Micron Technology
Memory Applications Group

8-Bit Color Video Board
for the Macintosh SE/30
640x480
Xceed SE/306-48

C. Snodgrass
L. Mills
Version 1.00
August 15, 1989

For Internal Use Only

Glen Atkins
Greg Blodgett
MARK DAY
Laura Mills
Chuck Snodgrass
Tom Stamm

CONTENTS

Chapter 1
Introduction

The Micron Xceed SE/306-48 is an 8-bit board for the Macintosh SE/30. The standard Macintosh SE/30 video is a built-in black-and-white screen, 512x342 pixels. For Micron's 8-bit color video board, the pixel data is represented by 1, 2, 4, or 8 data bits at 640 x 480 resolution. This allows a selection of over 16 million colors (16,777,216) for each pixel, but only 256 colors may be shown on the monitor at one time. The 24 bits of information required for the monitor are obtained from a look-up table, using the 8 bits of board data as the table index.

The eight bits representing a color are converted to an RGB value (RGB monitor = red, green, blue) using the look-up table. The pixel color is a combination of the red, green, and blue component values obtained from the table. Each component color requires eight bits of information, converted to an analog voltage level with a D/A converter. These three resulting voltages, one for each component color, go to the monitor to create the color for a single pixel. The pixels appear left-to-right, top-to-bottom on the monitor; scanning across each line then down to the next line. The screen is 640 x 480 pixels, requiring almost 1/2 MB of on-board memory (300KB) to store a single screen of data.

Besides the Video RAMs (VRAMs) to store the picture data, there is a ROM on the board that contains the video board driver and information that enables the Macintosh to identify the board. When the computer is powered up, it checks for any boards by reading the information in the ROM. Eleven PALs control most board functions. The video board is in slot A.

Chapter 2
Macintosh SE/30

The Macintosh SE/30 contains the Motorola 68030 microprocessor and one expansion slot, which may be numbered 9, A, or B hex. The SE/30 is essentially a Macintosh IIx computer in a smaller box. The expansion slot, though, comes off of the processor bus. The SE/30 does not have any NuBus expansion slots.

While the Macintosh SE/30 has 32 address lines, only 24 of these are used in the normal (default) mode. The computer can be placed in 24-bit or 32-bit mode similar to the Macintosh II. The 8-bit video board's ROM and VRAM array addresses, however, are all directly accessible in normal mode.

Address	Description
00000000 - 000FFFFF	RAM (minimum configuration)
00100000 - 00CFFFFF	RAM (expansion area)
00D00000 - 3FFFFFFF	RAM (undefined)
40000000 - 4007FFFF	ROM Bank 0 (minimum configuration)
40080000 - 4FFFFFFF	ROM (undefined)
50000000 - 50001FFF	VIA1 (x0200)
50002000 - 50003FFF	VIA2 (x0200)
50004000 - 50005FFF	SCC (x0002)
50006000 - 50007FFF	SCSI (Handshake)
50010000 - 50011FFF	SCSI (x0010)
50012000 - 50013FFF	SCSI (Pseudo DMA)
50014000 - 50015FFF	Sound
50016000 - 50017FFF	SWIM
50018000 - 57FFFFFF	(undefined)
58000000 - 5FFFFFFF	030 Direct Slot expansion (if pseudo-NuBus is not used)
60000000 - F8FFFFFF	expansion (undefined)
F9000000 - FBFFFFFF	expansion pseudo-NuBus slots
FC000000 - FDFFFFFF	expansion (undefined)
FE000000 - FE00FFFF	video RAM space
FEFF0000 - FEFFFFFF	video ROM space
FF000000 - FFFFFFFF	expansion (undefined)

Table 2-1 Address Mapping

The Macintosh SE/30 uses memory-mapped I/O. Each device in the system can be accessed by reading or writing to specific address locations in the address space of the computer. The addressing for a card in a slot is directly dependent on the slot number. The SE/30 has three slot numbers possible for a single slot. The slot number is defined in hardware on the expansion board and cannot be changed.

The slot in the Macintosh SE/30 can be directly accessed in the 32-bit mode at the address found by multiplying the slot ID by 10,000,000 hex (s000 0000-sFFF FFFF hex); this is the Super Slot Space. The slot can also be found at the Slot Space Fs00 0000-FsFF FFFF hex (s = slot ID, 9-B) when in the 24-bit default operating mode. Each physical slot has both the slot and super slot space allocated to it. In 24-bit mode the AMU (Address Mapping Unit) or PMMU (Paged Memory Management Unit), whichever is present, will convert s0 0000 to Fs00 0000.

2.1 Bus Signals

The Macintosh SE/30 Direct Slot expansion bus is based on the Motorola 68030 microprocessor. It is machine-specific, not a general-purpose bus like NuBus. The card design may be similar to NuBus (Pseudo-NuBus), with the same type of ROM and addressing, but the pinout and signals differ. The SE/30 Direct Slot uses a 120-pin 16MHz synchronous bus. Words, halfwords, and bytes can be read and written, but the bus is optimized for word transfers. All signals are active low.

Bit 31 Bit 0

Direct Slot Word			
Halfword 1		Halfword 0	
Byte 3	Byte 2	Byte 1	Byte 0

Table 2-2 Bus Data

The bus specification divides the signals into five types:

Power: +5V, +12V, -12V, and -5V.
Data and Address Lines: A0-A31, D0-D31.
Control Lines: \STERM, SIZ0-SIZ1, FC0-FC2, \RESET, \BERR, R/W, \AS
Clocks: CPUclock and C16M (for the SE/30, these are identical 16 MHz signals).
Machine-Specific Signals: PWROFF, \BUSLOCK, \IRQ1-\IRQ3, \TM0A-\TM1A, \NuBus, CPUclock.

The connector contains three rows of pins (see Appendix B). The Micron video board does not use all of the signals on the expansion bus. A 'high' level is > 2V, 'low' level < 0.8V.

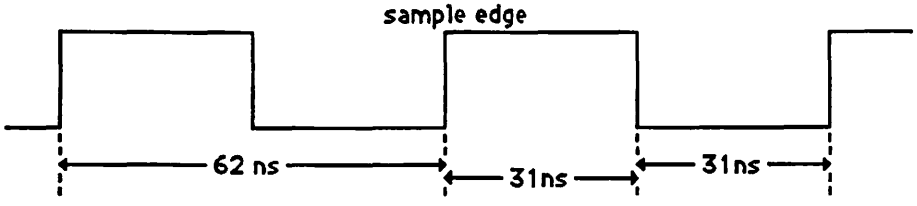


Figure 2-1 System Clock

2.2 READ-WRITE-READ CYCLE

The read-write-read transaction is shown below. It may take longer than shown, but the timer on the SE/30 main logic board will generate a bus error signal when the address strobe is asserted for longer than ≈20 microseconds.

SE/30 Read/Write Cycles
(All accesses to the video card are 32 Bit Synchronous)

Read Cycle

At state zero (S0) the SE/30 drives Ax, FCx, SIZx, and R\W with an address, type of function, size of data transfer, and read command. Half a clock cycle later, during state one (S1), \AS and \DS are asserted by the SE/30 to tell the external device that the address lines are now valid and data may be placed on the bus. By the end of S1 the external device will assert \STERM. \STERM tells the SE/30 that the read data will be valid by the next falling clock. At the start of state two (s2) the external device places data on the bus and terminates \STERM. The SE/30 latches in the data at the end of S2. State three (S3) provides the data hold time as required.

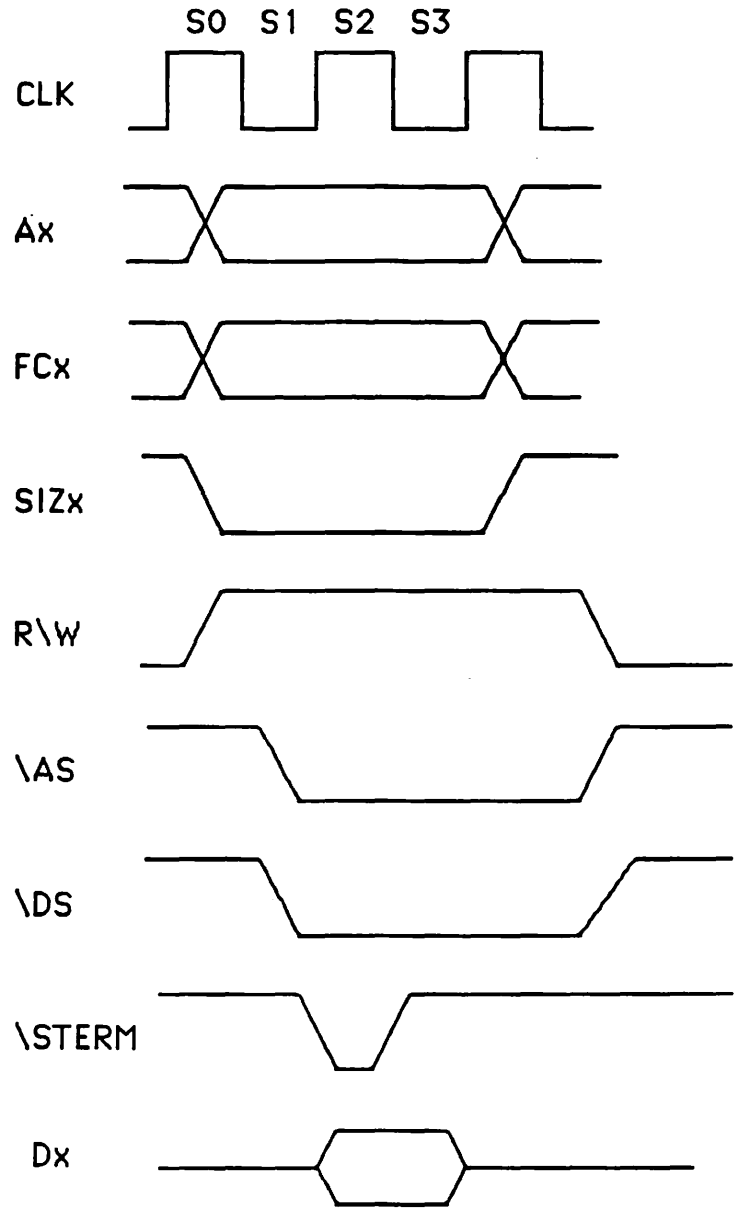


Figure 2-2 SE/30 32 Bit Synchronous Read

Write Cycle

At state zero (S0) the SE/30 drives Ax, FCx, SIZx, and R\W with an address, type of function, size of data transfer, and write command. Half a clock cycle later, during state one (S1), \AS is asserted by the SE/30 to tell the external device that the address lines are valid. At the beginning of state two (S2) the SE/30 looks for \STERM. If \STERM is not present the SE/30 will insert a wait state. Also during S2 the SE/30 will drive the data bus with valid data. During a wait state (SW) the SE/30 asserts \DS to tell the external device that the bus still contains valid data. At the start of each wait state the SE/30 will check for \STERM. If \STERM is not present the SE/30 will insert another wait state, if

\STERM is present, the SE/30 will terminate the write cycle in the following state (S3). During state three (S3) the address and data lines remain valid.

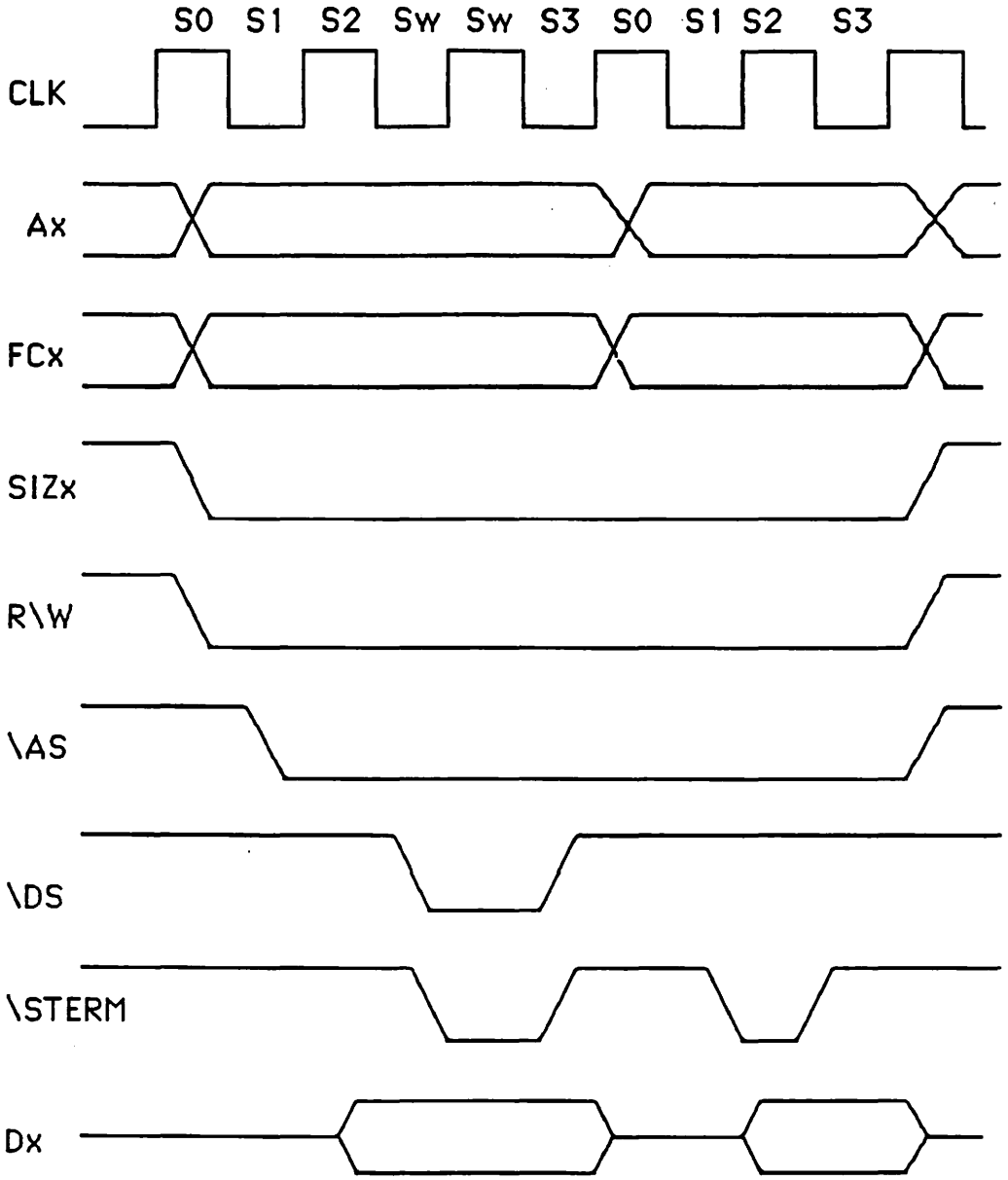


Figure 2-3 Read-Write-Read Cycles

2.3 System Board

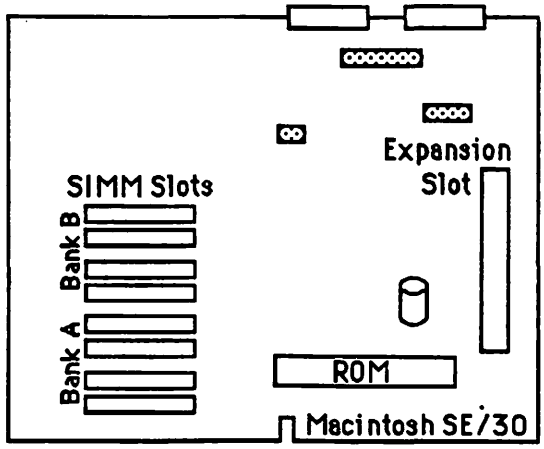


Figure 2-4 Macintosh SE/30 System Board

Chapter 3
Theory of Operation

August 11, 1989
Chuck Snodgrass x3841

Architecture

The video board can be broken down into three functional blocks: the bus interface, the video timing and the video output. The Macintosh SE/30 uses a Motorola 68030 processor which communicates with the video board over the bus. The video timing section of the board determines the video rate and format. The video output section drives out analog RGB video to a monitor.

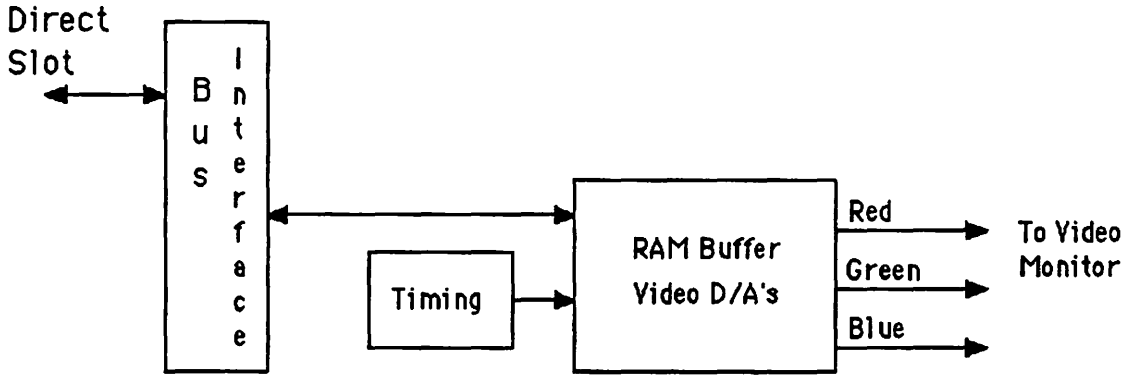


Figure 3-1 Block Diagram

3.1 Bus Interface

The SE/30 bus is a 32 bit 16MHz bus. In addition to 32 bits of address and data, there are several control lines. For write cycles, the address is also driven on the first rising edge of the clock and then the data is driven on the second and remains until a transfer acknowledge signal is received (\STERM low) or until a timeout occurs. Bus timeouts will occur if no acknowledge is received within ≈20 microseconds of a cycle start. Typically a bus timeout will result in a system error which will require the user to reboot the Macintosh. The \AS signal is used to indicate the beginning of a bus cycle if the \STERM signal is high. The \FC0 and \FC1 control lines are used at the beginning of a cycle to define the type of cycle to be performed, and at the end of the cycle

\STERM is used to determine when the cycle was completed. The board is accessed when the address (A24-A31) is either $FA of $AX and comes up in one of three modes. The mode is determined by A17, A18, and A19. Mode 1, RAMACC, (valid when A19=0 and A18=X) is used to read or write to Video RAM. Mode 2, DACACC, (valid when A19=1 and A18=0) is used to read or write data to the D/A's. Mode 3, MODACC, (valid when A19=1 and A18=1) determines the access mode of the ROM.

MODE	A19	A18	A17
Video	0	X	X
RAM	1	0	0
	1	0	1
D/As	1	1	0
	1	1	1

Table 3-1 Board Access

On power up the Bus reset signal is driven low to initialize all boards on the Bus. Reset is used by the video card to terminate any Bus cycles and disable the 66Hz interrupt. After reset the 68030 reads from the highest address of each Bus slot. By definition the declaration ROM (U35 on this card) must reside at this address. The ROM is only required to occupy eight of the 32 data bits on the bus. This puts valid ROM data at every fourth byte of address space. The video board ROM uses byte lane zero (the first byte lane, bus bits D0-D7). To determine the byte lane(s) being used, the processor reads all four bytes and looks for a valid data value in one or more of the byte lanes. For each valid value it continues reading from the specified byte lane(s) looking for a predefined sequence of numbers. If the numbers are correct, the processor will use only the specified byte lane(s) for future ROM accesses (refer to Cards and Drivers for more information concerning bus signals and byte lanes). The processor reads the ROM which contains information describing the board, and then copies the driver from the ROM into system RAM on the SE/30 motherboard. The driver open routine is then executed. This routine initializes the video buffer by writing a gray pattern (black and white dots) to the VRAM array. Then it sets the \VIDON signal TRUE which enables the 66Hz slot interrupt and the video output. The 66Hz interrupt signals the processor to update the cursor position. When the system detects a slot interrupt, control is transferred to the driver which clears the interrupt via a MODACC write, calls for a cursor position update and returns control to the system.

Information stored in the declaration ROM includes:

- byte lane #0 is used for reading the ROM
- verification code to demonstrate the existence of the ROM
- 1,2,4, or 8 bits per pixel
- 640 x 480 bits resolution
- single display page
- first pixel is located at an offset of 0 from the slot base address
- there is an offset of 400(Hex) between scan lines
- location of video driver in the ROM
- video driver
- checksum of the ROM

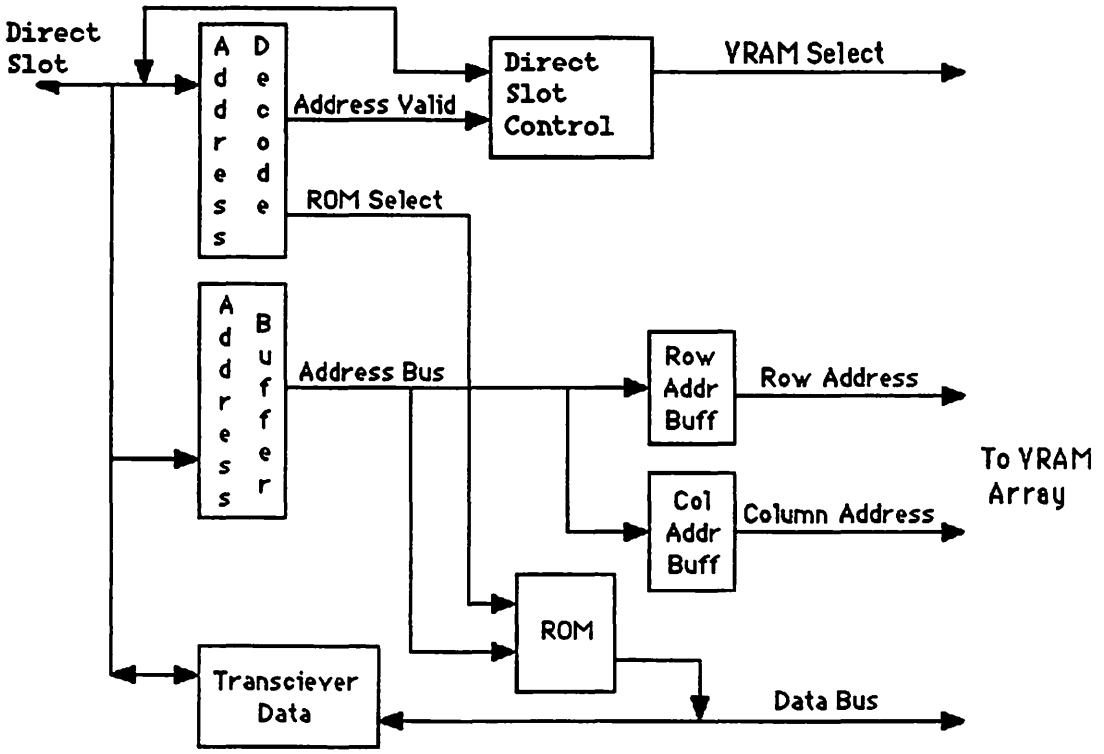


Figure 3-2 Bus Interface Block Diagram

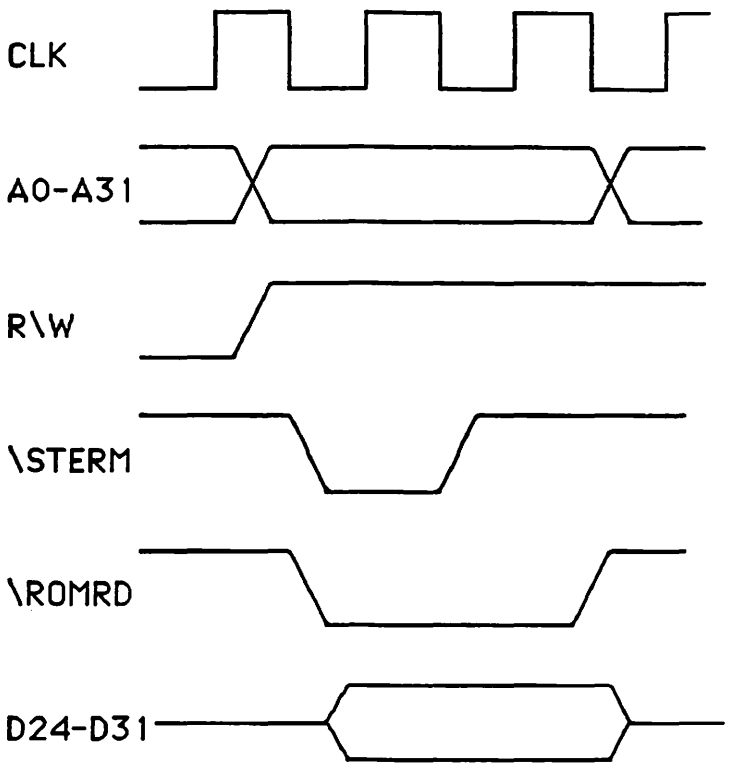


Figure 3-3 Declaration ROM Access from Bus

The address range of the ROM is FAXF8000 - FAXFFFFF. The ROM is actually only 8192 bytes long occupying 65536 bytes of address space due to being resident on only the zeroth byte lane. The ROM will appear to occupy multiple positions in this range of addresses (addresses wrap around) since only the MSB address lines are decoded to determine ROM access requests.

During normal operation the processor will write to and read from the video buffer. The accesses to the buffer are synchronized with the horizontal blanking intervals so that they do not interfere with the TRANSFER cycles which are required to update the SRAM portion of the VRAM and the video display.

The video RAMs are arranged in 2 banks, each one 32 bits wide. The video RAM consists of 2 parts, DRAM which can be updated at anytime (except during a transfer cycle) and SRAM which receives data via a TRANSFER cycle from the DRAM, which occurs for every horizontal trace. The TRANSFER cycle timing is independent of any bus timing and is generated around the horizontal synch signal. A TRANSFER cycle also generates a RAMHOLD which disables processor accesses to the RAM. Data enters the VRAMs through a 32 bit parallel path and leaves through one of four 8 bit serial paths. The serial pixel data is interleaved from the 4 latches. Data from these latches is formatted to the appropriate number of bits per pixel then output to the D/As.

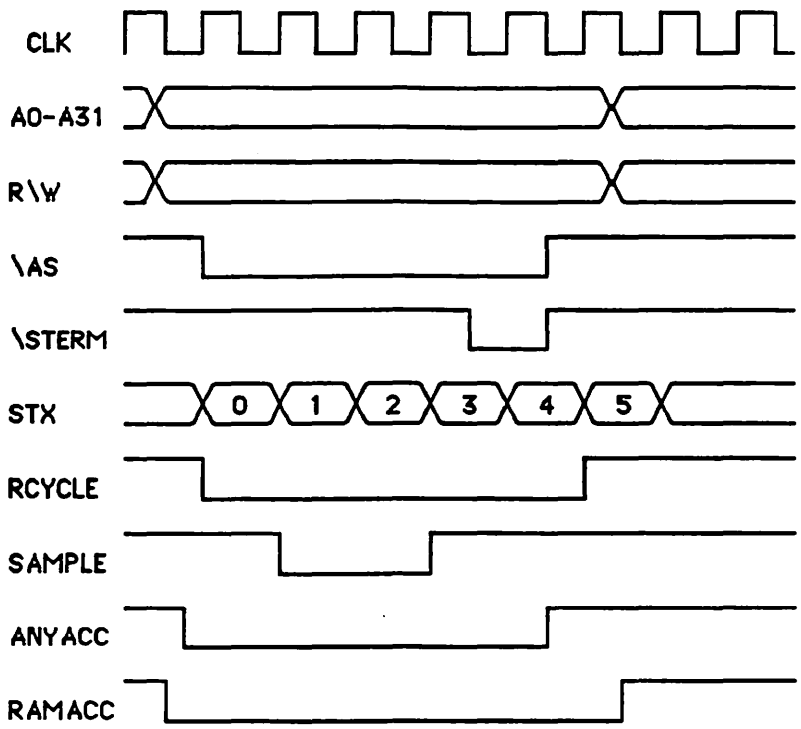


Figure 3-4 Bus Cycle

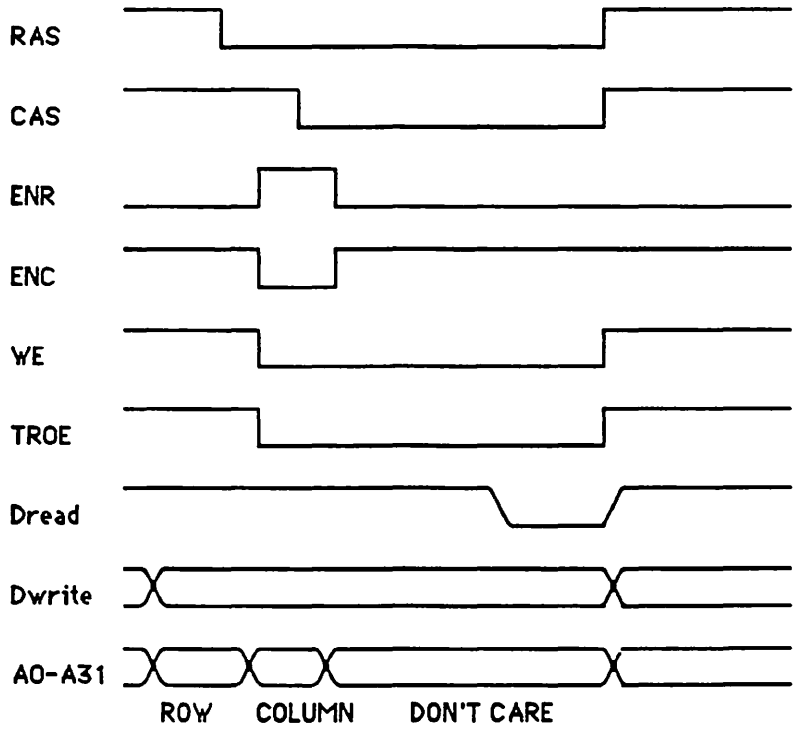


Figure 3-5 RAM Cycle

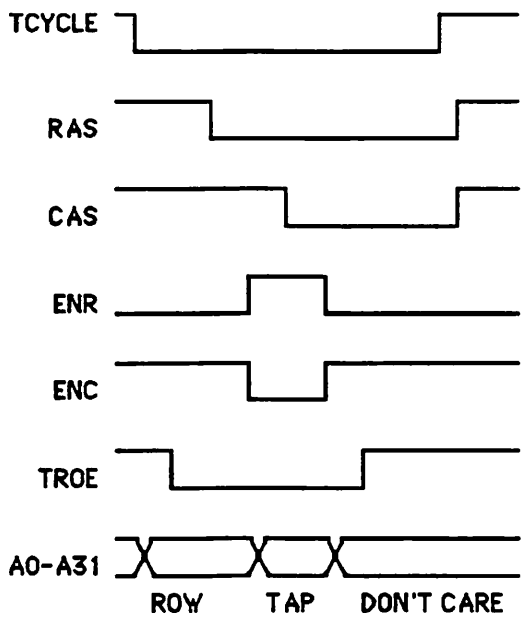


Figure 3-6 Transfer Cycle

The RAM occupies an address range of A00000 - Fs1FFFFF. However only the 640 x 480 pixel display rectangle is valid. This rectangle begins at A00000 (pixel #0 on line #1) and ends at A77E80 (pixel #639 on line #480). New lines start at multiples of 400H and end after 640 bytes (640 pixels). There will be RAM present throughout this entire range of addresses, however a partials program may be implemented that would allow RAMs with bad cells outside the video rectangle to be used. With this in mind, only the video rectangle should be tested in a RAM test.

Before shutting down the computer the operating system will execute a close routine also found in the video driver portion of the declaration ROM. This routine disables the slot interrupts which also disables the video output, and then returns control to the operating system.

To enable the slot interrupt and the video output, 4XXXXXX is written to any address in the range FAXE0000-FAXFFFFF. To disable the interrupt and the video output, 0 is written to any address in the range FAXE0000-FAXFFFFF. Enabling or disabling the interrupt will also clear the interrupt.

3.2 Video Timing

The video timing portion of the board generates the horizontal and vertical timing signals for the display. The horizontal and vertical sections are similar in design, using counters and PALs. On power up the output count of the counters is unknown, but they will count up to a point where they will cause the PAL to reload the counters to a known value. The horizontal counter is used to address pixels on a line (RAM column address). Counts 224-862 represent visible pixels. When the count reaches 863 the PAL drives the

\HBLNK (horizontal blank) signal low. This blanks the video. Several counts later the \SYNC (sync) signal goes low triggering the horizontal retrace of the monitor. The \HBLNK signal stays low during and after the \SYNC signal and goes high when the count reaches 224 for the first pixel of the current scan line. At the same time that the \HBLNK signal goes low a signal called \HSET also goes low causing the horizontal counters to reload. The \HSET signal also clocks the vertical counters and PAL. The clock frequency and the counter preset values determine the line frequency of the video.

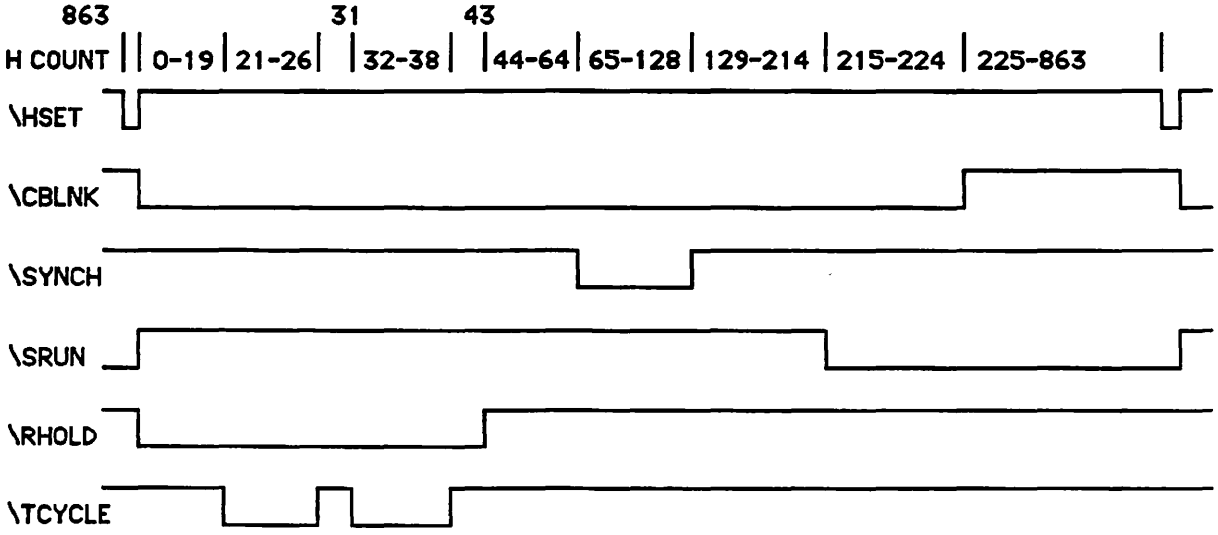


Figure 3-7 Horizontal Timing Signals

The vertical timing signals are generated in the same way. The vertical counters continually count up until reaching a count which causes the PAL to output a low level on the \VSET signal which reloads the counters. The vertical count represents the display line number. Lines 0-479 are visible lines. When the count reaches 480 the PAL outputs a low level on the \VBLNK (vertical blank) signal which is ORed into the \CBLNK signal forcing it low and causing the video to be blanked. Three line counts later the \VSYNC (vertical sync) signal is driven low and ORed into the \SYNC signal triggering the monitor to begin the vertical retrace. The \VSYNC signal is low for three lines. The total \VBLNK interval is approximately one millisecond. Also within the \VBLNK period the \VSET signal is driven low causing the vertical counters to reload. When the count again reaches 0, the \VBLNK signal will go high for the beginning of the next video frame. The frame rate is determined by the line rate and the preset value of the vertical counters. The frame rate is 66Hz.

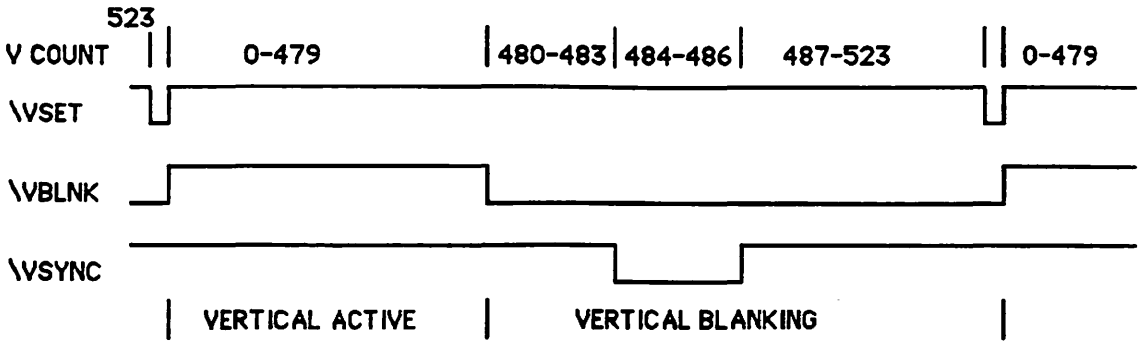


Figure 3-8 Vertical Timing Signals

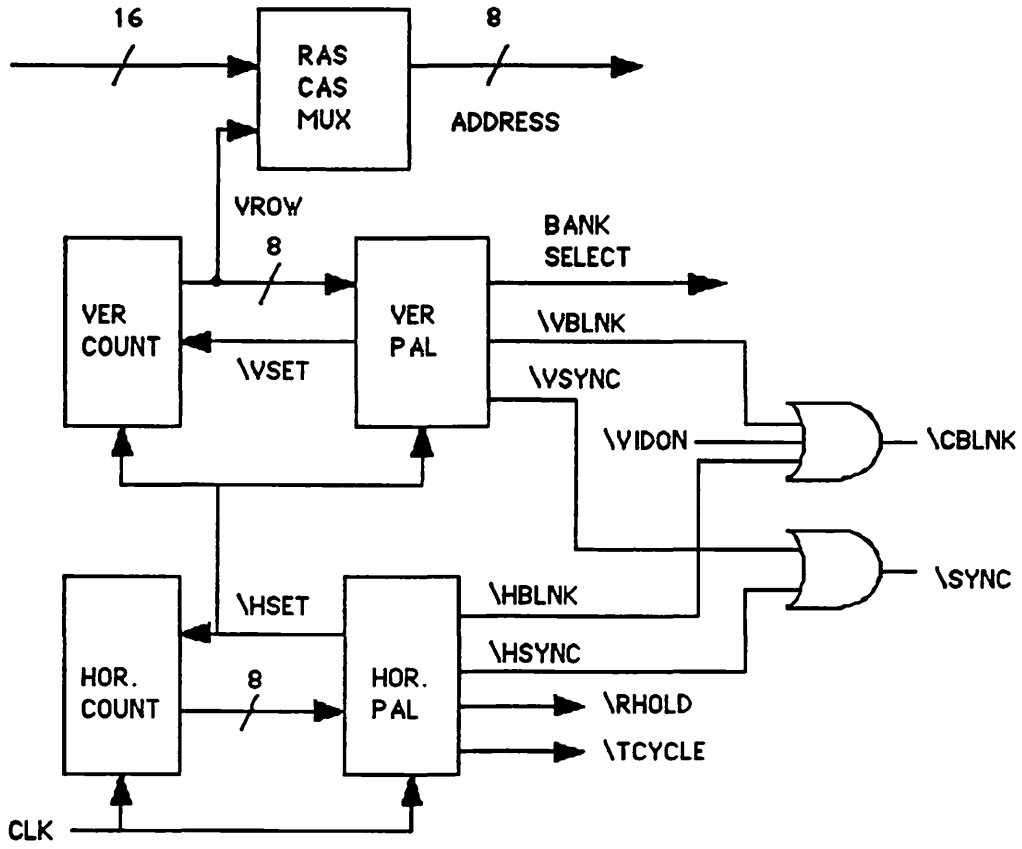


Figure 3-9 Timing Generation Block Diagram

Other signals from the horizontal timing PAL are used to prevent the processor access to the video buffer during a TRANSFER cycle.

3.3 RGB Video Output

To generate the actual analog video signals the RAM buffer is read out. There are four banks of RAM. Each bank is 1 out of every 4 pixels to the screen. In 8-bit mode the banks are read in an interleaved manner at up to 140 nanosecond cycle time for each bank which provides a pixel time of 35

nanoseconds. The data from the RAMs is converted to the proper format via U22, then sent into the LDAC (digital to analog) converter. The four VRAMs in each bank allow for 8 bits of information to be stored for each pixel. This allows 256 shades of red, green, blue, or some combination to be displayed; there is a possibility of over 16.7 million different colors. The video D/A converters also receive the \CBLNK and \SYNC signals. When both the blanking and sync signals are high the D/A's sample the digital data on the rising edge of the clock, and convert the digital RAM data into analog voltages. A data value of white will result in a voltage out of the D/A of approximately one volt which represents the brightest video level. Conversely a value of black will result in a voltage level of 0.3 volts. The \CBLNK signal also causes a black level to be output, and the \SYNC signal causes a voltage of zero volts to be output. The three video signals, one each representing red, green and blue, are separately transmitted to the video monitor. The \SYNC signal is also sent to the monitor independently at TTL voltage levels.

The R-data, as shown in Figure 3-10 Video Generation Block Diagram, is valid as follows: when LC0 is low, LD0-LD7 is valid; when LC1 is low, LD8-LD15 is valid; when LC2 is low, LD16-LD23 is valid; when LC 3 is low, LD24-LD31 is valid. When \LOAD is low, P-data is valid and when \LOAD is high, R-data is being shifted (reformatted) into a valid configuration.

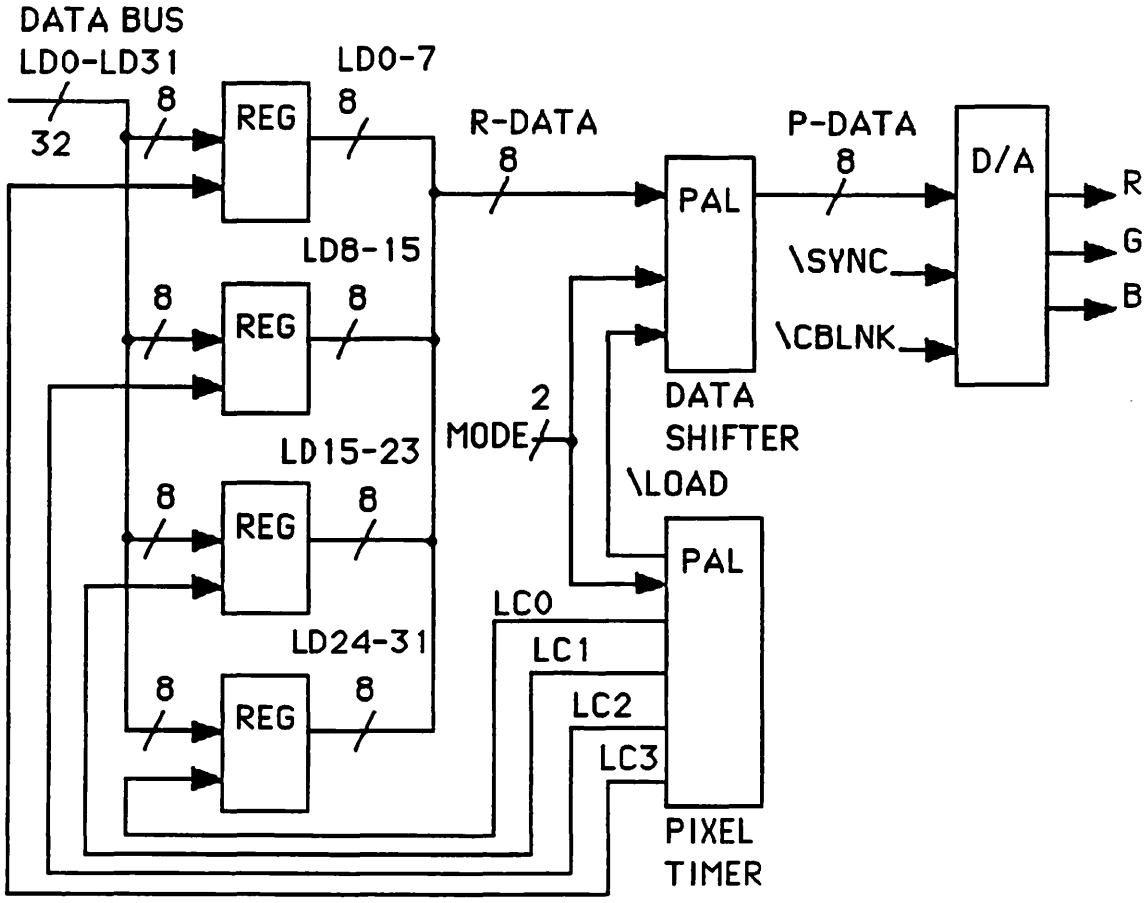


Figure 3-10 Video Generation Block Diagram

The following diagram illustrates the operation of a video D/A. In practice the blanking and sync intervals will be longer.

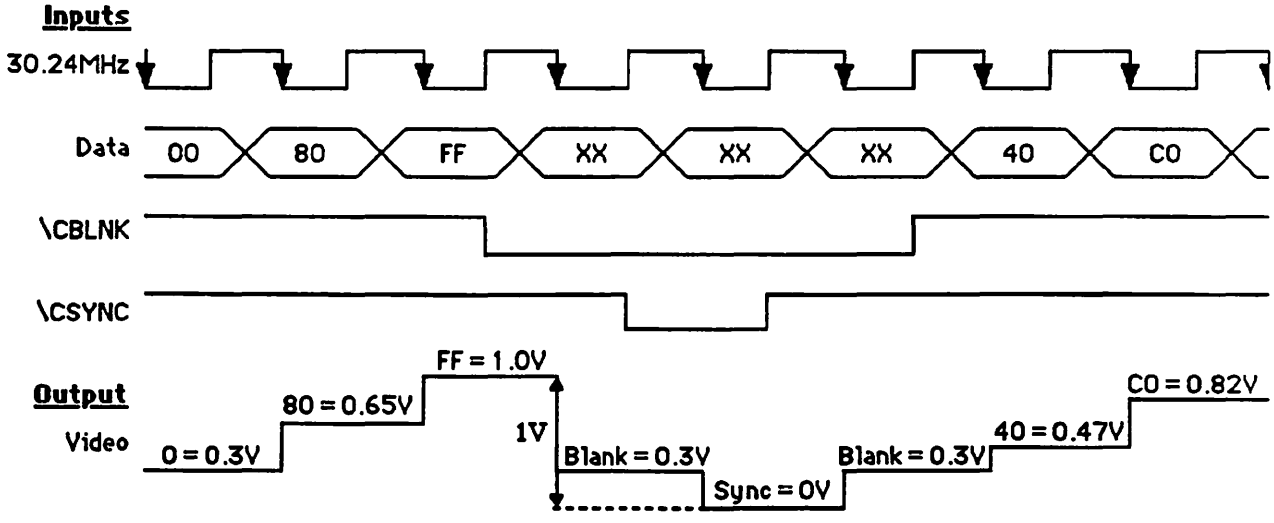


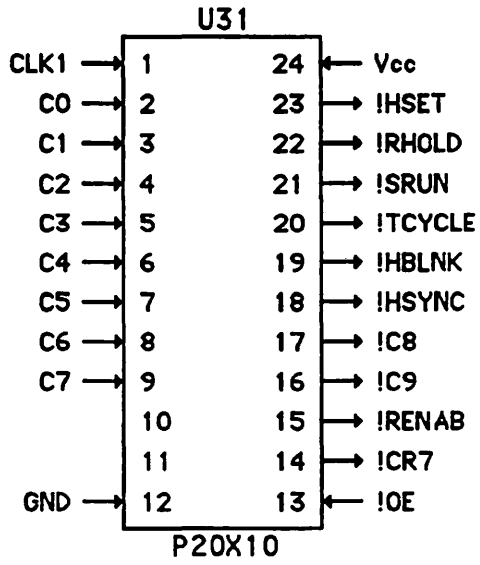
Figure 3-11 Video D/A Operation

Chapter 4
PAL Description

The Micron Xceed SE/306-48 contains eleven PALs. The boolean logic symbols are shown below:

&	AND	$	EXCLUSIVE OR
#	OR	!	NOT or logical INVERSION

```
Name        U31;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U31;
Device      P20X10;
Format      J;
```



/** Inputs **/

```
Pin  1      = CLK1;
Pin  2      = C0;
Pin  3      = C1;
Pin  4      = C2;
Pin  5      = C3;
Pin  6      = C4;
Pin  7      = C5;
Pin  8      = C6;
Pin  9      = C7;
Pin  13     = !OE;
```

/** Outputs **/

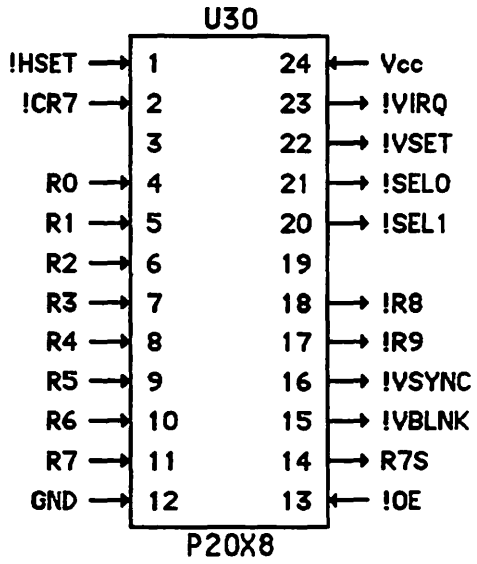
```
Pin  14     = !CR7;
Pin  15     = !RENAB;
Pin  16     = !C9;
Pin  17     = !C8;
Pin  18     = !HSYNC;
Pin  19     = !HBLNK;
Pin  20     = !TCYCLE;
Pin  21     = !SRUN;
Pin  22     = !RHOLD;
Pin  23     = !HSET;
```

/** Declarations and Intermediate Variable Definitions **/

```
COUNT14  = !C9 & !C8 & !C7 & !C6 & !C5 & !C4 & C3 & C2 & C1 & !C0;

COUNT16M  = !C9 & !C8 & !C7 & !C6 & !C5 & C4 & !C3;

COUNT32M  = !C9 & !C8 & !C7 & !C6 & C5 & !C4 & !C3;

COUNT40  = !C9 & !C8 & !C7 & !C6 & C5 & !C4 & C3 & !C2 & !C1 & !C0;

COUNT45  = !C9 & !C8 & !C7 & !C6 & C5 & !C4 & C3 & C2 & !C1 & C0;
```

```
COUNT61  = !C9 & !C8 & !C7 & !C6 & C5 & C4 & C3 & C2 & !C1 & C0;

COUNT125  = !C9 & !C8 & !C7 & C6 & C5 & C4 & C3 & C2 & !C1 & C0;

COUNT212  = !C9 & !C8 & C7 & C6 & !C5 & C4 & !C3 & C2 & !C1 & !C0;

COUNT221  = !C9 & !C8 & C7 & C6 & !C5 & C4 & C3 & C2 & !C1 & C0;

COUNT255M  = C7 & C6 & C5 & C4 & C3 & C2 & C1 & C0;

COUNT511  = !C9 & C8 & C7 & C6 & C5 & C4 & C3 & C2 & C1 & C0;

COUNT767  = C9 & !C8 & C7 & C6 & C5 & C4 & C3 & C2 & C1 & C0;

COUNT861  = C9 & C8 & !C7 & C6 & !C5 & C4 & C3 & C2 & !C1 & C0;

COUNT862  = C9 & C8 & !C7 & C6 & !C5 & C4 & C3 & C2 & C1 & !C0;

COUNT863  = C9 & C8 & !C7 & C6 & !C5 & C4 & C3 & C2 & C1 & C0;

/** Logic Equations **/

CR7.D  = COUNT16M;

RENAB.D  = COUNT40 # RENAB $ COUNT14;

SRUN.D  = COUNT212 # SRUN $ COUNT863;

TCYCLE.D  = COUNT16M # COUNT32M;

C9.D  = COUNT511 # C9 $ COUNT863;

C8.D  = COUNT255M # C8 $ COUNT511 # COUNT863;

HBLNK.D  = COUNT861 # HBLNK $ COUNT221;

HSYNC.D  = COUNT61 # HSYNC $ COUNT125;

RHOLD.D  = COUNT863 # RHOLD $ COUNT45;

HSET.D  = COUNT862;
```

```
Name        U30;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U30;
Device      P20X8;
Format      J;
```



/** Inputs **/

```
Pin  1      = !HSET;
Pin  2      = !CR7;
Pin  4      = R0;
Pin  5      = R1;
Pin  6      = R2;
Pin  7      = R3;
Pin  8      = R4;
Pin  9      = R5;
Pin  10     = R6;
Pin  11     = R7;
Pin  13     = !OE;
```

/** Outputs **/

```
Pin  14     = R7S;
Pin  15     = !VBLNK;
Pin  16     = !VSYNC;
Pin  17     = !R9;
Pin  18     = !R8;
Pin  20     = !SEL1;
Pin  21     = !SEL0;
Pin  22     = !VSET;
Pin  23     = !VIRQ;
```

/** Declarations and Intermediate Variable Definitions **/

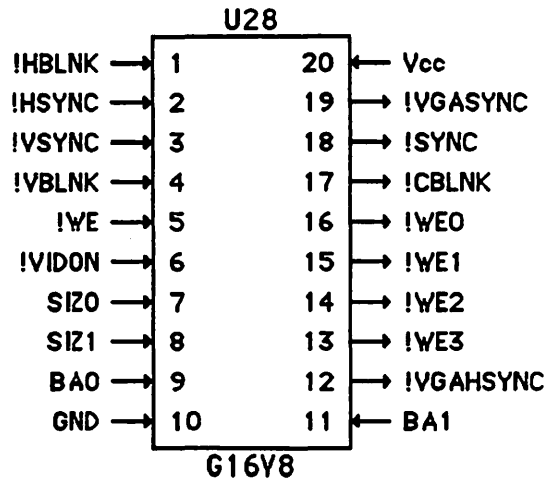
```
COUNT255  = !R9 & !R8 & R7 & R6 & R5 & R4 & R3 & R2 & R1 & R0;

COUNT479  = !R9 & R8 & R7 & R6 & !R5 & R4 & R3 & R2 & R1 & R0;

COUNT482  = !R9 & R8 & R7 & R6 & R5 & !R4 & !R3 & !R2 & R1 & !R0;

COUNT485  = !R9 & R8 & R7 & R6 & R5 & !R4 & !R3 & R2 & !R1 & R0;

COUNT511  = !R9 & R8 & R7 & R6 & R5 & R4 & R3 & R2 & R1 & R0;

COUNT523  = R9 & !R8 & !R7 & !R6 & !R5 & !R4 & R3 & !R2 & R1 & R0;

COUNT524  = R9 & !R8 & !R7 & !R6 & !R5 & !R4 & R3 & R2 & !R1 & !R0;

COUNT768  = R9 & R8 & !R7 & !R6 & !R5 & !R4 & !R3 & !R2 & !R1 & !R0;
```

/** Logic Equations **/

```
R7S  = R7 & !CR7 # !R7 & CR7;

VBLNK.D  = COUNT479 # VBLNK $ COUNT524;
```

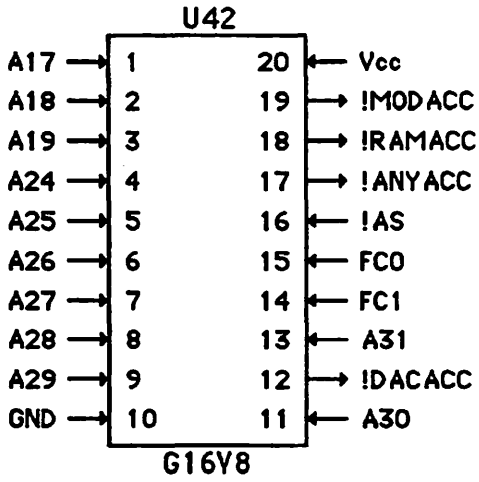
```
VSYNC.D  = COUNT482 # VSYNC $ COUNT485;

R9.D  = COUNT511 # R9 $ COUNT524 # COUNT768;

R8.D  = COUNT255 # R8 $ COUNT511 # COUNT768;

SEL1.D  = COUNT255 # SEL1 $ COUNT479;

SEL0.D  = COUNT524 # SEL0 $ COUNT255;

VSET.D  = COUNT523;

VIRQ  = COUNT479;
```

```
Name        U28;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U28;
Device      G16V8;
Format      J;
```




```
/** Inputs **/

Pin  1      = !HBLNK;
Pin  2      = !HSYNC;
Pin  3      = !VSYNC;
Pin  4      = !VBLNK;
Pin  5      = !WE;
Pin  6      = !VIDON;
Pin  7      = SIZ0;
Pin  8      = SIZ1;
Pin  9      = BA0;
Pin  11     = BA1;

/** Outputs **/

Pin  12     = !VGAHSYNC;
Pin  13     = !WE3;
Pin  14     = !WE2;
Pin  15     = !WE1;
Pin  16     = !WE0;
Pin  17     = !CBLNK;
Pin  18     = !SYNC;
Pin  19     = !VGAVSYNC;

/** Logic Equations **/

WE3  = WE & !BA0 & !BA1;

WE2  = WE & BA0 & ! BA1 # WE & !BA1 & !SIZ0 # WE & !BA1 & SIZ1;

WE1  = WE & !BA0 & BA1 # WE & !BA1 & !SIZ0 & !SIZ1 #
       WE & !BA1 & SIZ0 & SIZ1 # WE & !BA1 & BA0 & !SIZ0;

WE0  = WE & BA0 & BA1 # WE & BA0 & SIZ0 & SIZ1 #
       WE & !SIZ0 & !SIZ1 # WE & BA1 & SIZ1;

CBLNK  = !VIDON # VBLNK # HBLNK;

SYNC  = VSYNC # HSYNC;

VGAHSYNC  = HSYNC;

VGAVSYNC  = VSYNC;
```

```
Name        U42;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U42;
Device      G16V8;
Format      J;
```



/** Inputs **/

```
Pin  1      = A17;
Pin  2      = A18;
Pin  3      = A19;
Pin  4      = A24;
Pin  5      = A25;
Pin  6      = A26;
Pin  7      = A27;
Pin  8      = A28;
Pin  9      = A29;
Pin  11     = A30;
Pin  13     = A31;
Pin  14     = FC1;
Pin  15     = FC0;
Pin  16     = !AS;
```

/** Outputs **/

```
Pin  12     = !DACACC;
Pin  17     = !ANYACC;
Pin  18     = !RAMACC;
Pin  19     = !MODACC;
```

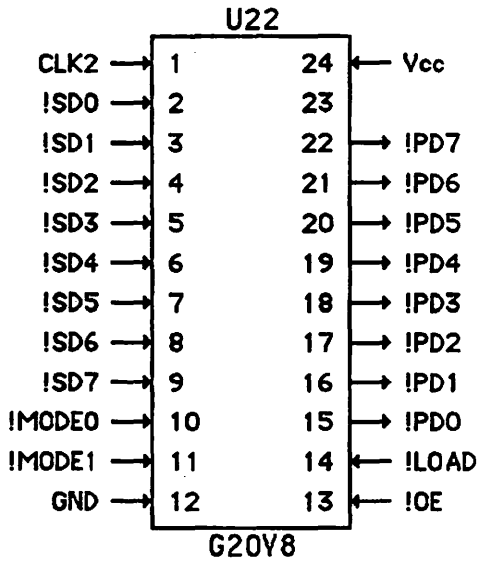
/** Declarations and Intermediate Variable Definitions **/

```
ARAM  = !A19 # !A18 & A19;

AMOD  = A17 & A18 & A19;

ADAC  = !A17 & A18 & A19;

AFA  = !A24 & A25 & !A26 & A27 & A28 & A29 & A30 & A31 & AS;

AA  = !A28 & A29 & !A30 & A31 & AS;
```

/** Logic Equations **/

```
RAMACC  = ARAM & AFA # ARAM & AA;

MODACC  = AMOD & AFA # AMOD & AA;

DACACC  = ADAC & AFA # ADAC & AA;

ANYACC  = AFA & !FC0 # AFA & !FC1 # AA & !FC0 #
          AA & !FC1 # ANYACC & AS;
```

```
Name     U22;
Partno   XXXXX;
Date     7/27/89;
Revision B;
Designer C. Snodgrass;
Company  Micron;
Assembly ;
Location U22;
Device   G20V8;
Format   J;
```



/** Inputs **/

```
Pin  1      = CLK2;
Pin  2      = !SD0;
Pin  3      = !SD1;
Pin  4      = !SD2;
Pin  5      = !SD3;
Pin  6      = !SD4;
Pin  7      = !SD5;
Pin  8      = !SD6;
Pin  9      = !SD7;
Pin  10     = !MODE0;
Pin  11     = !MODE1;
Pin  13     = !OE;
Pin  14     = !LOAD;
```

/** Outputs **/

```
Pin  15     = !PD0;
Pin  16     = !PD1;
Pin  17     = !PD2;
Pin  18     = !PD3;
Pin  19     = !PD4;
Pin  20     = !PD5;
Pin  21     = !PD6;
Pin  22     = !PD7;
```

/** Declarations and Intermediate Variable Definitions **/

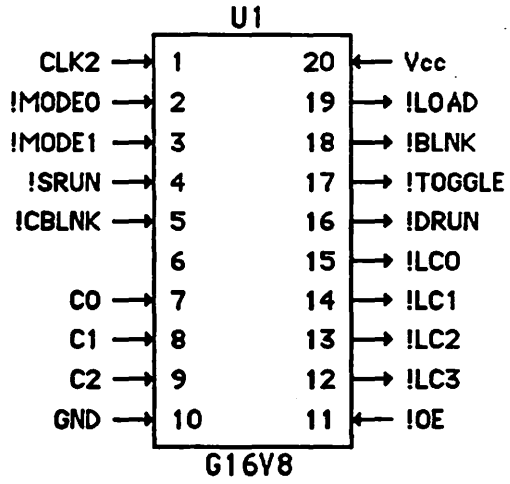
```
S4BIT  = !LOAD & !MODE0 & MODE1;

S2BIT  = !LOAD & MODE0 & !MODE1;

S1BIT  = !LOAD & !MODE0 & !MODE1;

L8BIT  = LOAD & MODE0 & MODE1;

L4BIT  = LOAD & !MODE0 & MODE1;

L2BIT  = LOAD & MODE0 & !MODE1;

L1BIT  = LOAD & !MODE0 & !MODE1;
```

/** Logic Equations **/

```
PD7.D  = SD7 & L8BIT # SD3 & L4BIT # SD1 & L2BIT # SD0 & L1BIT;

PD6.D  = SD6 & L8BIT # SD2 & L4BIT # SD0 & L2BIT # SD1 & L1BIT #
         PD7 & S1BIT;
```

```
PD5.D  = SD5 & L8BIT # SD1 & L4BIT # SD3 & L2BIT # SD2 & L1BIT #
         PD7 & S2BIT # PD6 & S1BIT;

PD4.D  = SD4 & L8BIT # SD0 & L4BIT # SD2 & L2BIT # SD3 & L1BIT #
         PD6 & S2BIT # PD5 & S1BIT;

PD3.D  = SD3 & L8BIT # SD7 & L4BIT # SD5 & L2BIT # SD4 & L1BIT #
         PD7 & S4BIT # PD5 & S2BIT # PD4 & S1BIT;

PD2.D  = SD2 & L8BIT # SD6 & L4BIT # SD4 & L2BIT # SD5 & L1BIT #
         PD6 & S4BIT # PD4 & S2BIT # PD3 & S1BIT;

PD1.D  = SD1 & L8BIT # SD5 & L4BIT # SD7 & L2BIT # SD6 & L1BIT #
         PD5 & S4BIT # PD3 & S2BIT # PD2 & S1BIT;

PD0.D  = SD0 & L8BIT # SD4 & L4BIT # SD6 & L2BIT # SD7 & L1BIT #
         PD4 & S4BIT # PD2 & S2BIT # PD1 & S1BIT;
```

```
Name      U1;
Partno    XXXXX;
Date      7/27/89;
Revision  B;
Designer  C. Snodgrass;
Company   Micron;
Assembly  ;
Location  U1;
Device    G16V8;
Format    J;
```



/** Inputs **/

```
Pin  1      = CLK2;
Pin  2      = !MODE0;
Pin  3      = !MODE1;
Pin  4      = !SRUN;
Pin  5      = !CBLNK;
Pin  7      = C0;
Pin  8      = C1;
Pin  9      = C2;
Pin  11     = !OE;
```

/** Outputs **/

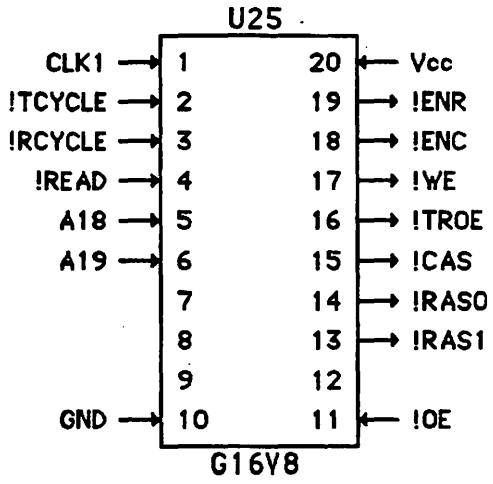
```
Pin  12     = !LC3;
Pin  13     = !LC2;
Pin  14     = !LC1;
Pin  15     = !LC0;
Pin  16     = !DRUN;
Pin  17     = !TOGGLE;
Pin  18     = !BLNK;
Pin  19     = !LOAD;
```

/** Declarations and Intermediate Variable Definitions **/

```
CYCLE0  = C2 & C1 & !C0;

CYCLE1  = C2 & C1 & C0;

CYCLE2  = !C2 & !C1 & !C0;

CYCLE3  = !C2 & !C1 & C0;

CYCLE4  = !C2 & C1 & !C0;

CYCLE5  = !C2 & C1 & C0;

CYCLE6  = C2 & !C1 & !C0;

CYCLE7  = C2 & !C1 & C0;

M8BIT   = MODE0 & MODE1;

M4BIT   = !MODE0 & MODE1;

M2BIT   = MODE0 & !MODE1;

M1BIT   = !MODE0 & !MODE1;
```

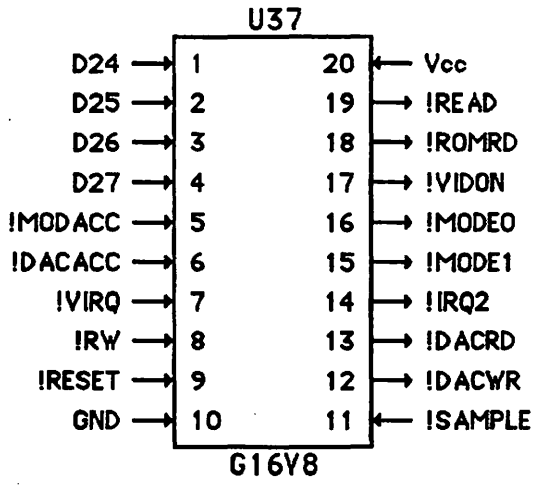
```
/** Logic Equations **/

LC0.D  = SRUN & LC0 & !TOGGLE & DRUN # SRUN & LC1 & TOGGLE #
         SRUN & !DRUN & CYCLE2 # SRUN & !DRUN & CYCLE6;

LC1.D  = SRUN & LC1 & !TOGGLE & DRUN # SRUN & LC2 & TOGGLE #
         SRUN & !DRUN & CYCLE1 # SRUN & !DRUN & CYCLE5;

LC2.D  = SRUN & LC2 & !TOGGLE & DRUN # SRUN & LC3 & TOGGLE #
         SRUN & !DRUN & CYCLE0 # SRUN & !DRUN & CYCLE4;

LC3.D  = SRUN & LC3 & !TOGGLE & DRUN # SRUN & LC0 & TOGGLE #
         SRUN & !DRUN & CYCLE7 # SRUN & !DRUN & CYCLE3;

DRUN.D  = SRUN & CYCLE6 # DRUN & SRUN;

TOGGLE.D  = DRUN & M1BIT & CYCLE7 # DRUN & M2BIT & CYCLE3 #
            DRUN & M2BIT & CYCLE7 # DRUN & M4BIT & CYCLE1 #
            DRUN & M4BIT & CYCLE3 # DRUN & M4BIT & CYCLE5 #
            DRUN & M4BIT & CYCLE7 # DRUN & M8BIT;

BLNK.D  = CBLNK;

LOAD.D  = DRUN & M1BIT & CYCLE0 # DRUN & M2BIT & CYCLE0 #
          DRUN & M2BIT & CYCLE4 # DRUN & M4BIT & CYCLE0 #
          DRUN & M4BIT & CYCLE2 # DRUN & M4BIT & CYCLE4 #
          DRUN & M4BIT & CYCLE6 # DRUN & M8BIT;
```

```
Name        U25;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U25;
Device      G16V8;
Format      J;
```




```
/** Inputs **/

Pin   1        = CLK1;
Pin   2        = !TCYCLE;
Pin   3        = !RCYCLE;
Pin   4        = !READ;
Pin   5        = A18;
Pin   6        = A19;
Pin   11       = !OE;

/** Outputs **/

Pin   13       = !RAS1;
Pin   14       = !RAS0;
Pin   15       = !CAS;
Pin   16       = !TROE;
Pin   17       = !WE;
Pin   18       = !ENC;
Pin   19       = !ENR;

/** Logic Equations **/

TROE.D  = RCYCLE & RAS0 & READ # RCYCLE & RAS1 & READ #
          TCYCLE & !CAS # TCYCLE & ENC;

CAS.D   = ENC # CAS & RCYCLE # CAS & TCYCLE;

RAS0.D  = RCYCLE & !A18 # TCYCLE & TROE # RAS0 & TCYCLE;

RAS1.D  = RCYCLE & A18 # TCYCLE & TROE # RAS1 & TCYCLE;

WE.D    = !READ & RCYCLE & RAS0 # !READ & RCYCLE & RAS1;

ENC.D   = RAS0 & !CAS & RCYCLE # RAS0 & !CAS & TCYCLE #
          RAS1 & !CAS & RCYCLE # RAS1 & !CAS & TCYCLE;

ENR.D   = !RAS0 & !RAS1 # CAS;
```

```
Name        U37;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U37;
Device      G16V8;
Format      J;
```



/** Inputs **/

```
Pin  1      = D24;
Pin  2      = D25;
Pin  3      = D26;
Pin  4      = D27;
Pin  5      = !MODACC;
Pin  6      = !DACACC;
Pin  7      = !VIRQ;
Pin  8      = !RW;
Pin  9      = !RESET;
Pin  11     = !SAMPLE;
```

/** Outputs **/

```
Pin  12     = !DACWR;
Pin  13     = !DACRD;
Pin  14     = !IRQ2;
Pin  15     = !MODE1;
Pin  16     = !MODE0;
Pin  17     = !VIDON;
Pin  18     = !ROMRD;
Pin  19     = !READ;
```

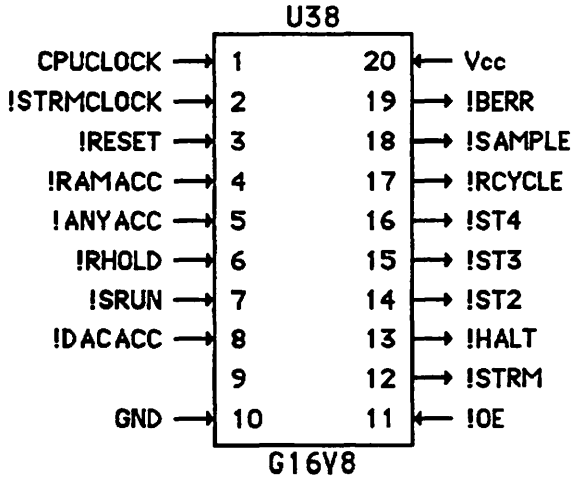
/** Declarations and Intermediate Variable Definitions **/

```
MODREAD  = !D27 & MODACC & SAMPLE & RW & !RESET;

MODKEEP  = D27 & MODACC & SAMPLE & RW & !RESET #
           MODACC & !SAMPLE & RW & !RESET #
           !MODACC & !RESET # MODACC & !RW & !RESET;
```

/** Logic Equations **/

```
IRQ2  = VIDON & VIRQ & !RESET # IRQ2 & VIDON & !MODACC & !RESET #
        IRQ2 & VIDON & MODACC & !RW & !RESET;

MODE0  = D24 & MODREAD # MODE0 & MODKEEP;

MODE1  = D25 & MODREAD # MODE1 & MODKEEP;

DACWR  = DACACC & SAMPLE & RW;

DACRD  = DACACC & SAMPLE & !RW # DACRD & DACACC;

VIDON  = D26 & MODREAD & !RESET # VIDON & MODKEEP & !RESET;

ROMRD  = MODACC & !RW;

READ  = !RW;
```

```
Name        U38;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U38;
Device      G16V8;
Format      J;
```



/** Inputs **/

```
Pin  1      = CPUCLOCK;
Pin  2      = !STRMCLOCK;
Pin  3      = !RESET;
Pin  4      = !RAMACC;
Pin  5      = !ANYACC;
Pin  6      = !RHOLD;
Pin  7      = !SRUN;
Pin  8      = !DACACC;
Pin  11     = !OE;
```

/** Outputs **/

```
Pin  12     = !STRM;
Pin  13     = !HALT;
Pin  14     = !ST2;
Pin  15     = !ST3;
Pin  16     = !ST4;
Pin  17     = !RCYCLE;
Pin  18     = !SAMPLE;
Pin  19     = !BERR;
```

/** Declarations and Intermediate Variable Definitions **/

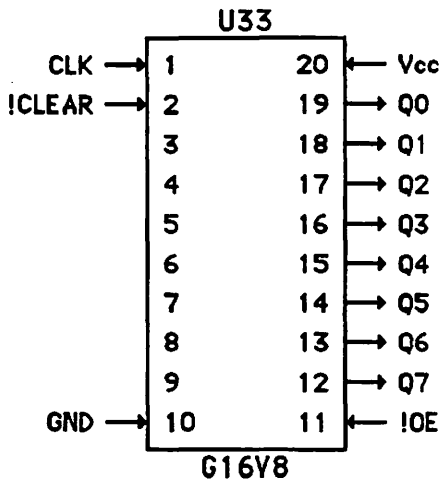
```
ACCESS  = ANYACC & !RESET;

COUNT0  = !ST4 & !ST3 & !ST2;

COUNT1  = !ST4 & !ST3 & ST2;

COUNT2  = !ST4 & ST3 & ST2;

COUNT3  = !ST4 & ST3 & !ST2;

COUNT4  = ST4 & ST3 & !ST2;

COUNT5  = ST4 & ST3 & ST2;

COUNT6  = ST4 & !ST3 & ST2;
```

/** Logic Equations **/

```
HALT  = COUNT4 & !SAMPLE & STRMCLOCK # COUNT4 & HALT #
        COUNT5 & HALT & !STRMCLOCK;

HALT.OE  = ACCESS;

BERR  = COUNT4 & !SAMPLE & STRMCLOCK # COUNT4 & BERR #
        COUNT5 & !STRMCLOCK;

BERR.OE  = ACCESS;
```

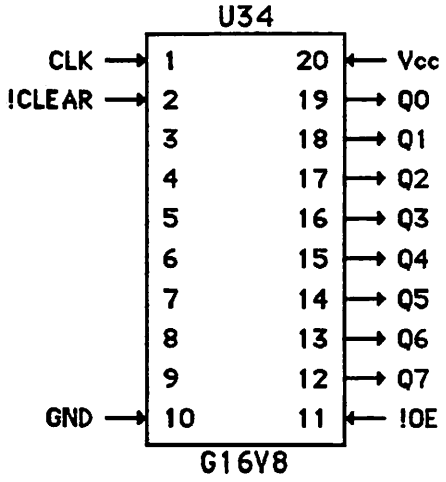
```
STRM  = COUNT4 & STRMCLOCK # COUNT4 & STRM # COUNT5 & !STRMCLOCK;

STRM.OE  = ACCESS;

ST2.D  = !RHOLD & ACCESS & COUNT0 # COUNT1 # COUNT4 # COUNT6;

ST3.D  = COUNT1 # COUNT2 # COUNT3 # COUNT4;

ST4.D  = COUNT3 # COUNT4 # COUNT5;

RCYCLE.D  = RAMACC & COUNT1 & !RESET # RCYCLE & COUNT2 & !RESET #
            RCYCLE & COUNT3 & !RESET # RCYCLE & COUNT4 & !RESET;

SAMPLE.D  = COUNT2 & !SRUN & DACACC # COUNT2 & !DACACC #
            COUNT3 & SAMPLE;
```

```
Name        U33;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U33;
Device      G16V8;
Format      J;
```




```
/** Inputs **/

Pin  1      = CLK;
Pin  2      = !CLEAR;
Pin  11     = !OE;

/** Outputs **/

Pin  12     = Q7;
Pin  13     = Q6;
Pin  14     = Q5;
Pin  15     = Q4;
Pin  16     = Q3;
Pin  17     = Q2;
Pin  18     = Q1;
Pin  19     = Q0;

/** Logic Equations **/

Q0.D  = !Q0 & !CLEAR;

Q1.D  = !Q1 & Q0 & !CLEAR # Q1 & !Q0 & !CLEAR;

Q2.D  = !Q2 & Q1 & Q0 & !CLEAR #
        Q2 & !Q1 & !CLEAR # Q2 & !Q0 & !CLEAR;

Q3.D  = !Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q3 & !Q2 & !CLEAR # Q3 & !Q1 & !CLEAR # Q3 & !Q0 & !CLEAR;

Q4.D  = !Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q4 & !Q3 & !CLEAR # Q4 & !Q2 & !CLEAR # Q4 & !Q1 & !CLEAR #
        Q4 & !Q0 & !CLEAR;

Q5.D  = !Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q5 & !Q4 & !CLEAR # Q5 & !Q3 & !CLEAR # Q5 & !Q2 & !CLEAR #
        Q5 & !Q1 & !CLEAR # Q5 & !Q0 & !CLEAR;

Q6.D  = !Q6 & Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q6 & !Q5 & !CLEAR # Q6 & !Q4 & !CLEAR # Q6 & !Q3 & !CLEAR #
        Q6 & !Q2 & !CLEAR # Q6 & !Q1 & !CLEAR # Q6 & !Q0 & !CLEAR;

Q7.D  = !Q7 & Q6 & Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q7 & !Q6 & !CLEAR # Q7 & !Q5 & !CLEAR # Q7 & !Q4 & !CLEAR #
        Q7 & !Q3 & !CLEAR # Q7 & !Q2 & !CLEAR # Q7 & !Q1 & !CLEAR #
        Q7 & !Q0 & !CLEAR;
```

```
Name        U34;
Partno      XXXXX;
Date        7/27/89;
Revision    B;
Designer    C. Snodgrass;
Company     Micron;
Assembly    ;
Location    U34;
Device      G16V8;
Format      J;
```




```
/** Inputs **/

Pin  1      = CLK;
Pin  2      = !CLEAR;
Pin  11     = !OE;

/** Outputs **/

Pin  12     = Q7;
Pin  13     = Q6;
Pin  14     = Q5;
Pin  15     = Q4;
Pin  16     = Q3;
Pin  17     = Q2;
Pin  18     = Q1;
Pin  19     = Q0;

/** Logic Equations **/

Q0.D  = !Q0 & !CLEAR;

Q1.D  = !Q1 & Q0 & !CLEAR # Q1 & !Q0 & !CLEAR;

Q2.D  = !Q2 & Q1 & Q0 & !CLEAR #
        Q2 & !Q1 & !CLEAR # Q2 & !Q0 & !CLEAR;

Q3.D  = !Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q3 & !Q2 & !CLEAR # Q3 & !Q1 & !CLEAR # Q3 & !Q0 & !CLEAR;

Q4.D  = !Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q4 & !Q3 & !CLEAR # Q4 & !Q2 & !CLEAR # Q4 & !Q1 & !CLEAR #
        Q4 & !Q0 & !CLEAR;

Q5.D  = !Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q5 & !Q4 & !CLEAR # Q5 & !Q3 & !CLEAR # Q5 & !Q2 & !CLEAR #
        Q5 & !Q1 & !CLEAR # Q5 & !Q0 & !CLEAR;

Q6.D  = !Q6 & Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q6 & !Q5 & !CLEAR # Q6 & !Q4 & !CLEAR # Q6 & !Q3 & !CLEAR #
        Q6 & !Q2 & !CLEAR # Q6 & !Q1 & !CLEAR # Q6 & !Q0 & !CLEAR;

Q7.D  = !Q7 & Q6 & Q5 & Q4 & Q3 & Q2 & Q1 & Q0 & !CLEAR #
        Q7 & !Q6 & !CLEAR # Q7 & !Q5 & !CLEAR # Q7 & !Q4 & !CLEAR #
        Q7 & !Q3 & !CLEAR # Q7 & !Q2 & !CLEAR # Q7 & !Q1 & !CLEAR #
        Q7 & !Q0 & !CLEAR;
```

Chapter 5
Parts List

Macintosh Video Board Parts List
Xceed SE/306-48 -- SE/30 8-Bit 640 x 480

	Part	Micron #	Location	Qty.
1.	GAL 16V8A-15	71-07114	U1,25,28,33,34,37,38,42	8
2.	GAL 20V8A-15	71-01306	U22	1
3.	TI PAL20X10-20CNT	71-01311	U31	1
4.	TI PAL20X8-20CNT	71-01301	U30	1
5.	IC, AMP AM29827	71-012555	U36,39	2
6.	IC, 74ALS541N	71-01316	U24,27,29,32	4
7.	IC, 74ALS574AN	71-01318	U2-4,21	4
8.	IC, 74ALS645AN	71-01303	U40,41,43,44	4
9.	IC, 74F241	71-01324	U23	1
10.	VRAM Micron MT42C4064Z-15(-25)		U5-20	16
11.	EPROM Signetics 27C64A-15FA	71-07106	U35	1
12.	VDAC Brooktree BT478KPJ35	71-07113	U26	1
13.	Oscillator, 30.24 MHz TTL	71-02044	Y2	1
14.	Diode, IN5908	71-02035	CR1	1
15.	Voltage regulator National LM385BZ-1.2	71-07250	Z1	1
16.	Capacitor, axial, 50V, 0.1 uF	71-04002	all bypass,C1,C3	31
17.	Capacitor, tant, 10V 33 uF	71-04031	C2,C4-6,C8	5
18.	Resistor, 147 ohm, 1/4 W 1%	71-03048	R1	1
19.	Resistor, 1 Kohm, 1/4 W 5%	71-03020	R2	1
20.	Resistor, 75 ohm, 1/4 W 1%	71-03049	R3-5	3
21.	Resistor, SIP, 10-pin 2.2 Kohm	71-03090	RN8-11	4
22.	Resistor, SIP, 10-pin 39 ohm	71-030915	RN1-5	5
23.	Ferrite bead filter #2743001112	71-07099	L1-10	10
24.	Connector, 120-pin AMP 535022-1	72-CON322	J1	1
25.	12-pin connector AMP 87631-8	71-07129	J2	1
26.	Pin for 12-pin connector	72-CON130	cable	12
27.	12-pin connector plug AMP 103168-4	71-07131	cable	1
28.	15-pin D-shell connector Vernitron 38-155	71-07146	cable	1

29.	PCB			1
30.	Mounting Bracket AKB120	71-07119	cable	1
31.	Mounting hardware			
32.	Oscillator 27.0 MHz TTL	71-02046	Y1	1
33.	Berg stick, 2-pin 65500-202	71-020844	CD	1
33.	Berg stick, 3-pin 65500-203	71-020846	IRE	1
34.	Resistor, SIP, 10-pin 270 ohm	72-RES235	RN6,7	2

Chapter 6
Video Board Testing

6.1 General Test Flow

The video board must first be examined visually for any missing or wrong components, solder bridges, etc. A quick resistance test should find any power-to-ground shorts. There are three phases of the test: Initial, Burn-in, and Final. Initial Test checks to see that the board functions properly. Burn-in is an extended test. Final Test is a quick functionality test before the board is shipped.

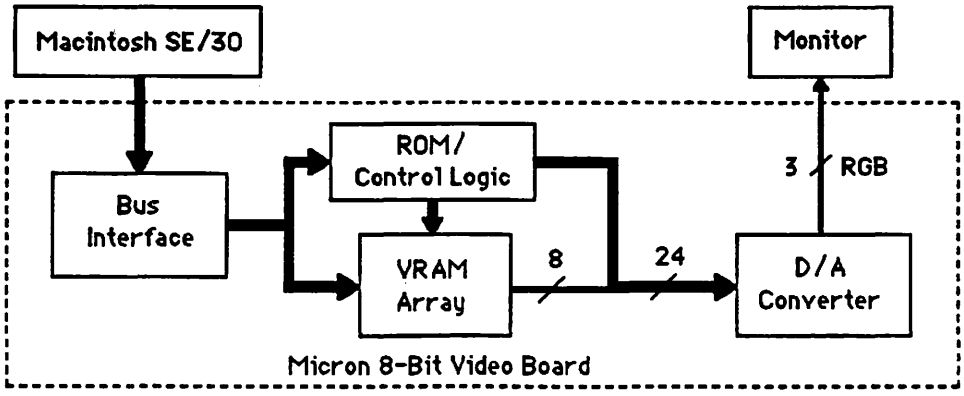


Figure 6-1 Video Board Block Diagram

For the purposes of testing, Micron's Macintosh SE/30 8-Bit Video Board consists primarily of four elements: The bus interface, the VRAM array to store the pixel data, the ROM and control logic, and the D/A converter. The bus interface is essentially invisible. The VRAM array and ROM can be accessed at specific addresses in the memory map. The D/A converter's operation can best be verified by checking the screen for the proper patterns or looking at the video signals on an oscilloscope.

6.2 Test Specification

ECN.

Chapter 7
Test Points/Oscilloscope

Red	R3	Pin 1
Red Return (ground)	R3	Pin 2
Green	R4	Pin 1
Green Return	R4	Pin 2
Blue	R5	Pin 1
Blue Return	R5	Pin 2

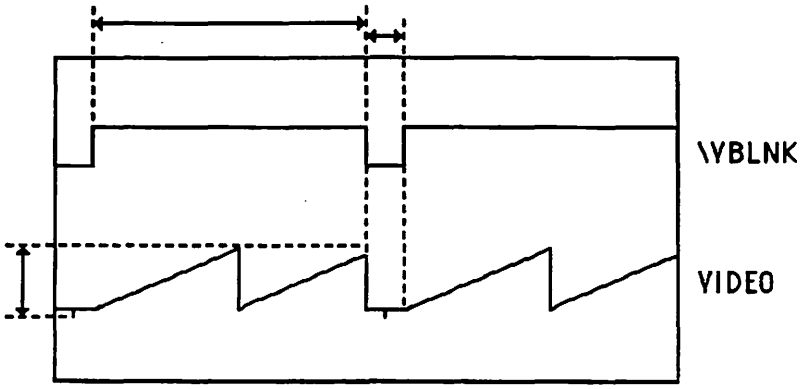


Figure 7-1 Oscilloscope

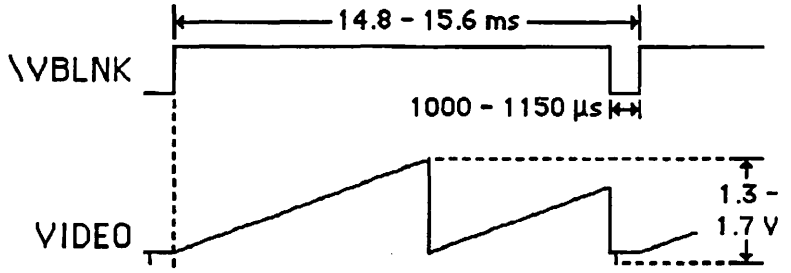


Figure 7-2 Measurements

Appendix A
Test Software

The test software is named '8Bit' and was written in-house. It may be modified as required. The following description covers the first release of the software, Version 1.00. On the Macintosh, software may be executed by double-clicking with the mouse on the icon with the program name.

When the program begins execution, it creates a file named '0.txt.' The old '0.txt' file will be deleted. If the '0.txt' file exists and is open (if the computer crashed before closing the file), then the new file will be named '1.txt' or '2.txt' etc. Once the computer has been rebooted, the file names will begin over with '0.txt.'

The file created by the program is of type 'text.' Word-processing programs such as MacWrite and Word can convert the file to a readable form. If the file is to be printed, it should be converted to 9 point text in a font that allows the data to line up correctly, such as Monaco or Courier (laser font). This is done by selecting (highlighting) all of the text and choosing the size and font menu choices.

If there are no errors during the automatic tests, nothing will be printed into the file. Otherwise, the test types, patterns, and other test information is printed in the file along with any errors found. A '0' or '1' (zero or one) in a bit position means that the bit was correct; a '.' or '/' (period or slash) signifies a bit error. The burn-in test stops testing a board once too many errors have been found.

expected	received	code
0	0	0
0	1	.
1	0	/
1	1	1

Table A-1 File Error Output

Initial Test, Burn-in, and Final Test are included as options in the main menu. These tests are used during the normal test flow. If the board fails any of the tests, then any part of the test may be run separately for debugging purposes. The remainder of the menu options allow the individual tests to be run.

The automatic tests may be halted at any point by holding down the mouse button. The program will complete the current loop and return to the main menu. The VRAM tests and video patterns may also be halted in this manner.

When the test is started, it checks the computer's data for any boards that were found. The variables for the slot and address selections are set to the

first (or lowest number) board found. If the user enters addresses or slots for a board that is not in the system, the computer may crash when the software tries to write to invalid addresses. Never write to a slot that does not contain a board.

If the program did not find any Micron boards, it will put up a dialog asking which board is installed. The board should be found by the system, and if it is not then the test has failed. The test should only be used for debugging purposes if the board was not found.

A.1 Test Options

To the right of the main window, there are several letters which represent the values of the test control flags. The flags control the general flow of the test: whether the data will be verified (checked for accuracy), whether information will be printed to the data file, etc. The setting of a flag is valid until it is either modified by the user's actions, or the test is halted. The automatic tests (initial, burn-in, final) may modify some of the flags. Flags may be changed through the 'Options' menu.

The flags that may be set by the user are listed below:

(H)elp will list information messages along with the menus as long as the flag is ON. Every menu selection changes the message written to the help window.
(V)erify ON will check the data that is read and print any errors to the file or screen if requested. If verify is OFF, then the data will not be checked.
(F)ile controls the test output. Errors will be printed into the file chosen by the user if the flag is ON.
(S)creen controls the test output to the screen. If the flag is OFF, most information will not be shown to the user.
The values (R)ead, (W)rite, and (.)Delay are set according to the 'Options' menu. They can not be individually changed by the user except through the menu.
'Number' of errors can be set to any number 1-999. If errors are found, then they will be reported up to this number of errors. After the number has been reached, the test will skip a block of memory and continue testing in this manner over the memory range.
'Number' of iterations is in the same dialog as Errors. Enter '0' for infinite iterations.

There are some exceptions to the flag settings. The ROTATE test will do a read/write test regardless of the flags, only using the Verify flag to determine if the data will be checked. The ROM verification test always reads the data regardless of the read/write flags, but follows the verify flag.

A.2 VRAM Tests

The VRAM tests are solid, increment, random, and rotate. The options include read/write, addressing, pattern selection, and iterations. Holding the mouse button down at any time during the test will terminate it.

Selecting the read/write option will run a loop where the data is written to each address and then immediately read back. If the delay is used, then the data is written to the entire board before being read. The rotate test is automatically a read/write selection but does not change those flags.

The first address is the address at which the test will start. The test will run until it hits the address just below the last address entered. For example, running a test from c0001000 to c0001001 is the same as c0001000 to c0001004. Both will only test the first address c0001000.

The VRAM tests should be used to test only the visible portion of the VRAM array, since the rest of the addresses do not affect the function of the board. The hex value for the distance across is used for this. Only 280 hex bytes are shown on the small screen; a total of 400 hex values exist per row. The large monitor shows all 400 hex bytes. The value in this variable selects the addresses in each row to test. (Note: The screen width 280 hex corresponds to 640 pixels per row in decimal; 8-bit mode, one byte per pixel. For 4-bit mode one byte represents two pixels, or 140 hex bytes visible.)

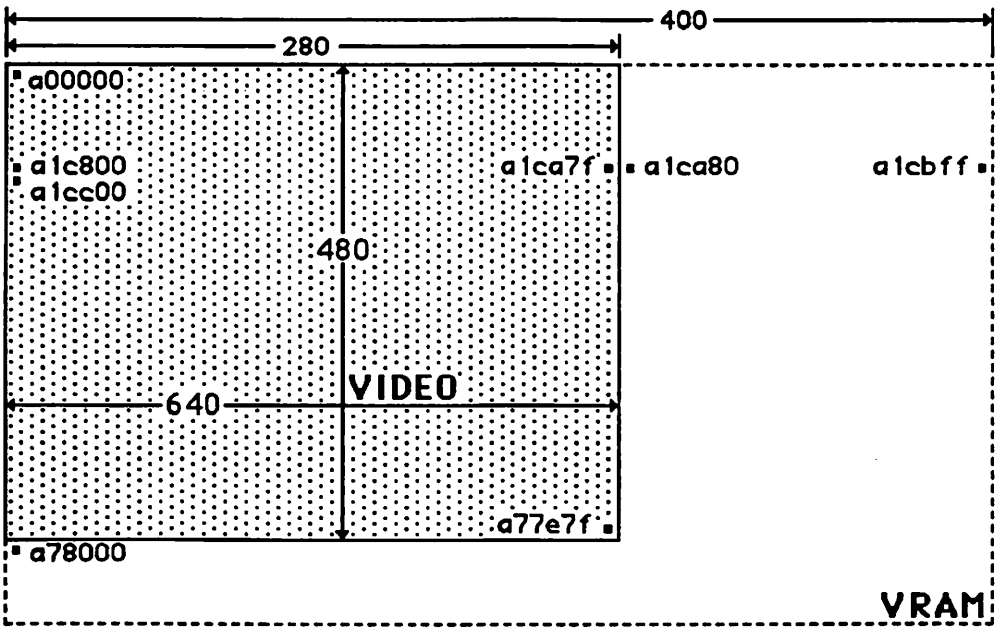


Figure A-1 VRAM Array and Video Output: 8-Bit 640x480

The Solid test uses a single value for the data, entered by the user. The Increment test starts with the user-selected pattern, but increments the data by 1 at every address. The Random data is generated from a number selected

by the user, and repeats every 4000 hex. Rotate uses its own data -- the value 111111 hex (1 hex = 0001 binary) is used, then 222222 (0010), 444444 (0100), and 888888 (1000) are written and read at a single address. The test begins at the next address with 111111 again. Rotate is a read/write test only.

All of the tests except Rotate allow the user to enter a starting value of one hex byte. All of the tests do not use bytes, though. Solid and Increment use byte values, Rotate uses words, and Random uses long words (8, 16, and 32 bits respectively).

A.3 Video Tests

The Video tests in the Miscellaneous menu are Color, Text, and Border. None of the video test data is verified. The important thing to note about the video patterns is if there are any pixels missing or if the pattern or colors are shown incorrectly.

The 'Color' test changes the color palette. A window is opened with blocks of different colors. The colors usually are different shades next to each other -- they should vary smoothly. If one of the colors doesn't match, the visual verification has failed.

The Text and Border tests put windows on the test screen. The alphabet is printed in several different sizes and fonts of text; it all should all be readable in the text window. The Border test should show a white border around the edge of the screen, with black in the center.

Windows may only be opened in active screens. When a board is first put in a slot or moved to a new slot, the computer must be told (using Direct Monitor in the Control Panel) that the board exists, and be rebooted. Otherwise the windows will not appear on the second screen. This is why all boards are tested in the same slot during Initial and Final tests.

A.4 Miscellaneous

The Miscellaneous menu also allows testing of the ROM and control logic. The ROM data is contained in the high byte of the upper 8000 hex memory locations in a slot. The data is aliased, so also appears at multiples of these addresses (Fs300000-FsFFFFFC). The ROM contains the driver for the board and information on the board type. The data is read from the ROM in the ROM test; the verify flag determines whether the data will be checked. In the test the ROM is read from the bottom to the top of the last possible address space, starting at FsFFE000.

Mode changes the bits/pixel information for the board. For visual verification, there is a small window that shows the explicit color palette. It has 256 blocks of color. If the board is in 1-bit mode, all of the blocks will be black or white. For 2- and 4-bit mode, vertical stripes of 4 or 16 different colors will be seen in

the window. In 8-bit mode, each block will be a different color for a total of 256 colors.

The switch option turns the video board of and on. The screen will flash when the switch selection is used. From software, the status of the board can be determined by the least significant bit (LSB) in the most significant byte of the VRAM data. This is NOT data actually stored in the VRAM array -- only the 3 bytes of data in the VRAMs are normally shown in the VRAM tests. If the board is on, this bit will be high; if the board is off the bit will be low. The switch test also requires user verification to be sure the board was turned off.

Appendix B
Schematics

PIN	ROW C	ROW B	ROW A
40	+12	-5	-12
39	GND	GND	GND
38	C16M	ECLK	CPUclock
37	+5	+5	+5
36	A0	A1	A2
35	A3	A4	A5
34	A6	GND	A7
33	A8	A9	A10
32	A11	A12	A13
31	A14	+5	A15
30	A16	A17	A18
29	A19	A20	A21
28	A22	GND	A23
27	A24	A25	A26
26	A27	A28	A29
25	A30	+5	A31
24	D31	D30	D29
23	D28	D27	D26
22	D25	GND	D24
21	D23	D22	D21
20	D20	D19	D18
19	D17	+5	D16
18	D15	D14	D13
17	D12	D11	D10
16	D9	GND	D8
15	D7	D6	D5
14	D4	D3	D2
13	D1	Reserved	D0
12	/HALT	/BERR	/RESET
11	FC0	FC1	FC2
10	/BR	/BG	/BGACK
9	/AS	SIZ0	SIZ1
8	/R/W	/DSACK0	/DSACK1
7	/CBREQ	/CBACK	/STERM
6	/RMC	/DS	/CIOUT
5	/IPL0	/IPL1	/IPL2
4	/IRQ1	/IRQ2	/IRQ3
3	/TM0A	/TM1A	/BUSLOCK
2	/NUBUS	Reserved	Reserved
1	PWROFF	Reserved	Reserved

Table B-1 Connector Pin List

Pin	Signal
1	GND
2	RED
3	/CSYNCH
4	GND
5	GREEN
6	GND
7	No connection
8	No connection
9	BLUE
10	No connection
11	GND
12	GREEN
13	GND
14	GND
15	No connection

Table B-2 Video Output Connector - computer

Pin	Signal
1	\VGASYNC
2	\VGAVSYNC
3	GND
4	\CSYNC
5	GND
6	RED
7	GND
8	GREEN
9	GND
10	BLUE
11	FREQ ADJ
12	CABLE OK

Table B-3 Video Output Connector - board