

GFX_SpriteController

Overview

This is a sprite or hardware cursor display controller.

The display controller interfaces as a slave device with the cpu via a 64-bit bus. It interfaces to the memory system as a bus master using a 64-bit bus.

Features

- up to 64 sprites or hardware cursors
- 256 entry color palette
- each sprite may contain three different colors plus transparent
- sprites may be linked together to increase color selection to 16
- sprite may be up to 32 pixels horizontally and 1024 scan lines vertically
- sprite display synchronized to externally supplied hsync, vsync
- bitmap image fetch triggered on horizontal sync.

Documentation Notes

Some registers are illustrated as if they were 32 bits in size for easier readability. The lower half of a register is presented first, followed by the upper half. These are given addresses that are four bytes apart. However, registers in the circuit are really 64-bit.

Clocks

The video dot clock rate is 40MHz for an 800x600 VGA display. The bus clock for interface to the cpu is independent of the video clock.

Cursor / Sprite Control

The controller has 64 hardware cursors or sprites. The cursors may be up to 32 pixels wide and 1024 pixels high. The cursors all share a common 256 entry color palette. However, each cursor may have its own set of colors. A sprite by itself may use three different colors simultaneously plus transparency. Note that the cursors are effectively 32 pixels wide, but pixels may be set to be transparent, so the apparent size of the cursor looks smaller.

Cursor Color Palette

The cursor color palette has 256 entries each of which is a 64-bit vector including additional attributes besides just the color. Attributes include alpha blending, reverse video and flashing.

The registers are organized into groups of four, a group of four registers present for each of the sprites. Thus, each sprite can have a different set of colors from other sprites. Only three of the four registers are used. (The color code 00 is transparent). The registers are further organized into sixteen groups of sixteen for linked sprites. A set of 16 color registers is used when sprites are linked together. The first group establishes a set of colors which are shared between sprite 0 and 1. The second group is shared between sprites 2 and 3, and so on. Note that color palette entry #0 is never used.

\$000	~ ₂₆	i	f	rate ₄	Alpha ₈	RGB888 ₂₄	Color0	
\$008 to \$7F8	255 more registers							

Alpha₈ determines how much of the cursor color is present in the output. A value of zero causes the cursor to be fully output. A value of all ones will make the cursor invisible. Alpha blending can be used to create shadows by selecting a cursor color of black then setting the alpha register to a none-zero value.

I - The I flag indicates to reverse the video output. The color under the cursor is xor'd with -1.

f – indicates to flash the cursor. The cursor will flash at a rate determined from the rate₄ field.

Cursor Link Register

The cursor link register indicates cursors which are linked to the next cursor to increase the apparent number of colors available to sixteen rather than four. Each bit in the register specifies the link state for the corresponding sprite. Linked cursors must have their coordinates maintained with the same values.

\$C08	Link _{63..0}
-------	-----------------------

Cursor Enable Register

The cursor enable register controls which sprites are visible on the screen. Bits in the register enable the sprite display when set to a one for the sprite corresponding to the bit number.

\$C00	Enable _{63..0}
-------	-------------------------




Collision Register

The collision register indicates which sprites are colliding with other sprites. Each bit in the register corresponds to a sprite. If the bit is set then the sprite has collided with another sprite. The bits will remain set in the register until the register is updated.

\$C10	SprCollision _{63..0}
-------	-------------------------------

Cursor Control Registers

The control register layout for all cursors is identical. The layout is shown only for the first cursor, cursor #0. Note that the count and position registers are 16-bit addressable. Any or all of the 16-bit fields may be updated.

\$800	Address _{31..6}	0 ₆	bitmap address low
\$804	(Reserved) Address _{63..32}		bitmap address high
\$808	 vpos ₁₂	 hpos ₁₂	vertical / horizontal position
\$80C	MCnt ₁₆	 zpos ₈	total number of pixels
\$810 to \$81C	Cursor #1 Registers		
...	...		
\$BE0 to \$BFC	Cursor #63 Registers		

Cursor Bitmap Address

This register contains the address of the cursor's bitmap in the core's memory. The cursor bitmap occupies contiguous words of memory. The memory address must be 64-byte aligned. The amount of memory required is determined by the MCnt field for the cursor. Each raster line of the cursor is composed of one sixty-four-bit value to allow bitmaps up to 32 pixels in width to be defined. The amount of memory required is one word, which is a fixed amount. Even if the sprite is only five pixels wide, a whole word of memory per scanline is still required.

- The system memory controller reads and caches 256 pixels at a time (8x64-bit words) as the performance of the memory system is increased substantially by reading multiple words in a single transfer. Most of the time pixel data is fetched from a read cache and not main memory. This is an aspect of the system's memory not the sprite controller. As far as the sprite controller is concerned it just wants to see data for a 64-bit transfer. Read caching is not required by the sprite controller. Another implementation of the system may handle sprite accesses differently.

MCnt

The size register controls the visible size of the sprite. Sprites may be up to 32 pixels in size horizontally, and up to 1024 scan lines vertically. The value placed in the register should be one less than the total count of pixels to display. The count should be a multiple of 32 pixels then minus one. For example, for a 32hx30v sprite the count would be 960-1 = 959.

Horizontal and Vertical Position

The horizontal and vertical position are relative to the sync position which is position (0, 0).

Z-Order

The z-order (zpos) register controls the appearance priority of the sprite compared to other graphics on-screen. If the sprite's z-order is less than the z-order of the current pixel it will appear in front of the pixel. Otherwise it will be hidden by the pixel.

Port Signals

Name	Width	I/O	
rst_i	1	i	This active high signal resets the core and WISHBONE bus interfaces
Slave Signals			
clk_i	1	i	Clock signal for slave peripheral interface (typically the cpu clock)
cs_i	1	i	circuit select
cyc_i	1	i	cycle is valid
stb_i	1	i	data transfer in progress
ack_o	1	o	data transfer acknowledge
sel_i	8	i	byte lane selects
we_i	1	i	write enable to register set
adr_i	12	i	addresses the registers of the core
dat_i	64	i	data input for registers
dat_o	64	o	data output of registers
Master Signals (read only)			
m_clk_i	1	i	clock signal for bus master interface (typically the memory clock)
m_cyc_o	1	o	cycle is valid
m_stb_o	1	o	data transfer is taking place
m_ack_i	1	i	data transfer acknowledge
m_adr_o	32	o	Memory address for bitmap data read
m_dat_i	64	i	data input from bitmap memory
m_spriteno_o	6	o	sprite number to aid in memory caching (acts like an ASID)
Video Port			
dot_clk_i	1	i	This is the video clock input (40 MHz)
hsync_i	1	o	horizontal sync signal
vsync_i	1	o	vertical sync signal
zrgb_i	32	o	color output video data in ZRGB (8,8,8) format