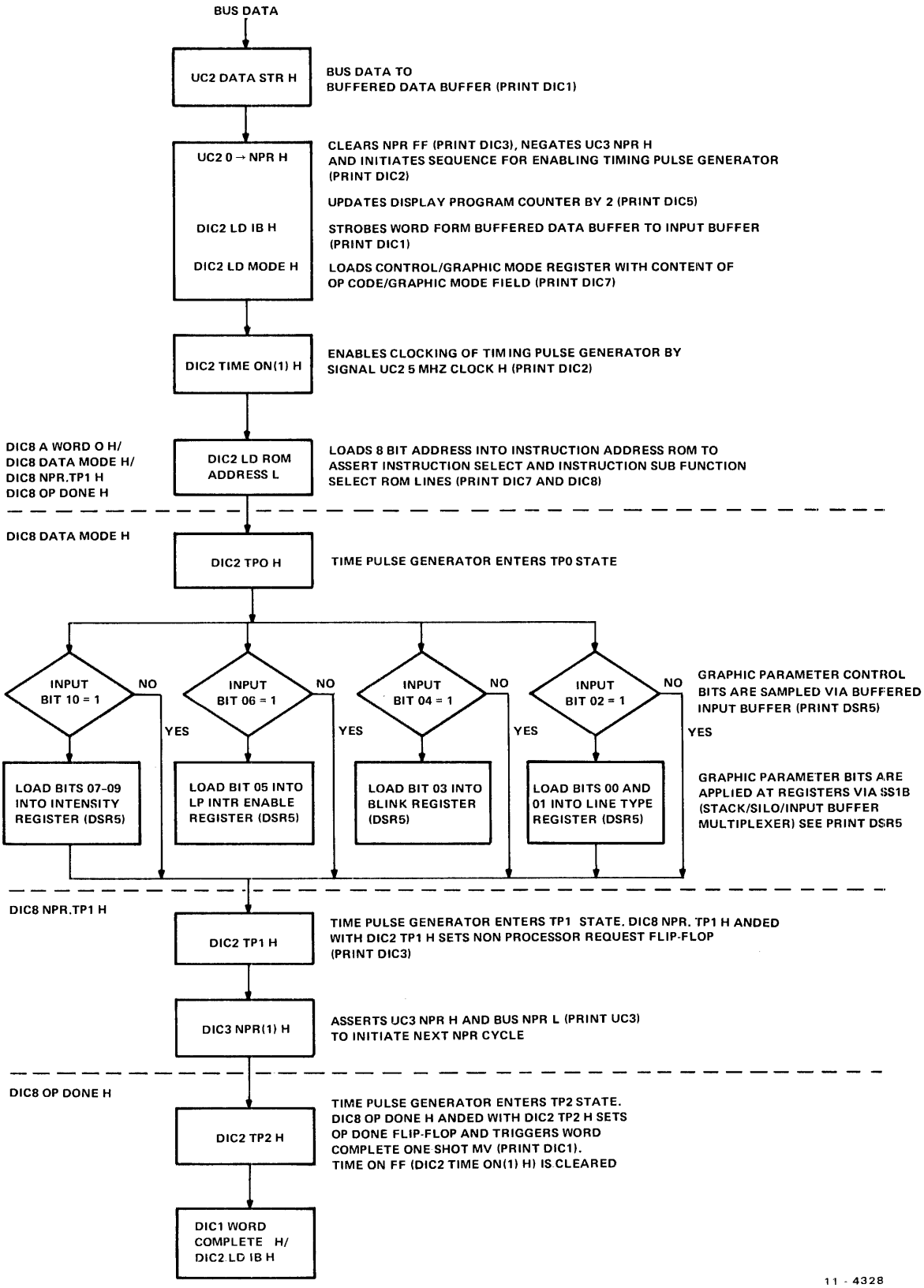


Figure 4-29 Set Graphic Mode Instruction Flow Diagram

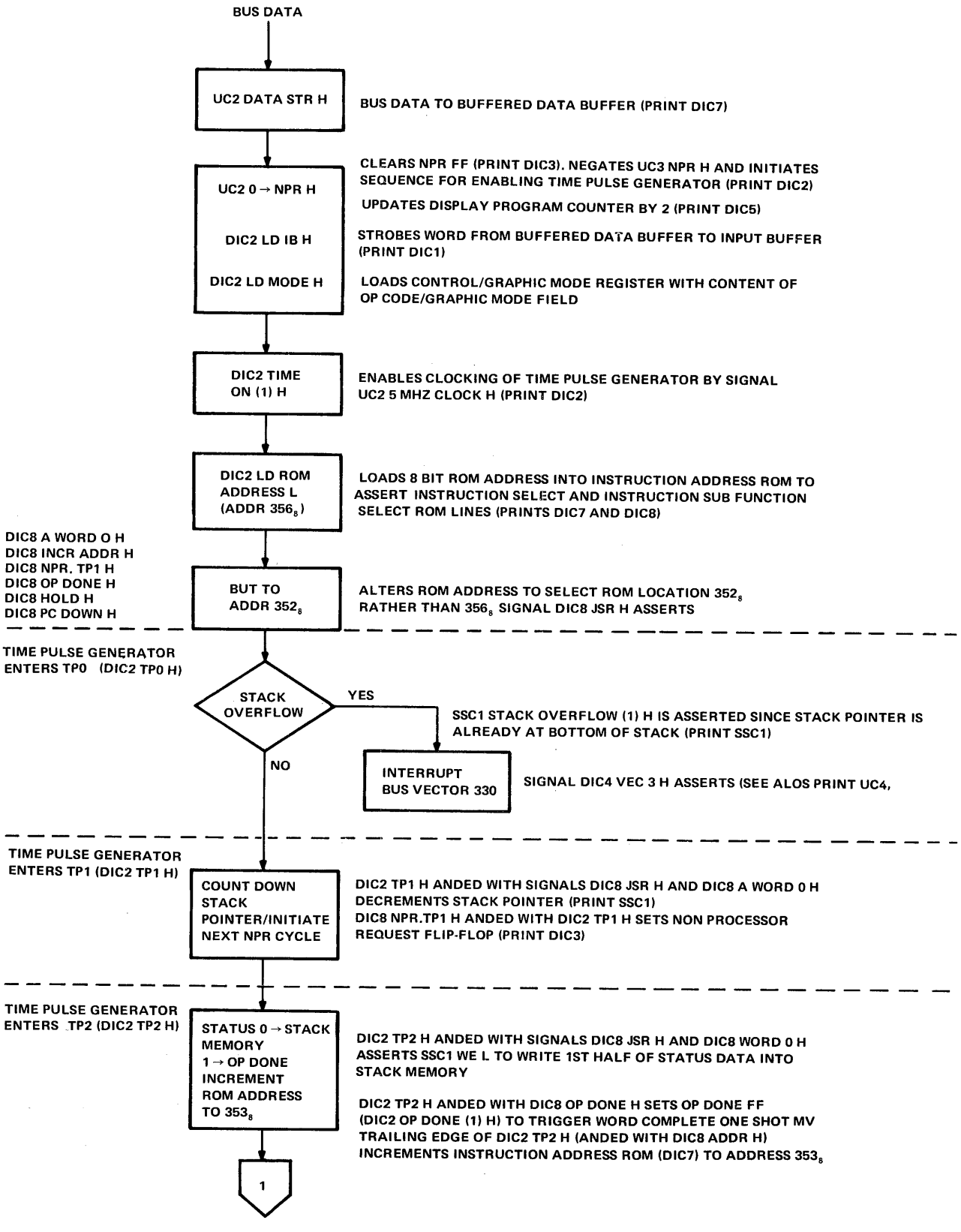


Figure 4-30 JSR Instruction Flow Diagram, Part 1

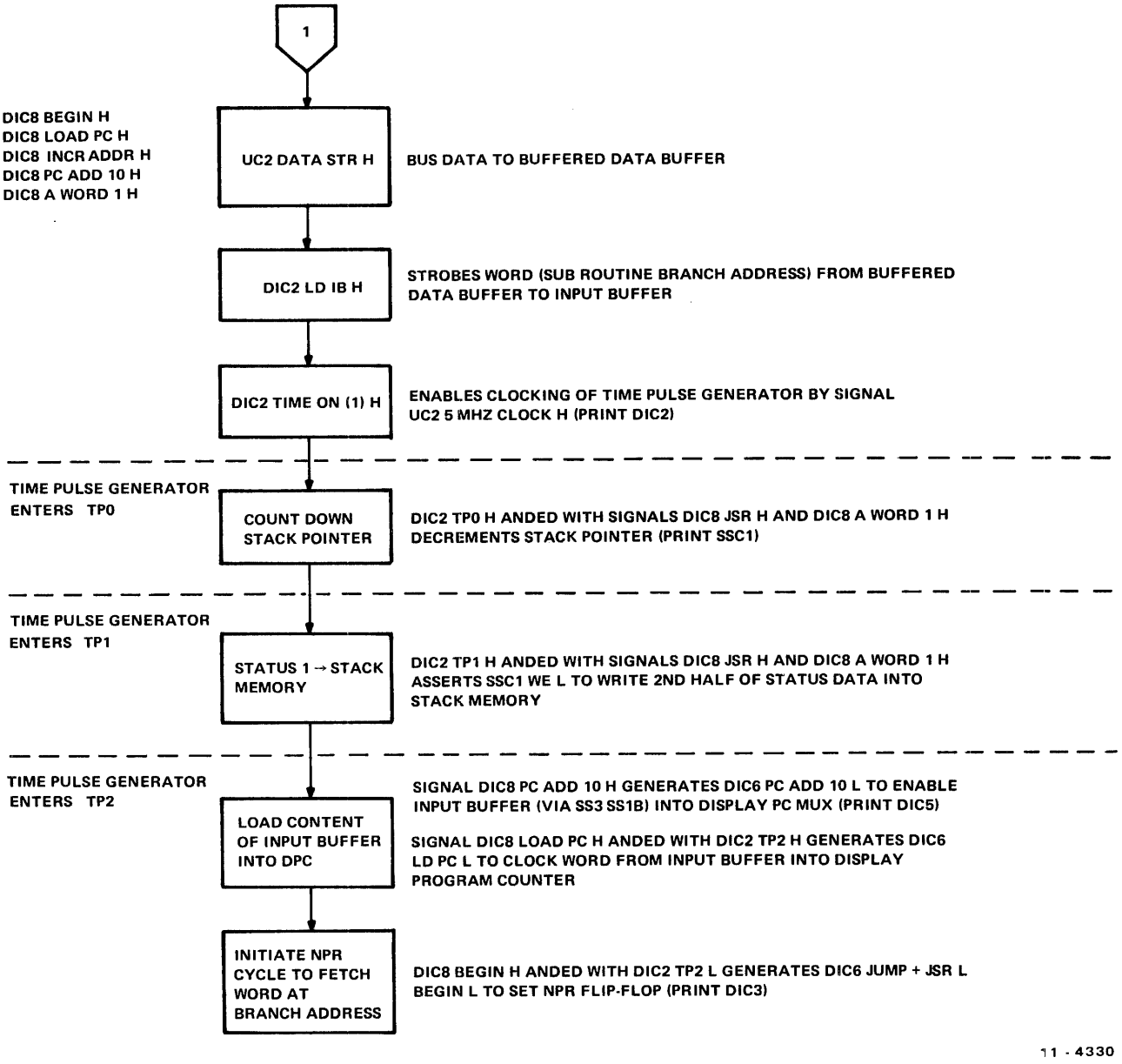
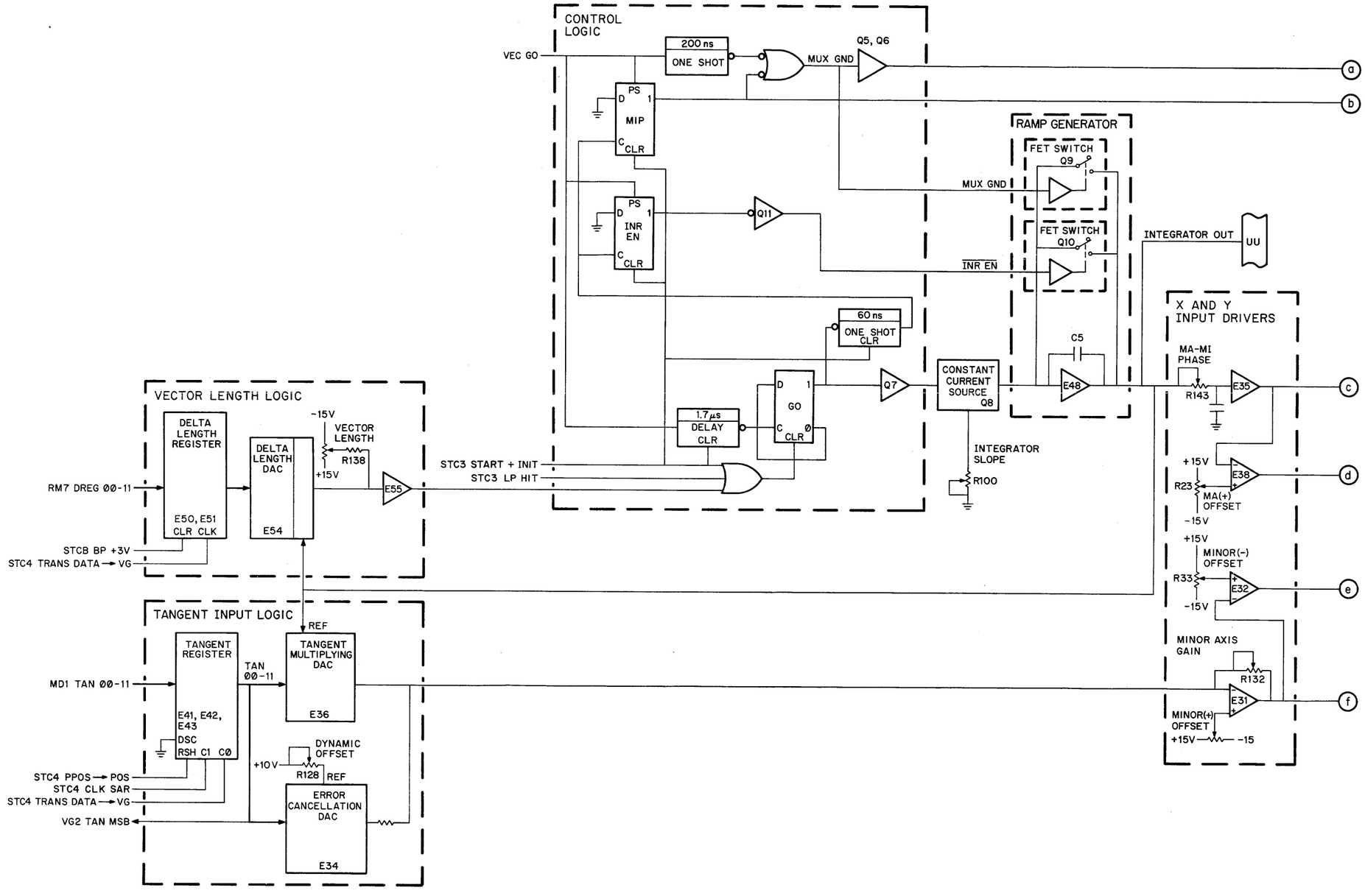


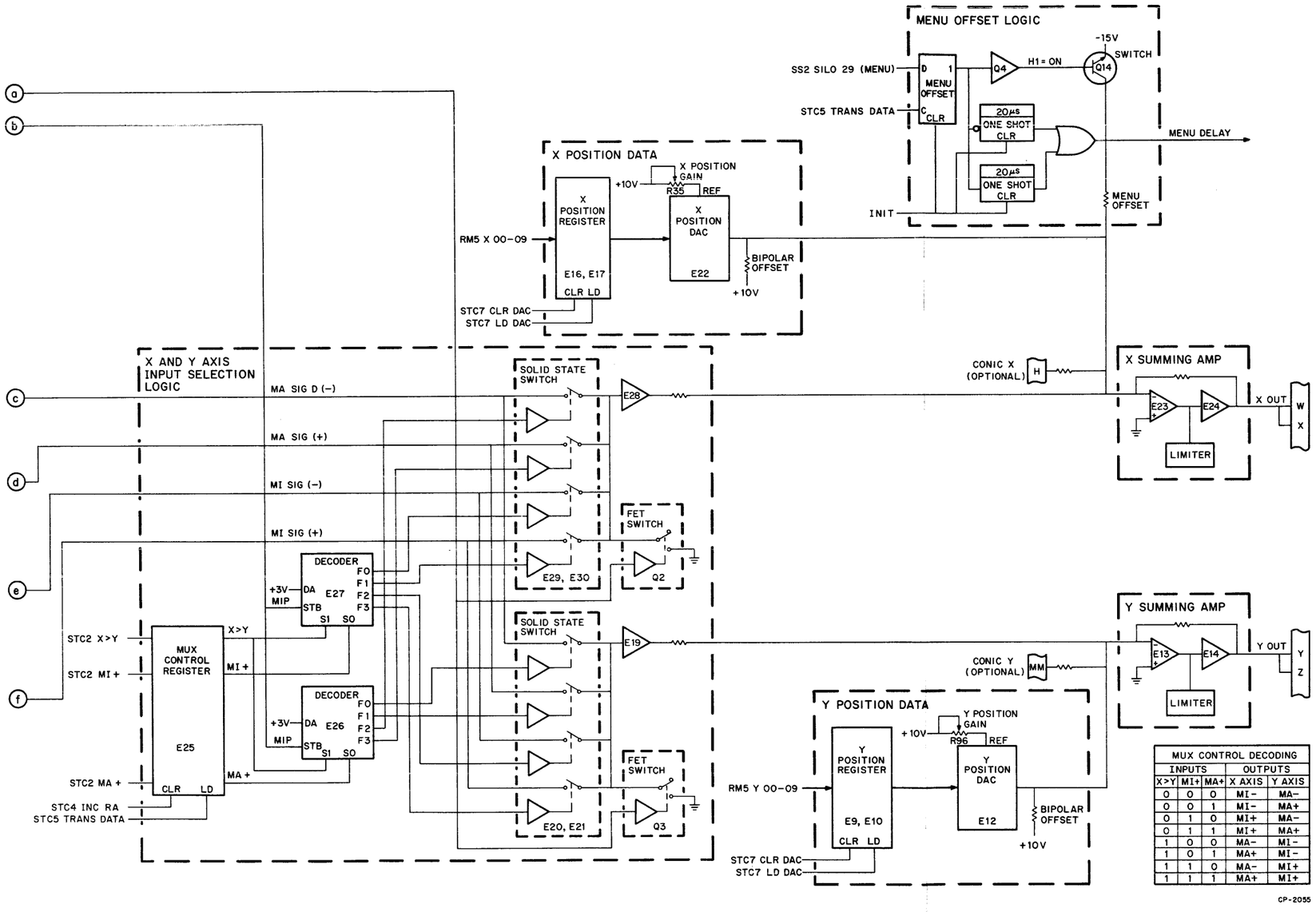
Figure 4-30 JSR Instruction Flow Diagram, Part 2

4.2.23.2 X and Y Initial Position Determination (Figure 4-31) – The initial starting position of a vector is determined by the digital values contained in the X and Y position registers. These registers hold an initial X, Y coordinate before a vector is drawn and during the vector stroke. When a vector is completed, the position registers are updated to the coordinates of the end of the vector. The output of each register is converted by a DAC to an analog signal which is summed into its respective power amplifier. The X and Y power amplifiers sum the signals from the position DACs, the X and Y vector data inputs, and, in X, the menu offset. The menu offset circuit consists of a special purpose switch which offsets the X coordinate system to the right side of the screen to write text beside a picture which is being drawn. Whenever vectors are not being drawn, the vector data inputs are held to zero volts. The output amplifiers provide the power to drive the VR48 Display Monitor inputs. These amplifiers also have limiters to protect the VR48 from over voltage. The normal operating range of the outputs is ±5 V.



CP-2054

Figure 4-31 Vector Generator, Detailed Block Diagram (Sheet 1 of 2)



MUX CONTROL DECODING

INPUTS			OUTPUTS	
X>Y	MI+	MA+	X AXIS	Y AXIS
0	0	0	MI-	MA-
0	0	1	MI-	MA+
0	1	0	MI+	MA-
0	1	1	MI+	MA+
1	0	0	MA-	MI-
1	0	1	MA+	MI-
1	1	0	MA-	MI+
1	1	1	MA+	MI+

CP-2055

Figure 4-31 Vector Generator, Detailed Block Diagram (Sheet 2 of 2)

4.2.23.3 Vector Determination

Major Axis – The control logic (Figure 4-31) provides the necessary timing and delays to allow settling time in the analog circuits and also to provide the control to interface with the rest of the VT48 Display Processor.

When a vector is to be drawn, the graphics calculation logic loads the digital value of the lengths of the major axis into the Delta Length register (E50, E51), the tangent of the vector into the Tangent register (E41, E42, E43), the control signals (X > Y, MI+, MA+) into the MUX Control register (E25), and asserts signal STC4 VEC GO L.

STC4 VEC GO L sets both the MIP (Move In Progress) and the INR ENA (Integrator Enable) flip-flops, while triggering a 200 ns one-shot and a 1.7 μs delay. When the 200 ns one-shot times out, the FET switches are turned off with the following results:

- Q9 and Q10 activate the ramp generator by removing the short between its input and output.
- Q2 enables the vector data input to the X summing amplifiers while Q3 does the same for the Y axis.

By this time, the MUX control signals (X > Y, MA+, MI+) have been decoded and the solid state switches are enabled that connect the vector data inputs to the proper X and Y input drives.

When the 1.7 μs delay times out, the GO flip-flop is set enabling the constant current source (Q8) that provides the input to the ramp generator (E48).

The integrating speed of the ramp generator is controlled by the current in Q8. This current is determined primarily by reference diodes D12 and D13, and is adjustable by the integrator slope control R100.

The output of the ramp generator provides the major axis output signal. An inverter at the output of the integrator generates the complementary signal so both positive and negative values (MA+ and MA–) are available.

The delta length DAC (E54) determines the final voltage level of the integrator and thus the length of the vector major axis. This is accomplished by comparing the digital output of the Delta Length register with the output of the ramp generator. When they are equal, comparator amplifier E55 triggers and resets the GO flip-flop.

Minor Axis – While the major axis signal is being developed, the minor axis signal is also being generated. The multiplying DAC (E36) attenuates the output of the ramp generator by the amount determined by the digital value stored in the Tangent register.

DAC E36 will not operate linearly if the reference current goes to zero. Therefore, the analog input to the multiplying DAC is offset +20 mV by a resistor network and +10 V source. This 200 mV offset is multiplied by the digital input and appears as an error current at the output summing junction. To cancel this error, a second multiplying DAC (E34) is used. This DAC effectively multiplies a –200 mV offset by the same digital input and sums the equal and opposite error into a summing junction.

4.2.23.4 Major and Minor Axis Outputs – The major and minor axes have inverting amplifiers so each signal and its complement are available at the multiplexer inputs. R-C circuits are used to match the phase of the + and – signals in each axis. Offset adjustment and precision feedback resistors are used to ensure the accuracy of the inverted signals. Phase errors between the major and minor axes are nulled by a major-minor phase adjustment circuit at the input of E35.

When the GO flip-flop is reset, a 100 ns one-shot is triggered, which in turn, resets the MIP and INR EN flip-flops. This action turns on Q2 and Q3, zeroing the vector data inputs, and Q9 and Q10 which reset the ramp generator. When the ramp generator is reset, its output goes to a level of -100 mV rather than 0 V. This allows the integrator and other amplifiers to get a "running start" before the visible portion of a vector is drawn. It also minimizes the effect of a second-order phase errors which occur during initial acceleration of the amplifiers.

4.2.23.5 Z Axis (Intensity) Circuits - Intensity on the VR48 Display Monitor screen is controlled by the Z output (Figure 4-32). A 3-bit value for intensity is received and stored in register E8. This digital Z information is converted to an analog voltage by DAC E5 and amplifier E2. The result of this conversion is applied to a second multiplying DAC (E1). This second DAC provides for the modulation of intensity for characters, where constant beam velocity cannot be maintained. The four digital inputs to this DAC are developed on the character generator board. The output of E1 drives a power amplifier (E6, E7) similar to those in the X and Y outputs. The range of the Z output is from 0 to +3 V.

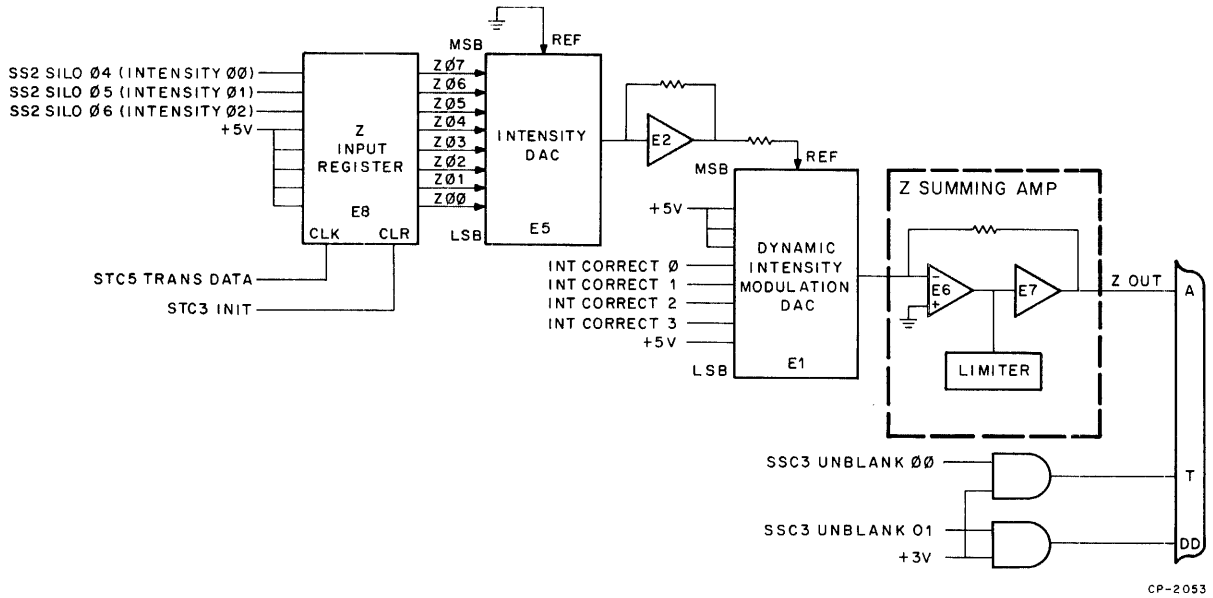


Figure 4-32 Z Axis (Intensity) Block Diagram

4.2.23.6 Voltage Regulators - Voltage regulators are used to generate +5 and +10 V. The 10 V output is used as a reference for the DACs while the +5 V source provides a logic compatible voltage which is referenced to the analog ground system.

4.2.24 Character Generator Detailed Block Diagram Discussion

The character generator (Figure 4-33) is designed to display characters and special symbols on a VR48 Display Monitor. This is accomplished by generating ramped voltages with specific amplitude and duration for both the X and Y axes. When these voltages are applied to the corresponding axis on a display monitor, a vector with a certain direction and magnitude will result. These vectors, connected sequentially, form a pattern which represents the character being drawn. Due to the method of generation, characters of almost any complexity and size can be drawn, while special symbols can be defined as a character and drawn on the screen by addressing the generator with only one ASCII code.

4.2.24.1 Character Generator Startup - A 7-bit ASCII word that defines the character to be drawn is presented to the address converter ROM, which in turn produces a 12-bit starting address for the

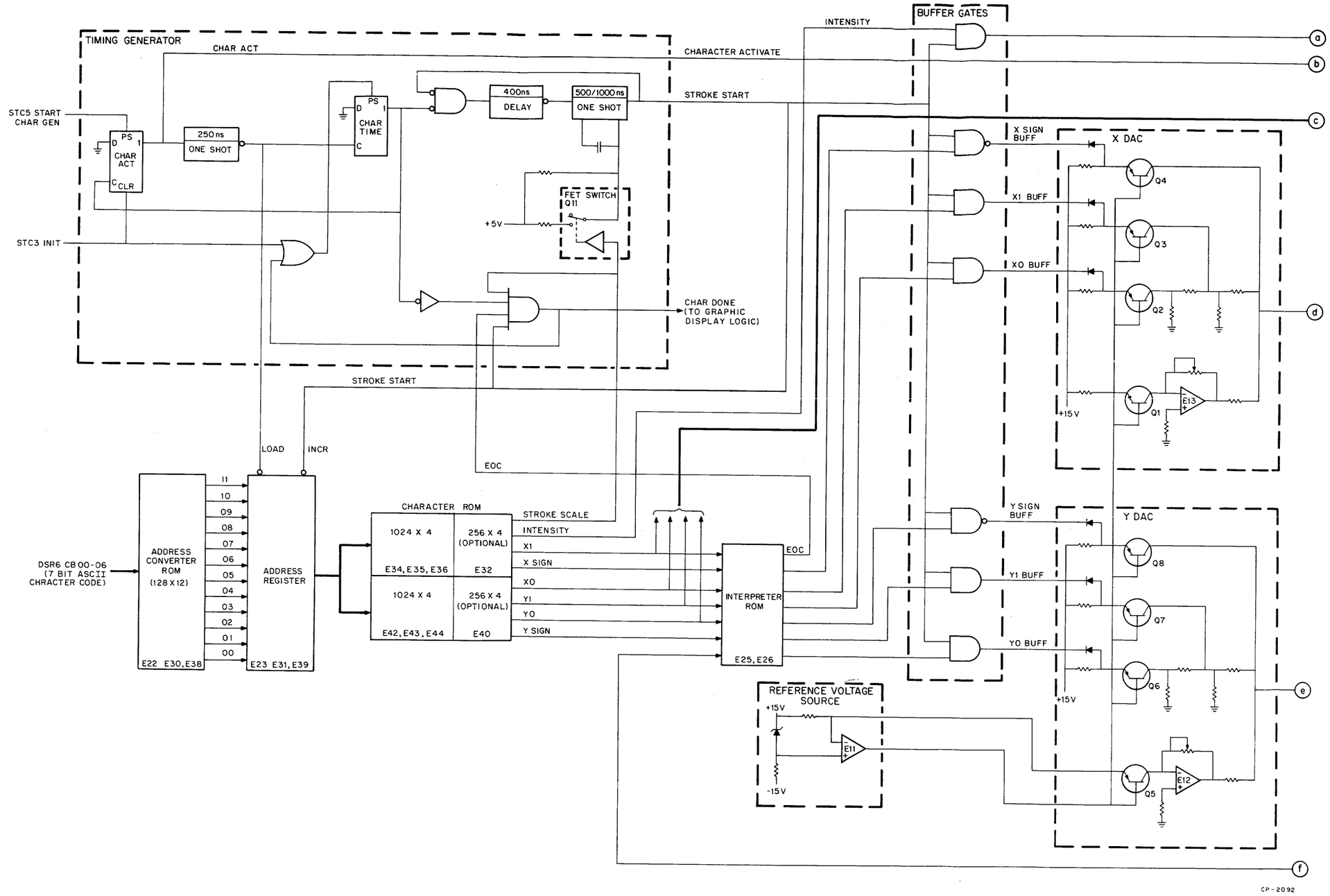


Figure 4-33 Character Generator, Detailed Block Diagram (Sheet 1 of 2)

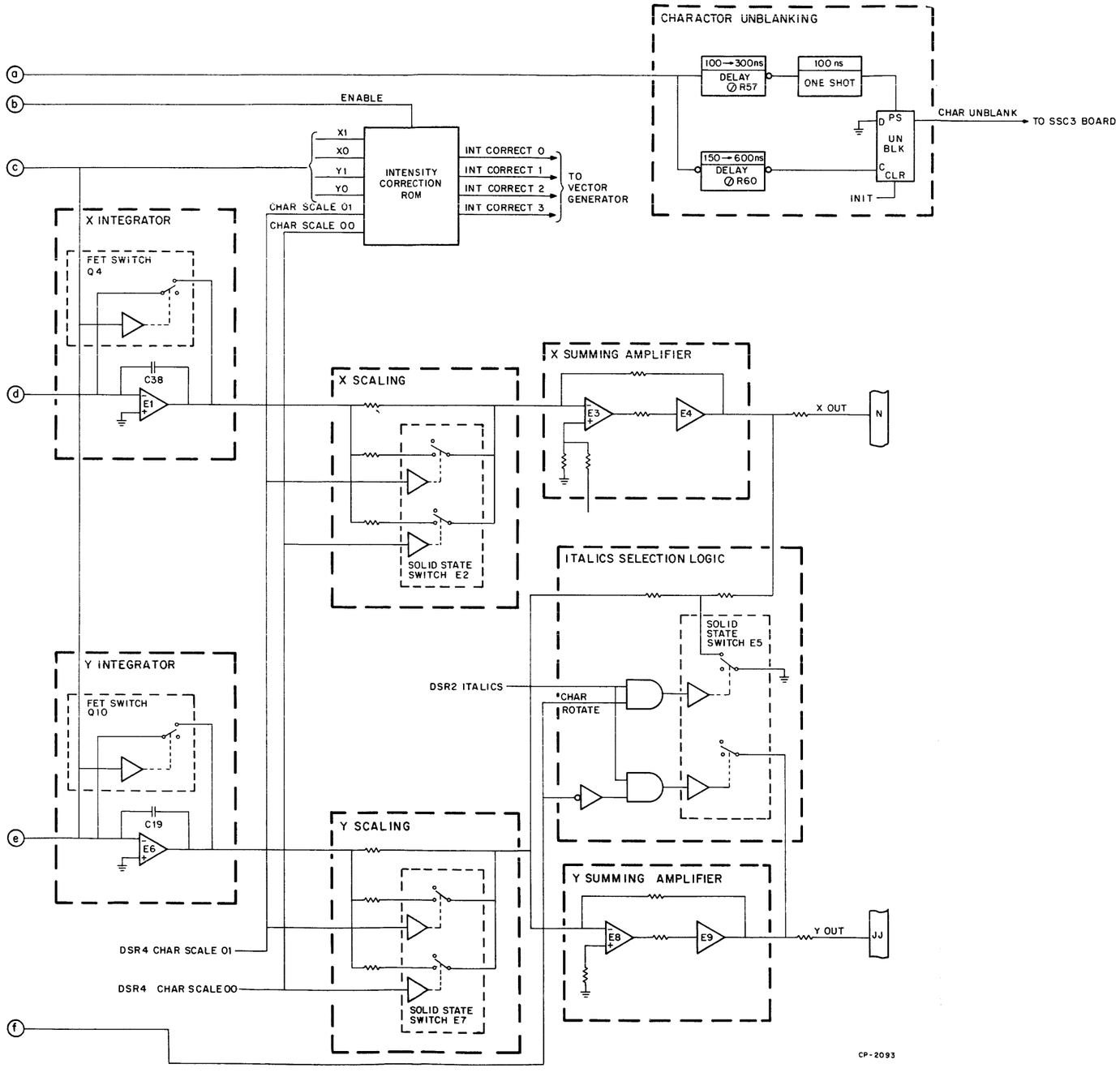


Figure 4-33 Character Generator, Detailed Block Diagram (Sheet 2 of 2)

character ROM sequence required to draw the character. Upon receipt of the start signal (STC5 START CHAR GEN L), the timing generator issues a signal (CHAR ACT) to enable the integrating and intensity logic and to load the output of the address converter ROM into the Address register. The address is then applied to the character ROM where it accesses the location of the first stroke (vector) of the character.

4.2.24.2 Character ROM Outputs – The output of the character ROM consist of six bits of data that define the X and Y coordinates of the vector. These bits are sent to the interpreter ROM. Two other bits of information are also generated by the character ROM. Their functions are as follows:

Stroke Scale – This signal allows the length of a vector to be doubled. This is beneficial because it saves a character ROM location if two sequential and identical strokes are required, while also saving the time required to address the second location. For example, the letter X can be drawn with two double strokes, i.e.,  ; instead of four single strokes, i.e., .

Intensity – This signal is gated by the timing logic to enable both leading and trailing edge delays that set and reset the character unblanking flip-flops. This flip-flop determines whether the vector is visible on the screen. Invisible vectors are drawn when it is necessary to relocate a stroke's starting position.

4.2.24.3 Interpreter ROM – The interpreter ROM, in addition to the X and Y information, receives a signal (DSR4 CHAR ROTATE 00) which, when asserted, makes the character appear to be rotated 90 degrees on the screen. The X and Y outputs of the interpreter ROM are sent to buffer gates. These gates, when disabled, condition the X and Y DACs to supply zero current. The buffers are enabled by the timing signal STROKE START which controls the amount of time (single or double stroke) the DACs are allowed to provide current to the X and Y integrators.

4.2.24.4 D/A Converters – The DACs are bipolar devices that convert a 3-bit binary input into a comparable value of current. Three digital inputs provide the possibility of eight different currents; however, only seven of these are used. They are 0, plus 1, 2, and 3 and minus 1, 2, and 3. These numbers are directly related to the magnitude and sign of the currents out of the DAC. A potentiometer is provided to adjust the output current of the DAC to zero when all digital inputs are negated.

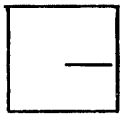
4.2.24.5 X/Y Integrators – An analog integrator utilizes the current from the DAC to produce a ramped voltage output. The rate at which the voltage increases is directly dependent on the amount of current supplied by the DAC. This relationship can be used to control the vector produced on the screen. If the input to the DAC causes a zero current, the integrator output will not move, but will stay in a hold mode. When the input to the DAC is then changed, and current is presented to the integrator, a ramp will result until the DAC is placed back in the hold mode at which time integration stops. Consecutive operations of this type appear as vector, hold, vector, hold, and etc. These vectors, therefore, define a character on the screen while the holds are needed to allow the scope yoke to settle between strokes.

4.2.24.6 Examples of Character Strokes – As was stated earlier, the current from the DACs determines the vector on the screen. Let us examine a vector drawn in a positive X direction. This is caused by a ramped voltage from the X axis and no voltage at all from the Y axis. If the current from the DAC causes a ramp of 1 V per μs and the current presented to the input of the integrator for 1 μs, a deflection equal to 1 V would be seen on the screen. If the current were cut in half, a vector half as long would result as long as the current was present for the same period of time. Therefore, it can be seen that DAC current will vary vector length. The vectors on the screen are directly related in direction and magnitude to the ramps out of the integrators, but only when the ramps of both integrators are combined vectorially in the scope.

The following examples illustrate this. Assume the center of the scope is the starting point of the vector.

X DAC = +3

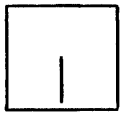
Y DAC = 0



Result: A vector in the X positive direction with a magnitude of 3 units.

X DAC = 0

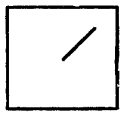
Y DAC = -3



Result: A vector in the Y negative direction with a magnitude of 3 units.

X DAC = +3

Y DAC = +3



Results: A vector at 45 angle with a magnitude of 1.414 times 3 units.

4.2.24.7 Scaling Circuits and Drivers – The output of the integrators is routed through a scaling circuit that consists of an analog multiplexer. Binary inputs (DSR4 CHAR SCALE 00–01) cause the voltages from the integrators to be attentuated. This allows four character sizes to be digitally selected.

The X and Y drivers receive the scaled voltages and amplify them to the point where they can drive the circuits in the monitor. A summing circuit is connected between the drivers that feeds a small percentage of the opposite axis voltage into either the X or Y amplifier when an italicized character is selected by the signal DSR2 ITALICS.

It can be seen that quite a number of different vectors are required to be drawn on the screen to make up a character. A word for each stroke of the character is sequentialy stored in the character ROM. These words are accessed as the timing generator periodically (900 ns, short stroke, 1400 ns, long stroke) increments the Address register. The last word in each character sequence is an end of character word. When this word is detected, the timing generator is initialized and the graphics calculation logic is notified, by the signal CG2 CHAR DONE L, that the character has been drawn on the screen.

CHAPTER 5
VT48 DISPLAY PROCESSOR UNIT DIAGNOSTICS, PREVENTIVE MAINTENANCE, AND ALIGNMENT PROCEDURES

5.1 INTRODUCTION

The subsequent paragraphs describe the diagnostics and adjustment procedures to be carried out on the VT48. Diagnostic programs are provided to verify proper operation following installation and to checkout and align the vector generator and character generator modules periodically.

5.2 DIAGNOSTIC MAINTENANCE PROGRAMS

There are five diagnostic maintenance programs available for use in checkout of VT48 operation. These programs, designators and equipment they serve to checkout are listed in Table 5-1.

Table 5-1 Diagnostic Maintenance Programs

Designation	Purpose
DEC-11-DZVSA	Instruction Test, Part I, VT48 only
DEC-11-DZVSB	Instruction Test, Part II, VT48 only
DEC-11-DZVSC	Instruction Test, Part III, VT48 only
DEC-11-DZVSD	Visual Test, VR48 and VT48
DEC-11-DXVSA	VS60 DECX11 Object Module, System Level Diagnostic

The three instruction test diagnostics are used to check all modules except the vector and character generator modules. The visual test is used to exercise the analog circuits on the vector and character generators so that alignment and adjustment procedures can be carried out. The final program is used for system level diagnostics.

5.2.1 Instruction Test Diagnostics Load and Run Procedures

5.2.1.1 Instruction Test Part I (11-DZVSA) – Proceed as follows:

1. Load the diagnostic into memory.
2. Load address 200 octal.
3. All switches down. Start.
4. Run for 10 minutes. (Multiple passes.)

5. Reference switch settings are:

    15 Halt on error
    14 Loop on test
    13 Inhibit error typeouts
    11 Inhibit iterations
    10 Bell on error
     9 Loop on error
     8 Loop on test in SWR (7:0)

NOTE

Pass 1 requires 30 seconds or less. Pass 2 requires approximately one minute.

5.2.1.2 Instruction Test Part II (11-DZVSB) – Proceed as follows:

1. Load the diagnostic into memory.
2. Load address 200 octal.
3. All switches down. Start.
4. Run for 10 minutes. (Multiple passes.)
5. Reference switch settings are:

    15 Halt on error
    14 Loop on test
    13 Inhibit error typeouts
    11 Inhibit iterations
    10 Bell on error
     9 Loop on error
     8 Loop on test in SWR (7:0)

5.2.1.3 Instruction Test Part III (11-DZVSC) – Proceed as follows:

1. Load the diagnostic into memory.
2. Load address 200 octal.
3. All switches down. Start.
4. Run for 10 minutes. (Multiple passes.)
5. Reference switch settings are:

    15 Halt on error
    14 Loop on test
    13 Inhibit error typeouts
    11 Inhibit iterations
    10 Bell on error
     9 Loop on error
     8 Loop on test in SWR (7:0)

5.2.2 Visual Test Diagnostic Load and Run Procedure

To load this diagnostic, proceed as follows:

1. Load the diagnostic into memory.
2. Load address 200 octal.
3. Check that index of test patterns appears in the menu area of the VR48 Display Monitor.
4. Individual test patterns may now be selected for display.

5.3 MAINTENANCE SWITCH PURPOSES

Table 5-2 summarizes the design purposes of the four maintenance switches in the VT48.

Table 5-2 Summary of Maintenance Switch Purposes

Maint Switch No.	Purpose
1	Used to enter contents of BDB register into display program counter (DPC). The contents of DPC can then be sampled to verify display file address.
2	Used to load the DPC and then execute one NPR cycle. That is, it serves to access a single word from PDP-11 core memory.
3	Affects graphics calculation logic only. Used to load delta length (larger axis) into X Position register and to load tangent (minor axis) into Y Position register. X/Y position registers can then be sampled to verify contents.
4	Used to single step the graphics calculation logic. That is, it inhibits generation of OP DONE signal to display instruction control until all calculation sequences are complete.

5.4 PREVENTIVE MAINTENANCE

Preventive maintenance for the VT48 consists of visual checks and periodic alignment of the two analog modules (vector generator and character generator).

5.4.1 Mechanical Checks

The VT48 should be visually checked periodically for mechanical defects. Check for the following:

1. Check that the VT48 is secure in its mounting.
2. Inspect the backplane for bent pins and loose wire wrap.
3. Check the power harness for opens or shorts.
4. Check that the cables to both vector generator and character generator are properly secured.
5. Check for the presence of damaged or charred components on all logic modules.

5.4.2 VT48 Alignment Procedures

The adjustment procedures set forth in the subsequent paragraphs presume that all alignment procedures specified in the *VR48 Technical Manual* have already been successfully executed. VR48 Display Monitor adjustments must precede the VT48 adjustments.

5.4.3 Vector Generator (A322) Adjustments

These adjustments require loading of the visual test (VISTEST) diagnostic maintenance program (Paragraph 5.2.2). It may be necessary to perform all adjustments a second time (after a second pass through the VR48 Display Monitor adjustments) if the system is significantly out of calibration at the start of alignment proceedings. The proper sequence of adjustments is as follows:

1. Minor axis (+) offset adjustment.
2. Minor axis (–) offset adjustment.
3. Major axis (–) offset adjustment.
4. Major axis (+) offset adjustment.
5. X–Y phase adjustment.
6. Delta length adjustment.
7. X position gain adjustment.
8. Y position gain adjustment.
9. Minor axis gain adjustment.
10. Major-minor phase adjustment.
11. Dynamic offset adjustment.
12. Light pen start coordinate.
13. Light pen end coordinate.

Locations of the adjustment potentiometers to be used in these checks are shown in Figures 5-1 and 5-2. After having loaded the visual test diagnostic program, execute each checkout procedure in the sequence given.

5.4.3.1 Minor Axis (+) Offset Adjustment (R135, on A322 Module) – To make this adjustment, proceed as follows:

1. Select test pattern E; then check that the test pattern shown in Figure 5-3 appears at left center of the VR48 Display Monitor.
2. If the short lines overlap the dots, adjust R138 (A322) to obtain small gaps between the vector ends and dots.
3. If the vector start points overlap the dots, adjust R7 on the M7054 to move the vector start points out of the way.
4. Adjust R135 (A322) so that lines 2, 4, and 6 align with the points as shown in Figure 5-3.

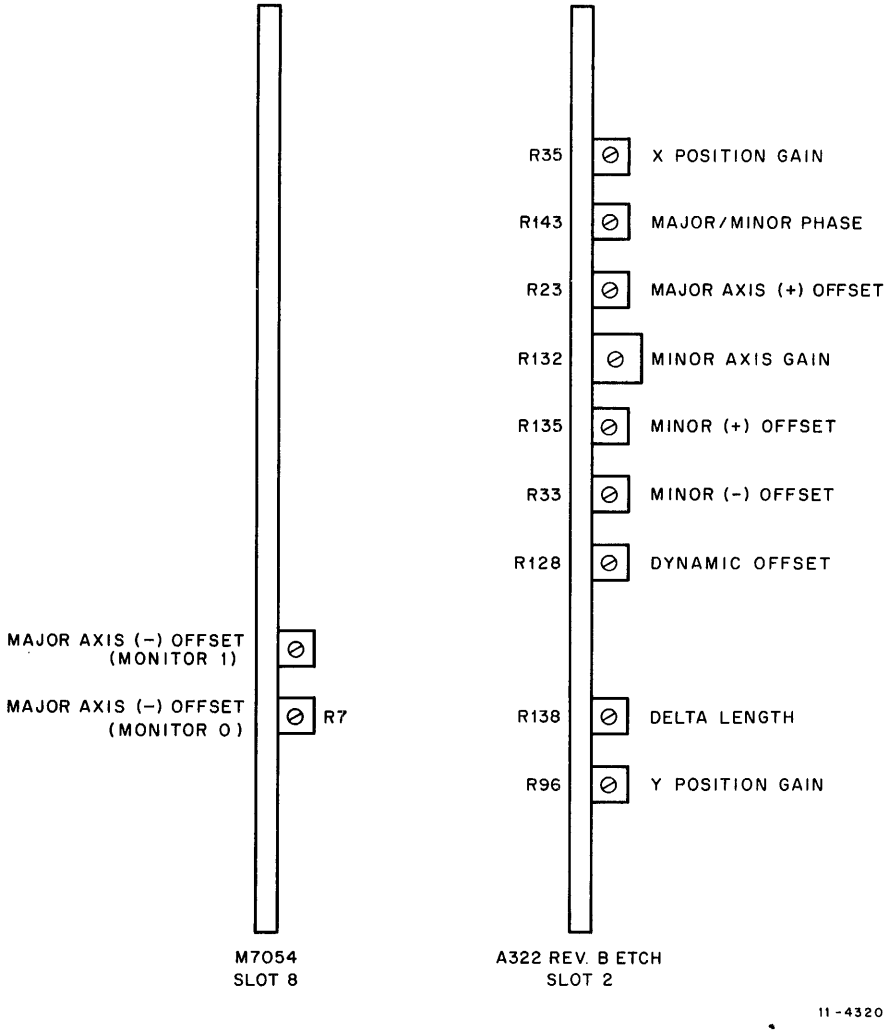


Figure 5-1 Vector Generator (Revision B)
Adjustment Potentiometer Locations

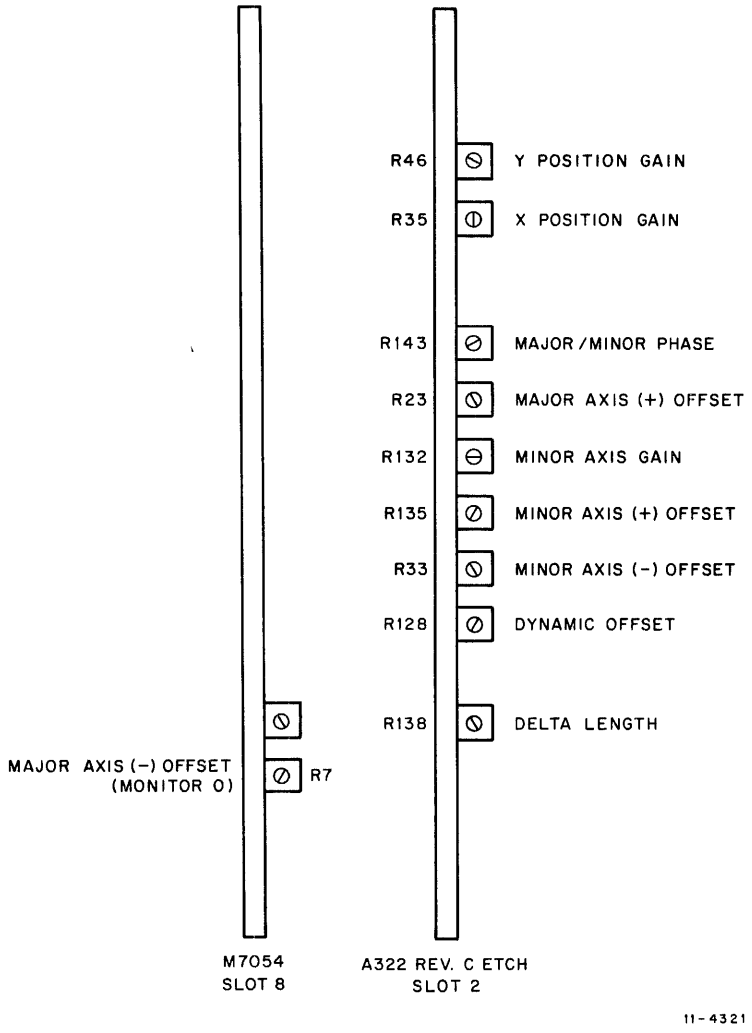


Figure 5-2 Vector Generator (Revision C)
Adjustment Potentiometer Locations

5.4.3.2 Minor Axis (-) Offset Adjustment (R33, on A322 Module) - To make this adjustment, proceed as follows:

1. Select test pattern E; then check that the test pattern shown in Figure 5-3 appears at left center of the VR48 Display Monitor.
2. Adjust R33 on the A322 module so that lines 1, 3, and 5 align with the points.
3. Check for a straight vertical line of short vectors and dots as shown in Figure 5-3.

5.4.3.3 Major Axis (-) Offset Adjustment - This adjustment is made using R7 on the M7054 module. Proceed as follows:

1. Select test pattern E; then observe the VR48 Display Monitor for the test pattern shown in Figure 5-4.

2. Adjust R7 for the following conditions as observed on the test pattern:

   a. Line 3 should start directly on top of the line causing a small bright spot where lines 3 and 1 intersect.

   b. A very small gap exists between lines 5 and 1. That is, line 5 ends just to the left of line 1.

NOTE
If lines 2 and 4 overlap line 1, thereby interfering with above procedures, adjust R23 on the A322 module to move lines 2 and 4 off to the right.

Figure 5-3 Minor Axis Offset Subpicture, Test Pattern E

5.4.3.4 Major Axis (+) Offset Adjustment – This is a two-pass procedure involving the setting of R23 on the vector generator. For the first pass of the procedure, proceed as follows:

1. Select test pattern E; then observe for the test pattern shown in Figure 5-4.

2. Adjust R23 for the following conditions as observed on the test pattern:

   a. Line 2 should just overlap line 1 in the same manner as line 3 in the preceding adjustment.

   b. A very small gap exists between lines 1 and 4. That is, line 4 starts just to the right of line 1.

The second pass of this adjustment is performed, if necessary, after all succeeding adjustments have been made and serves as a check on the interaction of R23 with the other adjustments.

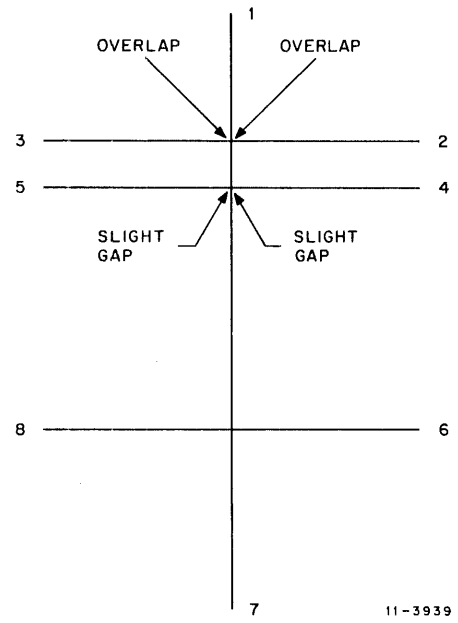


Figure 5-4 Major Axis Offset Subpicture, Test Pattern E

After all other adjustments have been completed, proceed as follows:

1. Select test pattern E, then observe the VR48 Display Monitor for the test pattern shown in Figure 5-5.
2. Check to see if line pairs (1) and (3) are the same distance apart so that they appear as mirror images of one another. If not, adjust R23 for the mirror image effect.

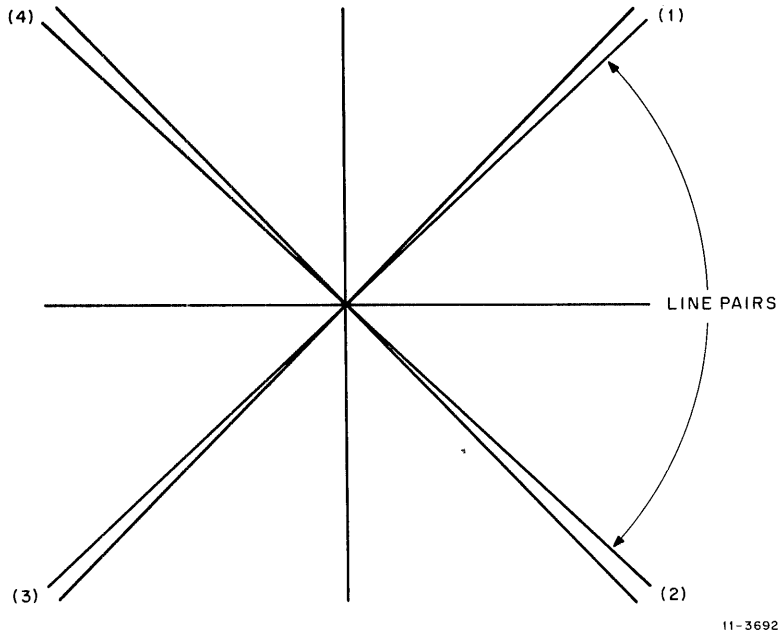


Figure 5-5 Dynamic Offset Subpicture

5.4.3.5 X-Y Phase Adjustment - This adjustment is carried out at the VR48 Display Monitor by adjusting R32 on the Y deflection amplifier module (phase) and R32 on the X-deflection amplifier (bandwidth). Refer to Figure 5-6. This is a two-pass test. For the first pass, proceed as follows:

1. Select test pattern E; then observe the VR48 Display Monitor for the pattern shown in Figure 5-7 (left side of CRT screen).

2. Adjust bandwidth potentiometer (R32 on X-Defl. Amp.) CCW until oscillation (or ringing) is observed on display. Re-adjust potentiometer CW until oscillation disappears. Then adjust potentiometer CW an additional 1/8 of a turn.

3. Adjust R32 Y-Phase potentiometer on the VR48 Display Monitor so that the vector start points in all directions are equal in the center.

For the second pass, proceed as follows:

1. Observe the VR48 Display Monitor for the subpicture shown in Figure 5-8.

2. Observe the diagonal lines in the small square.

3. Adjust the Y-phase potentiometer (R32 on VR48) so that these lines overlay one another without bowing.

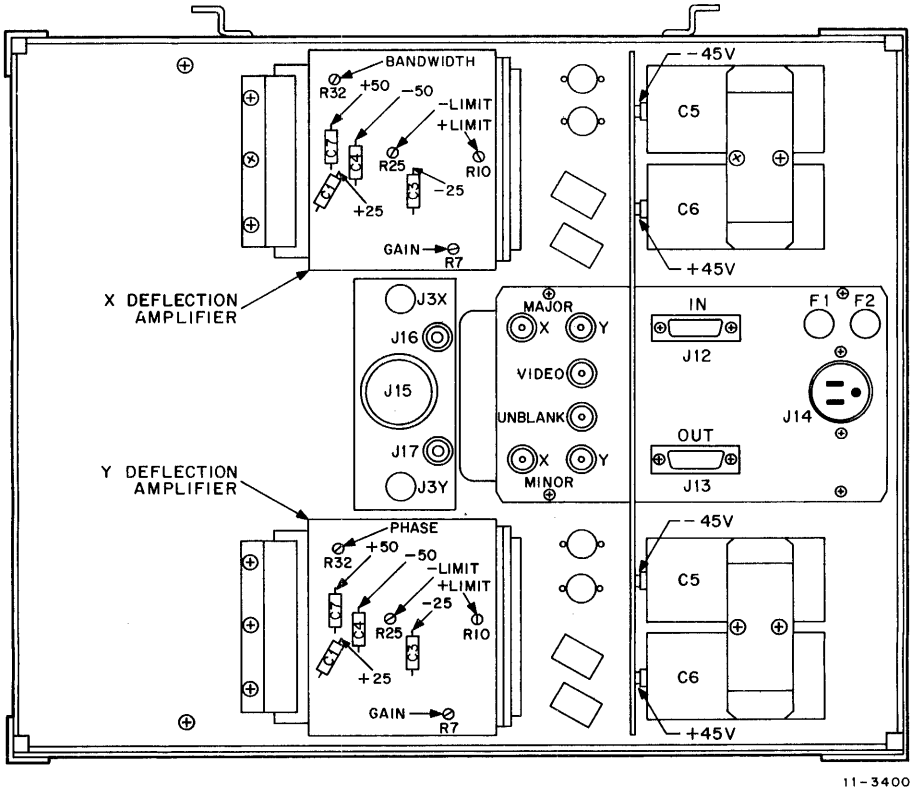


Figure 5-6 VR48 Monitor Phase Adjustment Location

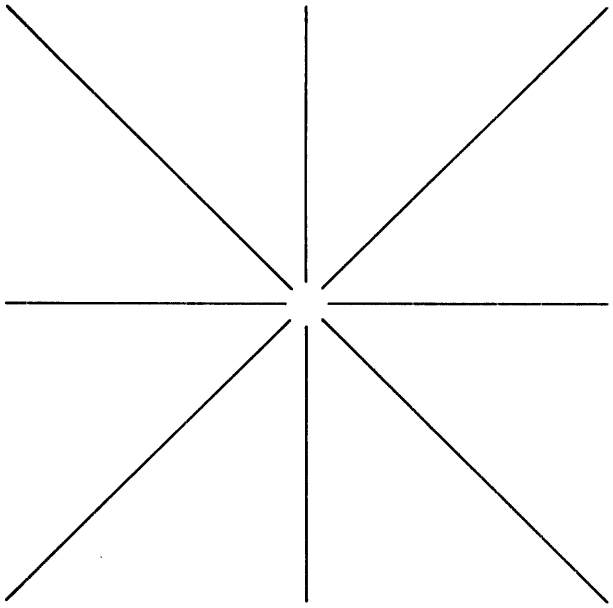


Figure 5-7 X–Y Phase Subpicture

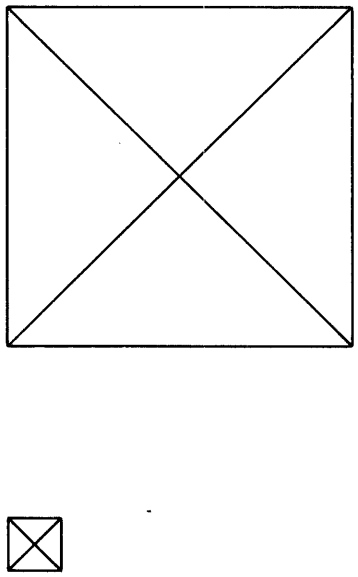


Figure 5-8 X–Y Phase/Delta Length Subpicture

5.4.3.6 Delta Length Adjustment – To carry out this adjustment, proceed as follows:

1. Select test pattern E; then observe the VR48 Display Monitor for the pattern shown in Figure 5-8.

2. Adjust R138 on the vector generator so that the horizontal and vertical lines on both boxes meet, but do not overlap (criss-cross).

NOTE

If the large box side lines fail to meet while the small box side lines overlap (meaning it is impossible to achieve the conditions defined in step 2), it indicates that the VR48 X–Y phase is out of adjustment.

5.4.3.7 X Position Gain – This involves adjustment of R35 on the vector generator. Proceed as follows:

1. Select test pattern G; then observe the VR48 Display Monitor for the pattern shown in Figure 5-9.

2. Observe the vertical vector on the right side of the screen.

3. Adjust R35 so that the ends of the horizontal vectors meet the vertical vector on the right side of the screen. There should be a slight overlay as indicated by small bright spots where the vectors meet.

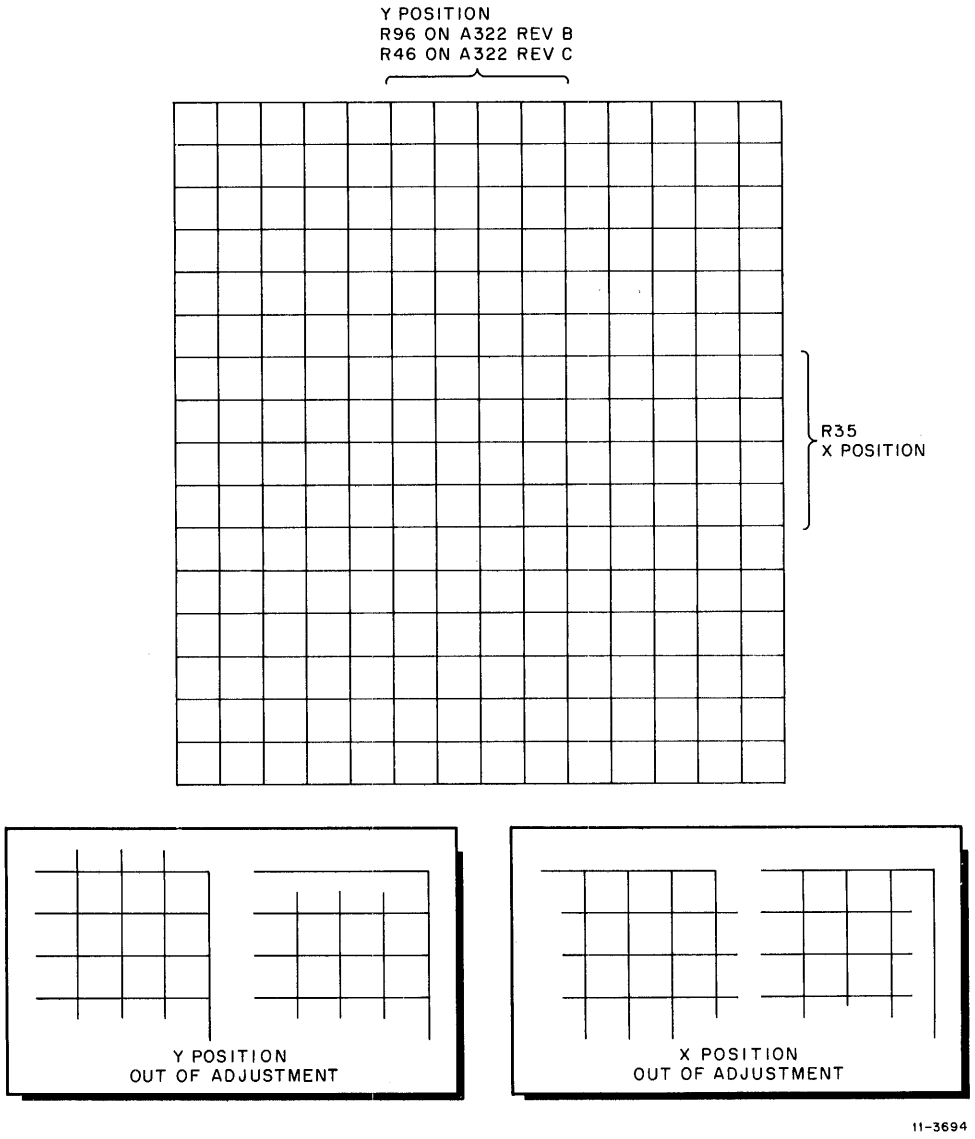


Figure 5-9 Pincushion Pattern

5.4.3.8 Y Position Gain – This adjustment involves potentiometer R96 (revision B) or R46 (revision C). Proceed as follows:

1. Select test pattern G; then observe the VR48 Display Monitor for the pattern shown in Figure 5-9.

2. Observe the horizontal vector across the top of the screen.

3. Adjust the Y-position gain potentiometer so that the ends of the vertical vectors meet the horizontal vector at the top of the screen. There should be a slight overlay as indicated by small bright spots where the vectors meet.

5.4.3.9 Minor Axis Gain – This is a two-pass procedure involving R132 on the vector generator. The first pass is made prior to conducting major-minor phase and offset adjustments (next two paragraphs). After completing the latter adjustments, the second pass is made. For the first pass, proceed as follows:

1. Select test pattern P; then observe the VR48 Display Monitor for the patterns shown in Figure 5-10.
2. Observe the vectors falling along the 45 degree angle area.
3. Adjust R132 so that adjacent 45 degree vectors are parallel and touching. That is, the error delta-2 value should equal the error delta-1 value. Both should be zero (not separated, and not overlapping).

For the second pass, adjust R132 so that test pattern P has no discontinuities in the 45 degree angle area. The vector line pattern should appear uniform from zero through 90 degrees.

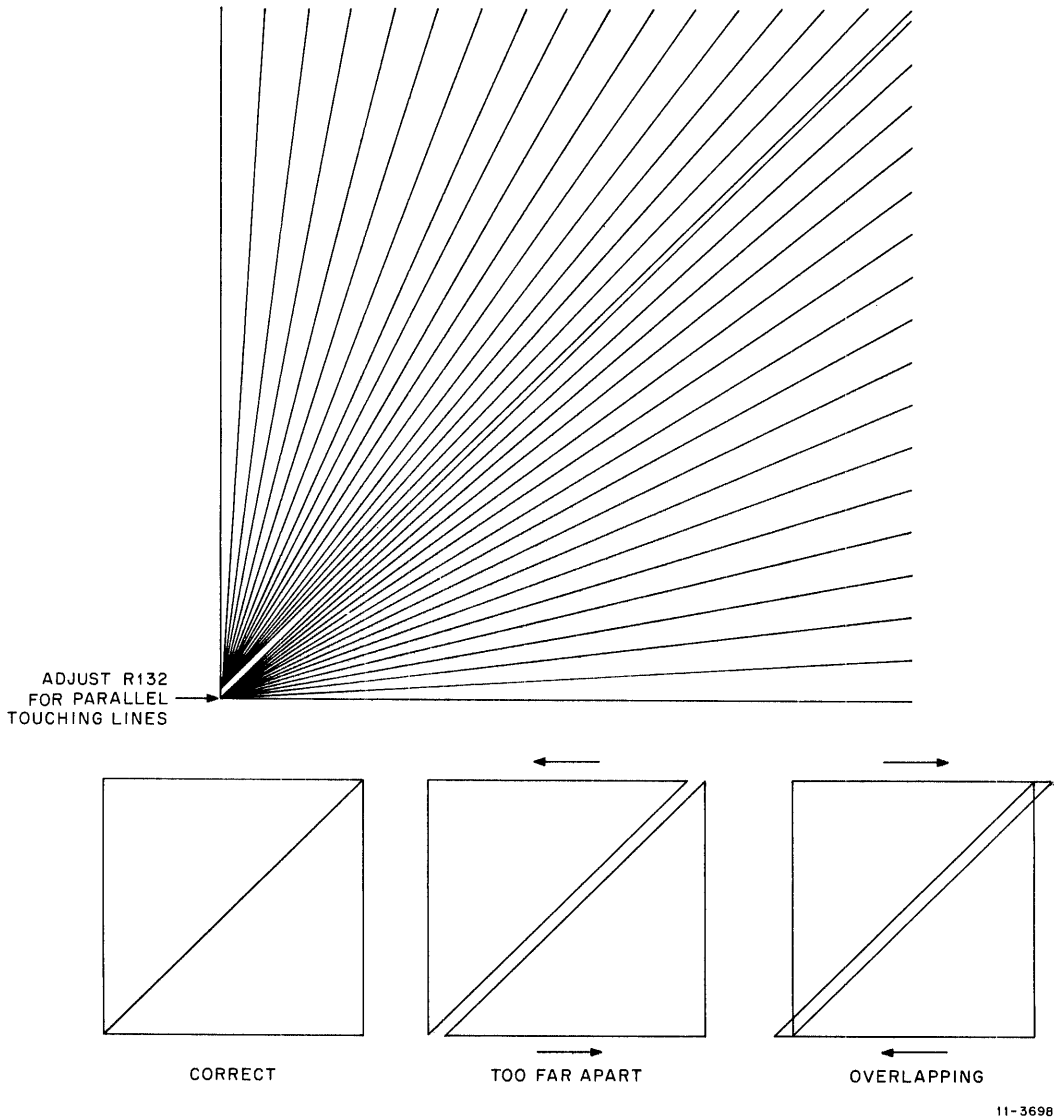


Figure 5-10 Vector Fans Test Pattern, Part 1

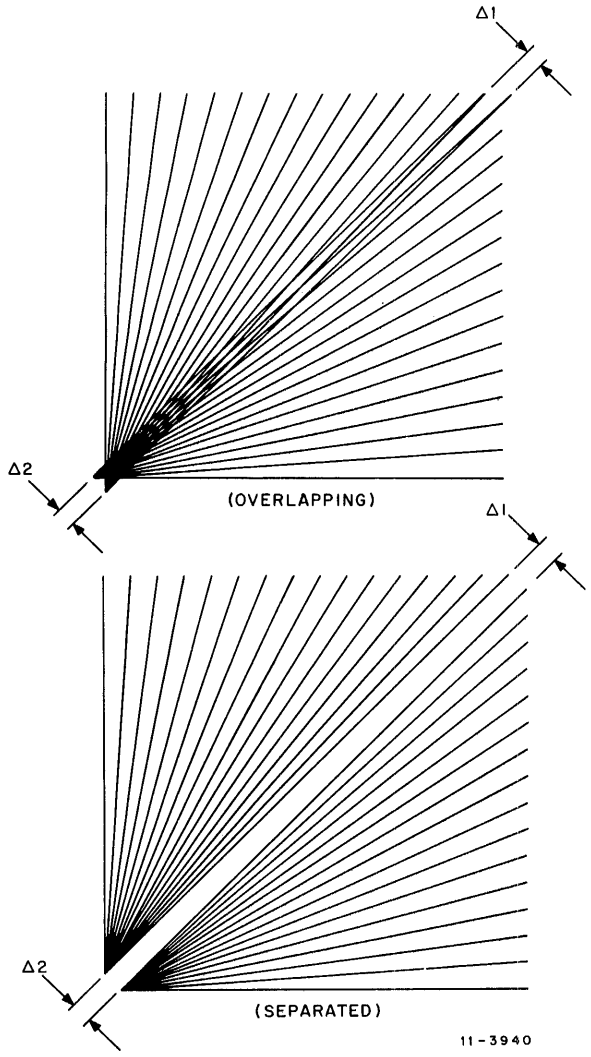


Figure 5-10 Vector Fans Test Pattern, Part 2

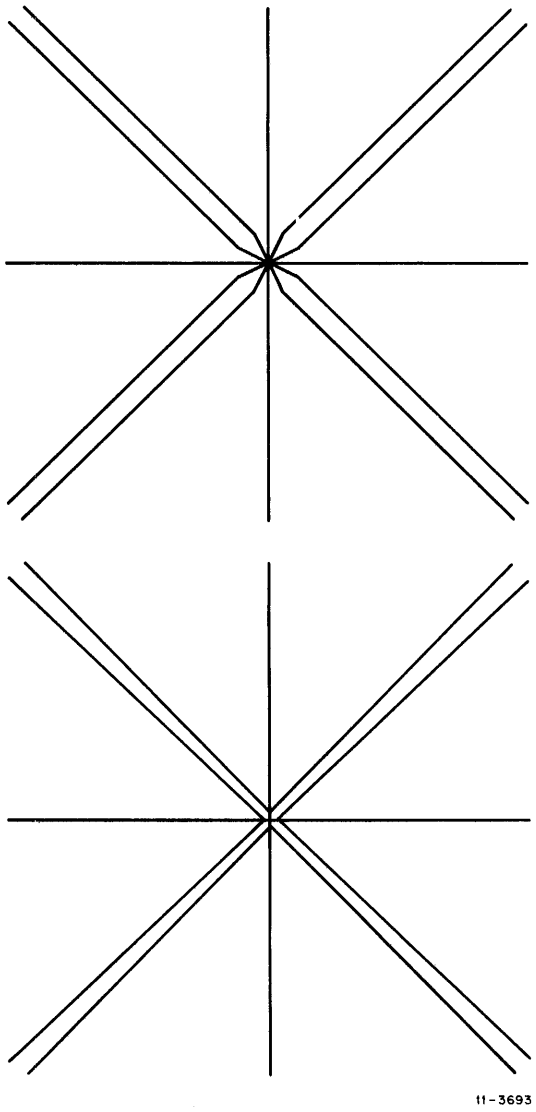


Figure 5-11 Examples of Bowed Vectors

5.4.3.10 Major-Minor Phase Adjustment – This adjustment procedure involves R143 on the vector generator. Proceed as follows:

1. Select test pattern E; then observe the VR48 Display Monitor for the dynamic offset pattern shown in Figure 5-5.
2. Adjust R143 so that the pairs of lines diverge evenly without bowing. Bowing is most easily detected when the pairs of lines are separated slightly.
3. Adjust R128 to slightly separate the pairs of lines; then observe the pattern for bowing effects as shown in Figure 5-11.
4. Adjust R143 so the vector pairs diverge evenly without bowing. When properly adjusted, both pairs should start at a common center and diverge to approximately 2 line-widths (24 mils) at the ends.

5.4.3.11 Dynamic Offset Adjustment - This adjustment involves R128 on the vector generator. Proceed as follows:

1. Select test pattern E; then observe the display monitor for the dynamic offset pattern shown in Figure 5-5.
2. Adjust R128 so that the diagonal line pairs numbered 2, 3, and 4 originate from the center as shown in Figure 5-5.

NOTE

If lines 1+3 do not converge at the center, readjust R23 major axis (+) offset to center the point of convergence.

3. Following this adjustment proceed to second pass of minor axis gain procedure under Paragraph 5.4.3.9.

5.4.3.12 Light Pen Start Coordinate Adjustment - This procedure involves rotary switch S1 on the M7053 module. An extender board is required to access the switch. Proceed as follows:

1. Select the light pen and intensity level test, pattern U.
2. Observe the VR48 Display Monitor for eight different intensity lines displayed at the left side of the screen.
3. Using the light pen, select the line designated Y = 1400. (The coordinates of the light pen may be read at the upper right side of the screen.)
4. Move the light pen to the left along the selected line until it reaches the end of the line.
5. Check for an X coordinate reading of X = 201 (Y = 1400).
6. Adjust S1 for an X coordinate reading as close as possible to X = 201.

5.4.3.13 Light Pen End Coordinate - This procedure involves R100 on the vector generator module. An extender board is required to access this potentiometer. Proceed as follows:

1. Select the light pen and intensity level test, pattern U.
2. Observe the VR48 Display Monitor for eight different intensity lines at the left of the screen.
3. Using the light pen, select the line designated Y = 1400.
4. Move the light pen slowly to the right end of the line. As the light pen passes over the last 1/4 inch, the X coordinate values should increase smoothly from 750_8 to 1000_8.
5. Adjust R100 for this condition.

NOTE

As the light pen is moved slowly to the right over the last quarter inch, the X coordinate readout should not skip any coordinate values. Also the X readout should not reach X = 1000 before the light pen is at the end of the line.

5.4.4 Sync Clock Adjustments

This adjustment is made using R18 and R19 on the M7057 module. No diagnostic or test pattern is required. Proceed as follows (Figure 5-12):

1. Extend the M7057 module.
2. Observe test point E64/p.6 (signal DSR2 30 FPS H) on an oscilloscope. Adjust R18 on the M7057 module for a pulse period of 33.3 ms.
3. Observe test point E64/p.8 (signal DSR2 40 FPS H). Adjust R19 on the M7057 module for a pulse period of 25 ms.
4. Replace the M7057 module.

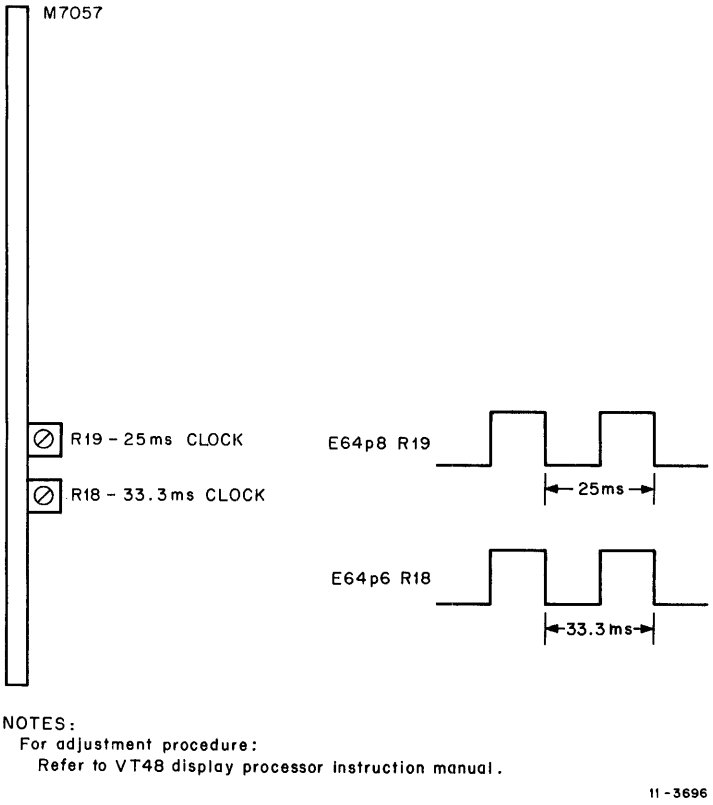


Figure 5-12 Sync Clock Adjustments

5.4.5 Character Generator Adjustments

These adjustments require the use of the VISTEST (Paragraph 5.2.2) diagnostic program. In addition, a short character display routine must be manually entered from the console. Proceed as follows (Figures 5-13 and 5-14):

1. Extend the A321 module.

2. Manually enter the following routine from the console:

Loc. (Octal)	Contents (Octal)
001000	012737
001001	002000
001002	172000
001003	000001
002000	114000
002001	001000
002003	100000
002004	000102
002005	160000
002006	002000

3. Observe that a 'B' is displayed on the VR48 Display Monitor.

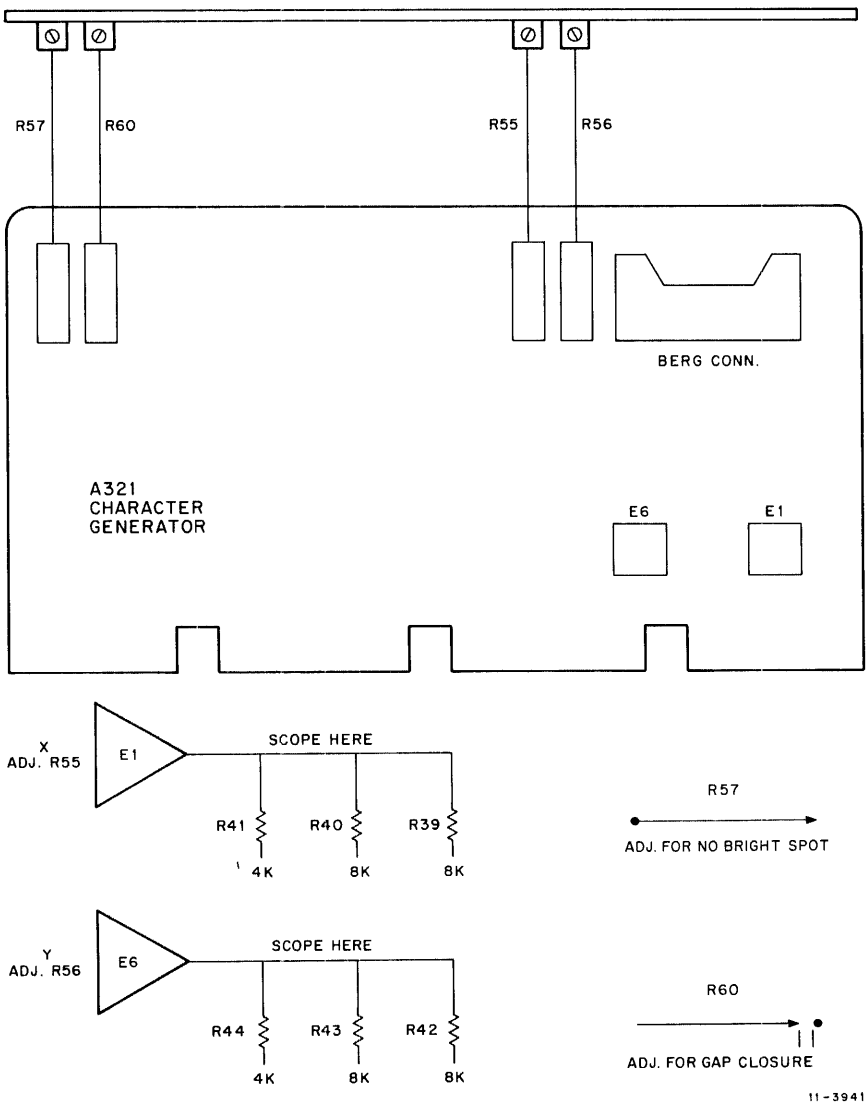


Figure 5-13 Character Generator Adjustments

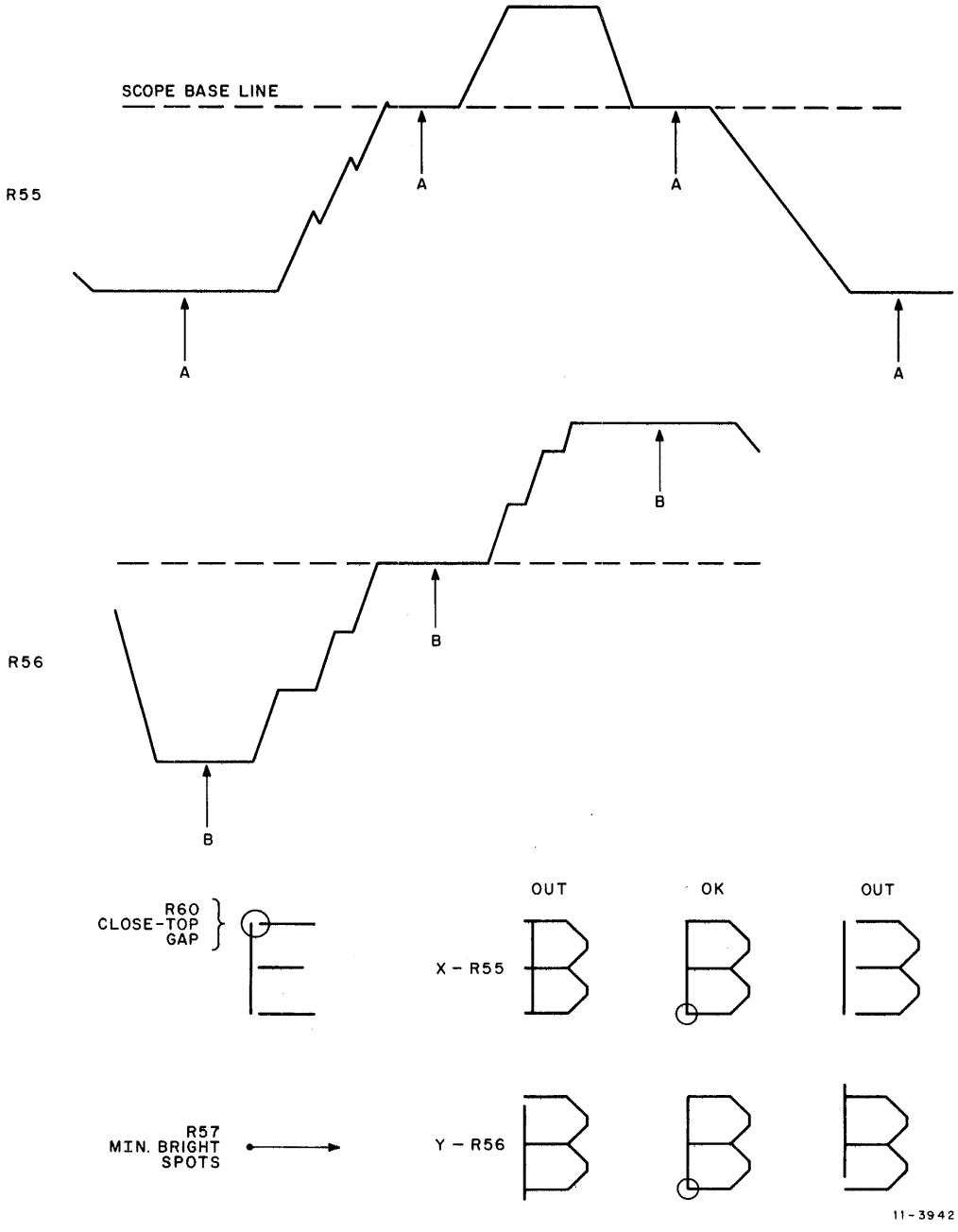


Figure 5-14 Character Generator Waveforms/Display

4. Connect the channel 1 oscilloscope probe to the junction of R39/R40/R41 on the A321 module.

5. Set the oscilloscope controls to the settings indicated below:

Control	Setting
TIME/DIV.	0.5 μs/DIV.
VOLTS/DIV.	0.5 V/DIV.
'TRIGGER' SEL SW.	Chan 1
'INPUT CHAN 1' SELECT SW.	DC
'SLOPE' CONTROL	(-)

6. Re-adjust controls as necessary to obtain waveform I in Figure 5-14.

7. Adjust R55 on the A321 module until the lines marked A in the waveform are parallel with the grid markings on the oscilloscope.

8. Reconnect the channel 1 scope probe to the junction of R42/R43/R44 on the A321 module.

9. Adjust oscilloscope controls as necessary to obtain waveform II in Figure 5-14.

10. Adjust R56 on the A321 module until the lines marked B in the waveform are parallel with the grid markings on the oscilloscope.

NOTE

Steps 7 and 10 above are approximate adjustments only. A fine adjustment of R55 and R56 will be made along with the R57 and R60 adjustments to follow.

11. Remove the A321 module extender and re-insert the module in slot 1.

12. Select VISTEST, pattern M (reload diagnostic if required).

13. Observe the large Es in the center of the VR48 display.

14. Adjust R60 on the A321 module to close the gap at the top of the Es as depicted in Figure 5-14.

15. Adjust R57 for minimum bright spots at the origin of the vectors as shown in Figure 5-14.

16. Re-adjust R55 so that the 3 portion of the B touches the vertical (or 1) portion as shown in Figure 5-14. (Note that R55 moves the 3 in a horizontal direction.)

17. Re-adjust R56 so that the 3 portion of the B lines up with the bottom and top of the 1 portion as shown in Figure 5-14. (Note that R56 moves the 3 vertically.)

18. Select VISTEST, pattern A.

19. "Fine tune" R55 and R56 for optimum display characteristics of the characters C, G, O, and B. Additional fine tuning of R57 and R60 may also be necessary at this point.

APPENDIX A
CHARACTER STROKE PATTERNS

This appendix presents illustrations showing the stroke patterns for all characters generated by the character generator. A breakdown of these illustrations is given below:

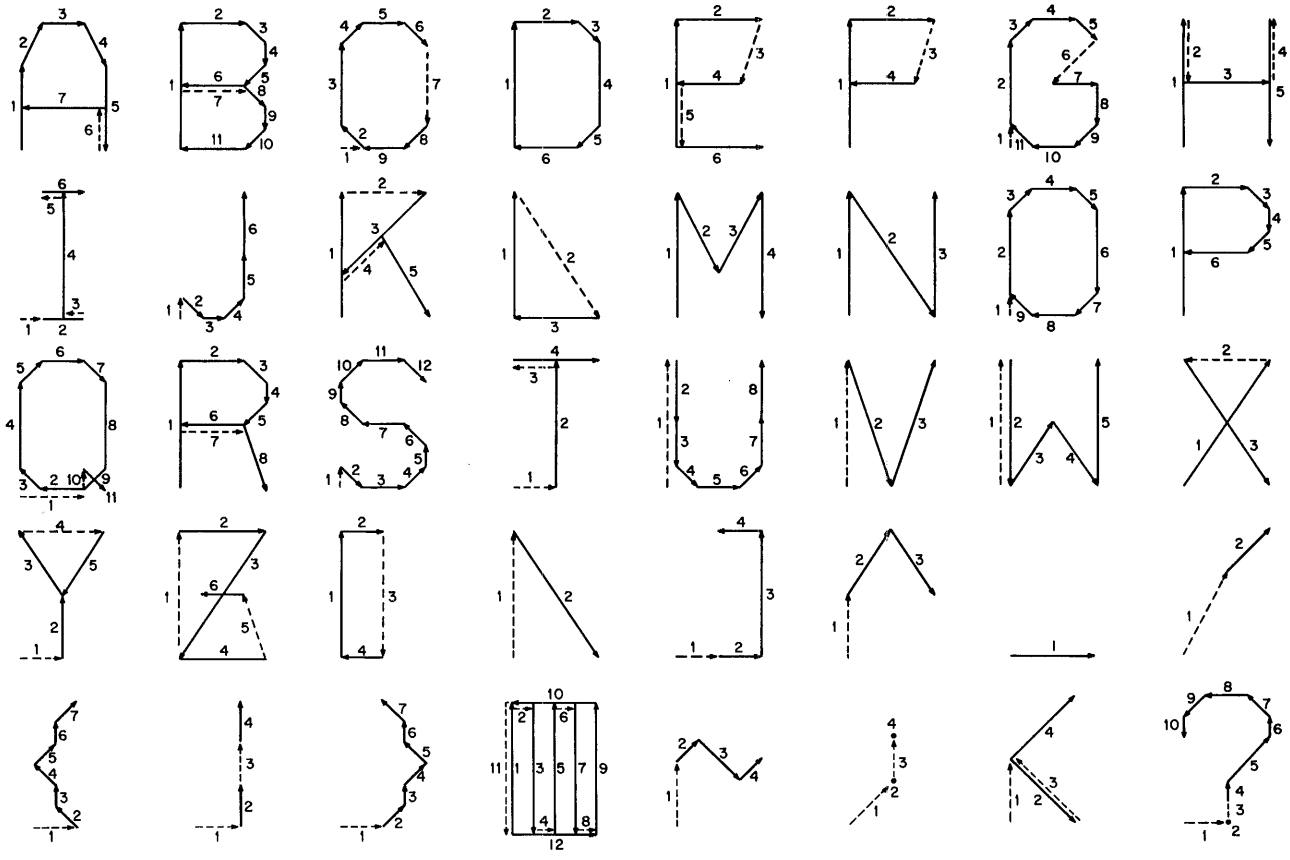


Figure A-1 Upper Case Font and Special Symbols

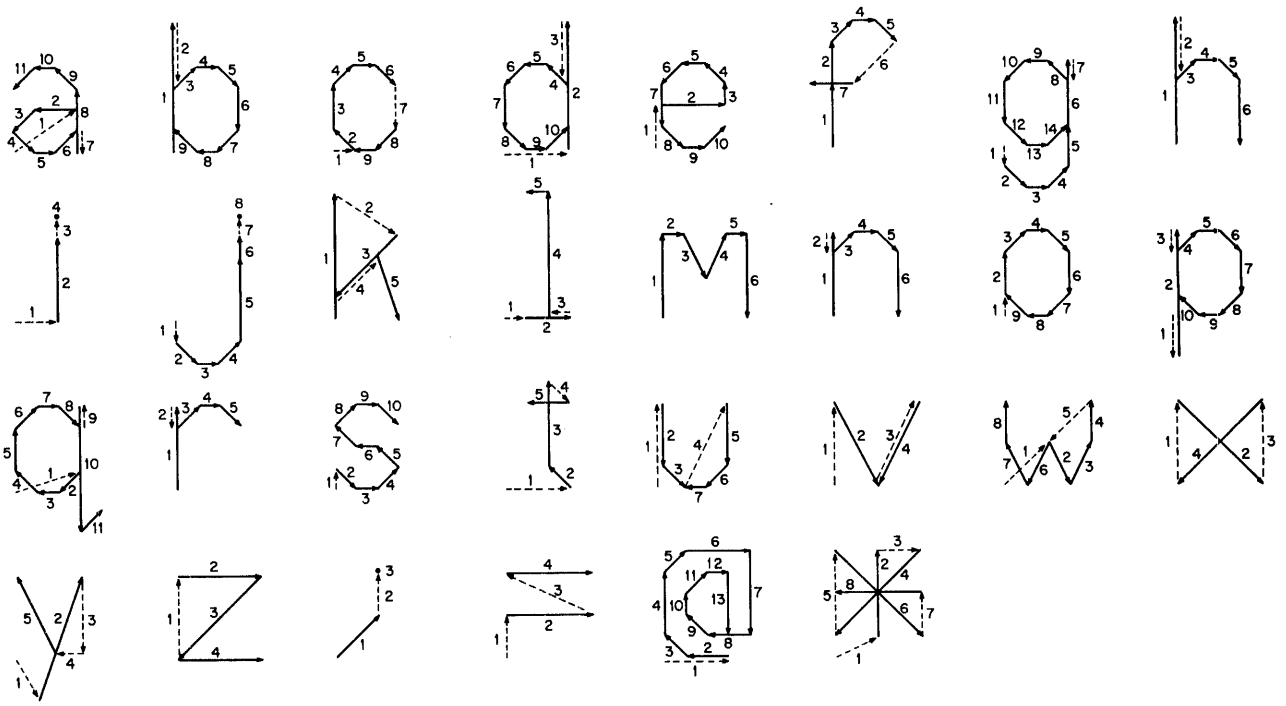


Figure A-2 Lower Case Font and Special Symbols

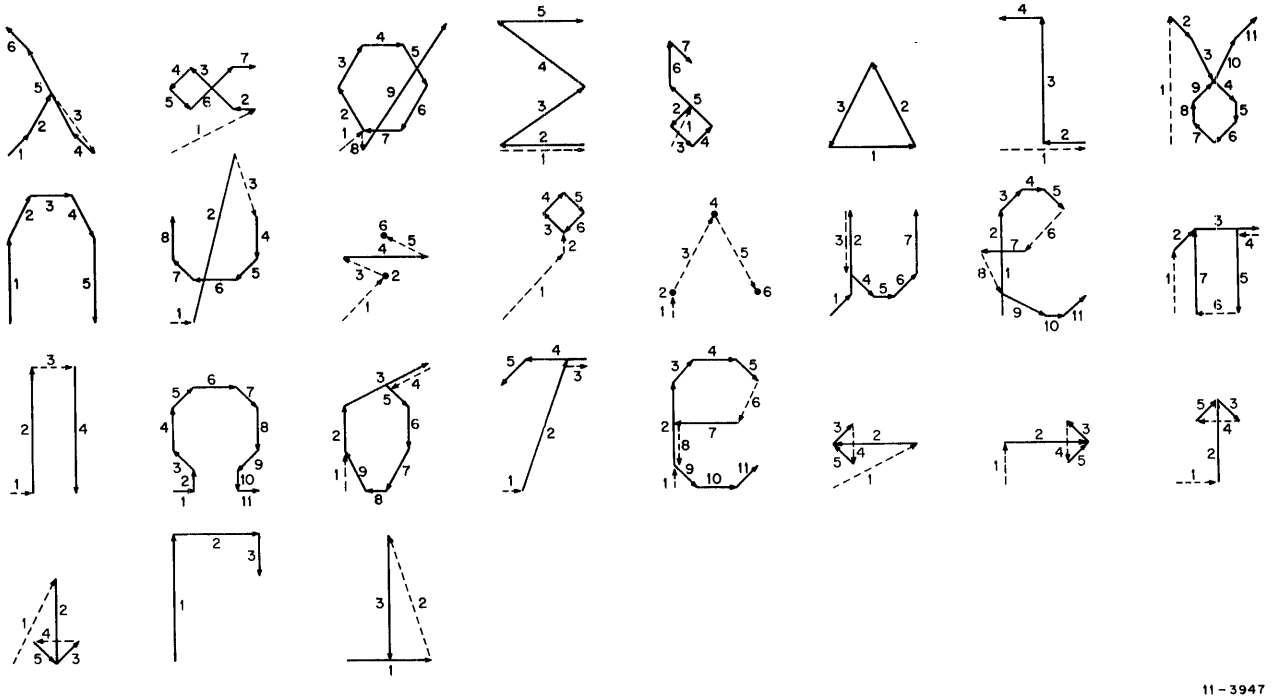


Figure A-3 Greek Character Font and Special Symbols

Figure A-4 Numeral Font

VT48 DISPLAY PROCESSOR UNIT
TECHNICAL MANUAL
EK-VT48-TM-001

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? _____

_____

_____

_____

What features are most useful? _____

_____

_____

_____

What faults do you find with the manual? _____

_____

_____

_____

Does this manual satisfy the need you think it was intended to satisfy? _____

Does it satisfy *your* needs? _____ Why? _____

_____

_____

_____

Would you please indicate any factual errors you have found. _____

_____

_____

_____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

— — — — — — — — — — — — Fold Here — — — — — — — — — — —

— — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
Maynard, Massachusetts 01754