Avalon LCD Controller

Version 1.0, December 2004 Application Note 372

Introduction

The Avalon™ LCD Controller provides a flexible solution, which may be implemented in Altera® Cyclone™ II, Cyclone, Stratix® II, Stratix GX, or Stratix devices, and has the following features:

- 5 layers:
  - Layer 0—backdrop layer, RGB565
  - Layer 1—video layer, RGB565, no pixel alpha
  - Layer 2 to 4—three drawing layers—palette modes, 8-bit data, RGB666 + 6-bit pixel alpha
- Layer alpha for all layers
- Picture-in-picture support
- Timing generation display
- Avalon direct memory access (DMA) masters to read image(s) from frame buffer memory
- Avalon register slave for control and status

Functional Description

Figure 1 shows the Avalon LCD controller block diagram.

Figure 1. Block Diagram

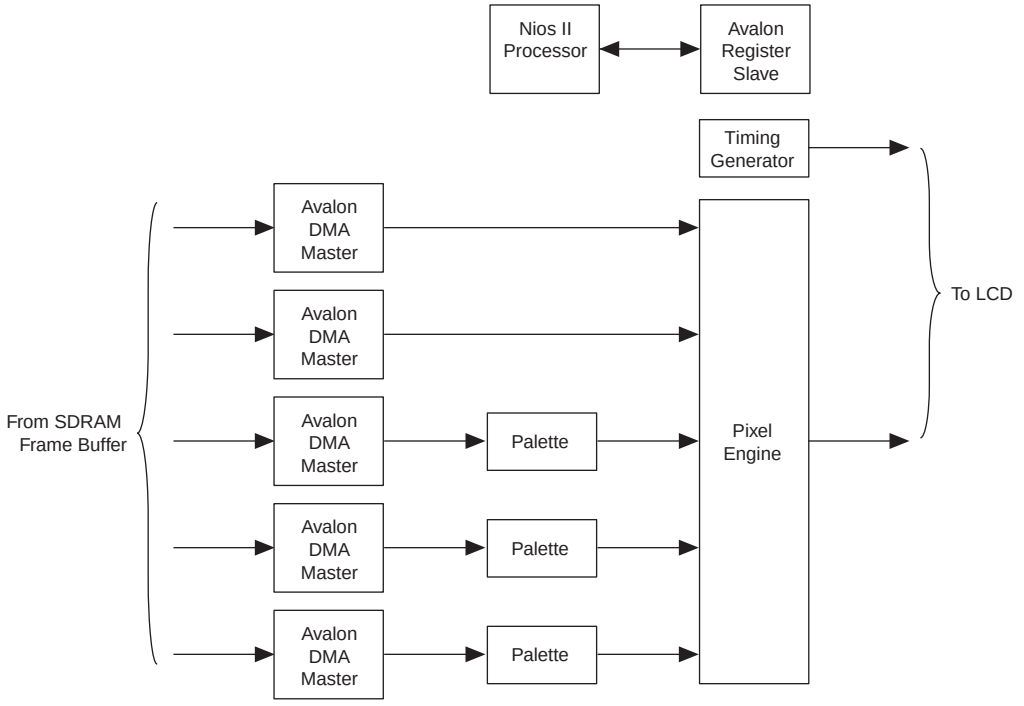


Figure 2 shows the signals.

Figure 2. Signals

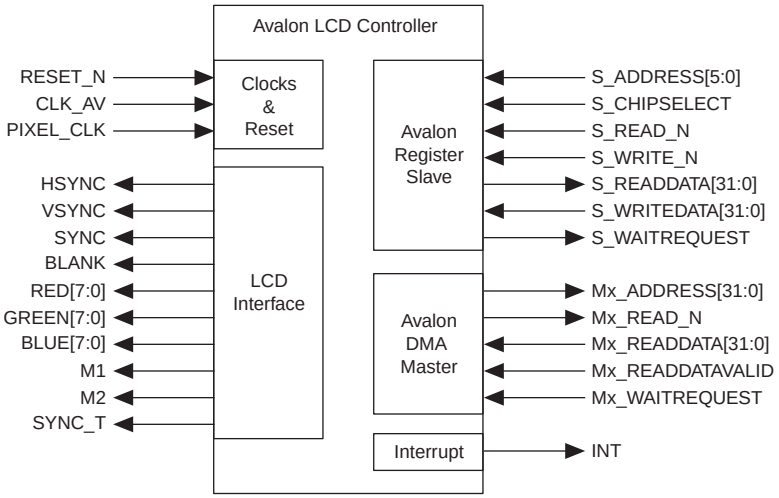


Table 1 shows the signals.

Table 1. Signals (Part 1 of 2)

Signal	Direction	Function
Clocks and Reset		
RESET_N	Input	Active low asynchronous reset.
CLK_AV	Input	Nios® II and Avalon clock.
PIXEL_CLK	Input	Pixel clock.
TFT Display Interface		
HSYNC	Output	Horizontal synchronization signal to display.
VSYNC	Output	Vertical synchronization signal to display.
SYNC	Output	Synchronization signal to display.
BLANK	Output	Blank signal to display.
RED[7:0]	Output	Red data output.
GREEN[7:0]	Output	Green data output.
BLUE[7:0]	Output	Blue data output.

Table 1. Signals (Part 2 of 2)

Signal	Direction	Function
M1	Output	Configuration signals for Lancelot VGA video controller, set to “1”.
M2	Output	
SYNC_T	Output	
Interrupts		
INT	Output	Interrupt request.
Avalon Register Slave		
S_ADDRESS[5:0]	Input	Register address.
S_CHIPSELECT	Input	Device select module input.
S_READ_N	Input	Avalon read enable.
S_WRITE_N	Input	Avalon write enable.
S_READDATA[31:0]	Output	Avalon read data.
S_WRITEDATA[31:0]	Input	Avalon write data.
S_WAITREQUEST[31:0]	Output	Avalon wait request.
Avalon DMA Master(s), *x*(0..4)		
Mx_ADDRESS[31:0]	Output	Avalon address to frame buffer.
Mx_READ_N	Output	Avalon write enable.
Mx_READDATA[31:0]	Input	Avalon read data from frame buffer.
Mx_READDATAVAILID	Input	Avalon read data valid signal.
Mx_WAITREQUEST	Input	Avalon wait request.

For information on the Lancelot VGA video controller, see **www.altera.com/products/devkits/partners/kit-lancelot.html**.

Clocks

The Avalon LCD controller has an Avalon clock and a pixel clock.

Avalon Clock (CLK_AV)

The Avalon DMA master and register slave must be driven from the same clock that the Nios II processor and Avalon bus fabric use.

Pixel Clock (PIXEL_CLK)

A clock of the required frequency for the display drives the timing generation, pixel engine, and is the clock that the controller uses to read the data from the DMA FIFO buffers. For VGA timings this clock is 25 MHz.

Timing Generation

The timing generation is handled by the **vga_timing.v** sub-module. This module generates the timing for the display (`vsync`, `hsync`, etc.), and the internal timing for the Avalon DMA masters read side. This module also supports the picture-in-picture functionality by controlling when the Avalon DMA master FIFO buffers are read from. Figures 3 and 4 show the `hsync` and `vsync` timing.

Figure 3. hsync Timing

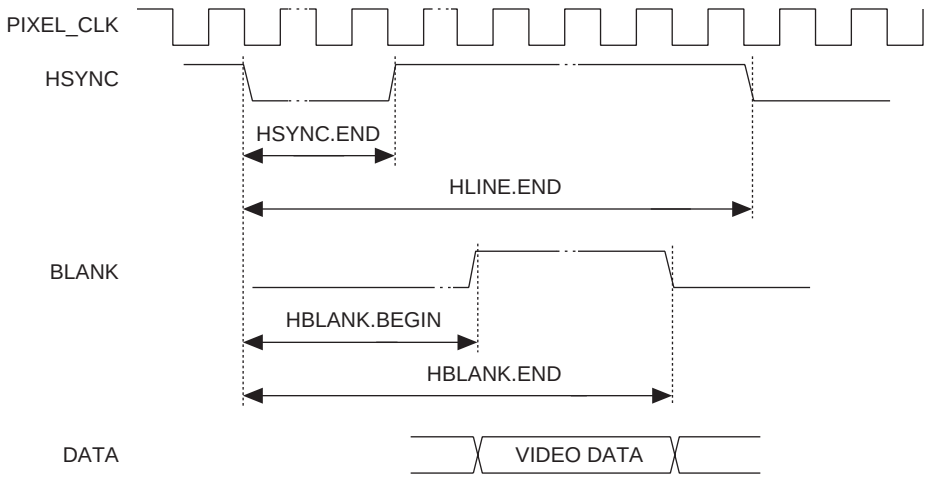
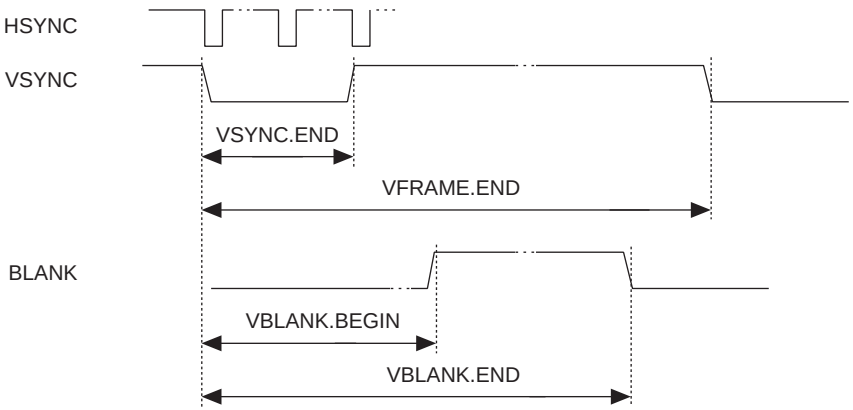


Figure 4. vsync Timing

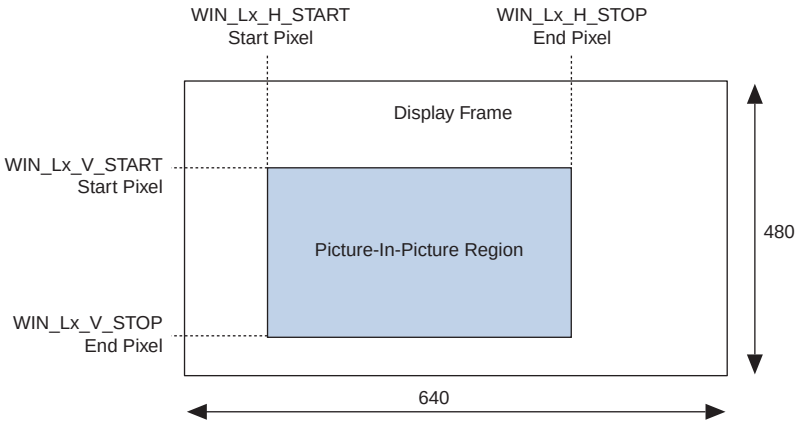


Different display resolutions can be catered for by setting different parameters within the following code:

```
// 640 by 480 VGA timing parameters
// horizontal timing parameters
parameter hsync_end = 96;
//sync pulse, 640x480 = 96

parameter hblank_begin = 144;
// = sync + back porch, 640x480 = 144

parameter hblank_end = 784;
// = sync + back porch + h_resolution, 640x480 = 784

parameter hline_end = 800;
// = sync + back porch + h_resolution + front porch, 640x480 = 800

// vertical timing parameters
parameter vsync_end = 2;
//sync pulse, 640x480 = 2

parameter vblank_begin = 33;
// = sync + back porch, 640x480 = 33

parameter vblank_end = 513;
// = sync + back porch + v_resolution, 640x480 = 513

parameter vframe_end = 524;
// = sync + back porch + v_resolution + front porch, 640x480 = 524
```

The timing engine supports picture-in-picture (PinP) for layers1 to 4 and is controlled by setting the appropriate start and end points (vertical and horizontal) for the PinP picture (see Figure 5).

Figure 5. PinP Example, Layer 1, VGA Display



Pixel Engine

The pixel engine blends the layers together according to the following equations:

Equation 1

$$Pixel\_Output = (Pixel_{L4} \times Alpha_{L4}) + (Pixel_{L3\_L2\_L1\_L0} \times (1 - Alpha_{L4})$$

Equation 2

$$Pixel_{L3\_L2\_L1\_L0} = (Pixel_{L3} \times Alpha_{L3}) + (Pixel_{L2\_L1\_L0} \times (1 - Alpha_{L3}))$$

Equation 3

$$Pixel_{L2\_L1\_L0} = (Pixel_{L2} \times Alpha_{L2}) + (Pixel_{L1\_L0} \times (1 - Alpha_{L2}))$$

Equation 4

$$Pixel_{L1\_L0} = (Pixel_{L1} \times Alpha_{L1}) + (Pixel_{L0} \times (1 - Alpha_{L1}))$$

Where the $Alpha_{Lx}$ is given by:

$$AlphaLx = Pixel\_Alpha_{Lx} \times Layer\_Alpha_{Lx}$$

Where $Pixel\_Alpha_{Lx}$ and $Layer\_Alpha_{Lx}$ are between 0 and 1.

The design is pipelined so it achieves the required clock frequency, hence there is different latency associated with each of the layers. This latency is accounted for in the generation of the read signals from the timing controller to the Avalon DMA masters.

Palette(s)

Layers 2 to 4 use a palette. For these layers all pixels are saved as 8-bits (byte) within the frame buffer. The LCD controller provides a dedicated look-up table (palette) for each layer with 256 entries. The look-up table outputs RGB666 with an additional 6 bits of pixel alpha.

A single M4K RAM block is used for each palette.

Avalon DMA Master

There are 5 Avalon DMA masters supporting the LCD controller—one for each layer. They are read only masters, which read from the frame buffer(s) in system memory (usually some form of SDRAM).

The master performs bursts of 32 words. An exception to this rule is at the end of the frame where the final burst read can be less than 32 words, depending upon the frame length.

The DMA master assumes the use of a linear frame buffer in which all lines are contiguous.

The organization of pixels in the frame buffer must be such that there is a whole number of 32-bit words per frame.

The start address and length of the DMA from the frame buffer is programmed via the Avalon register slave interface.

A single M4K RAM block is used for each Avalon DMA master.

Master Interrupt

The DMA master may be configured to generate an interrupt request at the end of each video frame written to memory.

Avalon Register Slave

The Avalon register slave interface gives access to control and status registers to configure the operation of the video input module.

Table 2 shows the registers. All registers are 32 bit aligned to a word boundary. Unused bits should be written as zero.

Table 2. Registers (Part 1 of 2)

Address (h)	Access	Mnemonic	Name
00	W	CR	Control register.
00	R	SR	Status register.
04	RW	const_alpha_lay1	Layer 1 constant alpha register.
08	RW	const_alpha_lay2	Layer 2 constant alpha register.
0C	RW	const_alpha_lay3	Layer 3 constant alpha register.
10	RW	const_alpha_lay4	Layer 4 constant alpha register.
14	RW	background	Background color register.
18	RW	win_l1_h_start	Layer 1 horizontal start position.
1C	RW	win_l1_h_stop	Layer 1 horizontal stop position.
20	RW	win_l1_v_start	Layer 1 vertical start position.
24	RW	win_l1_v_stop	Layer 1 vertical stop position.
28	RW	win_l2_h_start	Layer 2 horizontal start position.
2C	RW	win_l2_h_stop	Layer 2 horizontal stop position.
30	RW	win_l2_v_start	Layer 2 vertical start position.
34	RW	win_l2_v_stop	Layer 2 vertical stop position.
38	RW	win_l3_h_start	Layer 3 horizontal start position.
3C	RW	win_l3_h_stop	Layer 3 horizontal stop position.
40	RW	win_l3_v_start	Layer 3 vertical start position.
44	RW	win_l3_v_stop	Layer 3 vertical stop position.
48	RW	win_l4_h_start	Layer 4 horizontal start position.
4C	RW	win_l4_h_stop	Layer 4 horizontal stop position.
50	RW	win_l4_v_start	Layer 4 vertical start position.
54	RW	win_l4_v_stop	Layer 4 vertical stop position.
58	RW	layer_0_dma_start_addr	Layer 0 DMA start address.
5C	RW	layer_0_dma_length	Layer 0 DMA length.
60	RW	layer_1_dma_start_addr	Layer 1 DMA start address.
64	RW	layer_1_dma_length	Layer 1 DMA length.
68	RW	layer_2_dma_start_addr	Layer 2 DMA start address.
6C	RW	layer_2_dma_length	Layer 2 DMA length.
70	RW	layer_3_dma_start_addr	Layer 3 DMA start address.

Table 2. Registers (Part 2 of 2)			
Address (h)	**Access**	**Mnemonic**	**Name**
74	RW	layer_3_dma_length	Layer 3 DMA length.
78	RW	layer_4_dma_start_addr	Layer 4 DMA start address.
7C	RW	layer_4_dma_length	Layer 4 DMA length.
80	RW	dma_control_reg	DMA control register.
84	RW	int_enable_reg	Interrupt enable register.
88	RW	int_clear_reg	Interrupt clear register.

Control Register (CR)

Table 3 shows the control register format.

Table 3. Control Register Format		
Bit	**Mnemonic**	**Description**
0	L0_ON	0 = disables layer 0 data output from associated Avalon DMA FIFO buffer. 1 = enables layer 0 data output from associated Avalon DMA FIFO buffer.
1	L1_ON	0 = disables layer 1 data output from associated Avalon DMA FIFO buffer. 1 = enables layer 1 data output from associated Avalon DMA FIFO buffer.
2	L2_ON	0 = disables layer 2 data output from associated Avalon DMA FIFO buffer. 1 = enables layer 2 data output from associated Avalon DMA FIFO buffer.
3	L3_ON	0 = disables layer 3 data output from associated Avalon DMA FIFO buffer. 1 = enables layer 3 data output from associated Avalon DMA FIFO buffer.
4	L4_ON	0 = disables layer 4 data output from associated Avalon DMA FIFO buffer. 1= enables layer 4 data output from associated Avalon DMA FIFO buffer.
31:5	0	–

Status Register (SR)

Table 4 shows the status register format.

Table 4. Status Register Format

Bit	Mnemonic	Description
0	L0_ON	Returns current state of L0_ON bit in control register.
1	L1_ON	Returns current state of L1_ON bit in control register.
2	L2_ON	Returns current state of L2_ON bit in control register.
3	L3_ON	Returns current state of L3_ON bit in control register.
4	L4_ON	Returns current state of L4_ON bit in control register.
7:5	0	–
8	VBLANK	Returns current state of VBLANK.
31:9	0	–

Layer 1 Constant Alpha Register (const_alpha_lay1)

Table 5 shows the layer 1 constant alpha register format.

Table 5. Layer 1 Constant Alpha Register Format

Bit	Mnemonic	Description
5:0	CONST_ALPHA_LAY1	Specifies the layer alpha to be applied to layer 1.
31:6	0	–

Layer 2 Constant Alpha Register (const_alpha_lay2)

Table 6 shows the layer 2 constant alpha register format.

Table 6. Layer 2 Constant Alpha Register Format

Bit	Mnemonic	Description
5:0	CONST_ALPHA_LAY2	Specifies the layer alpha to be applied to layer 2.
31:6	0	–

Layer 3 Constant Alpha Register (const_alpha_lay3)

Table 7 shows the layer 3 constant alpha register format.

Table 7. Layer 3 Constant Alpha Register Format

Bit	Mnemonic	Description
5:0	CONST_ALPHA_LAY3	Specifies the layer alpha to be applied to layer 3.
31:6	0	–

Layer 4 Constant Alpha Register (const_alpha_lay4)

Table 8 shows the layer 4 constant alpha register format.

Table 8. Layer 4 Constant Alpha Register Format

Bit	Mnemonic	Description
5:0	CONST_ALPHA_LAY4	Specifies the layer alpha to be applied to layer 4.
31:6	0	–

Background Register (background)

Table 9 shows the background register format.

Table 9. Background Register Format

Bit	Mnemonic	Description
15:0	background	Color applied as a background if no other layers enabled (Reset value of 0hfa88).
31:16	0	–

Layer 1 Horizontal Start Position (win_l1_h_start)

Table 10 shows the layer 1 horizontal start position register format.

Table 10. Layer 1 Horizontal Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l1_h_start	Horizontal start position for layer 1.
31:12	0	–

Layer 1 Horizontal Stop Position (win_l1_h_stop)

Table 11 shows the layer 1 horizontal stop position register format.

Table 11. Layer 1 Horizontal Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l1_h_stop	Horizontal stop position for layer 1.
31:12	0	–

Layer 1 Vertical Start Position (win_l1_v_start)

Table 12 shows the layer 1 vertical start position register format.

Table 12. Layer 1 Vertical Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l1_v_start	Vertical start position for layer 1.
31:12	0	–

Layer 1 Vertical Stop Position (win_l1_v_stop)

Table 13 shows the layer 1 vertical stop position register format.

Table 13. Layer 1 Vertical Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l1_v_stop	Vertical stop position for layer 1.
31:12	0	–

Layer 2 Horizontal Start Position (win_l2_h_start)

Table 14 shows the layer 2 horizontal start position register format.

Table 14. Layer 2 Horizontal Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l2_h_start	Horizontal start position for layer 2.
31:12	0	–

Layer 2 Horizontal Stop Position (win_l2_h_stop)

Table 15 shows the layer 2 horizontal stop position register format.

Table 15. Layer 2 Horizontal Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l2_h_stop	Horizontal stop position for layer 2.
31:12	0	–

Layer 2 Vertical Start Position (win_l2_v_start)

Table 16 shows the layer 2 vertical start position register format.

Table 16. Layer 2 Vertical Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l2_v_start	Vertical start position for layer 2.
31:12	0	–

Layer 2 Vertical Stop Position (win_l2_v_stop)

Table 17 shows the layer 1 vertical stop position register format.

Table 17. Layer 2 Vertical Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l2_v_stop	Vertical stop position for layer 2.
31:12	0	–

Layer 3 Horizontal Start Position (win_l3_h_start)

Table 18 shows the layer 3 horizontal start position register format.

Table 18. Layer 3 Horizontal Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l3_h_start	Horizontal start position for layer 3.
31:12	0	–

Layer 3 Horizontal Stop Position (win_l3_h_stop)

Table 19 shows the layer 3 horizontal stop position register format.

Table 19. Layer 3 Horizontal Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l3_h_stop	Horizontal stop position for layer 3.
31:12	0	–

Layer 3 Vertical Start Position (win_l3_v_start)

Table 20 shows the layer 3 vertical start position register format.

Table 20. Layer 3 Vertical Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l3_v_start	Vertical start position for layer 3.
31:12	0	–

Layer 3 Vertical Stop Position (win_l3_v_stop)

Table 21 shows the layer 3 vertical stop position register format.

Table 21. Layer 3 Vertical Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l3_v_stop	Vertical stop position for layer 3.
31:12	0	–

Layer 4 Horizontal Start Position (win_l4_h_start)

Table 22 shows the layer 4 horizontal start position register format.

Table 22. Layer 4 Horizontal Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l4_h_start	Horizontal start position for layer 4.
31:12	0	–

Layer 4 Horizontal Stop Position (win_l4_h_stop)

Table 23 shows the layer 4 horizontal stop position register format.

Table 23. Layer 4 Horizontal Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l4_h_stop	Horizontal stop position for layer 4.
31:12	0	–

Layer 4 Vertical Start Position (win_l4_v_start)

Table 24 shows the layer 4 vertical start position register format.

Table 24. Layer 4 Vertical Start Position Register Format

Bit	Mnemonic	Description
11:0	win_l4_v_start	Vertical start position for layer 4.
31:12	0	–

Layer 4 Vertical Stop Position (win_l4_v_stop)

Table 25 shows the layer 4 vertical stop position register format.

Table 25. Layer 4 Vertical Stop Position Register Format

Bit	Mnemonic	Description
11:0	win_l4_v_stop	Vertical stop position for layer 4.
31:12	0	–

Layer 0 DMA Start Address (layer_0_dma_start_addr)

Table 26 shows the layer 0 DMA start address register format.

Table 26. Layer 0 DMA Start Address Register Format

Bit	Mnemonic	Description
31:0	layer_0_dma_start_addr	Start address of layer 0 frame (32-bit word aligned).

Layer 0 DMA Length (layer_0_dma_length)

Table 27 shows the layer 0 DMA length register format.

Table 27. Layer 0 DMA Length Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_0_dma_length	Length of frame stored in frame buffer (whole number of 32-bit words).

Layer 1 DMA Start Address (layer_1_dma_start_addr)

Table 28 shows the layer 1 DMA start address register format.

Table 28. Layer 1 DMA Start Address Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_1_dma_start_addr	Start address of layer 1 frame (32-bit word aligned).

Layer 1 DMA Length (layer_1_dma_length)

Table 29 shows the layer 1 DMA length register format.

Table 29. Layer 1 DMA Length Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_1_dma_length	Length of frame stored in frame buffer (whole number of 32-bit words).

Layer 2 DMA Start Address (layer_2_dma_start_addr)

Table 30 shows the layer 2 DMA start address register format.

Table 30. Layer 2 DMA Start Address Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_2_dma_start_addr	Start address of layer 2 frame (32-bit word aligned).

Layer 2 DMA Length (layer_2_dma_length)

Table 31 shows the layer 2 DMA length register format.

Table 31. Layer 2 DMA Length Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_2_dma_length	Length of frame stored in frame buffer (whole number of 32-bit words).

Layer 3 DMA Start Address (layer_3_dma_start_addr)

Table 32 shows the layer 3 DMA start address register format.

Table 32. Layer 3 DMA Start Address Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_3_dma_start_addr	Start address of layer 3 frame (32-bit word aligned).

Layer 3 DMA Length (layer_3_dma_length)

Table 33 shows the layer 3 DMA length register format.

Table 33. Layer 3 DMA Length Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_3_dma_length	Length of frame stored in frame buffer (whole number of 32-bit words).

Layer 4 DMA Start Address (layer_4_dma_start_addr)

Table 34 shows the layer 4 DMA start address register format.

Table 34. Layer 4 DMA Start Address Register Format		
Bit	**Mnemonic**	**Description**
31:0	layer_4_dma_start_addr	Start address of layer 4 frame (32-bit word aligned).

Layer 4 DMA Length (layer_4_dma_length)

Table 35 shows the layer 4 DMA length register format.

Table 35. Layer 4 DMA Length Register Format

Bit	Mnemonic	Description
31:0	layer_4_dma_length	Length of frame stored in frame buffer (whole number of 32-bit words).

DMA Control Register (dma_control_reg)

Table 36 shows the DMA control register format.

Table 36. DMA Control Register Format

Bit	Mnemonic	Description
0	dma_layer_0_on	0 = disables DMA for layer 0 from frame buffer. 1 = enables DMA for layer 0 from frame buffer.
1	dma_layer_1_on	0 = disables DMA for layer 1 from frame buffer. 1 = enables DMA for layer 1 from frame buffer.
2	dma_layer_2_on	0 = disables DMA for layer 2 from frame buffer. 1 = enables DMA for layer 2 from frame buffer.
3	dma_layer_3_on	0 = disables DMA for layer 3 from frame buffer. 1 = enables DMA for layer 3 from frame buffer.
4	dma_layer_4_on	0 = disables DMA for layer 4 from frame buffer. 1 = enables DMA for layer 4 from frame buffer.
31:5	0	–

Interrupt Enable Register (int_enable_reg)

Table 37 shows the interrupt enable register format.

Table 37. Interrupt Enable Register Format

Bit	Mnemonic	Description
0	int_enable	0 = disables interrupt generation. 1 = enables interrupt generation at end of frame.
31:1	0	–

Interrupt Clear Register (int_clear_reg)

Table 38 shows the interrupt clear register format.

Table 38. Interrupt Clear Register Format

Bit	Mnemonic	Description
0	int_clear	1 = clears interrupt source.
31:1	0	–

Resource Usage

The Avalon TFT Controller module consumes approximately 4,400 logic cells when implemented in a Cyclone device. This usage can be reduced by optimizing the pixel engine, especially appropriate for lower pixel clock rates (smaller display resolutions). A Stratix or Cyclone II device implementation uses fewer logic cells, if you employ hardware multipliers in the pixel engine.

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
www.altera.com
Applications Hotline:
(800) 800-EPLD
Literature Services:
lit_req@altera.com