5.4 Graphics Pipeline

The graphics pipeline of the GX1 processor contains a 2D graphics accelerator. This hardware accelerator has a BitBLT/vector engine which dramatically improves graphics performance when rendering and moving graphical objects. Overall operating system performance is improved as well. The accelerator hardware supports pattern generation, source expansion, pattern/source transparency, and 256 ternary raster operations. The block diagram of the graphics pipeline is shown in Figure 5-11.

5.4.1 BitBLT/Vector Engine

BLTs are initiated by writing to the GP_BLT_MODE register, which specifies the type of source data (none, frame buffer, or BLT buffer), the type of the destination data (none, frame buffer, or BLT buffer), and a source expansion flag.

Vectors are initiated by writing to the GP_VECTOR_MODE register (GX_BASE+8204h), which specifies the direction of the vector and a “read destination data” flag. If the flag is set, the hardware will read destination data along the vector and store it temporarily in the BLT Buffer 0.

The BLT buffers use a portion of the L1 cache, called “scratchpad RAM”, to temporarily store source and destination data, typically on a scan line basis. See Section 5.1.4.2 "Scratchpad RAM Utilization" on page 95 for an explanation of scratchpad RAM. The hardware automatically loads frame-buffer data (source or destination) into the BLT buffers for each scan line. The driver is responsible for making sure that this does not overflow the memory allocated for the BLT buffers. When the source data is a bitmap, the hardware loads the data directly into the BLT buffer at the beginning of the BLT operation.

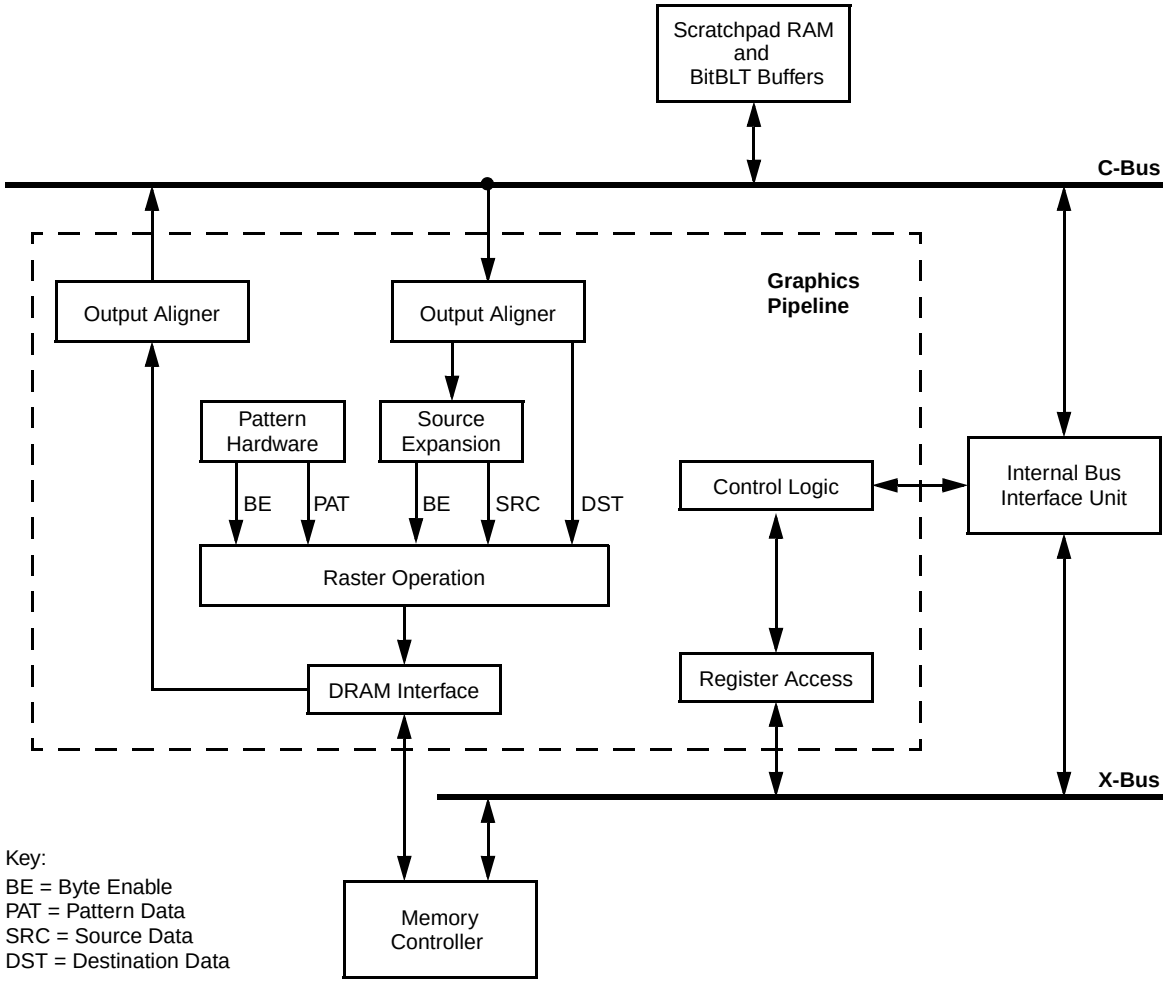


Figure 5-11. Graphics Pipeline Block Diagram

5.4.2 Master/Slave Registers

When starting a BitBLT or vector operation, the graphics pipeline registers are latched from the master registers to the slave registers. A second BitBLT or vector operation can then be loaded into the master registers while the first operation is rendered. If a second BLT is pending in the master registers, any write operations to the graphics pipeline registers will corrupt the values of the pending BLT. Software must prevent this from happening by checking the "BLT Pending" bit in the GP_BLT_STATUS register (GX_BASE+820Ch[2]).

Most of the graphics pipeline registers are latched directly from the master registers to the slave registers when starting a new BitBLT or vector operation. Some registers, however, use the updated slave values if the master registers have not been written, which allows software to render successive primitives without loading some of the registers as outlined in Table 5-20.

5.4.3 Pattern Generation

The graphics pipeline contains hardware support for 8x8 monochrome patterns (expanded to two colors), 8x8 dither patterns (expanded to four colors), and 8x1 color patterns. The pattern hardware, however, does not maintain a pattern origin, so the pattern data must be justified before it is loaded into the GX1 processor's registers. For solid primitives, the pattern hardware is disabled and the pattern color is always sourced from the GP_PAT_COLOR_0 register (GX_BASE+8110h).

Table 5-20. Graphics Pipeline Registers

Master	Function
GP_DST_XCOOR	Next X position along vector. Master register if written, otherwise: Unchanged slave if BLT, source mode = bitmap. Slave + width if BLT, source mode = text glyph
GP_DST_YCOOR	Next Y position along vector. Master register if written, otherwise: Slave +/- height if BLT, source mode = bitmap. Unchanged slave if BLT, source mode = text glyph.
GP_INIT_ERROR	Master register if written, otherwise: Initial error for the next pixel along the vector.
GP_SRC_YCOOR	Master register if written, otherwise: Slave +/- height if BLT, source mode = bitmap.

5.4.3.1 Monochrome Patterns

Setting the pattern mode to 01b (GX_BASE+8200h[9:8] = 01b) in the GP_RASTER_MODE register selects the monochrome patterns (see bit details on page 125). Those pixels corresponding to a clear bit (0) in the pattern are rendered using the color specified in the GP_PAT_COLOR_0 (GX_BASE+8110h**)** register, and those pixels corresponding to a set bit (1) in the pattern are rendered using the color specified in the GP_PAT_COLOR_1 register (GX_BASE+8112h).

If the pattern transparency bit is set high in the GP_RASTER_MODE register, those pixels corresponding to a clear bit in the pattern data are not drawn.

Monochrome patterns use registers GP_PAT_DATA_0 (GX_BASE+ Memory Offset 8120h) and GP_PAT_DATA_1 (GX_BASE+ Memory Offset 8124h) for the pattern data. Bits [7:0] of GP_PAT_DATA_0 correspond to the first row of the pattern, and bit 7 corresponds to the leftmost pixel on the screen. How the pattern and the registers fully relate is illustrated in Figure 5-12.

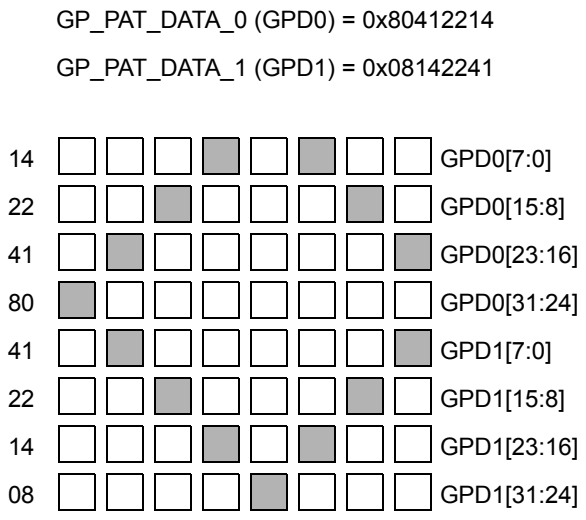


Figure 5-12. Example of Monochrome Patterns

5.4.3.2 Dither Patterns

Setting the pattern mode to 10b (GX_BASE+8200h[9:8] = 10b) in the GP_RASTER_MODE register selects the dither patterns. Two bits of pattern data are used for each pixel, allowing color expansion to four colors. The colors are specified in the GP_PAT_COLOR_0 through GP_PAT_COLOR_3 registers (Table 5-24 on page 123).

Dither patterns use all 128 bits of pattern data. Bits [15:0] of GP_PAT_DATA_0 correspond to the first row of the pattern (the lower byte contains the least significant bit of each pixel's pattern color and the upper byte contains the most significant bit of each pixel's pattern color). This is illustrated in Figure 5-13.

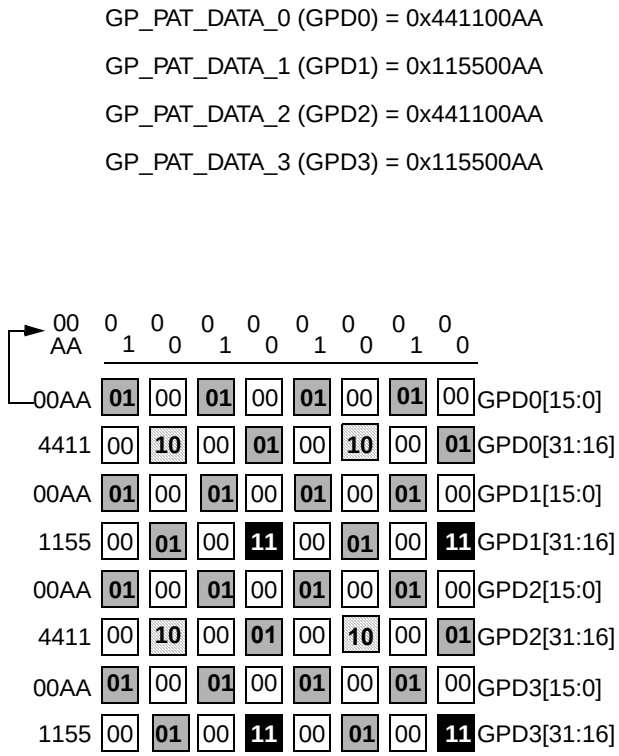


Figure 5-13. Example of Dither Patterns

5.4.3.3 Color Patterns

Setting the pattern mode to 11b (GX_BASE+8200h[9:8] = 11b), in the GP_RASTER_MODE register selects the color patterns. Bits [63:0] are used to hold a row of pattern data for an 8-bpp pattern, with bits [7:0] corresponding to the leftmost pixel of the row. Likewise, bits [127:0] are used for a 16-bpp color pattern, with bits [15:0] corresponding to the leftmost pixel of the row.

To support an 8x8 color pattern, software must load the pattern data for each row.

5.4.4 Source Expansion

The graphics pipeline contains hardware support for color expansion of source data (primarily used for text). Those pixels corresponding to a clear bit (0) in the source data are rendered using the color specified in the GP_SRC_COLOR_0 register (GX_BASE+810Ch), and those pixels corresponding to a set bit (1) in the source data are rendered using the color specified in the GP_SRC_COLOR_1 register (GX_BASE+810Eh).

If the source transparency bit is set in the GP_RASTER_MODE register, those pixels corresponding to a clear bit (0) in the source data are not drawn.

5.4.5 Raster Operations

The GP_RASTER_MODE register specifies how the pattern data, source data (color-expanded if necessary), and destination data are combined to produce the output to the frame buffer. The definition of the ROP value matches that of the Microsoft API (application programming interface). This allows Windows display drivers to load the raster operation directly into hardware. Table 5-21 illustrates this definition. Some common raster operations are described in Table 5-22.

Table 5-21. GP_RASTER_MODE Bit Patterns

Pattern (bit)	Source (bit)	Destination (bit)	Output (bit)
0	0	0	ROP[0]
0	0	1	ROP[1]
0	1	0	ROP[2]
0	1	1	ROP[3]
1	0	0	ROP[4]
1	0	1	ROP[5]
1	1	0	ROP[6]
1	1	1	ROP[7]

Table 5-22. Common Raster Operations

ROP	Description
F0h	Output = Pattern
CCh	Output = Source
5Ah	Output = Pattern XOR destination
66h	Output = Source XOR destination
55h	Output = ~Destination

5.4.6 Graphics Pipeline Register Descriptions

The graphics pipeline maps 200h locations starting at GX_BASE+8100h. However, only 72 bytes are defined and some of these registers will alias across the 200h space. Refer to Section 5.1.2 "Control Registers" on page 94 for instructions on accessing these registers. Table 5-23 summarizes the graphics pipeline registers and Table 5-24 gives detailed register/bit formats.

Table 5-23. Graphics Pipeline Configuration Register Summary

GX_BASE+ Memory Offset	Type	Name / Function	Default Value
8100h-8103h	R/W	**GP_DST/START_Y/XCOOR** Destination/Starting Y and X Coordinates Register: In BLT mode this register specifies the destination Y and X positions for a BLT operation. In Vector mode it specifies the starting Y and X positions in a vector.	00000000h
8104h-8107h	R/W	**GP_WIDTH/HEIGHT and GP_VECTOR_LENGTH/INIT_ERROR** Width/Height or Vector Length/Initial Error Register: In BLT mode this register specifies the BLT width and height in pixels. In Vector mode it specifies the vector initial error and pixel length.	00000000h
8108h-810Bh	R/W	**GP_SRC_X/YCOOR and GP_AXIAL/DIAG_ERROR** Source X/Y Coordinate Axial/Diagonal Error Register: In BLT mode this register specifies the BLT X and Y source. In Vector mode it specifies the axial and diagonal error for rendering a vector.	00000000h
810Ch-810Fh	R/W	**GP_SRC_COLOR_0 and GP_SRC_COLOR_1** Source Color Register: Determines the colors used when expanding monochrome source data in either the 8-bpp mode or the 16-bpp mode.	00000000h
8110h-8113h	R/W	**GP_PAT_COLOR_0 and GP_PAT_COLOR_1** Graphics Pipeline Pattern Color Registers 0 and 1: These two registers determine the colors used when expanding pattern data.	00000000h
8114h-8117h	R/W	**GP_PAT_COLOR_2 and GP_PAT_COLOR_3** Graphics Pipeline Pattern Color Registers 2 and 3:These two registers determine the colors used when expanding pattern data.	00000000h
8120h-8123h	R/W	**GP_PAT_DATA 0 through 3** Graphics Pipeline Pattern Data Registers 0 through 3: Together these registers contain 128 bits of pattern data. GP_PAT_DATA_0 corresponds to bits [31:0] of the pattern data. GP_PAT_DATA_1 corresponds to bits [63:32] of the pattern data. GP_PAT_DATA_2 corresponds to bits [95:64] of the pattern data. GP_PAT_DATA_3 corresponds to bits [127:96] of the pattern data.	00000000h
8124h-8127h	R/W		00000000h
8128h-812Bh	R/W		00000000h
812Ch-812Fh	R/W		00000000h
8140h-8143h[1]	R/W	**GP_VGA_WRITE** Graphics Pipeline VGA Write Patch Control Register: Controls the VGA memory write path in the graphics pipeline.	xxxxxxxxh
8144h-8147h[1]	R/W	**GP_VGA_READ** Graphics Pipeline VGA Read Patch Control Register: Controls the VGA memory read path in the graphics pipeline.	00000000h
8200h-8203h	R/W	**GP_RASTER_MODE** Graphics Pipeline Raster Mode Register: This register controls the manipulation of the pixel data through the graphics pipeline. Refer to Section 5.4.5 "Raster Operations" on page 121.	00000000h

Table 5-23. Graphics Pipeline Configuration Register Summary (Continued)

GX_BASE+ Memory Offset	Type	Name / Function	Default Value
8204h-8207h	R/W	**GP_VECTOR_MODE** Graphics Pipeline Vector Mode Register: Writing to this register initiates the rendering of a vector.	00000000h
8208h-820Bh	R/W	**GP_BLT_MODE** Graphics Pipeline BLT Mode Register: Writing to this initiates a BLT operation.	00000000h
820Ch-820Fh	R/W	**GP_BLT_STATUS** Graphics Pipeline BLT Status Register: Contains configuration and status information for the BLT engine. The status bits are contained in the lower byte of the register.	00000000h
8210h-8213h[1]	R/W	**GP_VGA_BASE** Graphics Pipeline VGA Memory Base Address Register: Specifies the offset of the VGA memory, starting from the base of graphics memory.	xxxxxxxxh
8214h-8217h[1]	R/W	**GP_VGA_LATCH** Graphics Pipeline VGA Display Latch Register: Provides a memory mapped way to read or write the VGA display latch.	xxxxxxxxh

1. The registers at GX_BASE+8140, 8144h, 8210h, and 8214h are located in the area designated for the graphics pipeline but are used for VGA emulation purposes. Refer to Table 5-39 on page 155 for these register's bit formats

Table 5-24. Graphics Pipeline Configuration Registers

Bit	Name	Description
GX_BASE+8100h-8103h		**GP_DST/START_X/YCOOR Register (R/W)** **Default Value = 00000000h**
31:16	**DESTINATION/STARTING Y POSITION (SIGNED):**	BLT Mode: Specifies the destination Y position for a BLT operation. Vector Mode: Specifies the starting Y position in a vector.
15:0	**DESTINATION/STARTING X POSITION (SIGNED):**	BLT Mode: Specifies the destination X position for a BLT operation. Vector Mode: Specifies the starting X position in a vector.
GX_BASE+8104h-8107h		**GP_WIDTH/HEIGHT and GP_VECTOR_LENGTH/INIT_ERROR Register (R/W)** **Default Value = 00000000h**
31:16	**PIXEL_WIDTH or VECTOR_LENGTH (UNSIGNED)**:	BLT Mode: Specifies the width, in pixels, of a BLT operation. No pixels are rendered for a width of zero. Vector Mode: Bits [31:30] are reserved in this mode allowing this 14-bit field to specify the length, in pixels, of a vector. No pixels are rendered for a length of zero. This field is limited to 14 bits due to a lack of precision in the registers used to hold the error terms.
15:0	**PIXEL_HEIGHT or VECTOR_INITIAL_ERROR (UNSIGNED):**	BLT Mode: Specifies the height, in pixels, of a BLT operation. No pixels are rendered for a height of zero. Vector Mode: Specifies the initial error for rendering a vector.
GX_BASE+8108h-810Bh		**GP_SCR_X/YCOOR and GP_AXIAL/DIAG_ERROR Register (R/W)** **Default Value = 00000000h**
31:16	**SRC_X_POS or VECTOR_AXIAL_ERROR (SIGNED):**	BLT Mode: Specifies the source X position for a BLT operation. Vector Mode: Specifies the axial error for rendering a vector.
15:0	**SRC_Y_POS or VECTOR_DIAG_ERROR (SIGNED):**	Source Y Position (Signed): Specifies the source Y position for a BLT operation. Vector Mode: Specifies the diagonal error for rendering a vector.

Table 5-24. Graphics Pipeline Configuration Registers (Continued)

Bit	Name	Description
GX_BASE+810Ch-810Dh	**GP_SRC_COLOR_0 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode**: 16-bpp color (RGB)
GX_BASE+810Eh-810Fh	**GP_SRC_COLOR_1 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode**: 16-bpp color (RGB)
Note:	The Graphics Pipeline Source Color register specifies the colors used when expanding monochrome source data in either the 8-bpp mode or the 16-bpp mode. Those pixels corresponding to clear bits (0) in the source data are rendered using GP_SRC_COLOR_0 and those pixels corresponding to set bits (1) in the source data are rendered using GP_SRC_COLOR_1.	
GX_BASE+8110h-8111h	**GP_PAT_COLOR_0 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode:** 16-bpp color (RGB)
Note:	The Graphics Pipeline Pattern Color 0-3 registers specify the colors used when expanding pattern data.	
GX_BASE+8112h-8113h	**GP_PAT_COLOR_1 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode:** 16-bpp color (RGB)
Note:	The Graphics Pipeline Pattern Color 0-3 registers specify the colors used when expanding pattern data.	
GX_BASE+8114h-8115h	**GP_PAT_COLOR_2 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode:** 16-bpp color (RGB)
Note:	The Graphics Pipeline Pattern Color 0-3 registers specify the colors used when expanding pattern data.	
GX_BASE+8116h-8117h	**GP_PAT_COLOR_3 Register (R/W)**	**Default Value = 0000h**
15:0		**8-bpp Mode:** 8-bpp color: The color index must be duplicated in the upper byte. **16-bpp Mode:** 16-bpp color (RGB)
Note:	The Graphics Pipeline Pattern Color 0-3 registers specify the colors used when expanding pattern data.	
GX_BASE+8120h-8123h	**GP_PAT_DATA_0 Register (R/W)**	**Default Value = xxxxxxxxh**
31:0		**GP Pattern Data Register 0:** The Graphics Pipeline Pattern Data registers 0 through 3 together contain 128 bits of pattern data. The GP_PAT_DATA_0 register corresponds to bits [31:0] of the pattern data.
GX_BASE+8124h-8127h	**GP_PAT_DATA_1 Register (R/W)**	**Default Value = xxxxxxxxh**
31:0		**GP Pattern Data Register 1:** The Graphics Pipeline Pattern Data registers 0 through 3 together contain 128 bits of pattern data. The GP_PAT_DATA_1 register corresponds to bits [63:32] of the pattern data.
GX_BASE+8128h-812Bh	**GP_PAT_DATA_2 Register (R/W)**	**Default Value = xxxxxxxxh**
31:0		**GP Pattern Data Register 2:** The Graphics Pipeline Pattern Data registers 0 through 3 together contain 128 bits of pattern data. The GP_PAT_DATA_2 register corresponds to bits [95:64] of the pattern data.
GX_BASE+812Ch-812Fh	**GP_PAT_DATA_3 Register (R/W)**	**Default Value = xxxxxxxxh**
31:0		**GP Pattern Data Register 3:** The Graphics Pipeline Pattern Data registers 0 through 3 together contain 128 bits of pattern data. The GP_PAT_DATA_3 register corresponds to bits [127:96] of the pattern data.
GX_BASE+8140h-8143h	**GP_VGA_WRITE Register (R/W)**	**Default Value = xxxxxxxxh**
Note that the register at GX_BASE+82140h is located in the area designated for the graphics pipeline but is used for VGA emulation purposes. Refer to Table 5-39 on page 155 for this register's bit formats.		
GX_BASE+8144h-8147h	**GP_VGA_READ Register (R/W)**	**Default Value = 00000000h**
Note that the register at GX_BASE+8144h is located in the area designated for the graphics pipeline but is used for VGA emulation purposes. Refer to Table 5-39 on page 155 for this register's bit formats.		

Table 5-24. Graphics Pipeline Configuration Registers (Continued)

Bit	Name	Description
GX_BASE+8200h-8203h	**GP_RASTER_MODE Register (R/W)**	**Default Value = 00000000h**
31:13	RSVD	**Reserved:** Set to 0.
12	TB	**Transparent BLT:** When set, this bit enables transparent BLT. The source color data will be compared to a color key and if it matches, that pixel will not be drawn. The color key value is stored in the BLT buffer as destination data. The raster operation must be set to C6h, and the pattern registers must be all F's for this mode to work properly.
11	ST	**Source Transparency:** Enables transparency for monochrome source data. Those pixels corresponding to clear bits in the source data are not drawn.
10	PT	**Pattern Transparency:** Enables transparency for monochrome pattern data. Those pixels corresponding to clear bits in the pattern data are not drawn.
9:8	PM	**Pattern Mode:** Specifies the format of the pattern data. 00 = Indicates a solid pattern. The pattern data is always sourced from the GP_PAT_COLOR_0 register. 01 = Indicates a monochrome pattern. The pattern data is sourced from the GP_PAT_COLOR_0 and GP_PAT_COLOR_1 registers. 10 = Indicates a dither pattern. All four pattern color registers are used. 11 = Indicates a color pattern. The pattern data is sourced directly from the pattern data registers.
7:0	ROP	**Raster Operation:** Specifies the raster operation for pattern, source, and destination data.
Note: Writing to this register launches a raster operation.		
GX_BASE+8204h-8207h	**GP_VECTOR_MODE Register (R/W)**	**Default Value = 00000000h**
31:4	RSVD	**Reserved:** Set to 0.
3	DEST	**Read Destination Data:** Indicates that frame-buffer destination data is required.
2	DMIN	**Minor Direction:** Indicates a positive minor axis step.
1	DMAJ	**Major Direction:** Indicates a positive major axis step.
0	YMAJ	**Major Direction:** Indicates a Y major vector.
GX_BASE+8208h-820Bh	**GP_BLT_MODE Register (R/W)**	**Default Value = 00000000h**
31:9	RSVD	**Reserved:** Set to 0.
8	Y	**Reverse Y Direction:** Indicates a negative increment for the Y position. This bit is used to control the direction of screen to screen BLTs to prevent data corruption in overlapping windows.
7:6	SM	**Source Mode:** Specifies the format of the source data. 00 = Source is a color bitmap. 01 = Source is a monochrome bitmap (use source color expansion). 10 = Unused. 11 = Source is a text glyph (use source color expansion). This differs from a monochrome bitmap in that the X position is adjusted by the width of the BLT and the Y position remains the same.
5	RSVD	**Reserved:** Set to 0.
4:2	RD	**Destination Data:** Specifies the destination data location. 000 = No destination data is required. The destination data into the raster operation unit is all ones. 010 = Read destination data from BLT Buffer 0. 011 = Read destination data from BLT Buffer 1. 100 = Read destination data from the frame buffer (store temporarily in BLT Buffer 0). 101 = Read destination data from the frame buffer (store temporarily in BLT Buffer 1).
1:0	RS	**Source Data:** Specifies the source data location. 00 = No source data is required. The source data into the raster operation unit is all ones. 01 = Read source data from the frame buffer (temporarily stored in BLT Buffer 0). 10 = Read source data from BLT Buffer 0. 11 = Read source data from BLT Buffer 1.
Note: Writing to this register launches a BLT operation.		

Table 5-24. Graphics Pipeline Configuration Registers (Continued)

Bit	Name	Description
GX_BASE+820Ch-820Fh		**GP_BLT_STATUS Register (R/W)** **Default Value = 00000000h**
31:11	RSVD	**Reserved:** Set to 0.
10:9	W	**Screen Width:** Selects a frame-buffer width. This register must be programmed correctly in order for compression to work. 00 = 1024 bytes 01 = 2048 bytes 10 = 4096 bytes 11 = 4096 bytes
8	M	**16-bpp Mode:** Selects a pixel data format of 16-bpp (default is 8-bpp).
7:3	RSVD	**Reserved:** Set to 0.
2	BP (RO)	**BLT Pending (Read Only):** Indicates that a BLT operation is pending in the master registers. The “BLT Pending” bit must be clear before loading any of the graphics pipeline registers. Loading registers when this bit is set high will destroy the values for the pending BLT.
1	PB (RO)	**Pipeline Busy (Read Only):** Indicates that the graphics pipeline is processing data. The “Pipeline Busy” bit differs from the “BLT Busy” bit in that the former only indicates that the graphics pipeline is processing data. The “BLT Busy” bit also indicates that the memory controller has not yet processed all of the requests for the current operation. The “Pipeline Busy” bit must be clear before loading a BLT buffer if the previous BLT operation used the same BLT buffer.
0	BB (RO)	**BLT Busy (Read Only):** Indicates that a BLT / vector operation is in progress. The “BLT Busy” bit must be clear before accessing the frame buffer directly.
GX_BASE+8210h-8213h		**GP_VGA_BASE (R/W)** **Default Value = xxxxxxxxh**
Note that the registers at GX_BASE+8210h are located in the area designated for the graphics pipeline but are used for VGA emulation purposes. Refer to Table 5-39 on page 155 for this register’s bit formats.		
GX_BASE+8214h-8217h		**GP_VGA_LATCH Register (R/W)** **Default Value = xxxxxxxxh**
Note that the registers at GX_BASE+8214h are located in the area designated for the graphics pipeline but are used for VGA emulation purposes. Refer to Table 5-39 on page 155 for this register’s bit formats.		

5.5 Display Controller

The GX1 processor incorporates a display controller that retrieves display data from the memory controller and formats it for output on a variety of display devices. The GX1 processor connects directly to the graphics of the AMD Geode™ CS5530A companion device. The display controller includes a display FIFO, compression/decompression (codec) hardware, hardware cursor, a 256-entry-by-18-bit palette RAM (plus three extension colors), display timing generator, dither and frame-rate-modulation circuitry for TFT panels, and versatile output formatting logic. A diagram of the display controller subsystem is shown in Figure 5-14.

5.5.1 Display FIFO

The display controller contains a large (64x64 bit) FIFO for queuing up display data from the memory controller as required for output to the screen. The memory controller must arbitrate between display controller requests and other requests for memory access from the microprocessor core, L1 cache controller, and the graphics pipeline.

Display data is required in real time, making it the highest priority in the system. Without efficient memory management, system performance would suffer dramatically due to the constant display-refresh requests from the display controller. The large size of the display FIFO is desirable so that the FIFO may primarily be loaded during times when there is no other request pending to the DRAM controller which allows the memory controller to stay in page mode for a longer period of time when servicing the display FIFO. When a priority request from the cache or graphics pipeline occurs, if the display FIFO has enough data queued up, the DRAM controller can immediately service the request without concern that the display FIFO will underflow. If the display FIFO is below a programmable threshold, a high-priority request will be sent to the DRAM controller, which will take precedence over any other requests that are pending.

The display FIFO is 64 bits wide to accommodate high-speed burst read operations from the DRAM controller at maximum memory bandwidth. In addition to the normal pixel data stream, the display FIFO also queues up cursor patterns.

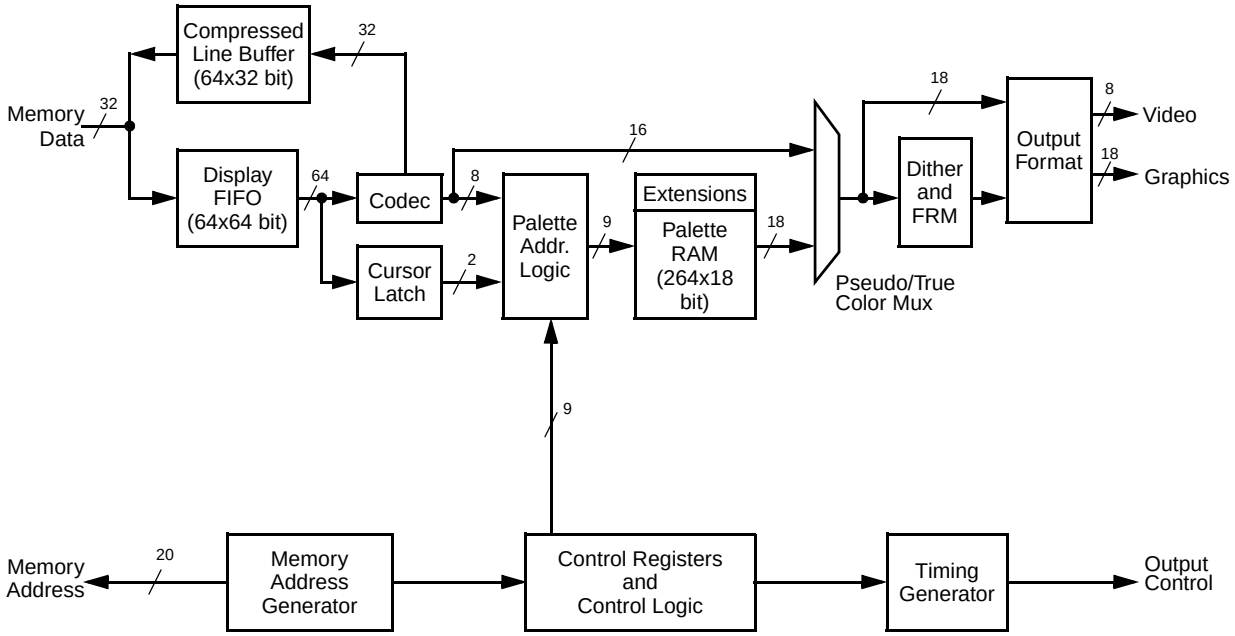


Figure 5-14. Display Controller Block Diagram

5.5.2 Compression Technology

To reduce the system memory contention caused by the display refresh, the display controller contains compression and decompression logic for compressing the frame buffer image in real time as it is sent to the display. It combines this compressed display buffer into the extra off-screen memory within the graphics memory aperture. Coherency of the compressed display buffer is maintained by use of dirty and valid bits for each line. The dirty and valid RAM is contained on-chip for maximum efficiency. Whenever a line has been validly compressed, it will be retrieved from the compressed display buffer for all future accesses until the line becomes dirty again. Dirty lines will be retrieved from the normal uncompressed frame buffer.

The compression logic has the ability to insert a programmable number of “static” frames, during which time dirty bits are ignored and the valid bits are read to determine whether a line should be retrieved from the frame buffer or compressed display buffer. The less frequently the dirty bits are sampled, the more frequently lines will be retrieved from the compressed display buffer. This allows a programmable screen image update rate (as opposed to refresh rate). Generally, an update rate of 30 frames per second is adequate for displaying most types of data, including real-time video. If a flat panel display is used that has a slow response time, such as 100 ms, the image need not be updated faster than ten frames per second, since the panel could not display changes beyond that rate.

The compression algorithm used in the GX1 processor commonly achieves compression ratios between 10:1 and 20:1, depending on the nature of the display data. This high level of compression provides higher system performance by reducing typical latency for normal system memory access, higher graphics performance by increasing available drawing bandwidth to the DRAM array, and much lower power consumption by significantly reducing the number of off-chip DRAM accesses required for refreshing the display. These advantages become even more pronounced as display resolution, color depth, and refresh rate are increased and as the size of the installed DRAM increases.

As uncompressed lines are fed to the display, they will be compressed and stored in an on-chip compressed line buffer (64x32 bits). Lines will not be written back to the compressed display buffer in the DRAM unless a valid compression has resulted, so there is no penalty for pathological frame buffer images where the compression algorithm breaks down.

5.5.3 Hardware Cursor

The display controller contains hardware cursor logic to allow overlay of the cursor image onto the pixel data stream. Overhead for updating this image on the screen is kept to a minimum by requiring that only the X and Y position be changed. This eliminates “submarining” effects commonly associated with software cursors. The cursor, 32x32 pixels with 2-bpp, is loaded into off-screen memory within the graphics memory aperture. The DC_CUR_ST_OFFSET programs the cursor start (see Table 5-30 on page 140). The 2-bit code selects color 0, color 1, transparent, or background-color inversion for each pixel in the cursor. The two cursor colors will be stored as extensions to the normal 256-entry palette at locations 100h and 101h.

The 2-bit cursor codes are as follows:

AND	XOR	Displayed
0	0	Cursor Color 0
0	1	Cursor Color 1
1	0	Transparent – Background Pixel
1	1	Inverted – Bit-wise Inversion of Background Pixel

The cursor overlay patterns are loaded to independent memory locations, usually mapped above the frame buffer and compressed display buffer (off-screen). The cursor buffer must start on a DWORD boundary. It is linearly mapped, and is always 256 bytes in size. If there is enough room (256 bytes) after the compression-buffer line but before the next frame-buffer line starts, the cursor pattern may be loaded into this area to make efficient use of the graphics memory.

Each pattern is a 32x32-pixel array of 2-bit codes. The codes are a combination of AND mask and XOR mask for a particular pixel. Each line of an overlay pattern is stored as two DWORDs, with each DWORD containing the AND masks for 16 pixels in the upper word and the XOR masks for 16 pixels in the lower word. DWORDs are arranged with the leftmost pixel block being least significant and the rightmost pixel block being most significant. Pixels within words are arranged with the leftmost pixels being most significant and the rightmost pixels being least significant. Multiple cursor patterns may be loaded into the off-screen memory. An application may simply change the cursor start offset to select a new cursor pattern. The new cursor pattern will become effective at the start of the next frame scan.

5.5.4 Display Timing Generator

The display controller features a fully programmable timing generator for generating all timing control signals for the display. The timing control signals include horizontal and vertical sync and blank signals in addition to timing for active and overscan regions of the display. The timing generator is similar in function to the CRTC of the original VGA, although programming is more straightforward. Programming of the timing registers is supported by AMD via a BIOS INT10 call during a mode set. When programming the timing registers directly, extreme care should be taken to ensure that all timing is compatible with the display device.

The timing generator supports overscan to maintain full backward compatibility with the VGA standard. This feature is supported primarily for CRT display devices since flat panel displays have fixed resolutions and do not provide for overscan. When a display mode is selected having a lower resolution than the panel resolution, the GX1 processor supports a mechanism to center the display by stretching the border to fill the remainder of the screen. The border color is at palette extension 104h.

5.5.5 Dither and Frame Rate Modulation

The display controller supports 2x2 dither and two-level frame rate modulation (FRM) to increase the apparent number of colors displayed on 9-bit or 12-bit TFT panels. Dither and FRM are individually programmable. With dithering and FRM enabled, 185,193 colors are possible on a 9-bit TFT panel, and 226,981 colors are possible on a 12-bit TFT panel.

5.5.6 Display Modes

The GX1 processor's display controller is programmable and supports resolutions up to 1280x1024 at 16 bits per pixel. This means the GX1 processor supports the standard display resolutions of 640x480, 800x600, 1024x768, and 1280x1024 at both 8 and 16 bits per pixel. Two 16-bit display formats are supported: RGB 5-6-5 and RGB 5-5-5. Table 5-26 on page 130 lists how the RGB data is mapped onto the pixel data bus for the CRT and various TFT interfaces. All CRT modes can have VESA-compatible timing. Table 5-25 lists some of the supported TFT panel display modes and Table 5-27 lists some of the supported CRT display modes.

Table 5-25. TFT Panel Display Modes[1]

Resolution	Simultaneous Colors	Refresh Rate (Hz)	DCLK[2] Rate (MHz)	PCLK[3] Rate (MHz)	Panel Type	Maximum Displayed Colors[4]
640x480[5]	8-bpp 256 colors out of a palette of 256	60	50.35	25.175	9-bit	57^3 = 185,193
					12-bit	61^3 = 226,981
					18-bit	4^3 = 262,144
	16-bpp 64 KB colors 5-6-5	60	50.35	25.175	9-bit	29x57x29 = 47,937
					12-bit	31x61x31 = 58,621
					18-bit	32x64x32 = 65,535
800x600[5]	8-bpp 256 colors out of a palette of 256	60	80.0	40.0	9-bit	57^3 = 185,193
					12-bit	61^3 = 226,981
					18-bit	64^3 = 262,144
	16-bpp 64 KB Colors 5-6-5	60	80.0	40.0	9-bit	29x57x29 = 47,937
					12-bit	31x61x31 = 58,621
					18-bit	32x64x32 = 65,535
1024x768	8-bpp 256 colors out of a palette of 256	60	65	65.0	9-bit/18-I/F	57^3 = 185,193
	16-bpp 64 KB colors 5-6-5	60	65	65.0	9-bit/18-I/F	29x57x29 = 47,937

1. This list is not meant to be an complete list of all the possible supported TFT display modes.
2. DCLK is the input clock from the AMD Geode™ CS5530A companion device. In some cases, DCLK is doubled to keep the CS5530A's PLL in a desired operational range.
3. PCLK is the graphics output clock to the Geode CS5530A companion device.
4. 9-bit and 12-bit panels use FRM and dither to increase displayed colors. (See Section 5.5.5 "Dither and Frame Rate Modulation" on page 129.)
5. All 640x480 and 800x600 modes can be run in simultaneous display with CRT.

Table 5-26. CRT and TFT Panel Data Bus Formats

Panel Data Bus Bit	CRT & 18-Bit TFT	12-Bit TFT	9-Bit TFT		
			640x480	1024x768	
17	R5	R5	R5	R5	Even
16	R4	R4	R4	R4	
15	R3	R3	R3	R3	
14	R2	R2		R5	Odd
13	R1			R4	
12	R0			R3	
11	G5	G5	G5	G5	Even
10	G4	G4	G4	G4	
9	G3	G3	G3	G3	
8	G2	G2		G5	Odd
7	G1			G4	
6	G0			G3	
5	B5	B5	B5	B5	Even
4	B4	B4	B4	B4	
3	B3	B3	B3	B3	
2	B2	B2		B5	Odd
1	B1			B4	
0	B0			B3	

Table 5-27. CRT Display Modes[1]

Resolution	Simultaneous Colors	Refresh Rate (Hz)	DCLK[2] Rate (MHz)	PCLK[3] Rate (MHz)
640x480	8-bpp 256 colors out of a palette of 256	60	50.35	25.175
		72	63.0	31.5
		75	63.0	31.5
		85	72.0	36.0
	16-bpp 64 KB colors RGB 5-6-5	60	50.35	25.175
		72	63.0	31.5
		75	63.0	31.5
		85	72.0	36.0

Table 5-27. CRT Display Modes[1]

Resolution	Simultaneous Colors	Refresh Rate (Hz)	DCLK[2] Rate (MHz)	PCLK[3] Rate (MHz)
800x600	8-bpp 256 colors out of a palette of 256	60	80.0	40.0
		72	100.0	50.0
		75	99.0	49.5
		85	112.5	56.25
	16-bpp 64 KB colors RGB 5-6-5	60	80.0	40.0
		72	100.0	50.0
		75	99	49.9
		85	112.5	56.25
1024x768	8-bpp 256 colors out of a palette of 256	60	65.0	65.0
		70	75.0	75.0
		75	78.5	78.5
		85	94.5	94.5
	16-bpp 64 KB colors RGB 5-6-5	60	65.0	65.0
		70	75.0	75.0
		75	78.5	78.5
		85	94.5	94.5
1280x1024	8-bpp 256 colors out of a palette of 256	60	108.0	108.0
		75	135.0	135
		85	157	157
	16 bpp 64 KB colors RGB 5-6-5	60	108	108
		75	135	135
		85	157	157

1. This list is not meant to be an complete list of all the possible supported CRT display modes.
2. DCLK is the input clock from the AMD Geode™ CS5530A companion device. In some cases, DCLK is doubled to keep the CS5530A's PLL in a desired operational range.
3. PCLK is the graphics output clock to the Geode CS5530A companion device.

5.5.7 Graphics Memory Map

The GX1 processor supports a maximum of 4 MB of graphics memory and will map it to an address space (see Figure 5-2 on page 93) higher than the maximum amount of installed RAM. The graphics memory aperture physically resides at the top of the installed system RAM. The start address and size of the graphics memory aperture are programmable on 512 KB boundaries. Typically, the system BIOS sets the size and start address of the graphics memory aperture during the boot process based on the amount of installed RAM, user defined CMOS settings, hard coded, etc. The graphics pipeline and display controller address the graphics memory with a 20-bit offset (address bits [21:2]) and four byte enables into the graphics memory aperture. The graphics memory stores several buffers that are used to generate the display: the frame buffer, compressed display buffer, VGA memory, and cursor pattern(s). Any remaining off-screen memory within the graphics aperture may be used by the display driver as desired or not at all.

5.5.7.1 DC Memory Organization Registers

The display controller contains a number of registers that allow full programmability of the graphics memory organization. This includes starting offsets for each of the buffer regions described above, line delta parameters for the frame buffer and compression buffer, as well as compressed line-buffer size information. The starting offsets for the various buffers are programmable for a high degree of flexibility in memory organization.

5.5.7.2 Frame Buffer and Compression Buffer Organization

The GX1 processor supports primary display modes 640x480, 800x600, 1024x768, and 1280x1024 at both 8-bpp and 16-bpp. Pixels are packed into DWORDs as shown in Figure 5-15.

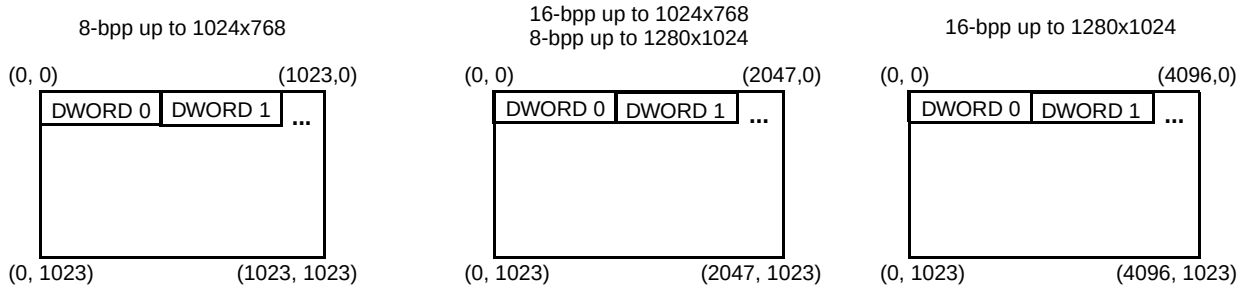
In order to simplify address calculations by the rendering hardware, the frame buffer is organized in an XY fashion where the offset is simply a concatenation of the X and Y pixel addresses. All 8-bpp display modes with the exception of the 1280x1024 resolution use a 1024-byte line delta between the starting offsets of adjacent lines. All 16-bpp display modes except 1280x1024 use a 2048-byte line delta between the starting offsets of adjacent lines. 1280x1024x16 uses a 4096-byte line offset. If there is room, the space between the end of a line and the start of the next line will be filled with the compressed display data for that line, thus allowing efficient memory utilization. For 1024x768 display modes, the frame-buffer line size is the same as the line delta, so no room is left for the compressed display data between lines. In this case, the compressed display buffer begins at the end of the frame buffer region and is linearly mapped.

5.5.7.3 VGA Display Support

The graphics pipeline contains full hardware support for the VGA front end. The VGA data is stored in a 256 KB buffer located in graphics memory. The main task for Virtual VGA (see Section 5.6 "Virtual VGA Subsystem" on page 149) is converting the data in the VGA buffer to an 8-bpp frame buffer that can be displayed by the display controller.

For some modes, the display controller can display the VGA data directly and the data conversion is not necessary. This includes standard VGA mode 13h and the variations of that mode used in several games; the display controller can also directly display VGA planar graphics modes D, E, F, 10, 11, and 12. Likewise, the hardware can directly display all of the higher-resolution VESA modes. Since the frame buffer data is written directly to memory instead of travelling across an external bus, the GX1 processor often outperforms VGA cards for these modes.

The display controller, however, does not directly support text modes. SoftVGA must convert the characters and attributes in the VGA buffer to an 8-bpp frame buffer image the hardware uses for display refresh.



DWORD

Bit Position	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Address	3h								2h								1h								0h							
Pixel Org - 8-bpp	(3,0)								(2,0)								(1,0)								(0,0)							
Pixel Org - 16-bpp	(1,0)																(0,0)															

Figure 5-15. Pixel Arrangement Within a DWORD

5.5.8 Display Controller Registers

The display controller maps 100h memory locations starting at GX_BASE+8300h for the display controller registers. However, only 116 bytes are defined and some of these registers will alias across the 100h space. Refer to Section 5.1.2 "Control Registers" on page 94 for instructions on accessing these registers.

The display controller registers are divided into six categories:

- Configuration and Status registers
- Memory Organization registers
- Timing registers
- Cursor and Line Compare registers
- Color registers
- Palette and RAM Diagnostic registers

Table 5-28 summarizes these registers and locations, and the following subsections give detailed register/bit formats.

Table 5-28. Display Controller Register Summary

GX_BASE+ Memory Offset	Type	Name/Function	Default Value
Configuration and Status Registers			
8300h-8303h	R/W	**DC_UNLOCK** Display Controller Unlock: This register is provided to lock the most critical memory-mapped display controller registers to prevent unwanted modification (write operations). Read operations are always allowed.	00000000h
8304h-8307h	R/W	**DC_GENERAL_CFG** Display Controller General Configuration: General control bits for the display controller.	00000000h
8308h-830Bh	R/W	**DC_TIMING_CFG** Display Controller Timing Configuration: Status and control bits for various display timing functions.	xx000000h
830Ch-830Fh	R/W	**DC_OUTPUT_CFG** Display Controller Output Configuration: Status and control bits for pixel output formatting functions.	xx000000h
Memory Organization Registers			
8310h-8313h	R/W	**DC_FB_ST_OFFSET** Display Controller Frame Buffer Start Address: Specifies offset at which the frame buffer starts.	xxxxxxxxh
8314h-8317h	R/W	**DC_CB_ST_OFFSET** Display Controller Compression Buffer Start Address: Specifies offset at which the compressed display buffer starts.	xxxxxxxxh
8318h-831Bh	R/W	**DC_CUR_ST_OFFSET** Display Controller Cursor Buffer Start Address: Specifies offset at which the cursor memory buffer starts.	xxxxxxxxh
831Ch-831Fh	--	**Reserved**	00000000h
8320h-8323h	R/W	**DC_VID_ST_OFFSET** Display Controller Video Start Address: Specifies offset at which the video buffer starts.	xxxxxxxxh
8324h-8327h	R/W	**DC_LINE_DELTA** Display Controller Line Delta: Stores line delta for the graphics display buffers.	xxxxxxxxh

Table 5-28. Display Controller Register Summary (Continued)

GX_BASE+ Memory Offset	Type	Name/Function	Default Value
8328h-832Bh	R/W	**DC_BUF_SIZE** Display Controller Buffer Size: Specifies the number of bytes to transfer for a line of frame buffer data and the size of the compressed line buffer. (The compressed line buffer will be invalidated if it exceeds the CB_LINE_SIZE, bits [15:9].)	xxxxxxxxh
832Ch-832Fh	--	**Reserved**	00000000h
Timing Registers			
8330h-8333h	R/W	**DC_H_TIMING_1** Display Controller Horizontal and Total Timing: Horizontal active and total timing information.	xxxxxxxxh
8334h-8337h	R/W	**DC_H_TIMING_2** Display Controller CRT Horizontal Blanking Timing: CRT horizontal blank timing information.	xxxxxxxxh
8338h-833Bh	R/W	**DC_H_TIMING_3** Display Controller CRT Sync Timing: CRT horizontal sync timing information. Note, however, that this register should also be programmed appropriately for flat panel only display since the horizontal sync transition determines when to advance the vertical counter.	xxxxxxxxh
833Ch-833Fh	R/W	**DC_FP_H_TIMING** Display Controller Flat Panel Horizontal Sync Timing: Horizontal sync timing information for an attached flat panel display.	xxxxxxxxh
8340h-8343h	R/W	**DC_V_TIMING_1** Display Controller Vertical and Total Timing: Vertical active and total timing information. The parameters pertain to both CRT and flat panel display.	xxxxxxxxh
8344h-8247h	R/W	**DC_V_TIMING_2** Display Controller CRT Vertical Blank Timing: Vertical blank timing information.	xxxxxxxxh
8348h-834Bh	R/W	**DC_V_TIMING_3** Display Controller CRT Vertical Sync Timing: CRT vertical sync timing information.	xxxxxxxxh
834Ch-834Fh	R/W	**DC_FP_V_TIMING** Display Controller Flat Panel Vertical Sync Timing: Flat panel vertical sync timing information.	xxxxxxxxh
Cursor and Line Compare Registers			
8350h-8353h	R/W	**DC_CURSOR_X** Display Controller Cursor X Position: X position information of the hardware cursor.	xxxxxxxxh
8354h-8357h	RO	**DC_V_LINE_CNT** Display Controller Vertical Line Count: This read only register provides the current scanline for the display. It is used by software to time update of the frame buffer to avoid tearing artifacts.	xxxxxxxxh
8358h-835Bh	R/W	**DC_CURSOR_Y** Display Controller Cursor Y Position: Y position information of the hardware cursor.	xxxxxxxxh

Table 5-28. Display Controller Register Summary (Continued)

GX_BASE+ Memory Offset	Type	Name/Function	Default Value
835Ch-835Fh	R/W	**DC_SS_LINE_CMP** Display Controller Split-Screen Line Compare: Contains the line count at which the lower screen begins in a VGA split-screen mode.	xxxxxxxxh
8360h-836Fh	--	**Reserved**	xxxxxxxxh
Palette and RAM Diagnostic Registers			
8370h-8373h	R/W	**DC_PAL_ADDRESS** Display Controller Palette Address: This register should be written with the address (index) location to be used for the next access to the DC_PAL_DATA register.	xxxxxxxxh
8374h-8377h	R/W	**DC_PAL_DATA** Display Controller Palette Data: Contains the data for a palette access cycle.	xxxxxxxxh
8378h-837Bh	R/W	**DC_DFIFO_DIAG** Display Controller Display FIFO Diagnostic: This register is provided to enable testability of the Display FIFO RAM.	xxxxxxxxh
837Ch-837Fh	R/W	**DC_CFIFO_DIAG** Display Controller Compression FIFO Diagnostic: This register is provided to enable testability of the Compressed Line Buffer (FIFO) RAM.	xxxxxxxxh

5.5.8.1 Configuration and Status Registers

The Configuration and Status registers group consists of four 32-bit registers located at GX_BASE+8300h-830Ch. These registers are described below and Table 5-29 gives their bit formats.

Table 5-29. Display Controller Configuration and Status Registers

Bit	Name	Description
GX_BASE+8300h-8303h		**DC_UNLOCK Register (R/W)** **Default Value = 00000000h**
31:16	RSVD	**Reserved:** Set to 0.
15:0	UNLOCK_ CODE	**Unlock Code:** This register must be written with the value 4758h in order to write to the protected registers. The following registers are protected by the locking mechanism. Writing any other value enables the write lock function. DC_GENERAL_CFG, DC_TIMING_CFG, DC_OUTPUT_CFG, DC_FB_ST_OFFSET, DC_CB_ST_OFFSET, DC_CUR_ST_OFFSET, DC_VID_ST_OFFSET, DC_LINE_DELTA, DC_BUF_SIZE, DC_H_TIMING_1, DC_H_TIMNG_2, DC_H_TIMING_3, DC_FP_H_TIMING, DC_V_TIMING_1, DC_V_TIMING_2, DC_V_TIMING_3, DC_FP_V_TIMING
GX_BASE+8304h-8307h		**DC_GENERAL_CFG (R/W) (Locked)** **Default Value = 00000000h**
31	RSVD	**Reserved:** Set to 0.
30	RSVD	**Reserved:** Set to 0.
29	VRDY	**Video Ready Protocol:** 0 = Low speed video port: 1 = High speed video port Always program to 1.
28	VIDE	**Video Enable:** Motion video port: 0 = Disable; 1 = Enable. **Note:** This bit should only be modified during vertical retrace (see DC_TIMING_CFG bits 31 and 30, GX_BASE_8308h).
27	SSLC	**Split-screen Line Compare:** VGA line compare function: 0 = Disable; 1 = Enable. When enabled, the internal line counter will be compared to the value programmed in the DC_SS _LINE_CMP register. If it matches, the frame buffer address will be reset to zero. This enables a split screen function.
26	CH4S	**Chain 4 Skip:** Allow display controller to read every 4th DWORD from the frame buffer for compatibility with the VGA: 0 = Disable; 1 = Enable.
25	DIAG	**FIFO Diagnostic Mode:** This bit allows testability of the on-chip Display FIFO and Compressed Line Buffer via the diagnostic access registers. A low-to-high transition will reset the Display FIFO's R/W pointers and the Compressed Line Buffer's read pointer. 0 = Normal operation; 1 = Enable.
24	LDBL	**Line Double:** Allow line doubling for emulated VGA modes: 0 = Disable; 1 = Enable. If enabled, this will cause each odd line to be replicated from the previous line as the data is sent to the display. Timing parameters should be programmed as if pixel doubling is not used, however, the frame buffer should be loaded with half the normal number of lines.
23:19	RSVD	**Reserved:** Set to 0.
18	FDTY	**Frame Dirty Mode:** Allow entire frame to be flagged as dirty whenever a pixel write occurs to the frame buffer (this is provided for modes that use a linearly mapped frame buffer for which the line delta is not equal to 1024 or 2048 bytes): 0 = Disable; 1 = Enable. When disabled, dirty bits are set according to the Y address of the pixel write.
17	RSVD	**Reserved:** Set to 0.
16	CMPI	**Compressor Insert Mode:** Insert one static frame between update frames: 0 = Disable; 1 = Enable. An update frame is a frame in which dirty lines are updated. Conversely, a static frame is a frame in which dirty lines are not updated (the display image may not actually be static, because lines that are not compressed successfully must be retrieved from the uncompressed frame buffer).

Table 5-29. Display Controller Configuration and Status Registers (Continued)

Bit	Name	Description
15:12	DFIFO HI-PRI END LVL	**Display FIFO High Priority End Level:** This field specifies the depth of the display FIFO (in 64-bit entries x 4) at which a high-priority request previously issued to the memory controller will end. The value is dependent upon display mode. This register should always be non-zero and should be larger than the start level.
11:8	DFIFO HI-PRI START LVL	**Display FIFO High Priority Start Level:** This field specifies the depth of the display FIFO (in 64-bit entries x 4) at which a high-priority request will be sent to the memory controller to fill up the FIFO. The value is dependent upon display mode. This register should always be nonzero and should be less than the high-priority end level.
7:6	DCLK_ DIV	**DCLK Divider:** This 2-bit field specifies the clock divider for the input DCLK pin. 00 = Forced Low 01 = DCLK ÷ 2 10 = DCLK 11 = DCLK
5	DECE	**Decompression Enable:** Allow operation of internal decompression hardware: 0 = Disable; 1 = Enable.
4	CMPE	**Compression Enable:** Allow operation of internal compression hardware: 0 = Disable; 1 = Enable
3	PPC	**Pixel Panning Compatibility:** This bit has the same function as that found in the VGA. Allow pixel alignment to change when crossing a split-screen boundary - it will force the pixel alignment to be 16-byte aligned: 0 = Disable; 1 = Enable. If disabled, the previous alignment will be preserved when crossing a split-screen boundary.
2	DVCK	**Divide Video Clock:** Selects frequency of VID_CLK pin: 0 = VID_CLK pin frequency is equal to one-half (½) the frequency of the core clock. 1 = VID_CLK pin frequency is equal to one-fourth (¼) the frequency of the core clock. Bit 28 (VIDE) must be set to 1 for this bit to be valid.
1	CURE	**Cursor Enable:** Use internal hardware cursor: 0 = Disable; 1 = Enable.
0	DFLE	**Display FIFO Load Enable:** Allow the display FIFO to be loaded from memory: 0 = Disable; 1 = Enable. If disabled, no write or read operations will occur to the display FIFO. If enabled, a flat panel should be powered down prior to setting this bit low. Similarly, if active, a CRT should be blanked prior to setting this bit low.
GX_BASE+8308h-830Bh		**DC_TIMING_CFG Register (R/W) (Locked)** **Default Value = xxx00000h**
31	VINT (RO)	**Vertical Interrupt (Read Only):** Is a vertical interrupt pending? 0 = No; 1 = Yes. This bit is provided to maintain backward compatibility with the VGA. It corresponds to VGA port 3C2h bit 7.
30	VNA (RO)	**Vertical Not Active (Read Only):** Is the active part of a vertical scan is in progress (i.e., retrace, blanking, or border)? 0 = Yes; 1 = No. This bit is provided to maintain backward compatibility with the VGA. It corresponds to VGA port 3BA/3DA bit 3.
29	DNA (RO)	**Display Not Active (Read Only):** Is the active part of a line is being displayed (i.e., retrace, blanking, or border)? 0 = Yes; 1 = No. This bit is provided to maintain backward compatibility with the VGA. It corresponds to VGA port 3BA/3DA bit 0.
28	RSVD	**Reserved:** Set to 0.
27	DDCI (RO)	**DDC Input (Read Only):** This bit returns the value from the DDCIN pin that should reflect the value from pin 12 of the VGA connector. It is used to provide support for the VESA Display Data Channel standard level DDC1.
26:20	RSVD	**Reserved:** Set to 0.
19:17	RSVD	**Reserved:** Set to 0.

Table 5-29. Display Controller Configuration and Status Registers (Continued)

Bit	Name	Description
16	BKRT	**Blink Rate**: 0 = Cursor blinks on every 16 frames for a duration of 8 frames (approximately 4 times per second) and VGA text characters will blink on every 32 frames for a duration of 16 frames (approximately 2 times per second). 1 = Cursor blinks on every 32 frames for a duration of 16 frames (approximately 2 times per second) and VGA text characters blink on every 64 frames for a duration of 32 frames (approximately 1 time per second). Blinking is enabled by BLNK bit 7.
15	PXDB	**Pixel Double:** Allow pixel doubling to stretch the displayed image in the horizontal dimension: 0 = Disable; 1 = Enable. If bit 15 is enabled, timing parameters should be programmed as if no pixel doubling is used, however, the frame buffer should be loaded with half the normal pixels per line. Also, the FB_LINE_SIZE parameter in DC_BUF_SIZE should be set for the number of bytes to be transferred for the line rather than the number displayed.
14	RSVD	**Reserved:** Set to 0.
13	PLNR	**VGA Planar Mode:** This bit must be set high for all VGA planar display modes.
12	FCEN	**Flat Panel Center:** Allows the border and active portions of a scan line to be qualified as “active” to a flat panel display via the ENADISP signal. This allows the use of a large border region for centering the flat panel display. 0 = Disable; 1 = Enable. When disabled, only the normal active portion of the scan line will be qualified as active.
11	FVSP	**Flat Panel Vertical Sync Polarity:** 0 = Causes TFT vertical sync signal to be normally low, generating a high pulse during sync interval. 1 = Causes TFT vertical sync signal to be normally high, generating a low pulse during sync interval.
10	FHSP	**Flat Panel Horizontal Sync Polarity**: 0 = Causes TFT horizontal sync signal to be normally low, generating a high pulse during sync interval. 1 = Causes TFT horizontal sync signal to be normally high, generating a low pulse during sync interval.
9	CVSP	**CRT Vertical Sync Polarity**: 0 = Causes CRT_VSYNC signal to be normally low, generating a high pulse during the retrace interval. 1 = Cause CRT_VSYNC signal to be normally high, generating a low pulse during the retrace interval.
8	CHSP	**CRT Horizontal Sync Polarity**: 0 = Causes CRT_HSYNC signal to be normally low, generating a high pulse during the retrace interval. 1 = Causes CRT_HSYNC signal to be normally high, generating a low pulse during the retrace interval.
7	BLNK	**Blink Enable:** Blink circuitry: 0 = Disable; 1 = Enable. If enabled, the hardware cursor will blink as well as any pixels. This is provided to maintain compatibility with VGA text modes. The blink rate is determined by the bit 16 (BKRT).
6	VIEN	**Vertical Interrupt Enable:** Generate a vertical interrupt on the occurrence of the next vertical sync pulse: 0 = Disable, vertical interrupt is cleared; 1 = Enable. This bit is provided to maintain backward compatibility with the VGA.
5	TGEN	**Timing Generator Enable:** Allow timing generator to generate the timing control signals for the display. 0 = Disable, the Timing registers may be reprogrammed, and all circuitry operating on the DCLK will be reset. 1 = Enable, no write operations are permitted to the Timing registers.
4	DDCK	**DDC Clock:** This bit is used to provide the serial clock for reading the DDC data pin. This bit is multiplexed onto the CRT_VSYNC pin, but in order for it to have an effect, the VSYE bit[1] must be set low to disable the normal vertical sync. Software should then pulse this bit high and low to clock data into the GX1 processor. This feature is provided to allow support for the VESA Display Data Channel standard level DDC1.

Table 5-29. Display Controller Configuration and Status Registers (Continued)

Bit	Name	Description
3	BLKE	**Blank Enable:** Allow generation of the composite blank signal to the display device: 0 = Disable; 1 = Enable. When disabled, the ENA_DISP output will be a static low level. This allows VESA DPMS compliance.
2	HSYE	**Horizontal Sync Enable:** Allow generation of the horizontal sync signal to a CRT display device: 0 = Disable; 1 = Enable. When disabled, the HSYNC output will be a static low level. This allows VESA DPMS compliance. Note that this bit only applies to the CRT; the flat panel HSYNC is controlled by the automatic power sequencing logic.
1	VSYE	**Vertical Sync Enable:** Allow generation of the vertical sync signal to a CRT display device: 0 = Disable; 1 = Enable. When disabled, the VSYNC output will be a static low level. This allows VESA DPMS compliance. Note that this bit only applies to the CRT; the flat panel VSYNC is controlled by the automatic power sequencing logic.
0	PPE	**Pixel Port Enable:** On a low-to-high transition this bit will enable the pixel port outputs. On a high-to-low transition, this bit will disable the pixel port outputs.
GX_BASE+830Ch-830Fh		**DC_OUTPUT_CFG Register (R/W) (Locked)** **Default Value = xxx00000h**
31:16	RSVD	**Reserved:** Set to 0.
15	DIAG	**Compressed Line Buffer Diagnostic Mode:** This bit allows testability of the Compressed Line Buffer via the diagnostic access registers. A low-to-high transition resets the Compressed Line Buffer write pointer. 0 = Disable (Normal operation); 1 = Enable.
14	CFRW	**Compressed Line Buffer Read/Write Select:** Enables the read/write address to the Compressed Line Buffer for use in diagnostic testing of the RAM. 0 = Write address enabled 1 = Read address enabled
13	PDEH	**Pixel Data Enable High**: 0 = The PIXEL [17:9] data bus to be driven to a logic low level.
12	PDEL	**Panel Data Enable Low**: 0 = This bit will cause the PIXEL[8:0] data bus to be driven to a logic low level.
11:8	RSVD	**Reserved:** Set to 0.
7:5	RSVD	**Reserved:** Set to 0.
4:3	RSVD	**Reserved:** Set to 0.
2	PCKE	**PCLK Enable**: 0 = PCLK is disabled and a low logic level is driven off-chip. 1 = Enable PCLK to be driven off-chip.
1	16FMT	**16-bpp Format:** Selects RGB display mode: 0 = RGB 5-6-5 mode 1 = RGB 5-5-5 display mode This bit is only significant if 8-bpp (OUTPUT_CONFIG, bit 0) is low, indicating 16-bpp mode.
0	8-bpp	**8-bpp / 16-bpp Select:** 0 = 16-bpp display mode is selected. 16FMT (OUTPUT_CONFIG, bit 1) will indicate the format of the 16-bit data.) 1 = 8-bpp display mode is selected. Used in VGA emulation.

5.5.9 Memory Organization Registers

The GX1 processor utilizes a graphics memory aperture that is up to 4 MB in size. The base address of the graphics memory aperture is stored in the DRAM controller Graphics Base Address register (see GBADD of MC_GBASE_ADD register, Table 5-15 on page 110). The graphics memory is made up of the normal uncompressed frame buffer, compressed display buffer, and cursor buffer. Each buffer begins at a programmable offset within the graphics memory aperture.

The various memory buffers are arranged so as to efficiently pack the data within the graphics memory aperture. The arrangement is programmable to efficiently accommodate different display modes. The cursor buffer is a linear block so addressing is straightforward. The frame buffer and compressed display buffer are arranged based upon scan lines. Each scan line has a maximum number of valid or active DWORDs, and a delta, which when added to the previous line offset, points to the next line. In this way, the buffers may either be stored as linear blocks, or as logical blocks as desired.

The Memory Organization registers group consists of six 32-bit registers located at GX_BASE+8310h-8328h. These registers are summarized in Table 5-28 on page 133, and Table 5-30 gives their bit formats.

Table 5-30. Display Controller Memory Organization Registers

Bit	Name	Description
GX_BASE+8310h-8313h	**DC_FB_ST_OFFSET Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:22	RSVD	**Reserved:** Set to 0.
21:0	FB_START _OFFSET	**Frame Buffer Start Offset:** This value represents the byte offset from the Graphics Base Address register (see GBADD of MC_GBASE_ADD register in Table 5-15 on page 110) of the starting location of the displayed frame buffer. This value may be changed to achieve panning across a virtual desktop or to allow multiple buffering. When this register is programmed to a nonzero value, the compression logic should be disabled. The memory address defined by bits [21:4] will take effect at the start of the next frame scan. The pixel offset defined by bits [3:0] will take effect immediately (in general, it should only change during vertical blanking).
GX_BASE+8314h-8317h	**DC_CB_ST_OFFSET Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:22	RSVD	**Reserved:** Set to 0.
21:0	CB_START _OFFSET	**Compressed Display Buffer Start Offset:** This value represents the byte offset from the Graphics Base Address register (see GBADD of MC_GBASE_ADD register in Table 5-15 on page 110) of the starting location of the compressed display buffer. Bits [3:0] must be programmed to zero so that the start offset is aligned to a 16-byte boundary. This value should change only when a new display mode is set due to a change in size of the frame buffer.
GX_BASE+8318h-831Bh	**DC_CUR_ST_OFFSET Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:22	RSVD	**Reserved:** Set to 0.
21:0	CUR_START _OFFSET	**Cursor Start Offset:** This register contains the byte offset from the Graphics Base Address register (see GBADD of MC_GBASE_ADD register in Table 5-15 on page 110) of the starting location of the cursor display pattern. Bits [1:0] should always be programmed to zero so that the start offset is DWORD aligned. The cursor data will be stored as a linear block of data.
GX_BASE+831Ch-831Fh	**Reserved**	**Default Value = 00000000h**
GX_BASE+8320h-8323h	**DC_VID_ST_OFFSET Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:22	RSVD	**Reserved:** Set to 0.
21:0	VID_START _OFFSET	**Video Buffer Start Offset Value:** This register contains the byte offset from the Graphics Base Address register (see GBADD of MC_GBASE_ADD register in Table 5-15 on page 110) of the starting location of the Video Buffer Start. Bits [3:0] must be programmed as zero so that the start offset is aligned to a 16 byte boundary. **Note:** This bit should only be modified during vertical retrace (see DC_TIMING_CFG bits 31 and 30, GX_BASE_8308h).

Table 5-30. Display Controller Memory Organization Registers (Continued)

Bit	Name	Description
GX_BASE+8324h-8327h	**DC_LINE_DELTA Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:23	RSVD	**Reserved:** Set to 0.
22:12	CB_LINE_ DELTA	**Compressed Display Buffer Line Delta:** This value represents number of DWORDs that, when added to the starting offset of the previous line, will point to the start of the next compressed line in memory. It is used to always maintain a pointer to the starting offset for the compressed display buffer line being loaded into the display FIFO.
11	RSVD	**Reserved:** Set to 0.
10:0	FB_LINE_ DELTA	**Frame Buffer Line Delta:** This value represents number of DWORDs that, when added to the starting offset of the previous line, will point to the start of the next frame buffer line in memory. It is used to always maintain a pointer to the starting offset for the frame buffer line being loaded into the display FIFO.
GX_BASE+8328h-832Bh	**DC_BUF_SIZE Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:30	RSVD	**Reserved:** Set to 0.
29:16	VID_BUF_ SIZE	**Video Buffer Size:** These bits set the video buffer size, in 64-byte segments. The maximum size is 1 MB. **Note:** This bit should only be modified during vertical retrace (see DC_TIMING_CFG bits 31 and 30, GX_BASE_8308h).
15:9	CB_LINE_ SIZE	**Compressed Display Buffer Line Size:** This value represents the number of DWORDs for a valid compressed line plus 1. It is used to detect an overflow of the compressed data FIFO. It should never be larger than 41h since the maximum size of the compressed data FIFO is 64 DWORDs.
8:0	FB_LINE_ SIZE	**Frame Buffer Line Size:** This value specifies the number of QWORD (8-byte segments) to transfer for each display line from the frame buffer. If panning is enabled, this value can generally be programmed to the displayed number of QWORD + 2 so that enough data is transferred to handle any possible alignment. Extra pixel data in the FIFO at the end of a line will automatically be discarded.
GX_BASE+832Ch-832Fh	**Reserved**	**Default Value = 00000000h**

5.5.10 Timing Registers

The Display Controller's timing registers control the generation of sync, blanking, and active display regions. They provide complete flexibility in interfacing to both CRT and flat panel displays. These registers will generally be programmed by the BIOS from an INT10h call or by the extended mode driver from a display timing file. Note that the horizontal timing parameters are specified in Character Clocks, which actually means pixels divided by 8, since all characters are bit mapped. For interlaced display the vertical counter will be incremented twice during each display line, so vertical timing parameters should be programmed with reference to the total frame rather than a single field.

The Timing registers group consists of six 32-bit registers located at GX_BASE+8330h-834Ch. These registers are summarized in Table 5-28 on page 133, and Table 5-31 gives their bit formats.

Table 5-31. Display Controller Timing Registers

Bit	Name	Description
GX_BASE+8330h-8333h	**DC_H_TIMING_1 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:19	H_TOTAL	**Horizontal Total:** The total number of Character Clocks for a given scan line minus 1. Note that the value is necessarily greater than the H_ACTIVE field because it includes border pixels and blanked pixels. For flat panels, this value will never change. The field [26:16] may be programmed with the pixel count minus 1, although bits [18:16] are ignored. The horizontal total is programmable on 8-pixel boundaries only.
18:16	IGRD	**Ignored**
15:11	RSVD	**Reserved:** Set to 0.
10:3	H_ACTIVE	**Horizontal Active:** The total number of Character Clocks for the displayed portion of a scan line minus 1. The field [10:0] may be programmed with the pixel count minus 1, although bits [2:0] are ignored. The active count is programmable on 8-pixel boundaries only. Note that for flat panels, if this value is less than the panel active horizontal resolution (H_PANEL), the parameters H_BLANK_START, H_BLANK_END, H_SYNC_START, and H_SYNC_END should be reduced by the value of H_ADJUST (or the value of H_PANEL – H_ACTIVE / 2) to achieve horizontal centering.
2:0	IGRD	**Ignored**
Note: For simultaneous CRT and flat panel display the H_ACTIVE and H_TOTAL parameters pertain to both.		
GX_BASE+8334h-8337h	**DC_H_TIMING_2 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:19	H_BLK_END	**Horizontal Blank End:** The Character Clock count at which the horizontal blanking signal becomes inactive minus 1. The field [26:16] may be programmed with the pixel count minus 1, although bits [18:16] are ignored. The blank end position is programmable on 8-pixel boundaries only.
18:16	IGRD	**Ignored**
15:11	RSVD	**Reserved:** Set to 0.
10:3	H_BLK_START	**Horizontal Blank Start:** The Character Clock count at which the horizontal blanking signal becomes active minus 1. The field [10:0] may be programmed with the pixel count minus 1, although bits [2:0] are ignored. The blank start position is programmable on 8-pixel boundaries only.
2:0	IGRD	**Ignored**
Note: A minimum of four Character Clocks are required for the horizontal blanking portion of a line in order for the timing generator to function correctly.		
GX_BASE+8338h-833Bh	**DC_H_TIMING_3 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:19	H_SYNC_END	**Horizontal Sync End:** The Character Clock count at which the CRT horizontal sync signal becomes inactive minus 1. The field [26:16] may be programmed with the pixel count minus 1, although bits [18:16] are ignored. The sync end position is programmable on 8-pixel boundaries only.
18:16	IGRD	**Ignored**
15:11	RSVD	**Reserved:** Set to 0.
10:3	H_SYNC_START	**Horizontal Sync Start:** The Character Clock count at which the CRT horizontal sync signal becomes active minus 1. The field [10:0] may be programmed with the pixel count minus 1, although bits [2:0] are ignored. The sync start position is programmable on 8-pixel boundaries only.
2:0	IGRD	**Ignored**
Note: This register should also be programmed appropriately for flat panel only display since the horizontal sync transition determines when to advance the vertical counter.		

Table 5-31. Display Controller Timing Registers (Continued)

Bit	Name	Description
GX_BASE+833Ch-833Fh	**DC_FP_H_TIMING Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:16	FP_H_SYNC_END	**Flat Panel Horizontal Sync End:** The pixel count at which the flat panel horizontal sync signal becomes inactive minus 1.
15:11	RSVD	**Reserved:** Set to 0.
10:0	FP_H_SYNC_START	**Flat Panel Horizontal Sync Start:** The pixel count at which the flat panel horizontal sync signal becomes active minus 1.
Note: These values are specified in pixels rather than Character Clocks to allow precise control over sync position. For flat panels which combine two pixels per panel clock, these values should be odd numbers (even pixel boundary) to guarantee that the sync signal will meet proper setup and hold times.		
GX_BASE+8340h-8343h	**DC_V_TIMING_1 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:16	V_TOTAL	**Vertical Total:** The total number of lines for a given frame scan minus 1. The value is necessarily greater than the V_ACTIVE field because it includes border lines and blanked lines. If the display is interlaced, the total number of lines must be odd, so this value should be an even number.
15:11	RSVD	**Reserved:** Set to 0.
10:0	V_ACTIVE	**Vertical Active:** The total number of lines for the displayed portion of a frame scan minus 1. For flat panels, if this value is less than the panel active vertical resolution (V_PANEL), the parameters V_BLANK_START, V_BLANK_END, V_SYNC_START, and V_SYNC_END should be reduced by the following value (V_ADJUST) to achieve vertical centering: V_ADJUST = (V_PANEL – V_ACTIVE) / 2. If the display is interlaced, the number of active lines should be even, so this value should be an odd number.
Note: These values are specified in lines.		
GX_BASE+8344h-8347h	**DC_V_TIMING_2 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:16	V_BLANK_END	**Vertical Blank End:** The line at which the vertical blanking signal becomes inactive minus 1. If the display is interlaced, no border is supported, so this value should be identical to V_TOTAL.
15:11	RSVD	**Reserved:** Set to 0.
10:0	V_BLANK_START	**Vertical Blank Start:** The line at which the vertical blanking signal becomes active minus 1. If the display is interlaced, this value should be programmed to V_ACTIVE plus 1.
Note: These values are specified in lines. For interlaced display, no border is supported, so blank timing is implied by the total/active timing.		
GX_BASE+8348h-834Bh	**DC_V_TIMING_3 Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:16	V_SYNC_END	**Vertical Sync End:** The line at which the CRT vertical sync signal becomes inactive minus 1.
15:11	RSVD	**Reserved:** Set to 0.
10:0	V_SYNC_START	**Vertical Sync Start:** The line at which the CRT vertical sync signal becomes active minus 1. For interlaced display, note that the vertical counter is incremented twice during each line and since there are an odd number of lines, the vertical sync pulse will trigger in the middle of a line for one field and at the end of a line for the subsequent field.
Note: These values are specified in lines.		
GX_BASE+834Ch-834Fh	**DC_FP_V_TIMING Register (R/W) (Locked)**	**Default Value = xxxxxxxxh**
31:27	RSVD	**Reserved:** Set to 0.
26:16	FP_V_SYNC_END	**Flat Panel Vertical Sync End:** The line at which the flat panel vertical sync signal becomes inactive minus 2. Note that the internal flat panel vertical sync is latched by the flat panel horizontal sync prior to being output to the panel.
15:11	RSVD	**Reserved:** Set to 0.
10:0	FP_VSYNC_START	**Flat Panel Vertical Sync Start:** The line at which the internal flat panel vertical sync signal becomes active minus 2. Note that the internal flat panel vertical sync is latched by the flat panel horizontal sync prior to being output to the panel.
Note: These values are specified in lines.		

5.5.11 Cursor and Line Compare Registers

The Cursor position registers contain pixel coordinate information for the cursor. These values are not latched by the timing generator until the start of the frame to avoid tearing artifacts when moving the cursor.

The Cursor Position group consists of two 32-bit registers located at GX_BASE+8350h and GX_BASE+8358h. These registers are summarized in Table 5-28 on page 133, and Table 5-32 gives their bit formats.

Table 5-32. Display Controller Cursor and Line Compare Registers

Bit	Name	Description
GX_BASE+8350h-8353h	**DC_CURSOR_X Register (R/W)**	**Default Value = xxxxxxxxh**
31:16	RSVD	**Reserved:** Set to 0.
15:11	X_OFFSET	**X Offset:** The X pixel offset within the 32x32 cursor pattern at which the displayed portion of the cursor is to begin. Normally, this value is set to zero to display the entire cursor pattern, but for cursors for which the "hot spot" is not at the left edge of the pattern, it may be necessary to display the rightmost pixels of the cursor only as the cursor moves close to the left edge of the display.
10:0	CURSOR_X	**Cursor X:** The X coordinate of the pixel at which the upper left corner of the cursor is to be displayed. This value is referenced to the screen origin (0,0) which is the pixel in the upper left corner of the screen.
GX_BASE+8354h-8357h	**DC_V_LINE_CNT Register (RO)**	**Default Value = xxxxxxxxh**
31:11	RSVD	**Reserved (Read Only)**
10:0	V_LINE _CNT (RO)	**Vertical Line Count (Read Only):** This value is the current scanline of the display.
Note: The value in this register is driven directly off of the DCLK, and is not synchronized with the CPU clock. Software should read this register twice and compare the two results to ensure that the value is not in transition.		
GX_BASE+8358h-835Bh	**DC_CURSOR_Y Register (R/W)**	**Default Value = xxxxxxxxh**
31:16	RSVD	**Reserved:** Set to 0.
15:11	Y_OFFSET	**Y Offset:** The Y line offset within the 32x32 cursor pattern at which the displayed portion of the cursor is to begin. Normally, this value is set to zero to display the entire cursor pattern, but for cursors for which the "hot spot" is not at the top edge of the pattern, it may be necessary to display the bottommost lines of the cursor only as the cursor moves close to the top edge of the display. If this value is nonzero, the CUR_START_OFFSET must be set to point to the first cursor line to be displayed.
10	RSVD	**Reserved:** Set to 0.
9:0	CURSOR_Y	**Cursor Y:** The Y coordinate of the line at which the upper left corner of the cursor is to be displayed. This value is referenced to the screen origin (0,0) which is the pixel in the upper left corner of the screen. This field is alternately used as the line-compare value for a newly-programmed frame buffer start offset. This is necessary for VGA programs that change the start offset in the middle of a frame. In order to use this function, the hardware cursor function should be disabled.
GX_BASE+835Ch-835Fh	**DC_SS_LINE_CMP Register (R/W)**	**Default Value = xxxxxxxxh**
31:11	RSVD	**Reserved:** Set to 0.
10:0	SS_LINE _CMP	**Split-Screen Line Compare:** This is the line count at which the lower screen begins in a VGA split-screen mode.
Note: When the internal line counter hits this value, the frame buffer address is reset to 0. This function is enabled with the SSLC bit in the DC_GENERAL_CFG register (see Table 5-29 on page 136).		

5.5.12 Palette Access Registers

These registers are used for accessing the internal palette RAM and extensions. In addition to the standard 256 entries for 8-bpp color translation, the GX1 processor palette has extensions for cursor colors and overscan (border) color.

The Palette Access register group consists of two 32-bit registers located at GX_BASE+8370h and GX_BASE+8374h. These registers are summarized in Table 5-28 on page 133, and Table 5-33 gives their bit formats.

Table 5-33. Display Controller Palette

Bit	Name	Description
GX_BASE+8370h-8373h		**DC_PAL_ADDRESS Register (R/W)** **Default Value = xxxxxxxxh**
31:9	RSVD	**Reserved:** Set to 0.
8:0	PALETTE _ADDR	**Palette Address:** The address to be used for the next access to the DC_PAL_DATA register. Each access to the data register automatically increments the palette address register. If non-sequential access is made to the palette, the address register must be loaded between each non-sequential data block. The address ranges are as follows. Address — Color 0h - FFh — Standard Palette Colors 100h — Cursor Color 0 101h — Cursor Color 1 102h — Reserved 103h — Reserved 104h — Overscan (Color Border) 105h - 1FFh — Not Valid
GX_BASE+8374h-8377h		**DC_PAL_DATA Register (R/W)** **Default Value = xxxxxxxxh**
31:18	RSVD	**Reserved:** Set to 0.
17:0	PALETTE _DATA	**Palette Data:** The read or write data for a palette access.[1]

1. When a read or write to the palette RAM occurs, the previous output value is held for one additional DCLK period. This effect should go unnoticed and provides for sparkle-free update. Prior to a read or write to this register, the DC_PAL_ADDRESS register should be loaded with the appropriate address. The address automatically increments after each access to this register, so for sequential access, the address register need only be loaded once

5.5.13 FIFO Diagnostic Registers

The FIFO Diagnostic register group consists of two 32-bit registers located at GX_BASE+8378h and GX_BASE+837Ch. These registers are summarized in Table 5-28 on page 133, and Table 5-34 gives their bit formats.

Table 5-34. FIFO Diagnostic Registers

Bit	Name	Description
GX_BASE+8378h-837Bh		**DC_DFIFO_DIAG Register (R/W)** **Default Value = xxxxxxxxh**
31:0	DISPLAY FIFO DIAGNOSTIC DATA	**Display FIFO Diagnostic Read or Write Data:** Before this register is accessed, the DIAG bit in DC_GENERAL_CFG register (see Table 5-29 on page 136) should be set high and the DFLE bit should be set low. Since, each FIFO entry is 64 bits, an even number of write operations should be performed. Each pair of write operations will cause the FIFO write pointer to increment automatically. After all write operations have been performed, a single read of don't care data should be performed to load data into the output latch. Each subsequent read will contain the appropriate data which was previously written. Each pair of read operations will cause the FIFO read pointer to increment automatically. A pause of at least four core clocks should be allowed between subsequent read operations to allow adequate time for the shift to take place.
GX_BASE+837Ch-837Fh		**DC_CFIFO_DIAG Register (R/W)** **Default Value = xxxxxxxxh**
31:0	COMPRESSED FIFO DIAGNOSTIC DATA	**Compressed Data FIFO Diagnostic Read or Write Data:** Before this register is accessed, the DIAG bit in DC_GENERAL_CFG (see Table 5-29 on page 136) register should be set high and the DFLE bit should be set low. Also, the DIAG bit in DC_OUTPUT_CFG (see Table 5-29 on page 139) should be set high and the CFRW bit in DC_OUTPUT_CFG should be set low. After each write, the FIFO write pointer will automatically increment. After all write operations have been performed, the CFRW bit of DC_OUTPUT_CFG should be set high to enable read addresses to the FIFO and a single read of don't care data should be performed to load data into the output latch. Each subsequent read will contain the appropriate data which was previously written. After each read, the FIFO read pointer will automatically increment.

5.5.14 CS5530A Display Controller Interface

As previously stated in Section 2.7 "AMD Geode™ GX1/ CS5530A System Designs" on page 18, the GX1 processor interfaces with the Geode CS5530A companion device. This section will discuss the specifics on signal connections between the two devices with regards to the display controller.

Because the GX1 processor is used in a system with the CS5530A companion device, the need for an external RAMDAC is eliminated. The CS5530A contains the DACs, a video accelerator engine, and a TFT interface.

A GX1 processor and CS5530A-based system supports both flat panel and CRT configurations. Figure 5-16 shows the signal connections for both types of systems.

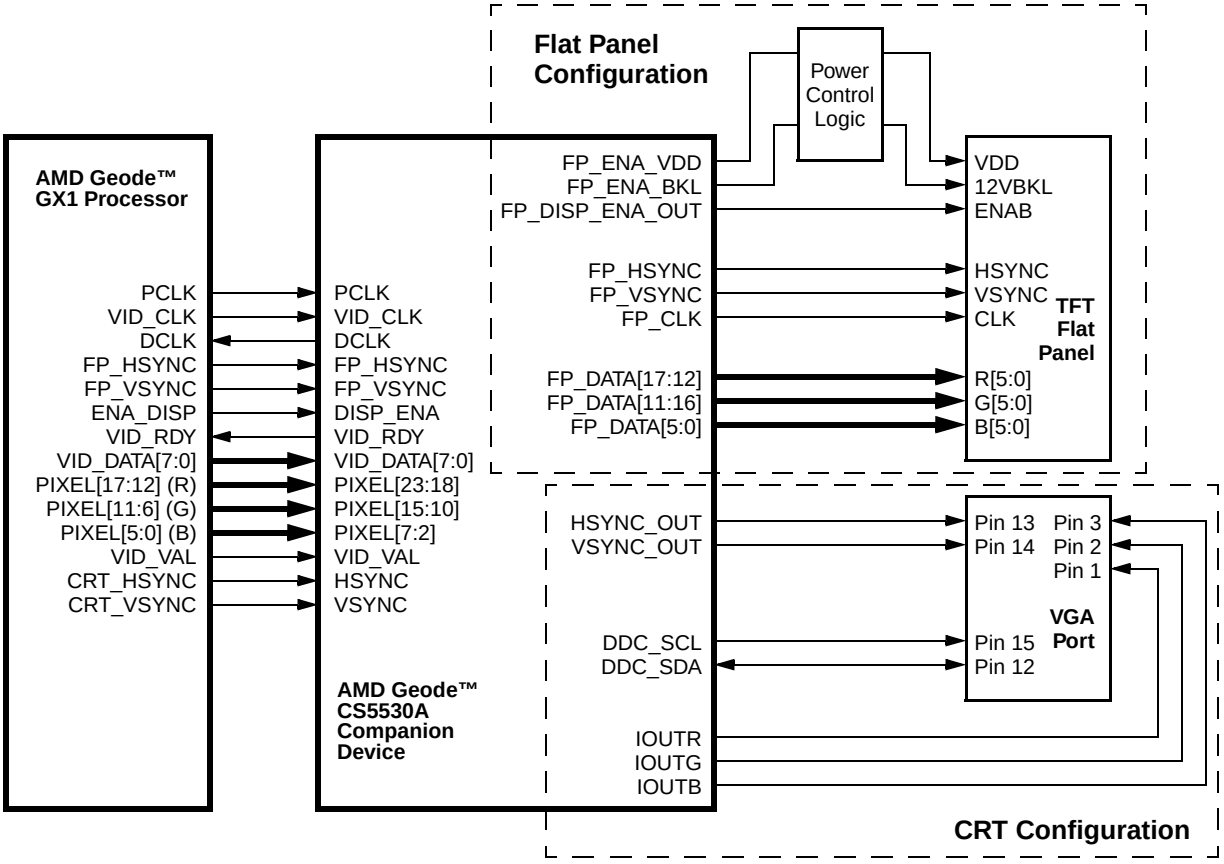


Figure 5-16. Display Controller Signal Connections

5.5.14.1 CS5530A Video Port Data Transfer

VID_VAL indicates that the GX1 processor has placed valid data on VID_DATA[7:0]. VID_RDY indicates that the CS5530A is ready to accept the next byte of video data.

VID_DATA[7:0] is advanced when both VID_VAL and VID_RDY are asserted. VID_RDY is driven one clock early to the GX1 processor while VID_VAL is driven coincident with VID_DATA[7:0]. A sample interface functional timing diagram is shown in Figure 5-17.

5.5.14.2 Video Port Maximum Transfer

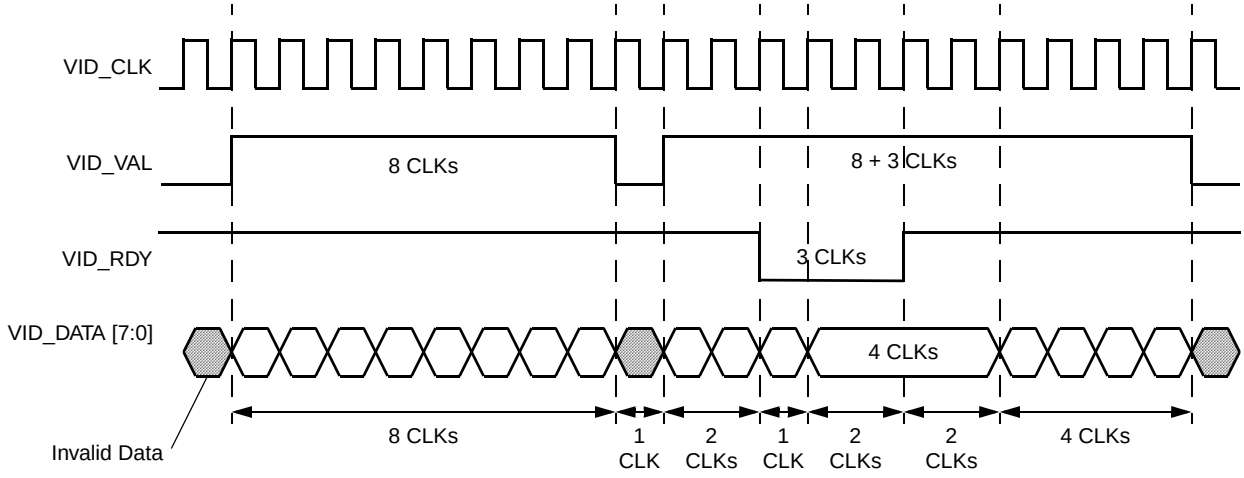
The video port can transfer 8 bytes per 9 clocks, or 4 video pixels per 9 video clocks. The total time available to fill a line buffer is H_{TOTAL} divided by the Dot Clock frequency. The time needed to fill the line buffer is (video_width)/(4/9 VID_CLK). Solving these two equations together:

$$\text{max\_video\_width} = \frac{4 \times \text{VID\_CLK} \times H_{TOTAL}}{9 \times \text{DCLK}}$$

At the higher resolutions and at 300 and 333 MHz CPU speeds, the video port can not be used. At 300 and 333 MHz, the video port must be set to run at 1/4 CPU speed instead of 1/2 CPU speed. For example, at a 1024x768x85 Hz graphics resolution and a 720x480 video frame buffer size:

$$\text{max\_video\_width} = \frac{4 \times 75 \times 1376}{9 \times 94.5} = 485 \text{ pixels}$$

Since 720 pixels is needed, 485 pixels is not enough.



Note: VID_CLK = CORE_CLK/2

Figure 5-17. Video Port Data Transfer (CS5530A)

5.6 Virtual VGA Subsystem

This section describes the Virtual System Architecture™ (VSA) technology implemented with the AMD Geode™ GX1 processor and VSA enhanced Geode CS5530A companion device. VSA provides a framework to enable software implementation of traditionally hardware-only components. VSA software executes in System Management Mode (SMM), enabling it to execute transparently to the operating system, drivers and applications.

The VSA design is based on a simple model for replacing hardware components with software. Hardware to be virtualized is merely replaced with simple access detection circuitry which asserts the processor's SMI# pin when hardware accesses are detected. The current execution stream is immediately preempted, and the processor enters SMM. The SMM system software then saves the processor state, initializes the VSA execution environment, decodes the SMI source and dispatches handler routines which have registered requests to service the decoded SMI source. Once all handler routines have completed, the processor state is restored and normal execution resumes. In this manner, hardware accesses are transparently replaced with the execution of SMM handler software.

Historically, SMM software was used primarily for the single purpose of facilitating active power management for notebook designs. That software's only function was to manage the power up and down of devices to save power. With high performance processors now available, it is feasible to implement, primarily in SMM software, PC capabilities traditionally provided by hardware. In contrast to power management code, this virtualization software generally has strict performance requirements to prevent application performance from being significantly impacted.

Several functions can be virtualized in a GX1 processor based design using the VSA environment. The VSA enhanced Geode CS5530A companion device provides programmable resources to trap both memory and I/O accesses. However, specific hardware is included to support the virtualization of VGA core compatibility and audio functionality in the system.

The hardware support for VGA emulation resides completely inside the GX1 processor. Legacy VGA accesses do not generate off-chip bus cycles. However, the VSA support hardware for the XpressAUDIO™ subsystem resides in the AMD Geode™ CS5530A companion device.

5.6.1 Traditional VGA Hardware

A VGA card consists of display memory and control registers. The VGA display memory shows up in system memory between addresses A0000h and BFFFFh. It is possible to map this memory to three different ranges within this 128 KB block.

The first range is
A0000h to AFFFFh for EGA and VGA modes,

the second range is
B0000h to B7FFFh for MDA modes,

and the third range is
B8000h to BFFFFh for CGA modes.

The VGA control registers are mapped to the I/O address range from 3B0h to 3DFh. The VGA registers are accessed with an indexing scheme that provides more registers than would normally fit into this range. Some registers are mapped at two locations, one for monochrome, and another for color.

The VGA hardware can be accessed by calling BIOS routines or by directly writing to VGA memory and control registers. DOS always calls BIOS to set up the display mode and render characters. Many other applications access the VGA memory and control registers directly. The VGA card can be set up to a virtually unlimited number of modes. However, many applications use one of the predefined modes specified by the BIOS routine which sets up the display mode. The predefined modes are translated into specific VGA control register setups by the BIOS. The standard modes supported by VGA cards are shown in Table 5-35 on page 149.

Table 5-35. Standard VGA Modes

Category	Mode	Text or Graphics	Resolution	Format	Type
Software	0,1	Text	40x25	Characters	CGA
	2,3	Text	80x25	Characters	CGA
	4,5	Graphics	320x200	2-bpp	CGA
	6	Graphics	640x200	1-bpp	CGA
	7	Text	80x25	Characters	MDA
Hardware	0Dh	Graphics	320x200	4-bpp	EGA
	0Eh	Graphics	640x200	4-bpp	EGA
	0Fh	Graphics	640x350	1-bpp	EGA
	10h	Graphics	640x350	4-bpp	EGA
	11h	Graphics	640x480	1-bpp	VGA
	12h	Graphics	640x480	4-bpp	VGA
	13h	Graphics	320x200	8-bpp	VGA

A VGA is made up of several functional units.

- The **frame buffer** is 256 KB of memory that provides data for the video display. It is organized as 64K 32-bit DWORDs.
- The **sequencer** decomposes word and DWORD CPU accesses into byte operations for the graphics controller. It also controls a number of miscellaneous functions, including reset and some clocking controls.
- The **graphics controller** provides most of the interface between CPU data and the frame buffer. It allows the programmer to read and write frame buffer data in different formats. Plus provides ROP (raster operation) and masking functions.
- The **CRT controller** provides video timing signals and address generation for video refresh. It also provides a text cursor.
- The **attribute controller** contains the video refresh datapath, including text rasterization and palette lookup.
- The **general registers** provide status information for the programmer as well as control over VGA-host address mapping and clock selection. This is all handled in hardware by the graphics pipeline.

It is important to understand that a VGA is constructed of numerous independent functions. Most of the register fields correspond to controls that were originally built out of discrete logic or were part of a dedicated controller such as the 6845. The notion of a VGA "mode" is a higher-level convention to denote a particular set of values for the registers. Many popular programs do not use standard modes, preferring instead to produce their own VGA setups that are optimal for their purposes.

5.6.1.1 VGA Memory Organization

The VGA memory is organized as 64K 32-bit DWORDs. This organization is usually presented as four 64 KB "planes". A plane consists of one byte out of every DWORD. Thus, plane 0 refers to the least significant byte from every one of the 64K DWORDs. The addressing granularity of this memory is a DWORD, not a byte; that is, consecutive addresses refer to consecutive DWORDs. The only provision for byte-granularity addressing is the four-byte enable signals used for writes. In C parlance,

```
single_plane_byte = (dword_fb[address] >>
(plane * 8)) & 0xFF;
```

When dealing with VGA, it is important to recognize the distinction between host addresses, frame buffer addresses, and the refresh address pipe. A VGA controller contains a lot of hardware to translate between these address spaces in different ways, and understanding these translations is critical to understanding the entire device. In standard four-plane graphics modes, a frame-buffer DWORD provides eight 4-bit pixels. The left-most pixel comes from bit 7 of each plane, with plane 3 providing the most significant bit.

```
pixel[i].bit[j] = dword_fb[address].bit[i*8 + (7-j)]
```

5.6.1.2 VGA Front End

The VGA front end consists of address and data translations between the CPU and the frame buffer. This functionality is contained within the graphics controller and sequencer components. Most of the front end functionality is implemented in the VGA read and write hardware of the GX1 processor. An important axiom of the VGA is that the front end and back end are controlled independently. There are no register fields that control the behavior of both pieces. Terms like "VGA odd/even mode" are therefore somewhat misleading; there are two different controls for odd/even functionality in the front end, and two separate controls in the refresh path to cause "sensible" refresh behavior for frame buffer contents written in odd/even mode. Normally, all these fields would be set up together, but they don't have to be. This sort of orthogonal behavior gives rise to the enormous number of possible VGA "modes". The CPU end of the read and write pipelines is one byte wide. WORD and DWORD accesses from the CPU to VGA memory are broken down into multiple byte accesses by the sequencer. For example, a WORD write to A0000h (in a VGA graphics mode) is processed as if it were two-byte write operations to A0000h and A0001h.

5.6.1.3 Address Mapping

When a VGA card sees an address on the host bus, bits [31:15] determine whether the transaction is for the VGA. Depending on the mode, addresses 000AXXXX, 000B{0xxx}XXX, or 000B{1xxx}XXX can decode into VGA space. If the access is for the VGA, bits [15:0] provide the DWORD address into the frame buffer (see odd/even and Chain 4 modes, next paragraph). Thus, each byte address on the host bus addresses a DWORD in VGA memory.

On a write transaction, the byte enables are normally driven from the sequencer's MapMask register. The VGA has two other write address mappings that modify this behavior. In odd/even (Chain 2) write mode, bit 0 of the address is used to enable bytes 0 and 2 (if zero) or bytes 1 and 3 (if one). In addition, the address presented to the frame buffer has bit 0 replaced with the PageBit field of the Miscellaneous Output register. Chain 4 write mode is similar; only one of the four byte enables is asserted, based on bits [1:0] of the address, and bits [1:0] of the frame buffer address are set to zero. In each of these modes, the MapMask enables are logically ANDed into the enables that result from the address.

5.6.1.4 Video Refresh

VGA refresh is controlled by two units: the CRT controller (CRTC) and the attribute controller (ATTR). The CRTC provides refresh addresses and video control; the ATTR provides the refresh datapath, including pixel formatting and internal palette lookup.

The VGA back end contains two basic clocks: the Dot Clock (or pixel clock) and the Character Clock. The Clock-Select field of the Miscellaneous Output register selects a "master clock" of either 25 MHz or 28 MHz. This master clock, optionally divided by two, drives the Dot Clock. The

Character Clock is simply the Dot Clock divided by eight or nine.

The VGA supports four basic pixel formats. Using text format, the VGA interprets frame buffer values as ASCII characters, foreground/background attributes, and font data. The other three formats are all “graphics modes”, known as APA (All Points Addressable) modes. These formats could be called CGA-compatible (odd/even 4-bpp), EGA-compatible (4-plane 4-bpp), and VGA-compatible (pixel-per-byte 8-bpp). The format is chosen by the ShiftRegister field of the Graphics Controller Mode register.

The refresh address pipe is an integral part of the CRTC, and has many configuration options. Refresh can begin at any frame buffer address. The display width and the frame buffer pitch (scan-line delta) are set separately. Multiple scan lines can be refreshed from the same frame buffer addresses. The LineCompare register causes the refresh address to be reset to zero at a particular scan line, providing support for vertical split-screen.

Within the context of a single scan line, the refresh address increments by one on every Character Clock. Before being presented to the frame buffer, refresh addresses can be shifted by 0, 1, or 2 bits to the left. These options are often mis-named BYTE, WORD, and DWORD modes. Using this shifter, the refresh unit can be programmed to skip one out of two or three out of four DWORDs of refresh data. As an example of the utility of this function, consider Chain 4 mode, described in Section 5.6.1.3 "Address Mapping" on page 150. Pixels written in Chain 4 mode occupy one out of every four DWORDs in the frame buffer. If the refresh path is put into “Doubleword” mode, the refresh will come only from those DWORDs writable in Chain 4. This is how VGA mode 13h works.

In text mode, the ATTR has a lot of work to do. At each Character Clock, it pulls a DWORD of data out of the frame buffer. In that DWORD, plane 0 contains the ASCII character code, and plane 1 contains an attribute byte. The ATTR uses plane 0 to generate a font lookup address and read another DWORD. In plane 2, this DWORD contains a bit-per-pixel representation of one scan line in the appropriate character glyph. The ATTR transforms these bits into eight pixels, obtaining foreground and background colors from the attribute byte. The CRTC must refresh from the same memory addresses for all scan lines that make up a character row; within that row, the ATTR must fetch successive scan lines from the glyph table so as to draw proper characters. Graphics modes are somewhat simpler. In CGA-compatible mode, a DWORD provides eight pixels. The first four pixels come from planes 0 and 2; each 4-bit pixel gets bits [3:2] from plane 2, and bits [1:0] from plane 0. The remaining four pixels come from planes 1 and 3. The EGA-compatible mode also gets eight pixels from a DWORD, but each pixel gets one bit from each plane, with plane 3 providing bit 3. Finally, VGA-compatible mode gets four pixels from each DWORD; plane 0 provides the first pixel, plane 1 the next, and so on. The 8-bpp mode uses an option to provide every pixel for two Dot Clocks, thus allowing the refresh pipe to keep up (it only increments on Character Clocks) and meaning that the 320-pixel-wide mode 13h really has 640 visible pixels per line. The VGA color model is unusual. The ATTR contains a 16-entry color palette with 6 bits per entry. Except for 8-bpp modes, all VGA configurations drive four bits of pixel data into the palette, which produces a 6-bit result. Based on various control registers, this value is then combined with other register contents to produce an 8-bit index into the DAC. There is a ColorPlaneEnable register to mask bits out of the pixel data before it goes to the palette; this is used to emulate four-color CGA modes by ignoring the top two bits of each pixel. In 8-bpp modes, the palette is bypassed and the pixel data goes directly to the DAC.

5.6.1.5 VGA Video BIOS

The video BIOS supports the VESA BIOS Extensions (VBE) Version 1.2 and 2.0, as well as all standard VGA BIOS calls. It interacts with Virtual VGA through the use of several extended VGA registers. These are virtual registers contained in the VSA code for Virtual VGA. (These registers are defined in a separate document.)

5.6.2 Virtual VGA

The GX1 processor reduces the burden of legacy hardware by using a balanced mix of hardware and software to provide the same functionality. The graphics pipeline contains full hardware support for the VGA “front-end”, the logic that controls read and write operations to the VGA frame buffer (located in graphics memory). For some modes, the hardware can also provide direct display of the data in the VGA buffer. Virtual VGA traps frame buffer accesses only when necessary, but it must trap all VGA I/O accesses to maintain the VGA state and properly program the graphics pipeline and display controller.

The processor core contains SMI generation hardware for VGA memory write operations. The bus controller contains SMI generation hardware for VGA I/O read and write operations. The graphics pipeline contains hardware to detect and process reads and writes to VGA memory. VGA memory is partitioned from system memory.

VGA functionality with the GX1 processor includes the standard VGA modes (VGA, EGA, CGA, and MDA) as well as the higher-resolution VESA modes. The CGA and MDA modes (modes 0 through 7) require that Virtual VGA convert the data in the VGA buffer to a separate 8-bpp frame buffer that the hardware can use for display refresh.

The remaining modes, VGA, EGA, and VESA, can be displayed directly by the hardware, with no data conversion required. For these modes, Virtual VGA often outperforms typical VGA cards because the frame buffer data does not travel across an external bus.

Display drivers for popular GUI (graphical user interface) based operating systems are provided by AMD that enable a full featured 2D hardware accelerator to be used instead of the emulated VGA core.

5.6.2.1 Datapath Elements

The graphics controller contains several elements that convert between host data and frame buffer data.

The rotator simply rotates the byte written from the host by 0 to 7 bits to the right, based on the RotateCount field of the DataRotate register. It has no effect in the read path.

The display latch is a 32-bit register that is loaded on every read access to the frame buffer. All 32 bits of the frame buffer DWORDs are loaded into the latch.

The **write-mode unit** converts a byte from the host into a 32-bit value. A VGA has four write modes:

- Write Mode 0:
  - Bit n of byte b comes from one of two places, depending on bit b of the EnableSetReset register. If that bit is zero, it comes from bit n of the host data. If that bit is one, it comes from bit b of the SetReset register. This mode allows the programmer to set some planes from the host data and the others from SetReset.
- Write Mode 1:
  - All 32 bits come directly out of the display latch; the host data is ignored. This mode is used for screen-to-screen copies.
- Write Mode 2:
  - Bit n of byte b comes from bit b of the host data; that is, the four LSBs of the host data are each replicated through a byte of the result. In conjunction with the BitMask register, this mode allows the programmer to directly write a 4-bit color to one or more pixels.
- Write Mode 3:
  - Bit n of byte b comes from bit b of the SetReset register. The host data is ANDed with the BitMask register to provide the bit mask for the write (see below).

The **read mode unit** converts a 32-bit value from the frame buffer into a byte. A VGA has two read modes:

- Read Mode 0:
  - One of the four bytes from the frame buffer is returned, based on the value of the ReadMapSelect register. In Chain 4 mode, bits [1:0] of the read address select a plane. In odd/even read mode, bit 0 of the read address replaces bit 0 of ReadMapSelect.
- Read Mode 1:
  - Bit n of the result is set to 1 if bit n in every byte b matches bit b of the ColorCompare register; otherwise it is set to 0. There is a ColorDon'tCare register that can exclude planes from this comparison. In four-plane graphics modes, this provides a conversion from 4-bpp to 1-bpp.

The ALU is a simple two-operand ROP unit that operates on writes. Its operating modes are COPY, AND, OR, and XOR. The 32-bit inputs are:

1) the output of the write-mode unit and
2) the display latch (not necessarily the value at the frame buffer address of the write).

An application that wishes to perform ROPs on the source and destination must first byte read the address (to load the latch) and then immediately write a byte to the same address. The ALU has no effect in Write Mode 1.

The bit mask unit does not provide a true bit mask. Instead, it selects between the ALU output and the display latch. The mask is an 8-bit value, and bit n of the mask makes the selection for bit n of all four bytes of the result (a zero selects the latch). No bit masking occurs in Write Mode 1.

The VGA hardware of the GX1 processor does not implement Write Mode 1 directly, but it can be indirectly implemented by setting the BitMask to zero and the ALU mode to COPY. This is done by the SMM code so there are no compatibility issues with applications.

5.6.2.2 GX1 VGA Hardware

The GX1 processor core contains hardware to detect VGA accesses and generate SMI interrupts. The graphics pipeline contains hardware to detect and process reads and writes to VGA memory. The VGA memory on the GX1 processor is partitioned from system memory. The GX1 processor has the following hardware components to assist the VGA emulation software:

- SMI Generation
- VGA Range Detection
- VGA Sequencer
- VGA Write/Read Path
- VGA Address Generator
- VGA Memory

5.6.2.3 SMI Generation

VGA emulation software is notified of VGA memory accesses by an SMI generated in dedicated circuitry in the processor core that detects and traps memory accesses. The SMI generation hardware for VGA memory addresses is in the second stage of instruction decoding on the processor core. This is the earliest stage of instruction decode where virtual addresses have been translated to physical addresses. Trapping after the execution stage is impractical, because memory write buffering will allow subsequent instructions to execute.

The VGA emulation code requires the SMI to be generated immediately when a VGA access occurs. The SMI generation hardware can optionally exclude areas of VGA memory, based on a 32-bit register which has a control bit for each 2 KB region of the VGA memory window. The control bit determines whether or not an SMI interrupt is generated for the corresponding region. The purpose of this hardware is to allow the VGA emulation software to disable SMI interrupts in VGA memory regions that are not currently displayed.

For direct display modes (8-bpp or 16-bpp) in the display controller, Virtual VGA can operate without SMI generation.

The SMI generation circuit on the GX1 processor has configuration registers to control and mask SMI interrupts in the VGA memory space.

5.6.2.4 VGA Range Detection

The VGA range detection circuit is similar to the SMI generation hardware, however, it resides in the internal bus interface address mapping unit. The purpose of this hardware is to notify the graphics pipeline when accesses to the VGA memory range A0000h to BFFFFh are detected. The graphics pipeline has VGA read and write path hardware to process VGA memory accesses. The VGA range detection can be configured to trap VGA memory accesses in one or more of the following ranges: A0000h to AFFFFh (EGA,VGA), B0000h to B7FFFh (MDA), or B8000h to BFFFFh (CGA).

5.6.2.5 VGA Sequencer

The VGA sequencer is located at the front end of the graphics pipeline. The purpose of the VGA sequencer is to divide up multiple-byte read and write operations into a sequence of single-byte read and write operations. 16-bit or 32-bit X-bus write operations to VGA memory are divided into 8-bit write operations and sent to the VGA write path. 16-bit or 32-bit X-bus read operations from VGA memory are accumulated from 8-bit read operations over the VGA read path. The sequencer generates the lower two bits of the address.

5.6.2.6 VGA Write/Read Path

The VGA write path implements standard VGA write operations into VGA memory. No SMI is generated for write path operations when the VGA access is not displayed. When the VGA access is displayed, an SMI is generated so that the SMI emulation can update the frame buffer. The VGA write path converts 8-bit write operations from the sequencer into 32-bit VGA memory write operations. The operations performed by the VGA write path include data rotation, raster operation (ALU), bit masking, plane select, plane enable, and write modes.

The VGA read path implements standard VGA read operations from VGA memory. No SMI is needed for read-path operations. The VGA read path converts 32-bit read operations from VGA memory to 8-bit data back to the sequencer. The basic operations performed by the VGA read path include color compare, plane-read select, and read modes.

5.6.2.7 VGA Address Generator

The VGA address generator translates VGA memory addresses up to the address where the VGA memory resides on the GX1 processor. The VGA address generator requires the address from the VGA access (A0000h to BFFFFh), the base of the VGA memory on the GX1 processor, and various control bits. The control bits are necessary because addressing is complicated by odd/even and Chain 4 addressing modes.

5.6.2.8 VGA Memory

The VGA memory requires 256 KB of memory organized as 64 KB by 32 bits. The VGA memory is implemented as part of system memory. The GX1 processor partitions system memory into two areas, normal system memory and graphics memory. System memory is mapped to the normal physical address of the DRAM, starting at zero and ending at memory size. Graphics memory is mapped into high physical memory, contiguous to the registers and dedicated cache of the GX1 processor. The graphics memory includes the frame buffer, compression buffer, cursor memory, and VGA memory. The VGA memory is mapped on a 256 KB boundary to simplify the address generation

5.6.3 VGA Configuration Registers

SMI generation can be configured to trap VGA memory accesses in one of the following ranges:

A0000h to AFFFFh (EGA,VGA),
B0000h to B7FFFh (MDA),
or B8000h to BFFFFh (CGA).

Range selection is accomplished through programmable bits in the VGACTL register (Index B9h). Fine control can be exercised within the range selected to allow off-screen accesses to occur without generating SMIs.

SMI generation can also separately control the following I/O ranges: 3B0h to 3BFh, 3C0h to 3CFh, and 3D0h to 3DFh. The BC_XMAP_1 register (GX_BASE+8004h) in the Internal Bus Interface Unit has an enable/disable bit for each of the address ranges above.

The VGA control register (VGACTL) provides control for SMI generation through an enable bit for memory address ranges A0000h to BFFFFh. Each bit controls whether or not SMI is generated for accesses to the corresponding address range. The default value of this register is zero so that VGA accesses will not be trapped on systems with an external VGA card.

The VGA Mask register (VGAM) has 32 bits that can selectively mask 2 KB regions within the VGA memory region A0000h to AFFFFh. If none of the three regions is enabled in VGACTL, then the contents of VGAM are ignored. VGAM can be used to prevent the occurrence of SMI when non-displayed VGA memory is accessed. This is an enhancement that improves performance for double-buffered applications only.

Table 5-36 summarizes the VGA Configuration registers. Detailed register/bit formats are given in Table 5-37. See Section 4.3.2.2 "Configuration Registers" on page 47 on how to access these registers.

Table 5-36. VGA Configuration Register Summary

Index	Type	Name/Function	Default Value
B9h	R/W	**VGACTL.** VGA Control Register	00h (SMI generation disabled)
BAh-BDh	R/W	**VGAM.** VGA Mask Register	xxxxxxxxh

Table 5-37. VGA Configuration Registers

Bit	Description	
Index B9h	**VGACTL Register (R/W)**	**Default Value = 00h**
7:3	**Reserved**: Set to 0.	
2	SMI generation for VGA memory range B8000h to BFFFFh: 0 = Disable; 1 = Enable.	
1	SMI generation for VGA memory range B0000h to B7FFFh: 0 = Disable; 1 = Enable.	
0	SMI generation for VGA memory range A0000h to AFFFFh: 0 = Disable; 1 = Enable.	
Index BAh-BDh	**VGAM Register (R/W)**	**Default Value = xxxxxxxxh**
31	SMI generation for address range AF800h to AFFFFh: 0 = Disable; 1 = Enable.	
30	SMI generation for address range AF000h to AF7FFh: 0 = Disable; 1 = Enable.	
29	SMI generation for address range AE800h to AEFFFh: 0 = Disable; 1 = Enable.	
28	SMI generation for address range AE000h to AE7FFh: 0 = Disable; 1 = Enable.	
27	SMI generation for address range AD800h to ADFFFh: 0 = Disable; 1 = Enable.	
26	SMI generation for address range AD000h to AD7FFh: 0 = Disable; 1 = Enable.	
25	SMI generation for address range AC800h to ACFFFh: 0 = Disable; 1 = Enable.	
24	SMI generation for address range AC000h to AC7FFh: 0 = Disable; 1 = Enable.	
23	SMI generation for address range AB800h to ABFFFh: 0 = Disable; 1 = Enable.	
22	SMI generation for address range AB000h to AB7FFh: 0 = Disable; 1 = Enable.	
21	SMI generation for address range AA800h to AAFFFh: 0 = Disable; 1 = Enable.	
20	SMI generation for address range AA000h to AA7FFh: 0 = Disable; 1 = Enable.	
19	SMI generation for address range A9800h to A9FFFh: 0 = Disable; 1 = Enable.	
18	SMI generation for address range A9000h to A97FFh: 0 = Disable; 1 = Enable.	
17	SMI generation for address range A8800h to A8FFFh: 0 = Disable; 1 = Enable.	
16	SMI generation for address range A8000h to A87FFh: 0 = Disable; 1 = Enable.	
15	SMI generation for address range A7800h to A7FFFh: 0 = Disable; 1 = Enable.	
14	SMI generation for address range A7000h to A77FFh: 0 = Disable; 1 = Enable.	
13	SMI generation for address range A6800h to A6FFFh: 0 = Disable; 1 = Enable.	
12	SMI generation for address range A6000h to A67FFh: 0 = Disable; 1 = Enable.	
11	SMI generation for address range A5800h to A5FFFh: 0 = Disable; 1 = Enable.	
10	SMI generation for address range A5000h to A57FFh: 0 = Disable; 1 = Enable.	
9	SMI generation for address range A4800h to A4FFFh: 0 = Disable; 1 = Enable.	
8	SMI generation for address range A4000h to A47FFh: 0 = Disable; 1 = Enable.	
7	SMI generation for address range A3800h to A3FFFh: 0 = Disable; 1 = Enable.	
6	SMI generation for address range A3000h to A37FFh: 0 = Disable; 1 = Enable.	
5	SMI generation for address range A2800h to A2FFFh: 0 = Disable; 1 = Enable.	
4	SMI generation for address range A2000h to A27FFh: 0 = Disable; 1 = Enable.	
3	SMI generation for address range A1800h to A1FFFh: 0 = Disable; 1 = Enable.	
2	SMI generation for address range A1000h to A17FFh: 0 = Disable; 1 = Enable.	
1	SMI generation for address range A0800h to A0FFFh: 0 = Disable; 1 = Enable.	
0	SMI generation for address range A0000h to A07FFh: 0 = Disable; 1 = Enable.	

5.6.4 Virtual VGA Register Descriptions

This section describes the registers contained in the graphics pipeline used for VGA emulation. The graphics pipeline maps 200h locations starting at GX_BASE+8100h. Refer to Section 5.1.2 "Control Registers" on page 94 for instructions on accessing these registers.

The registers are summarized in Table 5-38, followed by detailed bit formats in Table 5-39.

Table 5-38. Virtual VGA Register Summary

GX_BASE+ Memory Offset	Type	Name/Function	Default Value
8140h-8143h	R/W	**GP_VGA_WRITE** Graphics Pipeline VGA Write Patch Control register: Controls the VGA memory write path in the graphics pipeline.	xxxxxxxxh
8144h-8147h	R/W	**GP_VGA_READ** Graphics Pipeline VGA Read Patch Control register: Controls the VGA memory read path in the graphics pipeline.	00000000h
8210h-8213h	R/W	**GP_VGA_BASE VGA** Graphics Pipeline VGA Memory Base Address register: Specifies the offset of the VGA memory, starting from the base of graphics memory.	xxxxxxxxh
8214h-8217h	R/W	**GP_VGA_LATCH** Graphics Pipeline VGA Display Latch register: Provides a memory mapped way to read or write the VGA display latch.	xxxxxxxxh

Table 5-39. Virtual VGA Registers

Bit	Name	Description
GX_BASE+8140h-8143h		**GP_VGA_WRITE Register (R/W)** **Default Value = xxxxxxxxh**
31:28	RSVD	**Reserved:** Set to 0.
27:24	MAP_MASK	**Map Mask:** Enables planes 3 through 0 for writing. Combined with chain control to determine the final enables.
23:21	RSVD	**Reserved:** Set to 0.
20	W3	**Write Mode 3:** Selects write mode 3 by using the bit mask with the rotated data.
19	W2	**Write Mode 2:** Selects write mode 2 by controlling set/reset.
18:16	RC	**Rotate Count:** Controls the 8-bit rotator.
15:12	SRE	**Set/Reset Enable:** Enables the set/reset value for each plane.
11:8	SR	**Set/Reset:** Selects 1 or 0 for each plane if enabled.
7:0	BIT_MASK	**Bit Mask:** Selects data from the data latches (last read data).
GX_BASE+8144h-8147h		**GP_VGA_READ Register (R/W)** **Default Value = 00000000h**
31:18	RSVD	**Reserved:** Set to 0.
17:16	RMS	**Read Map Select:** Selects which plane to read in read mode 0 (Chain 2 and Chain 4 inactive).
15	F15	**Force Address Bit 15:** Forces address bit 15 to 0.
14	PC4	**Packed Chain 4:** Provides 64 KB of packed pixel addressing when used with Chain 4 mode. This bit causes the VGA addresses to be shifted right by 2 bits.
13	C4	**Chain 4 Mode:** Selects Chain 4 mode for both read operations and write operations. This overrides bits 10 and 9 of this register.
12	PB	**Page Bit:** Becomes LSB of address if COE is set high.
11	COE	**Chain Odd/Even:** Selects PB rather than A0 for least-significant VGA address bit.
10	W2	**Write Chain 2 Mode:** Selects Chain 2 mode for write operations. Bit 13 overrides this bit.
9	R2	**Read Chain 2 Mode:** Selects Chain 2 mode for read operations. Bit 13 overrides this bit.
8	RM	**Read Mode:** Selects between read mode 0 (normal) and read mode 1 (color compare).

Table 5-39. Virtual VGA Registers (Continued)

Bit	Name	Description
7:4	CCM	**Color Compare Mask:** Selects planes to include in the color comparison (read mode 1).
3:0	CC	**Color Compare:** Specifies value of each plane for color comparison (read mode 1).
GX_BASE+8210h-8213h	**GP_VGA_BASE (R/W)**	**Default Value = xxxxxxxxh**
31:14	RSVD	**Reserved:** Set to 0.
13:8	VGA_RD _BASE	**Read Base Address:** The VGA base address is added to the graphics memory base to specify where VGA memory starts. The VGA base address provides address bits [19:14] when mapping VGA accesses into graphics memory. This allows the VGA base address to start on any 64 KB boundary within the 4 MB of graphics memory. This register is used for reads to the VGA trace buffer.
7:6	RSVD	**Reserved:** Set to 0.
5:0	VGA_WR _BASE	**Write Base Address:** The VGA base address is added to the graphics memory base to specify where VGA memory starts. The VGA base address provides address bits [19:14] when mapping VGA accesses into graphics memory. This allows the VGA base address to start on any 64 KB boundary within the 4 MB of graphics memory. This register is used for writes to the VGA trace buffer.
GX_BASE+8214h-8217h	**GP_VGA_LATCH Register (R/W)**	**Default Value = xxxxxxxxh**
31:0	LATCH	**Display Latch:** Specifies the value in the VGA display latch. VGA read operations cause VGA frame buffer data to be latched in the display latch. VGA write operations can use the display latch as a source of data for VGA frame buffer write operations.