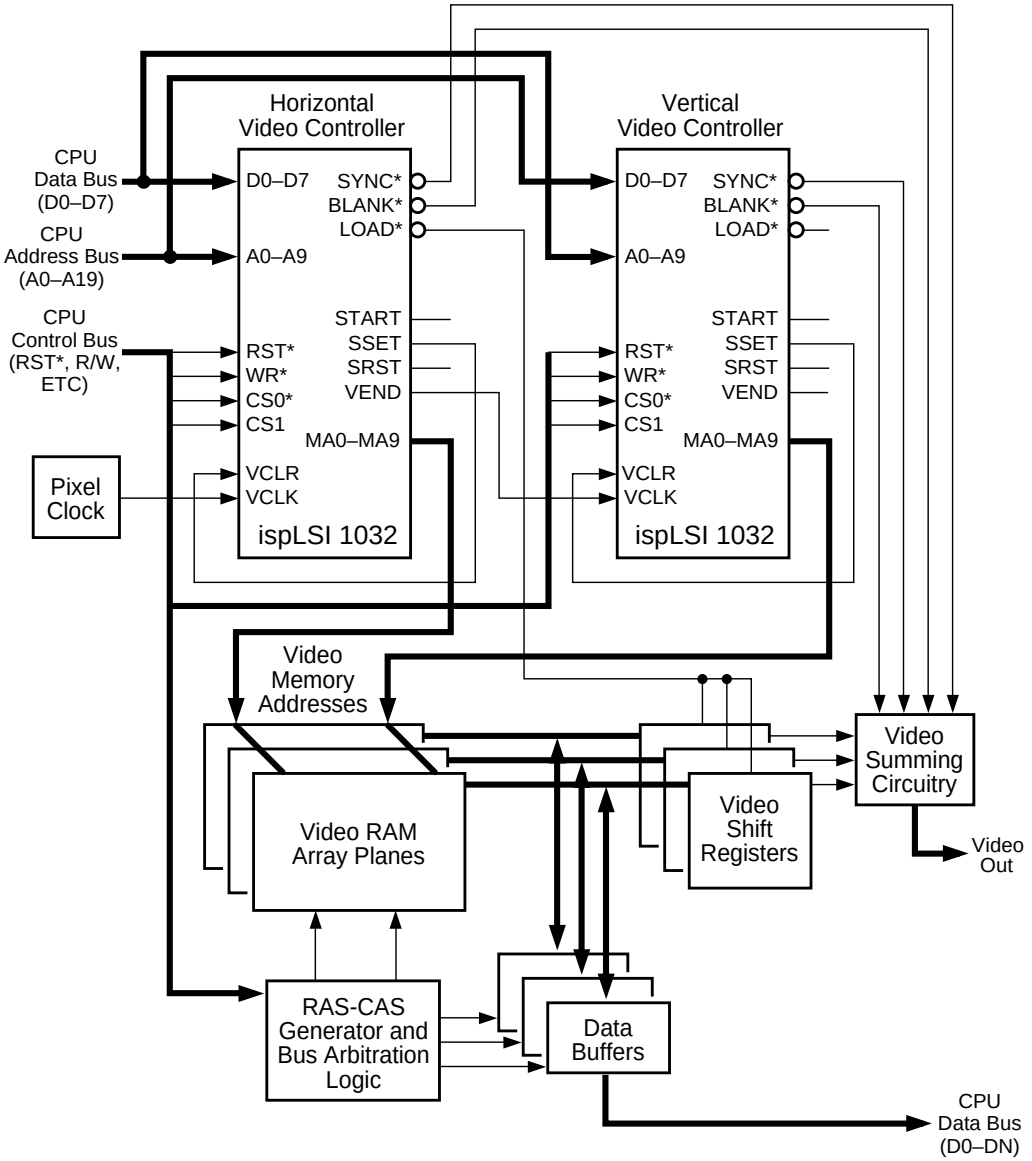
Video Graphics Controller

Introduction

This Graphics Controller design consists of two ispLSI® 1032 chips programmed identically to produce most of the basic video functions and timing signals associated with a general purpose graphics interface. The generic design of the controller allows customization by adding additional circuitry for a Graphics Controller System based on the design-specific requirements (see Figure 1). The ISP™ capability of the ispLSI device enables the design engineer to update the hardware via ISP programming software.

Figure 1. Video Graphics Controller System Block Diagram

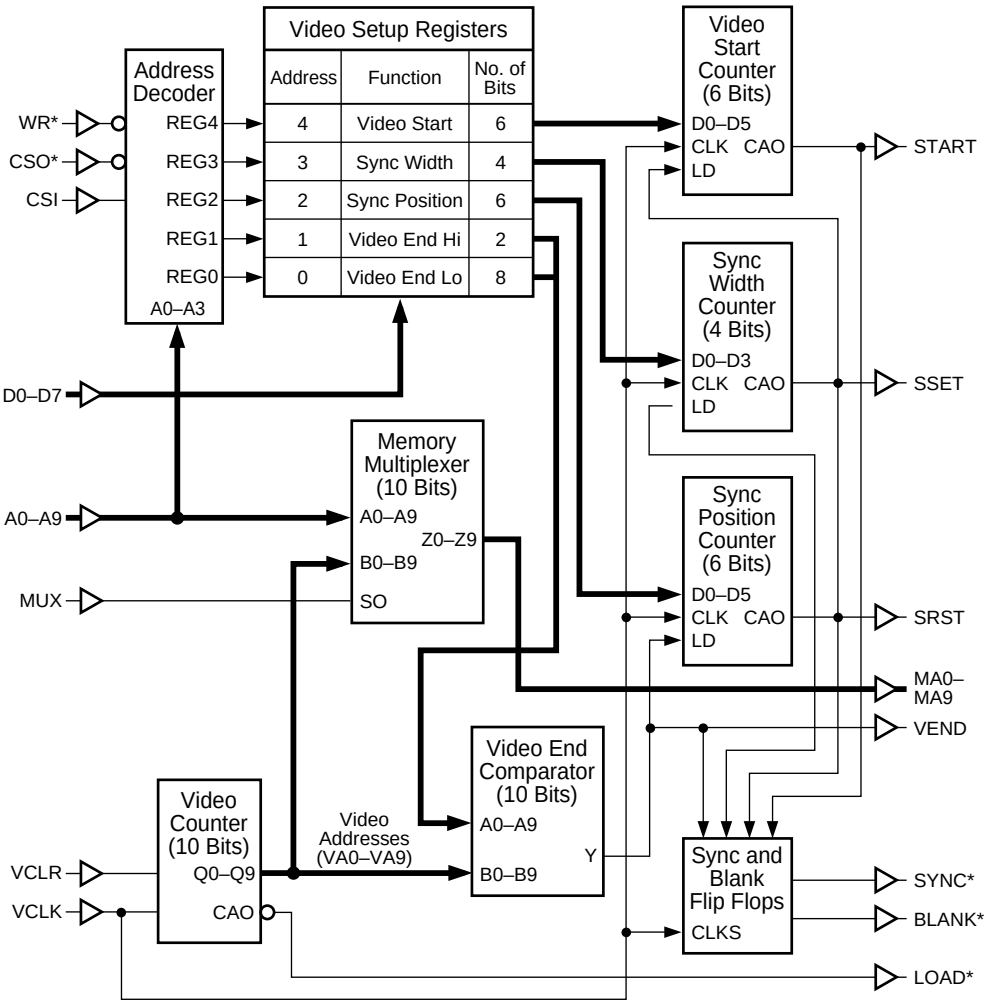


This design is capable of a maximum 1024 X 1024 non-interlaced display with programmable blanking and sync signal positioning. One of the ispLSI 1032 devices is used for Horizontal Video Control (HVC) and the other for Vertical Video Control (VVC). Because the two ispLSI 1032 devices are programmed identically, the LOAD* signal (Schematic 2) is redundant on the VVC chip and only used on the HVC chip.

Referencing Figure 2, the signals which the CPU sends to the Video Graphics Controller (VGC), are: WRITE (WR*), CHIP SELECT 0*/1 (CSO*/1), DATA BUS (D0-D7), ADDRESS BUS (A0-A9), and MULTIPLEXER SELECT (MUX). The Address Decoder receives an address from the CPU. Once decoded, this address enables one of the Video Setup Registers (VSRs) which then receives video information from the CPU data bus. This setup data is then fed to the appropriate counter or comparator, which actually controls that specific display parameter.

The CPU address bus is also interfaced to the Memory Multiplexer (MMUX) 'A' inputs. The 'B' inputs of the MMUX are connected to the outputs of the Video Counter (VCNTR). The MMUX allows either the CPU or the VCNTR to access video memory depending on the polarity of the MUX signal from the CPU. Additionally, the VCNTR produces the LOAD* signal to the video shift register, which is external to the ispLSI 1032.

Figure 2. Video Graphics Controller Chip Block Diagram



The VCNTR also feeds the Video End Comparator (VEC). The VEC compares the addresses from the VCNTR and the Video End Hi and Lo registers located in the VSRs. When true, the VEC outputs the Video End (VEND) signal and simultaneously enables the load for the Sync Position Counter (SPC), while clearing the Blanking flip-flop.

The SPC data is loaded from the Sync Position register which is located in the VSRs. The SPC counts down to zero at which point it outputs the Sync Reset (SRST) signal. SRST also enables the load for the Sync Width Counter (SWC), and clears the Sync flip-flop.

The SWC's data comes from the Sync Width register in the VSRs. The SWC counts down to zero. At zero, it enables the load for the Video Start Counter (VSC), and also sets the Sync flip-flop.

The VSC receives its data from the Video Start VSR. The VSC counts down to zero, and while at zero it produces the START signal simultaneously setting the Blanking flip-flop.

Address Decoder (Schematic 2)

The address decoder is enabled by the WR* and CS0*/1 signals and decodes address bits A0-A2 into one of five active high select output signals, R0-R4. These are the select lines to the video attribute setup registers (Schematic 3). The CS0* active low chip-select and CS1 active high chip-select are for differentiating between the horizontal controller and the vertical controller when interfacing to the CPU bus as two of these chips must be used in the system. The WR* is used to synchronize the access to the registers with CPU write cycle. All accesses to this block are write only.

Video Setup Registers (Schematic 3)

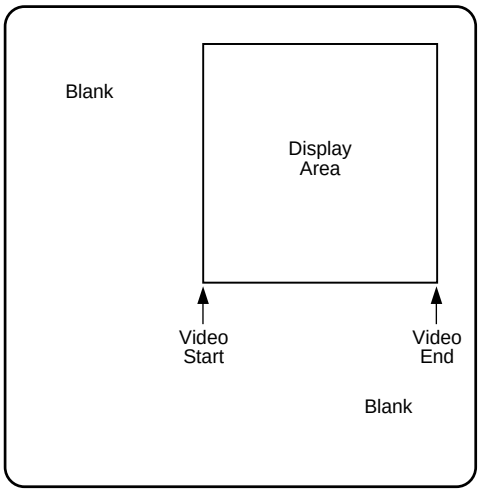
The circuit is designed to interface to an eight-bit data bus but could be easily redesigned to interface to a 16-bit bus. The Video Attribute Setup Register's addresses and widths are as shown in Table 1.

Table 1. Video Setup Register Addresses and Widths

Address	Name-Function	Number of bits
0	Video End Low (Ve 7:0)	8
1	Video End High (Ve 9:8)	2
2	Sync Position (Sp 5:0)	6
3	Sync Width (Sw 3:0)	4
4	Video Start (Vs 5:0)	6

These registers provide the data to be compared or loaded into one of the dead-end down counters used for positioning the display viewing area or sync pulse positions and widths (see Figure 3).

Figure 3. Typical Video Display Set up



Video End Low and High Registers

These registers combine to form the ten-bit address location of the video display endpoints. In the case of the horizontal display location, this is the right-hand side of the screen and the vertical display location is the bottom, or last visible scan line. In other words this is the point where video ends and blanking begins.

Sync Position Register

This six-bit register holds the value of the distance from where video ends and the horizontal or vertical sync pulses start thus allowing for sync pulse positioning relative to video end. This is counted in pixels in the horizontal plane and lines in the vertical plane. The value of this register cannot be less than one.

Sync Width Register

This four-bit register holds the value of the sync pulse width. This is counted in pixels in the horizontal plane and lines in the vertical plane. The value of this register cannot be less than one.

Video Start Register

This six-bit register holds the value of the distance from where the sync pulse or blanking ends and video starts. This is counted in pixels in the horizontal plane and lines in the vertical plane. The value of this register cannot be less than one.

Video Counter (Schematic 2)

This is a ten-bit counter which provides the video addresses VA0-VA9. In the case of the horizontal controller, this register provides the LOAD* signal for the video RAM shift registers. This register's synchronous outputs, clock, and asynchronous reset lines are accessible from the I/O pins of the chip for interfacing with the system's horizontal and vertical functions. The reset to the counter is VCLR and is typically connected externally to the SSET signal (Schematic 5). SSET resets the counter at the end of the sync pulse. This can be customized for the specific application. VCLK is the clock input to the counter. VCLK is connected to the pixel clock of the horizontal controller (HVC) and is driven by the VEND signal from the horizontal controller (HVC) in the case of the vertical controller (VVC). The LOAD* signal output is a one cycle-wide pulse every 16 pixels. This can be reduced to eight pixels by modifying the counter's boolean statements.

Video End Comparator (Schematic 5)

This is a ten-bit comparator which compares the ten-bit value in the Video End Low and High registers (R0-R1 Schematic 3), to the ten-bit value of the Video Counter. When the compare is true a one cycle-wide pulse is generated called VEND. This is the end of visible video and starts the sync position counter running while also clearing the blanking flip-flop.

Sync Position Counter (Schematic 4)

This is a six-bit loadable, dead-end down counter which counts until it reaches zero and then holds until it is loaded with a value greater than or equal to one. The load is activated by the VEND signal generated by the Video End Comparator. The count is a maximum of 64 pixels (horiz) or lines (vert) and is loaded each time with the value of the Sync Position Register (R2). When the count reaches zero, the counter produces the signal SRST which starts the Sync Width Counter and clears the Sync flip-flop (Schematic 5).

Sync Width Counter (Schematic 5)

This is a four-bit loadable, dead-end down counter which counts until it reaches zero and then holds until it is loaded with a value greater than or equal to one. The load is activated by the SRST signal which is generated by the sync position counter. The count is a maximum of 16 pixels (horiz) or lines (vert) and is loaded each time with the value of the Sync Width Register (R3 Schematic 3). When the count reaches zero, the counter produces the signal SSET which starts the Video Start Counter running and sets the sync flip-flop.

Video Start Counter (Schematic 4)

This is a six-bit loadable, dead-end down counter which counts until it reaches zero and then holds until it is loaded with a value greater than or equal to one. The load is activated by the SSET signal (Schematic 5), generated by the Sync Width Counter. The count is a maximum of 64 pixels (horiz) or lines (vert) and is loaded each time with the value of the Video Start Register (R4 Schematic 3). When the count reaches zero, the counter produces the signal START which sets the Blanking flip-flop (Schematic 5).

Sync Flip-Flop (Schematic 5)

This flip-flop is cleared by the signal SRST (Schematic 4), and set by the signal SSET to produce the sync pulse for either horizontal or vertical. It is a J-K flip-flop which is clocked by VCLK that delays the actual edges by one clock. This factor must be taken into account when calculating the sync position and sync width values as the value is one less than the true position or width. These values must be no less than one.

Blanking Flip-Flop (Schematic 5)

This flip-flop is cleared by the signal VEND and set by the signal START (Schematic 4), to produce the blanking signal for either horizontal or vertical controllers. It is a J-K flip-flop which is clocked by VCLK This flip-flop delays the actual edges by one clock. This must be taken into account when calculating the sync position and sync width values as the value is one less than the true position or width. Thus the Sync position and width values must be greater than or equal to one.

Memory Address Multiplexer (Schematic 6)

This is a dual input ten-bit multiplexer which outputs either the video addresses (VA0-VA9), or the CPU addresses (A0-A9), to the output pins (MA0-MA9). This allows for either the video counters or the CPU to directly address the video memory. The multiplexer is controlled by the signal MUX and when MUX is low selects the CPU address. When MUX is high it selects the video counters (horizontal and vertical).

This system design is generic in terms of the size and number of the video memory planes. It is based on the additional support of RAS-CAS logic, if multiplexed dynamic RAM is used, along with bus arbitration logic to allow for transparent accesses by the CPU. It also assumes that the shift registers (if used), are correctly chosen and interfaced to the video RAM. The final support circuitry is video summing which, depending on the type of display to be driven (analog or digital), and the polarity of the blanking and sync signals, has a wide variation of layouts. All of these functions, when finally chosen, can be easily incorporated into the additional 25 percent of each of the HVC and VVC chips remaining, or placed into additional ispLSI devices as needed. This design allows for quick and flexible programmable video graphic interface to numerous applications.

Table 2. Pin Functional Descriptions

NAME	TYPE	FUNCTION
WR*	Input	Allow strobe used to write data into video attribute set up register. Selected by address lines A0-A2. Also qualified with CS0*/1.
CS0*/1	Input	Active low/high chip select used to enable writes to attribute set up registers.
AO-A9	Input	A0-A2 are used to select one of the video attribute set up registers. A0-A9 are used to address the video memory.
D0-D7	Input	Data input to the video attribute set up registers.
MUX	Input	Mux select line for video memory access. High select CPU addresses (A0-A9), low select video counter addresses (VA0-VA9).
MA0-MA9	Output	Video memory address lines.
VEND	Output	Active high signal used to indicate the end of a horizontal or vertical scan.
SRST	Output	Active high signal used to indicate the end of horizontal or vertical Sync.
SSET	Output	Active high signal used to indicate the beginning of a horizontal or vertical Sync.
START	Output	Active high signal used to indicate the start of a horizontal or vertical visible scan.
LOAD*	Output	Active low signal used to load the external video shift registers with data from the video memory.
BLANK*	Output	Active low signal used to indicate the blanking of horizontal or vertical display.
SYNC*	Output	Active low signal used to indicate the horizontal or vertical Sync pulse.
VCLK	Input	System clock running at same frequency as the monitor.

Table 3. Video Attribute Formulas

The following are the formulas for calculating the display characteristics:

tc = pixel clock time period (i.e., 10Mhz = 100ns)
Ve = video end (0-1024)
Sp = sync position (1-63)
Sw = sync width (1-15)
Vs = video start (1-63)

Horizontal (HVC):

- horizontal scan line period = [Ve+(Sp+1)+(Sw+1)+(Vs+1)] * tc
- horizontal scan rate = 1/horizontal scan line period
- horizontal display period = [Ve-(Vs+1)] * tc
- LOAD* frequency = tc * 1

Vertical (VVC):

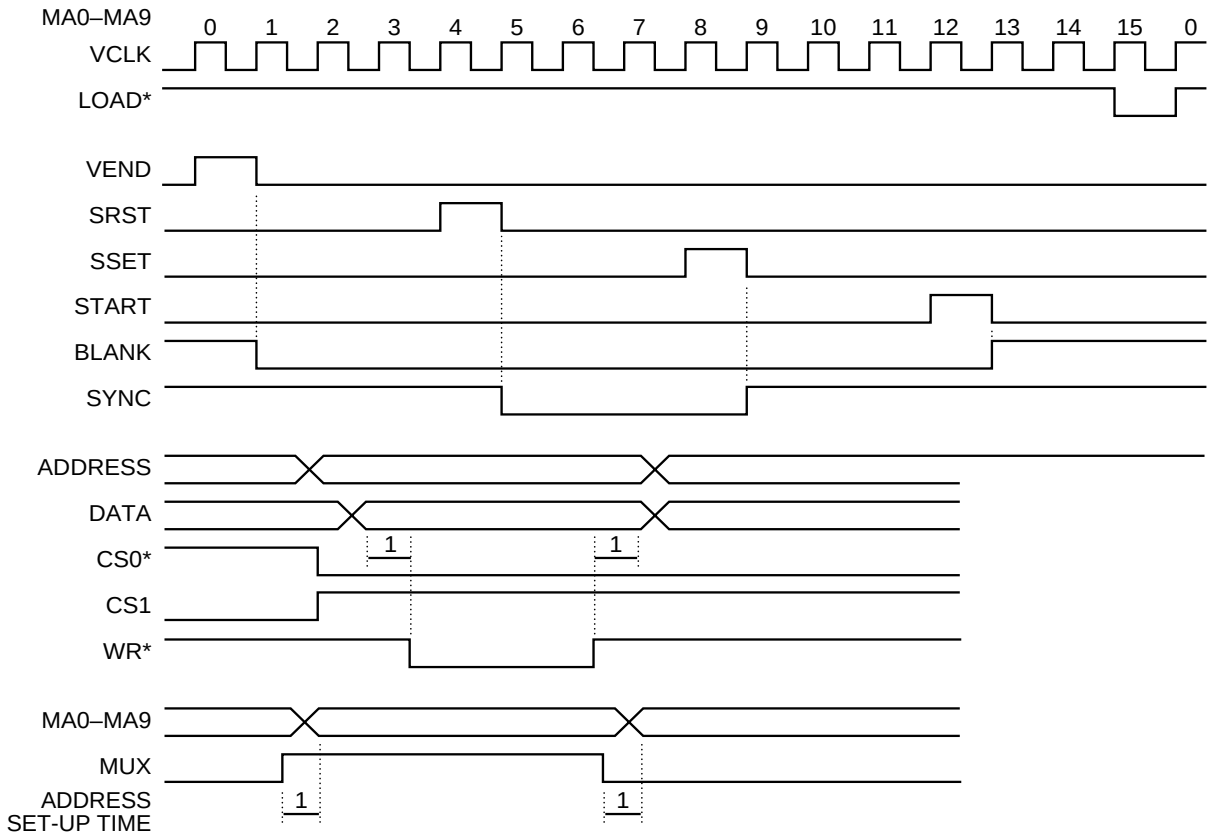
- vertical scan line period = [Ve+(Sp+1)+(Sw+1)+(Vs+1)] * horizontal scan line period
- vertical scan rate = 1/vertical scan line period
- vertical display period = [Ve-(Vs+1)] *horizontal scan line period

The ispLSI Advantage

The ispLSI 1032 is an excellent choice for this type of design because of its density, flexibility and speed. The device utilization percentages for this particular design are: 75% GLB, 66% GLB output, and 61% I/O. This means that there is enough of the device left to interface to a 16-bit bus or to add glue logic which might be associated with a specific design. The I/O assignment in the ispLSI 1032 is extremely flexible. I/Os can be fixed to a specific pin, or left for the router to decide the best connection. With no fixed pins, this design took 1.5 minutes to route, and re-routing with all pins fixed was completed in a matter of seconds.

The rest of this design example consists of an appendix containing the schematics and a hard copy of the LDF file for this design.

Figure 4. Video Graphics Controller Timing



1. Note:
   See Sidebar for Description

The major timing relationships for this device are shown in Figure 4. All signals are shown in relation to VCLK.

As can be seen from the diagram, LOAD* is generated every 16 VCLKs. LOAD* loads the video shift registers with data from the video memory. BLANK is activated by the falling edge of VEND and is inactivated at the falling edge of START. SYNC goes low at the falling edge of SRST and rises with the falling edge of SSET.

The CPU related signals are shown in Waveforms 9 to 13. CSO* and CS1 are really complimentary versions of the same signal. Because two ispLSI 1032s are used in the design, CS0*, for example, would be used as the chip select for the horizontal controller chip CS1 would then be used as the chip select for the vertical controller chip. In any case, there is a set-up and hold time associated with a data write into the chip. This is indicated by the short solid lines bounded by the dashed lines in between the DATA and CS0* waveforms. The actual set-up and hold times involved are dependent upon the frequency of VCLK, but the relationship to VCLK is clearly shown.

The last two waveforms on the diagram show the delay from MUX rising or falling and the validity of the addresses on MA0 to MA9. This delay employs the same caveat as above. The actual time depends upon the frequency of VCLK.

Appendix

```
// graphfix.ldf generated using Lattice pDS Software V2.50
LDF 1.00.00 DESIGNLDF;
DESIGN GRAPHICS 1.00;
PROJECTNAME
DESCRIPTION
This is one of two identical chips used for either horizontal or vertical control
in
the graphics controller design.  Two of these chips produce most of the basic
video functions and timing signals associated with a general purpose graphics
interface.  The design is capable of up to a 1024 X 1024 non-interlaced display
with programmable blanking and sync signal positioning.  One of the chips is
used for horizontal video control (HVC) and the other, vertical video control
(VVC).;
PART ispLSI1032-90LJ;
DECLARE
END;  //DECLARE

SYM GLB  D4  1  MISC. SIGNALS 2;
// SSET signal generation, SYNC & BLANK;
// intermediate signal generation;
SIGTYPE SYNC REG OUT;
SIGTYPE BLANK REG OUT;
SIGTYPE SSET OUT;
     EQUATIONS
         SYNC.CLK=VCLK
         SSET=!SSET1&SSET0;
         SYNC.D = !(!(!SYNC.Q & SSET) & (!SYNC.Q # SRST));
         BLANK.D = !(!(!BLANK.Q & START) & (!BLANK.Q # VEND));
      END;
END;

SYM GLB  C7  1  ENABLE - !WR&!CS0&CS1;
// Write enable qualification for address decoder;
SIGTYPE ENABLE OUT;
     EQUATIONS
         ENABLE = !WR & !CS0 & CS1;
     END;
END;

SYM GLB  A1  1  VIDEO COUNTERS;
// Video memory address counter bits VA4-VA7;
SIGTYPE [VA4..VA7] REG OUT;
      EQUATIONS
         VA4.CLK = VCLK;
         VA4.RE = VCLR;
         VA4=(VA0 & VA1& VA2 & VA3) $$ VA4;
         VA5=(VA0 & VA1 & VA2 & VA3 & VA4) $$ VA5;
         VA6=(VA0 & VA1 & VA2 & VA3 & VA4 & VA5) $$ VA6;
         VA7=(VA0 & VA1 & VA2 & VA3 & VA4 & VA5 & VA6) $$ VA7;
      END;
END;
```

```
SYM GLB  A0  1  VIDEO COUNTERS;
// Video memory address counter bits VA0-VA3;
SIGTYPE [VA0..VA3] REG OUT;
      EQUATIONS
         VA0.CLK = VCLK;
         VA0.RE = VCLR;
         VA0 = VA0 $$ VCC;
         VA1 = VA0 $$ VA1;
         VA2 = (VA0 & VA1) $$ VA2;
         VA3 = (VA0 & VA1& VA2) $$ VA3;
      END;
END;

SYM GLB  A2  1  VIDEO COUNTERS;
// Video memory address counter bits VA8,VA9;
//  and LOAD signal output generation;
SIGTYPE VA8 REG OUT;
SIGTYPE VA9 REG OUT;
SIGTYPE LOAD OUT;
      EQUATIONS
         VA8.CLK = VCLK;
         VA8.RE = VCLR;
         VA8=(VA0 & VA1 & VA2 & VA3 & VA4 & VA5 & VA6 & VA7) $$ VA8;
         VA9=(VA0 & VA1 & VA2 & VA3 & VA4 & VA5 & VA6 & VA7 & VA8) $$ VA9;
         LOAD=VA0 & VA1 & VA2 & VA3;
      END;
END;

SYM GLB  A3  1  ADDRESS DECODE;
// Register address decoder;
SIGTYPE [R0..R3] OUT;
      EQUATIONS
         R0 = ENABLE & !A0 & !A1 & !A2;
         R1 = ENABLE & A0 & !A1 & !A2;
         R2 = ENABLE & !A0 & A1 & !A2;
         R3 = ENABLE & A0 & A1 & !A2;
      END;
END;

SYM GLB  A4  1  END HI (VIDEO);
// R4 of register address decoder and video;
// data registers (video end hi);
SIGTYPE R4 OUT;
SIGTYPE [R1Q0..R1Q1] OUT;
EQUATIONS
         R4 = !WR & !CS0 & CS1 & !A0 & !A1 & A2;
         [R1Q0..R1Q1] = [D4..D5] & R1;
   END;
END;
```

```
SYM GLB  A5  1  END LO 1 (VIDEO);
// Video data registers (video end lo);
SIGTYPE [R0Q0..R0Q3] OUT;
     EQUATIONS
         [R0Q0..R0Q3] = [D0..D3] & R0;
     END;
END;

SYM GLB  A6  1  END LO 2 (VIDEO);
// Video data registers (video end lo);
SIGTYPE [R0Q4..R0Q7] OUT;
     EQUATIONS
         [R0Q4..R0Q7] = [D4..D7] & R0;
     END;
END;

SYM GLB  A7  1  POSITION, SYNC 1;
// Video data registers (sync position);
SIGTYPE [R2Q0..R2Q3] OUT;
     EQUATIONS
         [R2Q0..R2Q3] = [D0..D3] & R2;
     END;
END;

SYM GLB  B0  1  START & POSITION 2;
// Video data registers (sync position);
// Video data registers (video start);
SIGTYPE [R2Q4..R2Q5] OUT;
SIGTYPE [R4Q4..R4Q5] OUT;
     EQUATIONS
         [R2Q4..R2Q5] = [D4..D5] & R2;
         [R4Q4..R4Q5] = [D4..D5] & R4;
     END;
END;

SYM GLB  B1  1  WIDTH, SYNC;
// Video data registers (sync width);
SIGTYPE [R3Q0..R3Q3] OUT;
     EQUATIONS
         [R3Q0..R3Q3] = [D0..D3] & R3;
     END;
END;
SYM GLB  B2  1  START, VIDEO 1;
// Video data registers (video start);
SIGTYPE [R4Q0..R4Q3] OUT;
     EQUATIONS
         [R4Q0..R4Q3] = [D0..D3] & R4;
     END;
END;
```

```
SYM GLB  B3  1  SYNC POSITION CNTR 1;
// Low four bits of sync position counter;
SIGTYPE [Q0..Q3] REG OUT;
      EQUATIONS
         [Q0..Q3].CLK = VCLK;
         Q0 = (Q0&!VEND)$$((R2Q0&VEND)#(!VEND&!SRST0));
         Q1 = (Q1&!VEND)$$((R2Q1&VEND)#(!Q0&!VEND&!SRST0));
         Q2 = (Q2&!VEND)$$((R2Q2&VEND)#(!Q0&!Q1&!VEND&!SRST0));
         Q3 = (Q3&!VEND)$$((R2Q3&VEND)#(!Q0&!Q1&!Q2&!VEND&!SRST0));
      END;
END;

SYM GLB  B4  1  SYNC POSITION CNTR 2;
// Upper two bits of sync position counter;
// and sync reset signal generation;
SIGTYPE [Q4..Q5] REG OUT;
SIGTYPE SRST0 OUT;
SIGTYPE SRST1 REG OUT;
      EQUATIONS
         Q4.CLK = VCLK;
         SRST1.CLK=VCLK;
         Q4 = (Q4&!VEND)$$((R2Q4&VEND)#(!Q0&!Q1&!Q2&!Q3&!VEND&!SRST0));
         Q5 = (Q5&!VEND)$$((R2Q5&VEND)#(!Q0&!Q1&!Q2&!Q3&Q4&!VEND&!SRST0));
         SRST0=!Q0&!Q1&!Q2&!Q3&!Q4&!Q5;
         SRST1.D=SRST0;
      END;
END;

SYM GLB  B5  1  VIDEO START CNTR 1;
// Low four bits of video start counter;
SIGTYPE [QQ0..QQ3] REG OUT;
      EQUATIONS
         [QQ0..QQ3].CLK = VCLK;
         QQ0 = (QQ0&!SSET)$$((R4Q0&SSET)#(!SSET&!START0));
         QQ1 = (QQ1&!SSET)$$((R4Q1&SSET)#(!QQ0&!SSET&!START0));
         QQ2 = (QQ2&!SSET)$$((R4Q2&SSET)#(!QQ0&!QQ1&!SSET&!START0));
         QQ3 = (QQ3&!SSET)$$((R4Q3&SSET)#(!QQ0&!QQ1&!QQ2&!SSET&!START0));
      END;
END;

SYM GLB  B6  1  VIDEO START CNTR 2;
// Upper four bits of video start counter and;
// START signal generation;
SIGTYPE [QQ4..QQ5] REG OUT;
SIGTYPE START0 OUT;
SIGTYPE START1 REG OUT;
      EQUATIONS
         QQ4.CLK = VCLK;
         START0=!QQ0&!QQ1&!QQ2&!QQ3&!QQ4&!QQ5;
         QQ4 = (QQ4&!SSET)$$((R4Q4&SSET)#(!QQ0&!QQ1&!QQ2&!QQ3&!SSET&!START0));
```

```
         QQ5 =
(QQ5&!SSET)$$((R4Q5&SSET)#(!QQ0&!QQ1&!QQ2&!QQ3&QQ4&!SSET&!START0));
         START1.D=START0
      END;
END;

SYM GLB  B7  1  SYNC WIDTH COUNTER;
// Sync width counter;
SIGTYPE [QQQ0..QQQ3] REG OUT;
      EQUATIONS
         [QQQ0..QQQ3].CLK = VCLK;
         QQQ0 = (QQQ0&!SRST)$$((R3Q0&SRST)#(!SRST&!SSET0));
         QQQ1 = (QQQ1&!SRST)$$((R3Q1&SRST)#(!QQQ0&!SRST&!SSET0));
         QQQ2 = (QQQ2&!SRST)$$((R3Q2&SRST)#(!QQQ0&!QQQ1&!SRST&!SSET0));
         QQQ3 = (QQQ3&!SRST)$$((R3Q3&SRST)#(!QQQ0&!QQQ1&!QQQ2&!SRST&!SSET0));
      END;
END;

SYM GLB  C1  1  MISC. LOGIC 1;
// Sync width counter SSet signal set-up;
// Sync reset signal generation, video START;
// signal generation;
SIGTYPE SSET0 OUT;
SIGTYPE SSET1 REG OUT;
SIGTYPE SRST OUT;
SIGTYPE START OUT;
      EQUATIONS
         SSET1.CLK=VCLK;
         SSET0=!QQQ0&!QQQ1&!QQQ2&!QQQ3;
         SSET1.D=SSET0;
         SRST=!SRST1&SRST0;
         START=!START1&START0;
      END;
END;

SYM GLB  C2  1  COMPARE, VIDEO END1;
// First eight bits of video end (VEND) comparator;
SIGTYPE VEND1 OUT;
     EQUATIONS
        VEND1 = !((R0Q0$VA0) # (R0Q1$VA1) # (R0Q2$VA2) # (R0Q3$VA3) # (R0Q4$VA4)
# (R0Q5$VA5) # (R0Q6$VA6) # (R0Q7$VA7));
     END;
END;

SYM GLB  C3  1  COMPARE, VIDEO END 2;
// Last two bits of video end (VEND) comparator;
// and VEND signal generation;
SIGTYPE VEND OUT;
     EQUATIONS
         VEND = !((R1Q0$VA8) # (R1Q1$VA9)) & VEND1;
     END;
END;
```

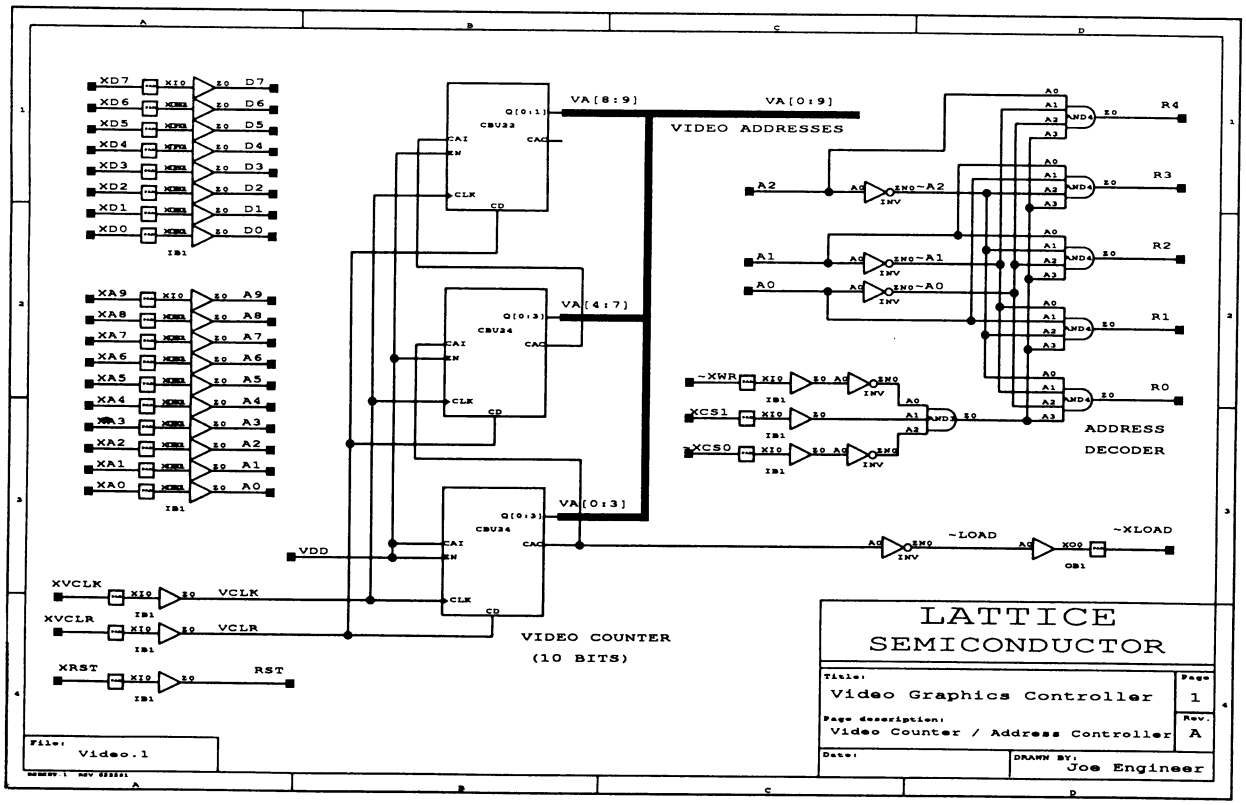
```
SYM GLB  C4  1  MEM ADDR MUX 1;
// Video memory address multiplexer bits;
// MA0-MA3;
SIGTYPE [MA0..MA3] OUT;
      EQUATIONS
         [MA0..MA3] = ([A0..A3] & !MUX) # ([VA0..VA3] & MUX);
      END;
END;

SYM GLB  C5  1  MEM ADDR MUX 2;
// Video memory address multiplexer bits;
// MA4-MA7;
SIGTYPE [MA4..MA7] OUT;
      EQUATIONS
         [MA4..MA7] = ([A4..A7] & !MUX) # ([VA4..VA7] & MUX);
      END;
END;

SYM GLB  C6  1  MEM ADDR MUX 3;
// Video memory address multiplexer bits;
// MA8 & MA9;
SIGTYPE [MA8,MA9] OUT;
      EQUATIONS
         [MA8,MA9] = ([A8,A9] & !MUX) # ([VA8,VA9] & MUX);
      END;
END;

SYM IOC  IO21  1  ;
// Read/Write control signal;
XPIN IO XWR       LOCK 48 ;
IB11 (WR,XWR);
END;

SYM IOC  IO20  1  ;
// Active high chip select;
XPIN IO XCS1         LOCK 3 ;
IB11 (CS1,XCS1);
END;

SYM IOC  IO19  1  ;
// Active low chip select;
XPIN IO XCS0         LOCK 4 ;
IB11 (CS0,XCS0);
END;

SYM IOC  IO0  1  ;
// Address bus A0;
XPIN IO XA0           LOCK 14 ;
IB11 (A0,XA0);
END;
SYM IOC  IO1  1  ;
```

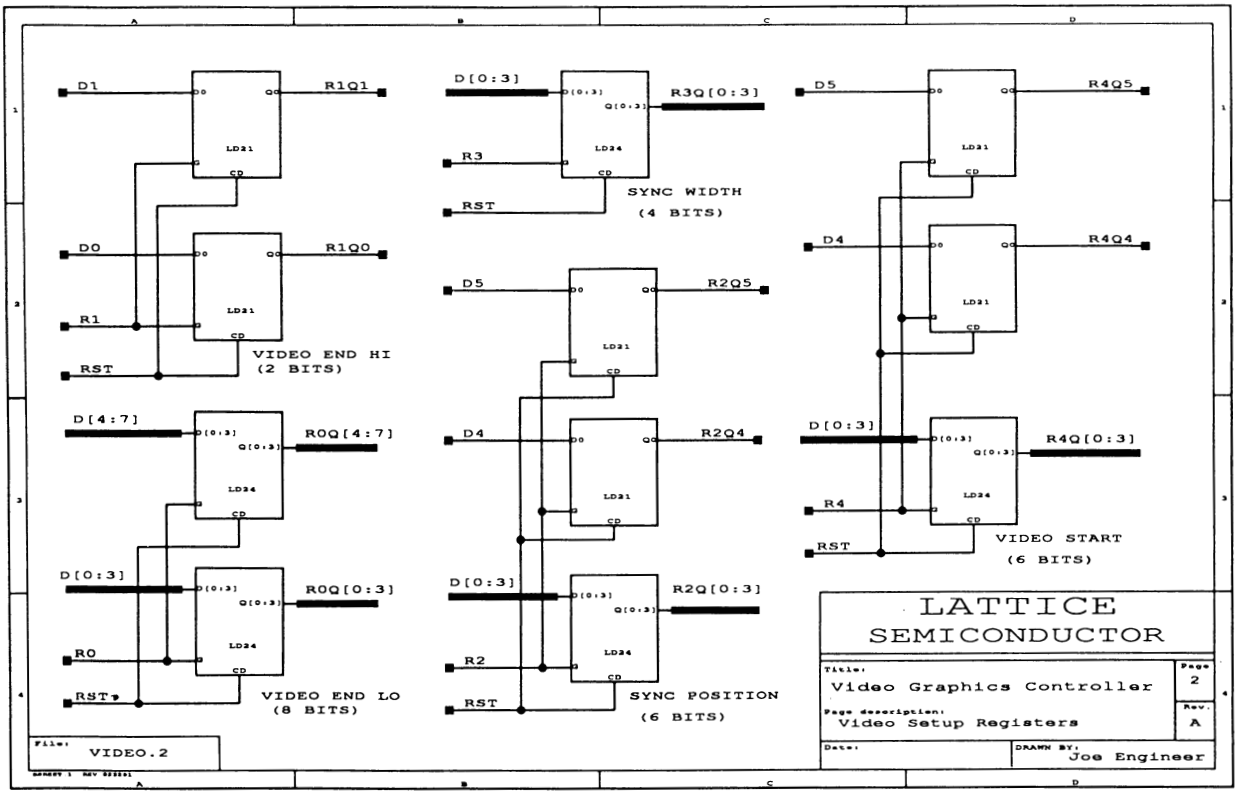
```
// Address bus A1;
XPIN IO XA1           LOCK 72 ;
IB11 (A1,XA1);
END;

SYM IOC  IO2  1  ;
// Address bus A2;
XPIN IO XA2           LOCK 15 ;
IB11 (A2,XA2);
END;

SYM IOC  IO3  1  ;
// Address bus A3;
XPIN IO XA3           LOCK 71 ;
IB11 (A3,XA3);
END;

SYM IOC  IO4  1  ;
// Address bus A4;
XPIN IO XA4       LOCK 16 ;
IB11 (A4,XA4);
END;

SYM IOC  IO5  1  ;
// Address bus A5;
XPIN IO XA5           LOCK 70 ;
IB11 (A5,XA5);
END;

SYM IOC  IO6  1  ;
// Address bus A6;
XPIN IO XA6           LOCK 17 ;
IB11 (A6,XA6);
END;

SYM IOC  IO7  1  ;
// Address bus A7;
XPIN IO XA7           LOCK 69 ;
IB11 (A7,XA7);
END;

SYM IOC  IO8  1  ;
// Address bus A8;
XPIN IO XA8           LOCK 18 ;
IB11 (A8,XA8);
END;

SYM IOC  IO9  1  ;
// Address bus A9;
XPIN IO XA9           LOCK 68 ;
IB11 (A9,XA9);
END;

SYM IOC  IO10  1  ;
// Data bus D0;
XPIN IO XD0           LOCK 26 ;
IB11 (D0,XD0);
END;

SYM IOC  IO11  1  ;
// Data bus D1;
XPIN IO XD1           LOCK 60 ;
IB11 (D1,XD1);
END;

SYM IOC  IO12  1  ;
// Data bus D2;
XPIN IO XD2           LOCK 27 ;
IB11 (D2,XD2);
END;

SYM IOC  IO13  1  ;
// Data bus D3;
XPIN IO XD3           LOCK 59 ;
IB11 (D3,XD3);
END;

SYM IOC  IO14  1  ;
// Data bus D4;
XPIN IO XD4           LOCK 28 ;
IB11 (D4,XD4);
END;

SYM IOC  IO15  1  ;
// Data bus D5;
XPIN IO XD5           LOCK 58 ;
IB11 (D5,XD5);
END;

SYM IOC  IO16  1  ;
// Data bus D6;
XPIN IO XD6           LOCK 29 ;
IB11 (D6,XD6);
END;

SYM IOC  IO17  1  ;
// Data bus D7;
XPIN IO XD7           LOCK 57 ;
IB11 (D7,XD7);
END;

SYM IOC  IO18  1  ;
// Video memory address multiplexer;
XPIN IO XMUX       LOCK 55 ;
IB11 (MUX,XMUX);
END;
```

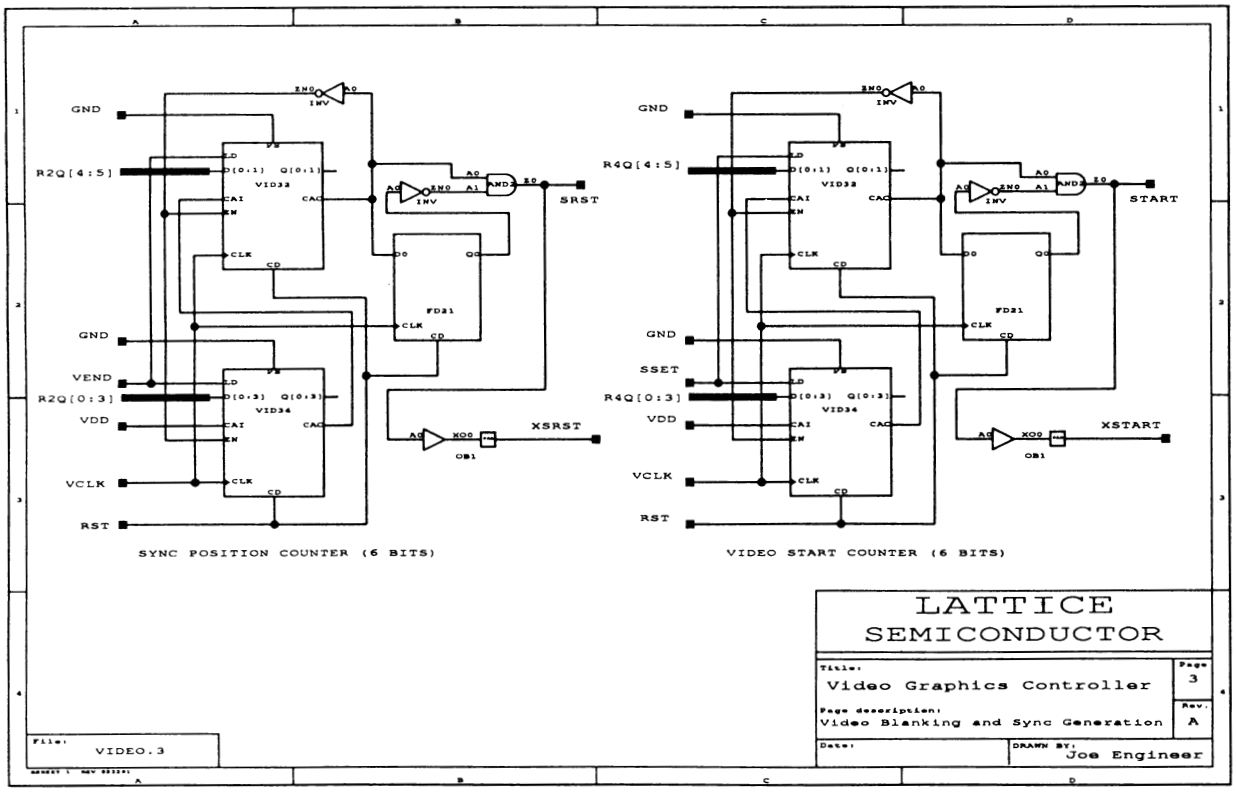
```
SYM IOC  IO25  1  ;
// Video memory address MA0;
XPIN IO XMA0           LOCK 49 ;
OB11 (XMA0,MA0);
END;

SYM IOC  IO26  1  ;
// Video memory address MA1;
XPIN IO XMA1           LOCK 79 ;
OB11 (XMA1,MA1);
END;

SYM IOC  IO27  1  ;
// Video memory address MA2;
XPIN IO XMA2           LOCK 50 ;
OB11 (XMA2,MA2);
END;

SYM IOC  IO28  1  ;
// Video memory address MA3;
XPIN IO XMA3           LOCK 78 ;
OB11 (XMA3,MA3);
END;

SYM IOC  IO29  1  ;
// Video memory address MA4;
XPIN IO XMA4           LOCK 51 ;
OB11 (XMA4,MA4);
END;

SYM IOC  IO30  1  ;
// Video memory address MA5;
XPIN IO XMA5           LOCK 77 ;
OB11 (XMA5,MA5);
END;

SYM IOC  IO31  1  ;
// Video memory address MA6;
XPIN IO XMA6           LOCK 52 ;
OB11 (XMA6,MA6);
END;

SYM IOC  IO32  1  ;
// Video memory address MA7;
XPIN IO XMA7           LOCK 76 ;
OB11 (XMA7,MA7);
END;

SYM IOC  IO33  1  ;
// Video memory address MA8;
XPIN IO XMA8           LOCK 53 ;
OB11 (XMA8,MA8);
END;
```

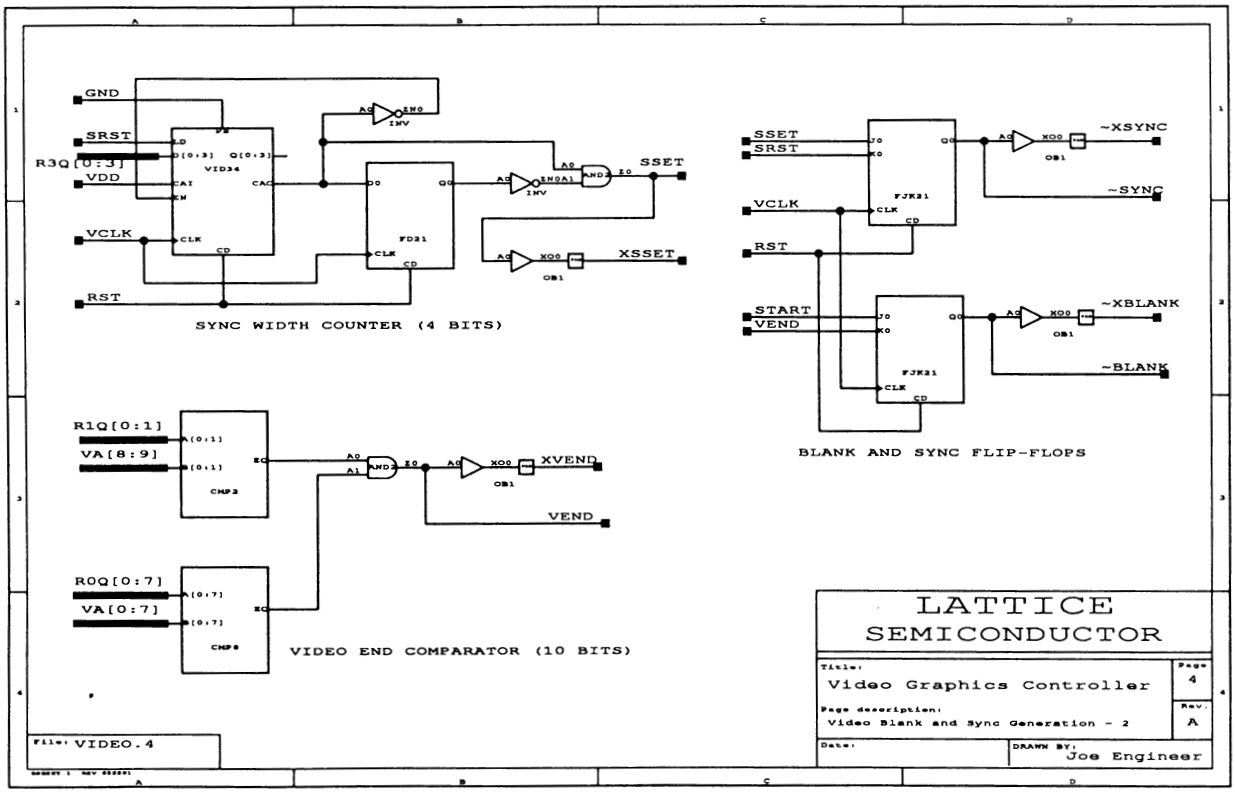
```
SYM IOC  IO34  1  ;
// Video memory address MA9;
XPIN IO XMA9           LOCK 75 ;
OB11 (XMA9,MA9);
END;

SYM IOC  IO35  1  ;
// Video end output signal;
XPIN IO XVEND       LOCK 45 ;
OB11 (XVEND,VEND);
END;

SYM IOC  IO36  1  ;
// Video sync reset signal;
XPIN IO XSRST       LOCK 46 ;
OB11 (XSRST,SRST);
END;

SYM IOC  IO37  1  ;
// Video sync width set output signal;
XPIN IO XSSET       LOCK 30 ;
OB11 (XSSET,SSET);
END;

SYM IOC  IO38  1  ;
// Video start output signal;
XPIN IO XSTART       LOCK 47 ;
OB11 (XSTART,START);
END;

SYM IOC  IO39  1  ;
// Video load output signal;
XPIN IO XLOAD     LOCK 32 ;
OB11 (!XLOAD,!LOAD);
END;

SYM IOC  IO40  1  ;
// Video Blanking output signal;
XPIN IO XBLANK     LOCK 36 ;
OB11 (XBLANK,BLANK);
END;

SYM IOC  IO41  1  ;
// Video sync output signal;
XPIN IO XSYNC       LOCK 31 ;
OB11 (XSYNC,SYNC);
END;

SYM IOC  Y0  1  ;
// Video clock input signal;
XPIN CLK XVCLK         LOCK 20 ;
IB11 (VCLK,XVCLK);
END;
```

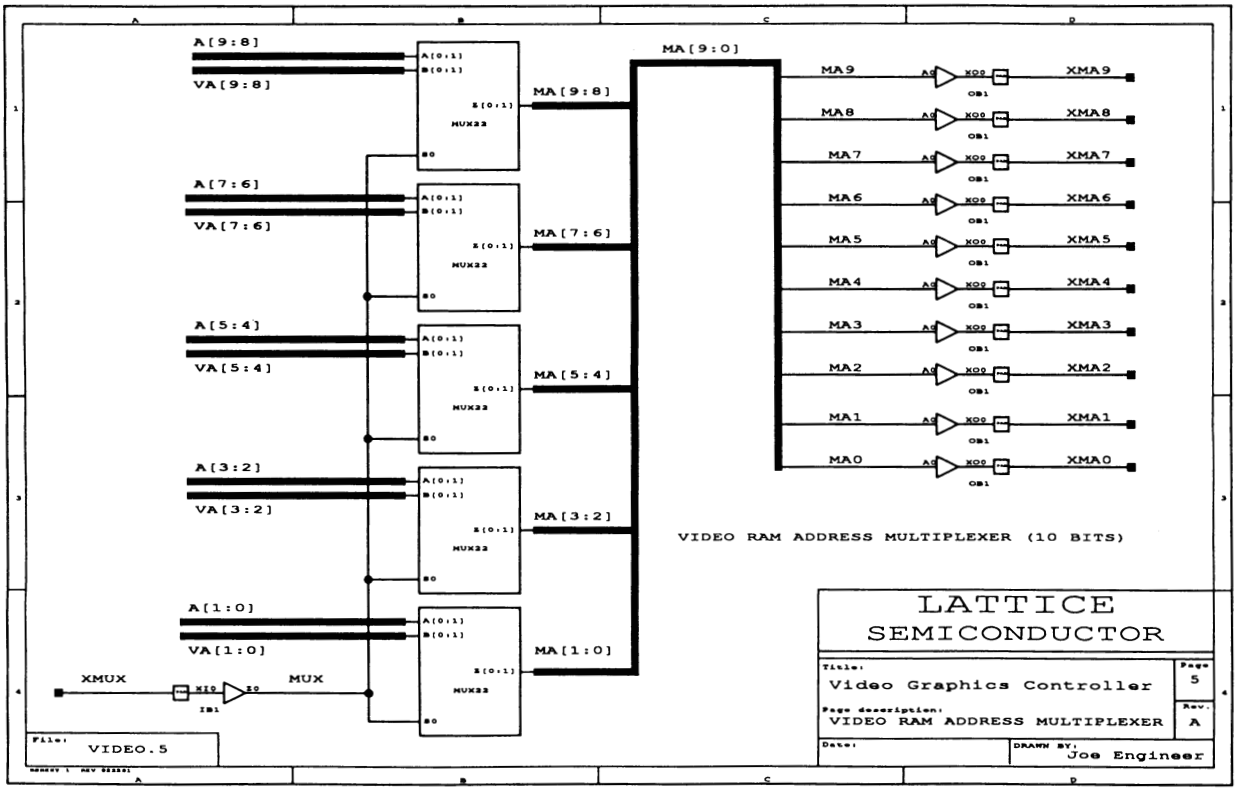
```
SYM IOC  I0  1  ;
// Video counter clear input;
XPIN I XVCLR       LOCK 25 ;
IB11 (VCLR,XVCLR);
END;
END;  //LDF DESIGNLDF
```

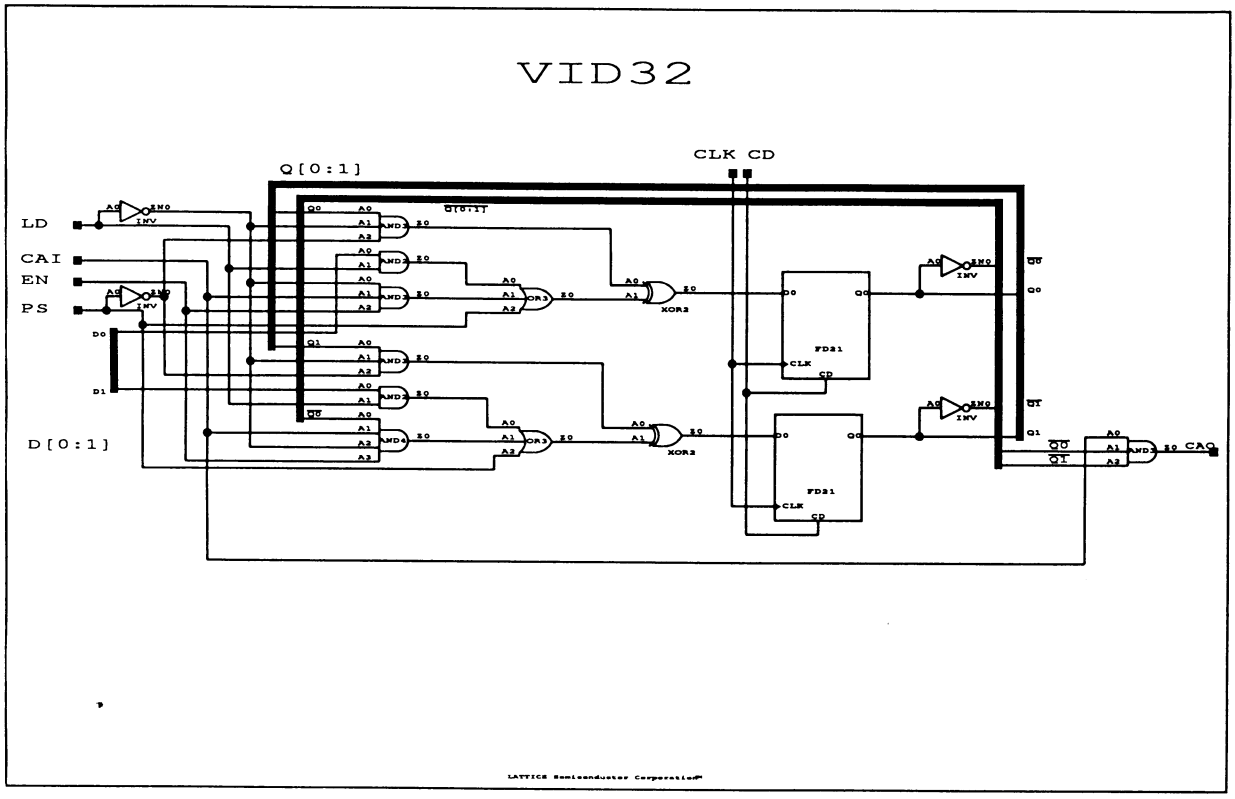


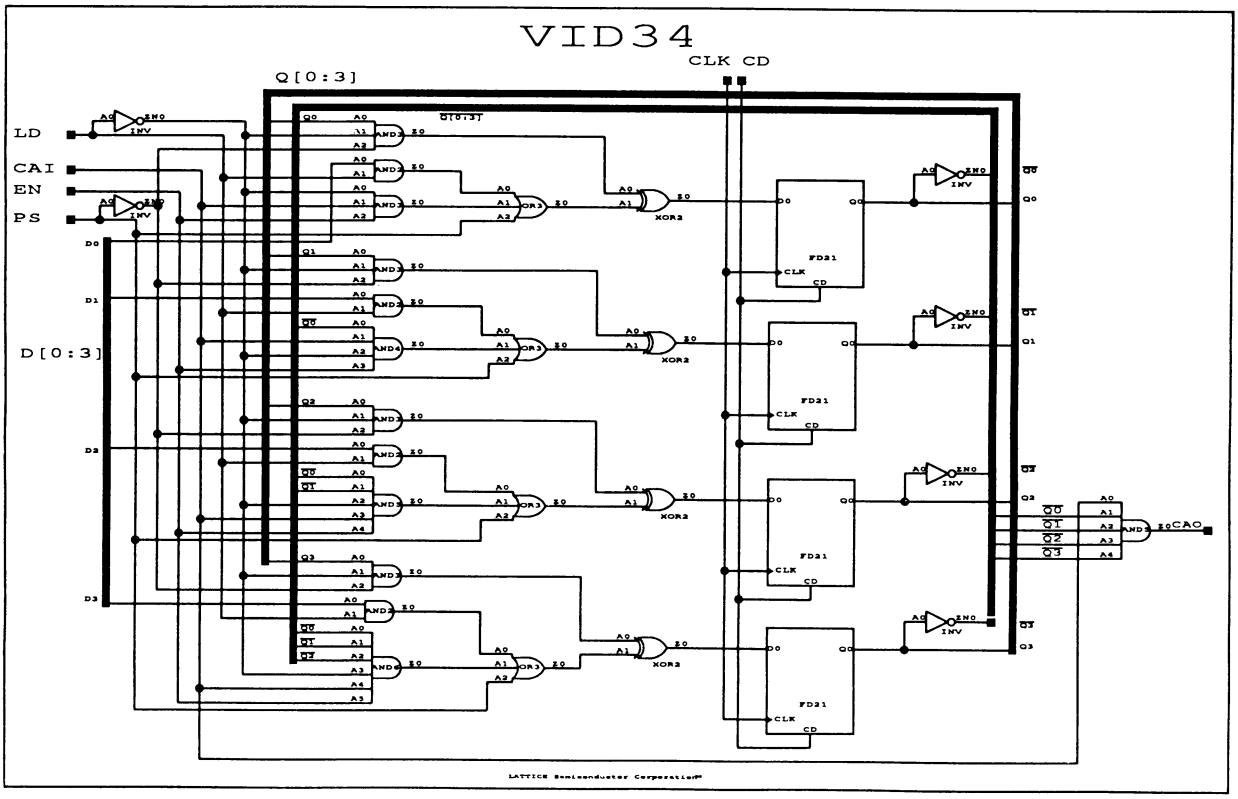

D1
R1Q1
D0
R1Q0
R1
RST
VIDEO END HI
(2 BITS)
D[4:7]
R0Q[4:7]
D[0:3]
R0Q[0:3]
R0
RST
VIDEO END LO
(8 BITS)
D[0:3]
R3Q[0:3]
R3
RST
SYNC WIDTH
(4 BITS)
D5
R2Q5
D4
R2Q4
D[0:3]
R2Q[0:3]
R2
RST
SYNC POSITION
(6 BITS)
D5
R4Q5
D4
R4Q4
D[0:3]
R4Q[0:3]
R4
RST
VIDEO START
(6 BITS)
LATTICE
SEMICONDUCTOR
Title:
Video Graphics Controller
Page description:
Video Setup Registers
Page
2
Rev.
A
Date:
DRAWN BY:
Joe Engineer
File:
VIDEO.2


GND
R2Q[4:5]
GND
VEND
R2Q[0:3]
VDD
VCLK
RST
SRST
XSRST
SYNC POSITION COUNTER (6 BITS)
GND
R4Q[4:5]
GND
SSET
R4Q[0:3]
VDD
VCLK
RST
START
XSTART
VIDEO START COUNTER (6 BITS)
LATTICE
SEMICONDUCTOR
Title:
Video Graphics Controller
Page description:
Video Blanking and Sync Generation
Page
3
Rev.
A
Date:
DRAWN BY:
Joe Engineer
File:
VIDEO.3


GND
SRST
R3Q[0:3]
VDD
VCLK
RST
SSET
XSSET
SYNC WIDTH COUNTER (4 BITS)
SSET
SRST
VCLK
RST
START
VEND
~XSYNC
~SYNC
~XBLANK
~BLANK
BLANK AND SYNC FLIP-FLOPS
R1Q[0:1]
VA[8:9]
XVEND
VEND
R0Q[0:7]
VA[0:7]
VIDEO END COMPARATOR (10 BITS)
LATTICE SEMICONDUCTOR
Title: Video Graphics Controller
Page description: Video Blank and Sync Generation - 2
Page 4
Rev. A
Date:
DRAWN BY: Joe Engineer
File: VIDEO.4


A[9:8]
VA[9:8]
A[7:6]
VA[7:6]
A[5:4]
VA[5:4]
A[3:2]
VA[3:2]
A[1:0]
VA[1:0]
XMUX
MUX
MA[9:8]
MA[7:6]
MA[5:4]
MA[3:2]
MA[1:0]
MA[9:0]
MA9
MA8
MA7
MA6
MA5
MA4
MA3
MA2
MA1
MA0
XMA9
XMA8
XMA7
XMA6
XMA5
XMA4
XMA3
XMA2
XMA1
XMA0
VIDEO RAM ADDRESS MULTIPLEXER (10 BITS)
LATTICE SEMICONDUCTOR
Title: Video Graphics Controller
Page description: VIDEO RAM ADDRESS MULTIPLEXER
Page 5
Rev. A
Date:
DRAWN BY: Joe Engineer
File: VIDEO.5


VID32
CLK CD
Q[0:1]
LD
CAI
EN
PS
D[0:1]
CAO
FD21
XOR2
INV


VID34
CLK CD
Q[0:3]
LD
CAI
EN
PS
D[0:3]
CAO
FD21
XOR2
INV

LATTICE SEMICONDUCTOR CORPORATION
5555 Northeast Moore Court
Hillsboro, Oregon 97124 U.S.A.
Tel.: (503) 681-0118
FAX: (503) 681-3037
http://www.latticesemi.com

November 1996