CHAPTER 7

DISPLAY SUBSYSTEM

The Display Subsystem is responsible for converting the pixel data in the various bitplanes to analog RGB signals which are used to display images on the high resolution monitor. The pixel data in the Window ID (WID) bitplanes is used to control how the data in the Frame Buffer, PUP and UAUX bitplanes are used to calculate the images to be displayed. The following sections describe the external interfaces, the major components, the registers, the interrupts, the basic operations and the programming considerations for the Display Subsystem.

External Interfaces

The Display Subsystem has external interfaces with the Host System, the Raster Subsystem, the high resolution monitor and the optional genlock card.

Host Interface

The Display Subsystem has an interface with the host via the Host Interface and Geometry Subsystems. The host can read and write the display registers as well as the registers in the five XMAP2s, the five XPC1s, the three RAMDACs and the RGB RAMDAC. The register addressing and the bit definitions are described in later paragraphs.

Raster Subsystem Interface

The interface to the Raster Subsystem consists of the Frame Buffer, the PUP, the UAUX and the WID bitplane pixel data. The Raster Subsystem also provides the two bits of cursor data which is used to display the cursor image. The bitplane oata is an input to the five XPC1s or the XMAP2s in the Display Subsystem. The two bits of cursor data are input to the five XPC1s in the base configuration and are input to the three RAMDACs in the enhanced configuration. The Display State Machine (DSM) is used to control the Display Subsystem timing. The DSM sends timing control signals to the Raster Subsystem to cause the data for each pixel to be sent to the XPC1 or XMAP2 chips.

High Resolution Monitor Interface

The interface to the high resolution color monitor consists of the analog RGB signals. These signals can conform to four different selectable timing configurations and to custom display timings via the genlock interface. The signals output by the RAMDACs conform to the RS-343A standard.

Genlock Interface

The genlock interface consists of synchronization signals to and from an optional genlock adapter. There is also a Pixel Clock input from the genlock adapter.

Major Components

The major components of the Display Subsystem depends on the hardware configuration. The following sections describe the major components of the base and enhanced configurations of the Display Subsystem.

Base Configuration

The major components of the base configuration Display Subsystem are shown in the block diagram of Figure 7.1. The Display State Machine controls the timing of the components in the Display Subsystem as well as the outputs from the VRAM in the Raster Subsystem. The Display Registers are used to control various aspects of the Display State Machine and to report status information to the host. The base configuration uses five XPC1 chips to form a five channel pixel pipeline. The outputs of the five XPC1 chips are connected to the five input ports on the RGB RAMDAC. The RGB RAMDAC then generates the analog RGB signals which drive the high resolution color monitor.

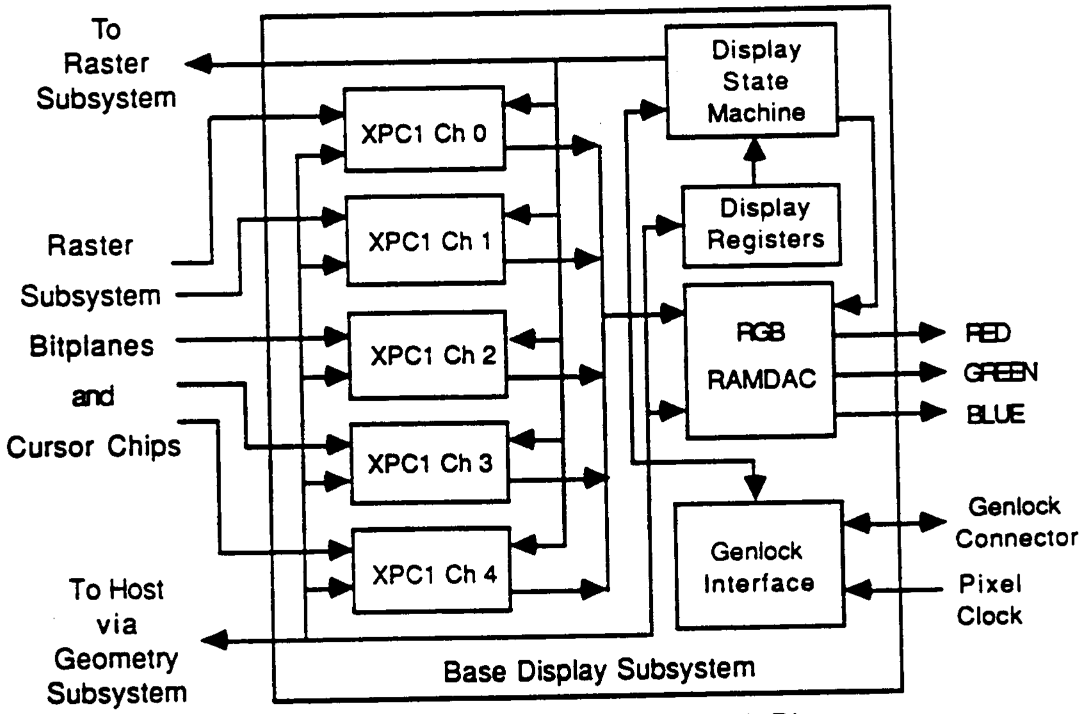

Figure 7.1 Base Display Subsystem Block Diagram

The genlock interface circuitry connects to an optional genlock adapter card which allows the outputs of multiple systems to be synchronized and displayed on a single monitor. The following paragraphs describe the components of the base configuration.

Display State Machine

The Display State Machine (DSM) controls the timing of the Display Subsystem. The DSM generates the necessary signals for the horizontal and vertical SYNC and BLANK signals which control the raster scan on the monitor. The DSM can be programmed to generate either an interlaced or non-interlaced raster scan. As the DSM generates the raster scan it also generates the necessary signals to transfer the data for each pixel from the bitplanes to the pixel formatting chips in the Display Subsystem. Each pixel is formatted and the resulting pixel data is transferred to the DACs which convert it to analog RGB signals for output to the monitor. The analog RGB signals are used to display the appropriate colors for each pixel as the scan line is generated. This process continues for each line of the raster scan. The DSM timing signals also control the operation of the cursor chip outputs which are sent to the pixel formatting chips on the base configuration or to the DACs on the enhanced configuration.

For each line of the raster scan the DSM generates a data transfer request signal to the RE2 chip during the horizontal blanking period preceding the pixel display portion of the scan line. The data transfer request causes the RE2 to generate the necessary signals to the VRAM chips to cause them to transfer the data for the desired row to a shift register inside the VRAMs. The RE2 uses the value in it's TOPSCAN register to generate the proper page address and shift start address for the current row to be displayed. Once the pixel data is in the shift register the DSM generates clock pulses which cause the VRAMs to shift the individual pixel data out to the pixel formatting chips in the Display Subsystem. For the base configuration each pixel consists of 8 bits of Frame Buffer data, 2 bits of PUP data and 2 bits of WID data. For the enhanced configuration each pixel consists of 24 bits of Frame Buffer data, 2 bits of PUP data, 2 bits of UAUX data and 4 bits of WID data.

The bitplanes and the components in the Display Subsystem have been designed to form a five channel pixel pipeline to support the very high clock rates necessary for the 1280 x 1024 high resolution monitor. As the scan line is displayed the pixels are processed by the five pixel pipelines. The first pixel on the scan line is processed by channel 1, the second pixel is processed by channel 2, the third pixel is processed by channel 3, the fourth pixel is processed by channel 4 and finally the fifth pixel is processed by channel 5. This process is then repeated for each group of five pixels on the current scan line. Each channel processes every fifth pixel on the scan line.

The DSM allows one of four built-in display timing modes to be selected. The display timing modes are selected using the Display Registers. The following timings are supported:

- 1280 x 1024 pixel non-interlaced 60 Hz
- 1280 x 1024 pixel interlaced 30 Hz
- 645 x 485 pixel RS170 30 Hz (NTSC broadcast standard)
- 780 x 575 pixel EURO 30 Hz (PAL - SECAM)

There are three oscillators that reside on the MGR board that are used to provide the timing. The first provides a 107.352 Mhz clock to support both 1280 X 1024 modes. The second provides a 12.27 MHz clock for RS170, and the third runs at 15.00 Mhz to support PAL and SECAM. There is also a Pixel Clock input from the optional genlock adapter which can be used for customized timing support.

For a selected timing mode the DSM allows variation in the following parameters:

- SYNC and BLANK width (within a five pixel resolution)

- frame buffer line length (pixels/line)
- frame buffer line count
- choice of interlaced or noninterlaced display
- vertical retrace latency timing

The DSM is built out of a Xilinx 2018 programmable logic array, an 8K by 8-bit PROM, a 2K by 8-bit RAM, a Bt438 clock generator chip and other miscellaneous clock control circuitry. The 8K PROM contains the Xilinx logic configuration data and the four different monitor timing tables. On system reset the logic configuration is loaded into the Xilinx chip and the monitor timing tables are copied to the 2K RAM for faster access. The operation of the Display State Machine is controlled by various bits in the Display Registers.

Display Registers

The MGR contains five display registers in discrete logic on the MRV2 card. These registers are used to provide status bits to the host and the other MGR subsystems and to control the operation of the Display State Machine. The host accesses the registers via the Host Interface and Geometry Subsystems. The host address decoding for these registers is shown in Table 7.1. To access these registers the HQMMSB register must be set to one. All five registers can be read or written via the host addresses shown. The Registers section defines the R/W capabilities for each bit in the registers.

Table 7.1 Display Registers Address Map

Host Address	Source/Destination
4E0 Hex	Display Register 0
4C0 Hex	Display Register 1
5E0 Hex	Display Register 2
5C0 Hex	Display Register 3
5A0 Hex	Display Register 4

HQMMSB register must be set to 1

These status information available to the other MGR subsystems and to the host includes:

- Z buffer (MZB1) daughter board installed or not installed
- Enhanced Video (MEV2) daughter board installed or not installed
- FIFO empty status

The display registers are also used to control the following Display State Machine functions:

- Sync enable/disable
- Monitor standby mode enable/disable.
- Xilinx configuration loading
- Monitor timing selection
- External Pixel Clock enable/disable
- Asynchronous clock enable/disable
- Synchronous clock enable/disable

The format and bit definitions of the Display Registers is contained in the Registers section of this chapter.

XPC1 Chips

The XPC1 is a proprietary Silicon Graphics chip which is to used to control the pixel display formatting. The XPC1 chip processes the input Frame Buffer bitplane data, the PUP bitplane data and the output data from the two cursor chips. The pixel formatting is controlled by the Window ID bitplane data. The XPC1 chip produces formatted Frame Buffer pixel data and overlay or underlay data which is output to the RGB RAMDAC which uses the data to generate the analog RGB output to the high resolution monitor.

To display a 1280 x 1024 pixel high resolution color image the Display Subsystem requires a five channel pixel pipeline to handle the very high data rates. In the base configuration the five channels are formed by the interleaved VRAM chips, the five XPC1 chips and a single RGB RAMDAC which has five input channels. Each XPC1 chip processes every fifth pixel along the scan line.

The XPC1 chip has programmable registers which are programmed by the host software. These values written into the registers then control the pixel formatting operations of the XPC1. The following sections describe the host software access to the XPC1 registers and the pixel formatting operations of the XPC1 chip.

Host Access to the XPC1 Registers

Each XPC1 chip contains an address register and four mode registers. The address register is used to address the mode registers. The mode registers are used to control the pixel formatting operations. The host software programs the address and mode registers in the XPC1 chips by reading or writing the addresses shown in Table 7.2. To read or write the registers in the XPC1 chips the HQMMSB register must be set to 1.

Table 7.2 XPC1 Address Map

XPC1 Channel	Host Address	Command
0	510 Hex	Write Address Reg
	514 Hex	R/W Data Byte
1	530 Hex	Write Address Reg
	534 Hex	R/W Data Byte
2	550 Hex	Write Address Reg
	554 Hex	R/W Data Byte
3	570 Hex	Write Address Reg
	574 Hex	R/W Data Byte
4	590 Hex	Write Address Reg
	594 Hex	R/W Data Byte
Broadcast	5B0 Hex	Write Address Reg
	5B4 Hex	Write Data Byte

HQMMSB register must be set to 1

The addresses shown in Table 7.2 are the offset from the base address of the adapter programmed into the GRP variable. Refer to the Host Interface Subsystem chapter for additional details on programming the base address of the adapter. A special broadcast mechanism allows the host software to write to all five XPC1 chips in a single write operation.

Figure 7.2 shows the registers in the five XPC1 chips and the addresses the host reads or writes to access the registers. The special broadcast mechanism is shown as dashed lines since writes to these locations actually go to the other five XPC1 chips.

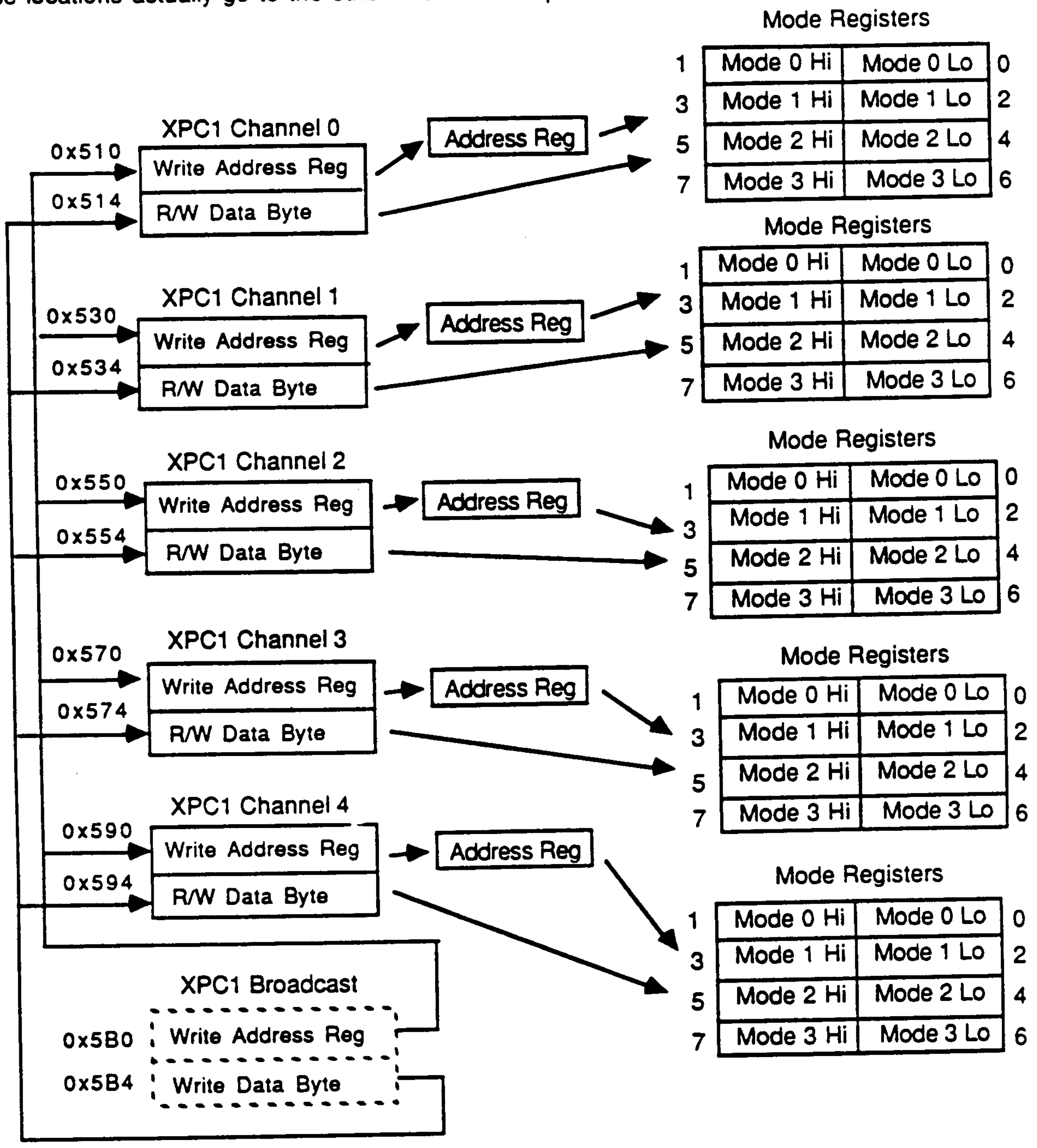

Figure 7.2 XPC1 Registers

Address Register

The address register is used to address the individual bytes of the mode registers. Since the four mode registers are addressed as 8 bytes only the low three bits of the address register are valid. The host writes the mode register byte address into the address register and then reads or writes the selected mode register byte.

Mode Registers

The mode registers control the pixel formatting operations of the XPC1 chips. The WID bitplane data is used as an index to access the individual mode registers. The Window IDs are used to divide the screen into one to four windows on the base configuration. Since each XPC1 chip processes every fifth pixel on the current scan line the host should program the corresponding mode registers in the five XPC1 to be the same. This means that all pixels which have the same window ID will be formatted the same regardless of which XPC1 does the formatting operations.

Each mode register is 14 bits wide and is accessed as two bytes. Only the right most 6 bits of the upper byte are valid. To write a mode register in all five XPC1 chips the host software writes the address of the low byte of the mode register into the address register by writing to the broadcast Write Address Register address. This causes the same address to be written into the address register of all five XPC1 chips. The selected mode register low byte is then written by writing the mode register low byte data to the broadcast R/W Data Byte address. This causes the mode register low byte data to be written into the selected mode register byte in all five XPC1 chips. The host software then writes the address of the upper byte of the mode register into the address register by writing to the broadcast Write Address Register address. The upper mode register byte is then written by writing to the broadcast R/W Data Byte address. When accessing the upper byte of a mode register only the low 6 bits are valid since each mode register is only 14 bits wide.

To read a mode register the low byte address is written into the address register by writing to the Write Address register of just the single XPC1 chip that is being read. The low mode register byte is then read by reading the R/W Data Byte address of the single XPC1 chip being read. The same procedure is followed to read the upper byte of the mode register. The host software should mask the upper two bits of the upper mode register byte since only the right most six bits are valid.

The format of the address and mode registers is defined in the Registers section of this chapter. Refer to the programming considerations section of this chapter for an example of how to program the address and mode registers of the XPC1 chips. The next section describes how the XPC1 chips perform the pixel formatting operations.

XPC1 Pixel Formatting Operations

The XPC1 chips are used to perform the pixel formatting operations specified by the mode registers. The base configuration of the Display Subsystem contains five XPC1 chips which form part of the five channel pixel pipeline. The pixels in the Raster Subsystem bitplanes are divided into five pixel groups and each of the five XPC1 chips processes one of the five pixels. The first pixel is processed by the channel 0 XPC1 chip. The second pixel is processed by the channel 1 XPC1 chip. The third pixel is processed by the channel 2 XPC1 chip. The fourth pixel is processed by the channel 3 XPC1 chip. Finally the fifth pixel is processed by the channel 4 XPC1 chip. This processing is continued for each five pixel group along the current scan line. Each XPC1 chip receives input data from the Raster Subsystem bitplanes and generates outputs which are sent to one of the five inputs on the RGB RAMDAC chip.

As shown in Figure 7.3, the input data to the XPC1 chips consists of 8 bits of Frame Buffer bitplane pixel data (FB_PIXEL), 2 bits of PUP bitplane data (PUP), 2 bits of cursor data (CURSOR) and 2 bits of Window ID (WID) bitplane data from the Raster Subsystem for each pixel it processes. The WID data is used as an index to select one of the four mode registers. The mode word in the selected mode register contains the necessary information to allow the XPC1 to format the input pixel data and generate as outputs an 8 bit formatted Frame Buffer pixel value (XOUT) and a 2 bit overlay value (OVL). The XOUT and OVL outputs are sent to the RGB RAMDAC which use this data to generate the analog RGB signals to drive the high resolution monitor.

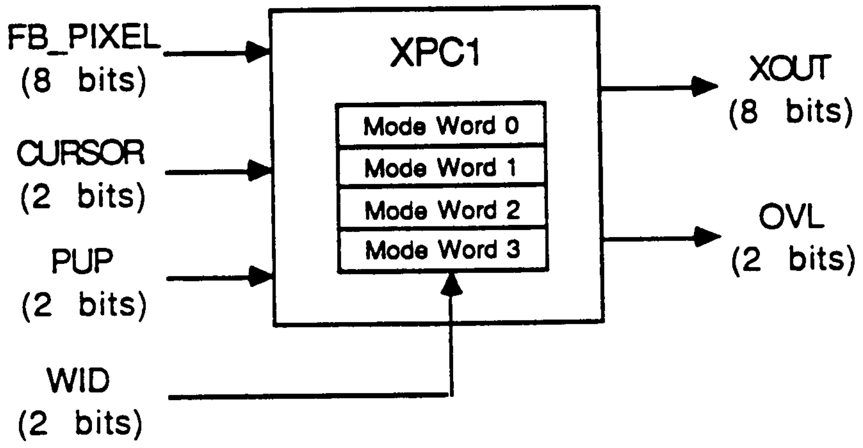

Figure 7.3 XPC1 Input and Output Data

The 2 bits of WID data allow the four mode registers to be indexed. The WID bitplane data are used to divide the screen into four windows. All pixels in a window will have the same Window ID written into the WID bitplanes for each pixel in the window. Since the pixels on each scan line are processed in groups of five by the five XPC1 chips the corresponding mode registers in each XPC1 chip are programmed with the same values as described in the previous section. This allows all pixels with the same WID to be formatted the same. The following paragraphs describe the pixel formatting operations to form the XOUT and OVL outputs of the XPC1 chips.

XOUT Output Formation

Each XPC1 chip formats a single pixel at a time. The XOUT output of the XPC1 chip is dependent on the FB_PIXEL input and the contents of the mode register selected by the WID bits of the pixel being processed. The formation of the formatted pixel output byte (XOUT) is a function of the following mode register bits from the mode register selected by the WID data bits:

- Display Mode (Bit 0)
- Buffer Select (Bit 3)
- Multimap Mode Enable (Bit 9)
- Multimap constant (Bits 13-10)

The Display mode bit controls the pixel type and should be set to match the pixel type selected with the GE_PIXTYPE token. The XPC1 chips support 8 bit Color Index singe buffer and 4 bit Color Index double buffer pixels. The 8 bit RGB pixels are supported as 8 bit Color Index single buffer pixels by the XPC1 chips.

For double buffer pixels the buffer select bit controls whether the front or back buffer is displayed and is correlated to the pixel write mask as set with the GE_PIXWRITEMASK token. Usually when a double buffer pixel type is specified the host software sets the pixel write mask so that one buffer is being written into while the buffer select bit in the mode register is set so that the opposite buffer is being displayed. When the host software has finished writing into the current buffer the buffer select bit is switched to display the newly written buffer and the pixel write mask is changed to allow the host software to update the opposite buffer. The Buffer select bit is ignored for single buffer pixels and should be set to 0 to select the front buffer. The host software should also set the pixel write mask to allow only the front buffer to be written for single buffer pixels.

The multimap enable bit is used to control whether the multimap constant is used to increase the number of available color indexes. When enabled the multimap constant nibble is placed in the upper nibble of the XOUT byte. This increases the range of colors that the host software can access in the color palette in the RGB RAMDAC.

The use of these mode register bits in forming the XOUT output byte are shown in Table 7.3.

Table 7.3 XOUT Formation by XPC1

Display Mode Bit 0	Buffer Select Bit 3	Multimap Mode Enable Bit 9	Multimap Constant Bits 13-10	XOUT Bits 7-4	XOUT Bits 3-0	Description
0	X	X	X	FB_PIXEL Bits 7-4	FB_PIXEL Bits 3-0	8-bit CI or RGB
1	0	0	X	0	FB_PIXEL Bits 3-0	4-bit CI front buf
1	1	0	X	0	FB_PIXEL Bits 7-4	4-bit CI back buf
1	0	1	m[0-3]	m[0-3]	FB_PIXEL Bits 3-0	4-bit CI front buf multimap
1	1	1	m[0-3]	m[0-3]	FB_PIXEL Bits 7-4	4-bit CI back buf multimap

The XPC1 chip supports 8 bit single buffer and 4 bit double buffer color index pixel formats. The following paragraphs describe these two pixel formats.

8 Bit CI Single Buffer Pixels

If the display mode bit (bit 0) is a 0 then the XPC1 is in 8 bit single buffer mode and all 8 bits of the FB_PIXEL input data are passed unchanged through the XPC1 to form the 8 bits of the XOUT output. This mode supports both 8 bit RGB pixels and 8 bit color index pixels. The XPC1 chip does treats the 8 bit RGB pixel data as an 8 bit color index and does no special processing for the 8 bit RGB pixels. The 8 bit Color Index pixels are used by the RGB RAMDAC to select a color palette word.

The 8 bit RGB pixels also are used as an 8 bit index into the RGB RAMDAC color palette. However, the 8 bit RGB pixel data contains the special 233 formatting of the RGB bits formed from the original 24 bit RGB value by the RE2 chip. The bits have the following format:

- FB_PIXEL bits 2-0 represent the three most significant bits from the original Red byte of the 24 bit RGB value
- FB_PIXEL bits 5-3 represent the three most significant bits from the original Green byte of the 24 bit RGB value
- FB_PIXEL bits 7-6 represent the two most significant bits of the original Blue byte of the 24 bit RGB value

When the Frame Buffer contains 8 bit RGB pixels the color palette in the RGB RAMDAC must be programmed with a special RGB color palette.

For 8 bit Color Index single buffer pixels the buffer select bit, the multimap constant enable bit and the multimap constant bits of the selected mode register are ignored.

4 Bit CI Double Buffer Pixels

If the display mode bit is set to 1, then the XPC1 is in 4 bit color index double buffer mode and the value of the buffer select bit (bit 3) from the selected mode register is used to determine which of the 4 FB_PIXEL bits will become the low nibble of XOUT. If the buffer select bit is a 0 then the low nibble of PIXEL (front buffer) becomes the low nibble of XOUT. If the buffer select bit is 1 then the high nibble of PIXEL (back buffer) becomes the low nibble of XOUT. The high nibble of XOUT is dependent on the multimap enable bit (bit 9) from the selected mode register. If the multimap enable bit is 0 then the high nibble of XOUT is a zero. If the multimap enable bit is a 1, then the high nibble of XOUT is the multimap constant nibble (bits 13-10) from the selected mode register.

OVL Output Formation

The formation of the 2 bit Overlay output (OVL), is shown in Table 7.4, and is a function of the 2 bit Cursor (CURSOR) input, the 2 bit PUP (PUP) input and the 8 bit FB_PIXEL input. It is also dependent on the two Overlay Enable bits (bits 4-5) and the Underlay Enable bit (bit 8) from the selected mode register.

The XPC1 first checks the two CURSOR bits first to see if a cursor condition is true. If either or both CURSOR bits is a one the cursor condition is true and the OVL output is set to the two CURSOR input bits. If both cursor bits are 0 then the cursor condition is false.

If the cursor condition is false then the XPC1 checks for a PUP overlay condition. If either of the two PUP bits is set and its corresponding overlay enable bit (mode register bits 4 and 5) is also set then the PUP overlay condition is true and the OVL output is set to the two input PUP bits. The PUP overlay condition is false if the results of ANDing the PUP bits with their corresponding enable bits are both 0.

If the cursor and PUP overlay conditions are both false then the XPC1 checks for a PUP underlay condition. If the FB_PIXEL input value is 0 and the Underlay Enable bit (mode register bit 8) is set then the PUP underlay condition is true and the OVL output is set to the two input PUP bits. If the FB_PIXEL value is nonzero or the underlay enable bit is 0 the PUP underlay condition is false. The FB_PIXEL value which is checked for zero is either the full 8 bits for 8 bit color index single buffer pixels or the selected 4 bit buffer for 4 bit color index double buffer pixels. The selected 4 bit buffer value depends on the buffer select bit as described above.

Table 7.4 OVL Formation by XPC1

CURSOR [1:0]	PUP[1] & OE[1]	PUP[0] & OE[0]	Underlay Enable	FB_PIXEL	OVL [1:0]	Description
01	X	X	X	X	CURSOR [1:0]	Cursor
10	X	X	X	X		
11	X	X	X	X		
00	1	X	X	X	PUP [1:0]	PUP Overlay
00	X	1	X	X		
00	0	0	1	0	PUP [1:0]	PUP Underlay
00	0	0	0	X	00	Transparent
00	0	0	1	NZ		

If the cursor, the PUP overlay or the PUP underlay condition is true then the XOUT output of the XPC1 will be ignored by the RGB RAMDAC and the OVL output of the XPC1 will be used to select one of the three overlay palette colors which will be displayed at the current pixel location.

If the cursor, PUP overlay and PUP underlay conditions are all false then the OVL bits are set to 0 to select the transparent case. In the transparent case the XOUT output will be used by the RGB RAMDAC to select one of the color palette entries to display the pixel color at the current pixel location.

RGB RAMDAC

The RGB RAMDAC is a Brooktree Bt458 chip, which is used to convert the 8-bit pixel data into 24-bit RGB values These values are then converted by the three DACs to analog RGB values and are output to the high resolution color monitor.

The RGB RAMDAC contains an input latch and multiplexer, a read mask, a blink mask, a 256 x 24 bit color palette, a 4 x 24 bit overlay palette. It also contains a red, green and blue DAC which are used to convert the RGB color values into analog signals which are output to the analog monitor. The internal representation of the RGB RAMDAC is shown in Figure 7.4.

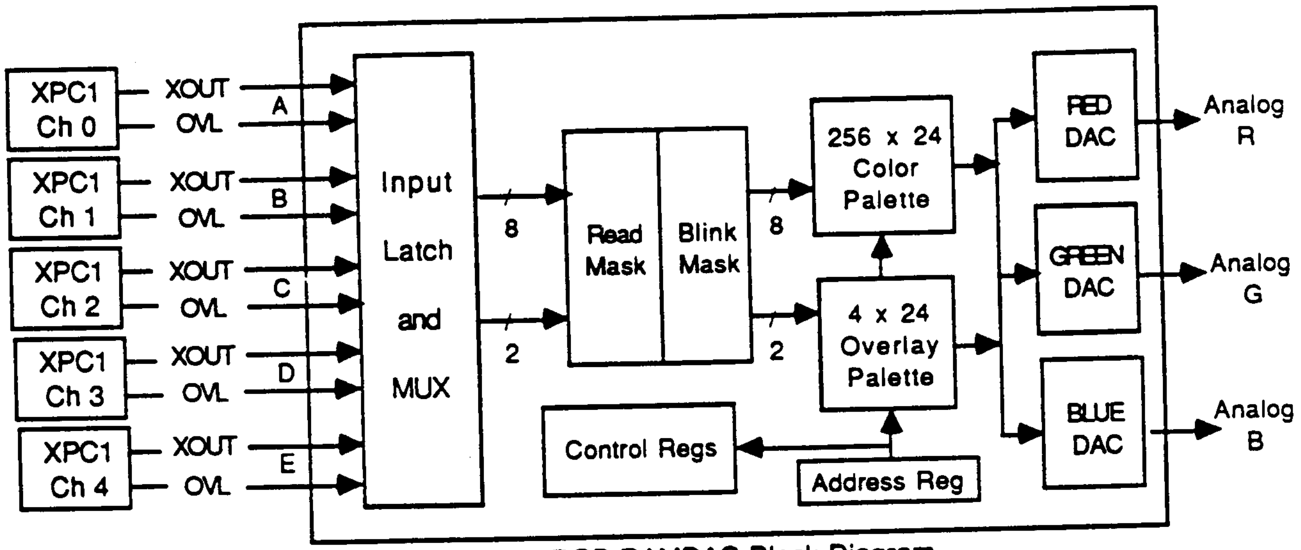

Figure 7.4 RGB RAMDAC Block Diagram

The following sections describe the host access to the programmable registers and the basic operations of the RGB RAMDAC.

Host Access to the RGB RAMDAC Registers

The RGB RAMDAC contains an address register and four control registers. The control registers include a command register, a read mask register, a blink mask register and a test register. The RGB RAMDAC also contains a 256 x 24 bit color palette and a 4 x 24 bit overlay palette. Table 7.5 shows the address decoding to access the registers, color palette and overlay palette. The HQMMSB Register must be set to 1 to read or write the RGB RAMDAC components.

Address Register

The address register is used to specify a location in the color palette, the overlay palette or select one of the four control registers. The address register is loaded with the values shown in Table 7.5 and then the palette entry or register is accessed.

Color Palette

To read color values from or write color values to the color palette the desired entry in the color palette is written into the address register and then three color bytes are read from or written to the color palette address. The three color bytes are the red, green and blue color values. After the third color byte is read or written the address register is automatically incremented. This allows the host software to write the starting address and then read or write groups of three colors to the the color palette. If the address register contains a 0xFF it wraps to 0x00 after the third color byte is read from or written to the color palette.

Table 7.5 RGB RAMDAC Address Map

Host Address	RGB RAMDAC Address Register Value	Source/Destination
5D0 Hex	XX	RGB RAMDAC Address Register
5D4 Hex	00 - FF Hex	Color Palette RAM
5D8 Hex	04	Read Mask Register
5D8 Hex	05	Blink Mask Register
5D8 Hex	06	Command Register
5D8 Hex	07	Test Register
5DC Hex	00 - 03	Overlay Palette RAM

HQMMSB Register must be set to 1

Overlay Palette

To read color values from or write color values to the overlay palette the desired entry in the overlay palette is written into the address register and then three color bytes are read from or written to the overlay palette address. The three color bytes are the red, green and blue color values. After the third color byte is read or written the address register is automatically incremented. This allows the host software to write the starting address and then read or write groups of three colors to set the colors to the overlay palette. If the address register contains a 3 it increments to 4 after the third color byte is read from or written to the overlay palette.

Command Register

The command register is used to control the operation of the input multiplexer, the overlay transparency, the blink rate, the overlay blink and the overlay display. The command register is read or written by writing the value 6 into the address register and then reading or writing the control register address.

The input multiplexer control bit can select either four or five input sources. The MGR always selects five since it has five XPC1s which are connected to the RGB RAMDACs inputs.

The overlay transparency control bit allows the overlay input equal to zero to be made transparent or to use the color value stored in overlay palette location 0. If it is made transparent then the color value in the color palette will be used instead of the value in the overlay palette. This bit is set for transparent mode in the MGR since a zero or nonzero value on the OVL inputs is used to determine if a word from the color palette or the overlay palette is used to get the RGB color values.

The blink rate control bits allow the number of vertical retrace periods the pixels will be on and off to be specified.

The overlay display control allows the overlay bits to be used or ignored. The MGR is programmed to utilize both overlay bits.

Read Mask Register

The read mask register allows bits in the input pixel bytes to be forced to zero. The MGR is programmed to not mask any of the eight bits. The read register is read or written by writing the value 4 into the address register and then reading or writing the control register address.

Blink Mask Register

The blink mask register controls which of the pixel bitplanes will blink. The MGR is programmed to not allow any of the bitplanes to blink. Blink is accomplished by varying the color palette values. The overlay blink control allows the overlay bit blinking capability. If enabled the overlay colors could be blinked between the overlay color and the transparent color. This capability is not used on the MGR. The blink mask register is read or written by writing the value 5 into the address register and then reading or writing the control register address.

Test Register

The test register allows a nibble of the output data to be read. The format of this register is described in the Registers section of this chapter. The test register is read or written by writing the value 7 into the address register and then reading or writing the control register address.

RGB RAMDAC Basic Operations

The input multiplexer can accept the 8-bit pixel input and the 2-bit overlay input from five different sources. The MGR takes the XOUT and OVL outputs from the five XPC1 chips and routes them to the five different input channels on the RGB RAMDAC. These five inputs are referred to as the A through E inputs. Each time the Display State Machine generates a load signal to the RGB RAMDAC the 10-bits on the five inputs are latched. The DSM then generates clock pulses which cause the RGB RAMDAC to process the latched input data. On the first clock the A input is processed then on the second clock the B input is processed and so on until finally the E input is processed. This processing is then repeated for each group of five pixels.

On each clock cycle generated by the Display State Machine, the selected eight bits of pixel data and two bits of overlay data are processed by the RGB RAMDAC using the read mask, the blink mask and the various fields in the control register. The read mask is used to force some of the input bits to zero. The host software should program the read mask to not mask any of the input bits. The blink mask is used to cause some of the input bits to vary between zero and one at the blink rate programmed into the control register. The host software should program the blink register and the control register so that blinking is disabled. The control register determines how the overlay bits are used. The host software should program the host software so that the overlay bits are not masked and the overlay bits equal to zero are treated as a transparency. This allows the pixel bits to select a color palette entry. The 8 bits of pixel data and the 2 bits of overlay data are left unchanged by the read and blink mask registers and are applied directly as an index to the color palette and the overlay palette.

The RGB RAMDAC contains a 256 x 24 bit color palette and a 4 x 24 bit overlay palette. The 24 bit values in the color palette and the overlay palette consist of 8 bits of Red, 8 bits of Green and 8 bits of Blue data. The color values stored in the color palette depend on which display mode is being used. If a 4 or 8 bit Color Index mode is in effect then the color palette will have the graphics application specified color map values which have had a gamma correction factor applied to them. If 8 bit RGB mode is in effect then a special RGB color map must be used. This map will also have the gamma correction factors applied by the host software before it is loaded into the color palette.

If the 2 bit overlay input (OVL) is nonzero the 2 bit overlay input will be used to select an entry in the overlay palette. If the overlay input is zero then the 8 bit pixel input data (XOUT) is used as an index into the color palette. The selected entry in the overlay or color palette outputs the 24 bit RGB value to the three DACs. The red color value is output to the Red DAC, the green color value to the Green DAC and the blue color value to the Blue DAC. The three DACs convert the bytes to analog signals which conform to RS-343A and are capable of driving doubly-terminated 75-ohm coaxial cable directly, without requiring external buffering. The RGB RAMDAC also merges the SYNC and BLANK signals onto the green analog signal.

Genlock Interface

The genlock interface allows the video output of the MGR adapter to be synchronized with the video output of a second video source which is the same type as the currently selected MGR video output. The video output types can be any of the four video types described for the Display State Machine. When in genlock mode, the pixel clock comes off an optional genlock card allowing the two system to clock at the same rate. The optional genlock card provides a gensync signal which is used to synchronize the display vertically. The external pixel clock is divided by constant and is output back to the genlock card to provide a horizontal sync signal. The constant is dependent on the currently selected video type. The sync signals on the Green output should be turned off if the color encoding is going to be done on the genlock card. If color encoding is not done then the sync can be left on.

Enhanced Configuration

The major components of the enhanced configuration Display Subsystem are shown in the block diagram of Figure 7.5. The enhanced configuration uses five XMAP2 chips and five Color Maps to form the five channel pixel pipeline. The outputs of the five XMAP2 chips are connected to the five input ports on the three RAMDACs. The RAMDACs then generate the analog RGB signals which drive the high resolution color monitor.

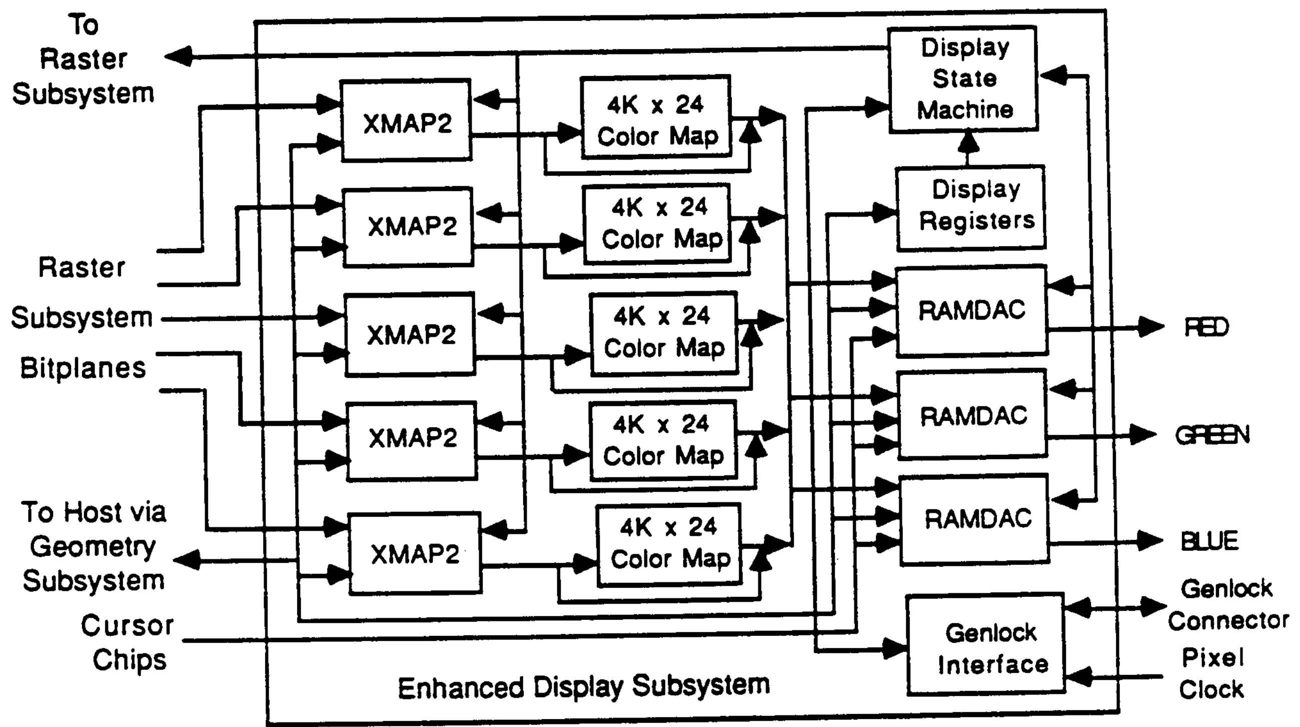

Figure 7.5 Enhanced Display Subsystem Block Diagram

The Display State Machine, Display Registers and the Genlock Interface are the same as in the base configuration. The following paragraphs describe the XMAP2s, the Color Maps and the RAMDACs.

XMAP2

The XMAP2 is a proprietary Silicon Graphics chip which is to used to control the pixel display formatting. The XMAP2 chip processes the input Frame Buffer bitplane data, the PUP bitplane data and the UAUX bitplane data. The pixel formatting is controlled by the Window ID bitplane data. The XMAP2 chip produces a byte of red, green and blue color data which is output to the three RAMDAC chips which use the data to generate the analog RGB output to the high resolution monitor.

To display a 1280 x 1024 pixel high resolution color image the Display Subsystem requires a five channel pixel pipeline to handle the very high data rates. In the enhanced configuration the five channels are formed by the interleaved VRAM chips, the five XMAP2 chips, the five Color Maps and three RAMDACs which each have five input channels. Each XMAP2 chip processes every fifth pixel along the current scan line. The block diagram of the XMAP2 chip is shown in Figure 7.6.

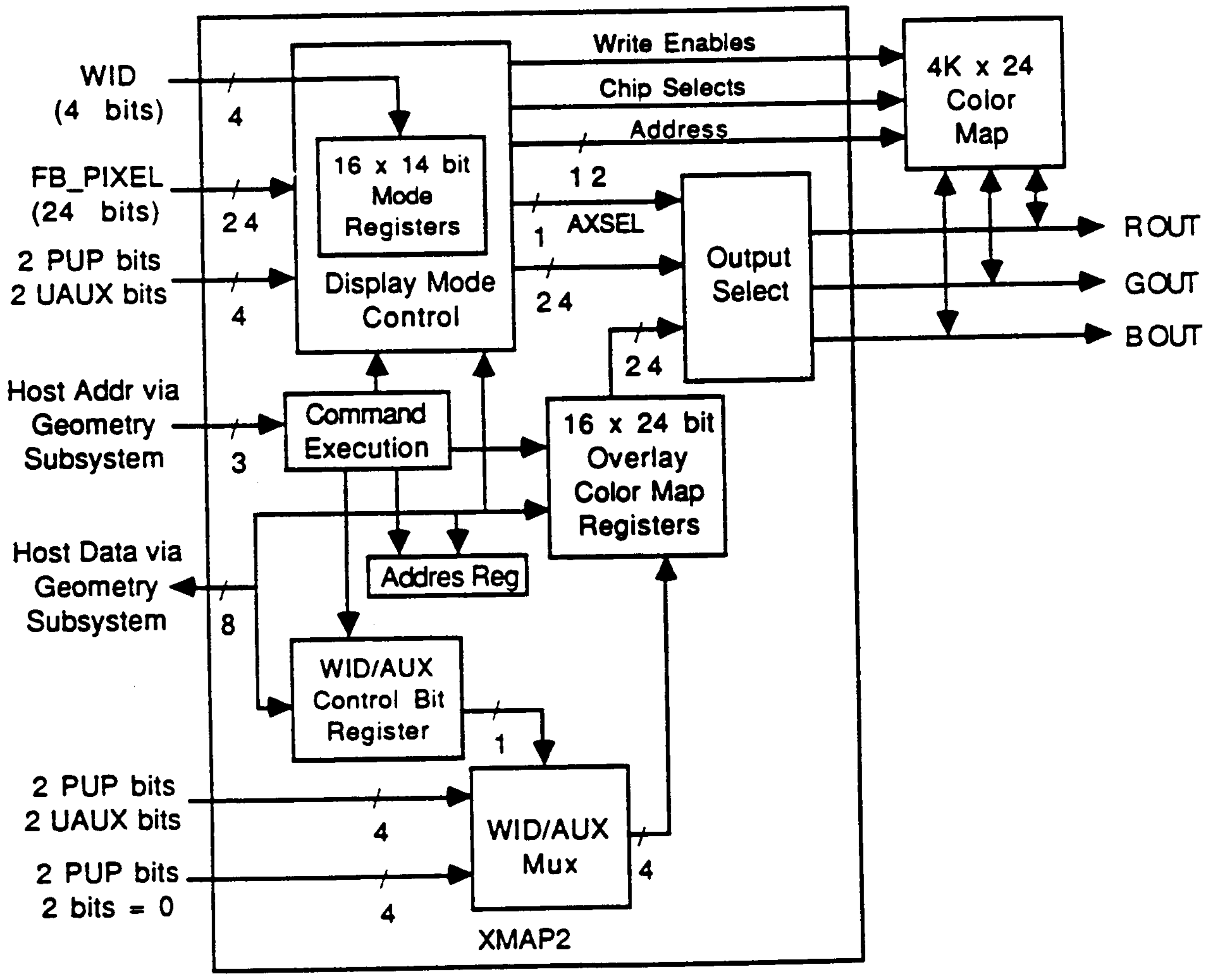

Figure 7.6 XMAP2 Block Diagram

The XMAP2 chip has programmable registers which are programmed by the host software. The values written into the registers control the pixel formatting operations of the XMAP2. The following sections describe the host software access to the XMAP2 registers and the pixel formatting operations of the XMAP2 chip.

Host Access to the XMAP2 Registers

The XMAP2 chip contains sixteen mode registers whose contents determine the display mode and control the outputs from the XMAP2. In addition to the 16 mode registers, each XMAP2 contains 16 overlay color map registers, a WID/AUX control bit register and a 13 bit address register.

The XMAP2 accepts eight commands from the host which are used to read or write the registers and the external color map. The eight commands are the No Operation command, the R/W Blue data byte command, the R/W Green data byte command, the R/W Red data byte command, the INCR address register command, the R/W other data byte command, the write MSB address byte command and the write LSB address byte command. Each command is issued by the host by reading or writing the address specified in Table 7.6.

Table 7.6 XMAP2 Address Map

XMAP2 Channel	Host Address	Command or Display Reg
0	400 Hex	No Operation
	404 Hex	R/W Blue Data Byte
	408 Hex	R/W Green Data Byte
	40C Hex	R/W Red Data Byte
	410 Hex	INCR Address Register
	414 Hex	R/W Other Data Byte
	418 Hex	R/W MSB Address Byte
	41C Hex	R/W LSB Address Byte
1	420 Hex	No Operation
	424 Hex	R/W Blue Data Byte
	428 Hex	R/W Green Data Byte
	42C Hex	R/W Red Data Byte
	430 Hex	INCR Address Register
	434 Hex	R/W Other Data Byte
	438 Hex	Write MSB Address Byte
	43C Hex	Write LSB Address Byte
2	440 Hex	No Operation
	444 Hex	R/W Blue Data Byte
	448 Hex	R/W Green Data Byte
	44C Hex	R/W Red Data Byte
	450 Hex	INCR Address Register
	454 Hex	R/W Other Data Byte
	458 Hex	Write MSB Address Byte
	45C Hex	Write LSB Address Byte

HQMMSB Register must be set to 1

Table 7.6 XMAP2 Address Map (continued)

XMAP2 Channel	Host Address	Command or Display Reg
3	460 Hex	No Operation
	464 Hex	R/W Blue Data Byte
	468 Hex	R/W Green Data Byte
	46C Hex	R/W Red Data Byte
	470 Hex	INCR Address Register
	474 Hex	R/W Other Data Byte
	478 Hex	Write MSB Address Byte
	47C Hex	Write LSB Address Byte
4	480 Hex	No Operation
	484 Hex	R/W Blue Data Byte
	488 Hex	R/W Green Data Byte
	48C Hex	R/W Red Data Byte
	490 Hex	INCR Address Register
	494 Hex	R/W Other Data Byte
	498 Hex	Write MSB Address Byte
	49C Hex	Write LSB Address Byte
Broadcast	4A0 Hex	No Operation
	4A4 Hex	Write Blue Data Byte
	4A8 Hex	Write Green Data Byte
	4AC Hex	Write Red Data Byte
	4B0 Hex	INCR Address Register
	4B4 Hex	Write Other Data Byte
	4B8 Hex	Write MSB Address Byte
	4BC Hex	Write LSB Address Byte

HQMMSB Register must be set to 1

Each command causes a data byte to be read from or written to the XMAP2 registers or the external color map entries. The exceptions to this are the No Operation command which does nothing and the INCR Address Register command which increments the address already in the XMAP2 address register. To access the command addresses the HQMMSB register must be set to 1.

The addresses shown in Table 7.6 are the offset from the base address of the adapter programmed into the GRP variable. Refer to the Host Interface Subsystem chapter for additional details on programming the base address of the adapter. A special broadcast mechanism allows the host software to write to all five XMAP2 chips in a single write operation.

The use of the commands to access the registers in the XMAP2 or the external color map, is shown in Figure 7.7. The other four channels, which are not shown, have the same configuration as channel 0. The only differences are the host addresses used for the commands to the other channels.

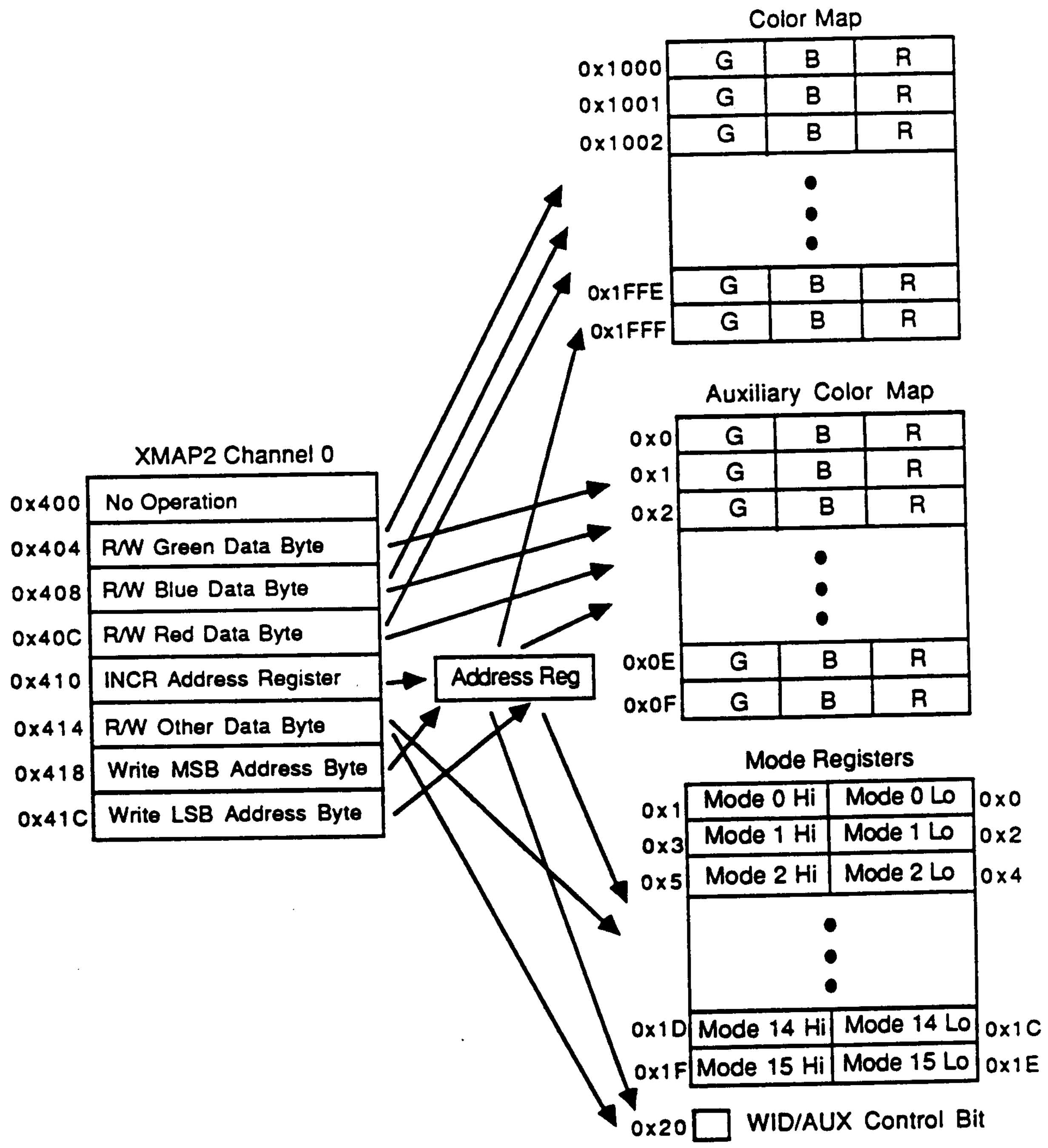

Figure 7.7 XMAP2 Register Access

Refer to the programming considerations section of this chapter for an example of how to program the XMAP2 registers. The following paragraphs describe the use of the registers.

Address Register

The address register is 13 bits wide and is used to select the other registers in the XMAP2 and the external color map entries. The address of the mode register, the WID/AUX control bit register, the overlay color map register or the external color map word to be accessed is written into the address register. The Write LSB Address byte command is used to write the low byte of the address register. The Write MSB Address byte command is used to write the upper five bits of the address register. The INCR address command is used to increment the address already in the address register. The format of the address is defined in the Registers section of this chapter.

Mode Registers

The 16 mode registers control the pixel formatting operations of the XMAP2 chips. The WID bitplane data is used as an index to access the individual mode registers. The Window IDs are used to divide the screen into one to sixteen windows on the enhanced configuration. Since each XMAP2 chip processes every fifth pixel on the current scan line the host should program the corresponding mode registers in the five XMAP2 chips to be the same. This means that all pixels which have the same window ID will be formatted the same regardless of which XMAP2 does the formatting operations. If the fast Z clear mode is used in the Raster Subsystem to invalidate the Z value in the Z buffer then the LSB bit of the Window ID is used for the fast Z clear. This means that only 8 different windows can be displayed on the screen. When fast Z clear mode is enabled the host software should allocate two window IDs to each window. This means that both mode registers in all five XMAP2 registers should be set to the same values.

Each mode register is 14 bits wide and is accessed as two bytes. Only the right most 6 bits of the upper byte are valid. The sixteen mode registers are addressed as 32 bytes. The mode registers occupy addresses 0x0 - 0x1F.

To write a mode register in all five XMAP2 chips the host software must first write the address of the low byte of the mode register into the address register. The address register is written by using the broadcast Write LSB Address Byte command and the broadcast Write MSB Address byte command. These commands cause the same address to be written into the address register in all five XMAP2 chips. Once the mode register low byte is written into the address registers the host then writes the low byte of the mode register by using the broadcast Write Other Data Byte address. Writing this command causes the mode register low byte data to be written into the selected mode register byte in all five XMAP2 chips. The host software then writes the address of the upper byte of the mode register into the address register by writing to the broadcast INCR Address Register address. This causes the address register to be incremented in all five XMAP2 chips. The upper mode register byte is then written by writing to the broadcast Write Other Data Byte command. When accessing the upper byte of a mode register only the low 6 bits are valid since each mode register is only 14 bits wide.

To read a mode register the low byte address is written into the address register by writing to the Write Address LSB byte command and the Write Address MSB byte command for just the single XMAP2 chip that is being read. The low mode register byte is then read by reading the R/W Other Data Byte Command address of the single XMAP2 chip being read. The same procedure is followed to read the upper byte of the mode register. The host software should mask the upper two bits of the upper mode register byte since only the right most six bits are valid.

WID/AUX Control Bit Register

The WID/AUX multiplexer allows the host software to control which four bits are used as an index into the overlay palette registers. This multiplexer is controlled by the WID/AUX Control Bit register. In the base configuration the WID/AUX control bit register is set to 1 so that the overlay color map is indexed using four bits which consist of the 2 PUP bits as the lower 2 bits and the upper 2 bits being 0. For the MGR adapter the XMAP2 chips are never used in the base configuration. In the enhanced configuration the WID/AUX control bit is set to 0 to select the 2 PUP bits as the lower 2 bits of the index and the 2 UAUX bits as the upper 2 bits of the index. These four bits then select one of the sixteen overlay color map words for possible output from the XMAP2.

The WID/AUX Control bit register is accessed by writing address 0x20 in the address register and then issuing the broadcast Write Other Data byte command. Only bit 0 is valid for the WID/AUX Control bit register. The format of the WID/AUX Control Bit register is defined in the Registers section of this chapter.

Overlay Color Map Registers

The overlay color map contains sixteen 24 bit words which each are programmed with the red, green and blue color values used for overlay or underlay pixels. To write a word in the overlay color map the address (0x0 through 0xF) is written into the address register. The broadcast Write Red Data Byte command is then issued to write the red color byte to all five XMAP2 chips. The broadcast Write Green Data Byte command is issued to write the green color byte to all five XMAP2 chips. The broadcast Write Blue Data Byte command is issued to write the blue color byte to all five XMAP2 chips. The broadcast INCR Address command could then be issued to point to the next overlay color map address. The red, green and blue data bytes would then be written in the same manner.

The overlay color map registers are read by writing the address of the register to be read into the address register and then issuing the R/W Red Data Byte, the R/W Green Data Byte and the R/W Blue Data Byte commands to the XMAP2 chip which contains the register to be read. The commands are issued by reading from the appropriate addresses as shown in Table 7.6. The format of the overlay color map registers is defined in the Registers section of this chapter.

External Color Map Words

The XMAP2 contains the necessary address, data and control signals to access an external Color Map which contains 4K words of color data. Each of the five XMAP2 chips has it's own external 4K color map which it controls. Each word in the color map is 24 bits wide and contains a Red, Green and Blue color value. The external color map words are indexed by the Color Index pixels to get the 24 bit RGB values.

The Color Maps are implemented on the MGR adapter with three 8K by 8 bit 35 nsec Static RAM (SRAM) chips. They are controlled by the XMAP2 chips as described above. The host software can use the map select bit (bit 7) in Display Register 4 to select either the low 4K words or the upper 4K words as the current color map.

To write a word in the external color map the address 0x1000 through 0x1FFF is written into the address register. The broadcast Write Red Data Byte command is issued to write the red color byte to all five XMAP2 chips. The broadcast Write Green Data Byte command is issued to write the green color byte to all five XMAP2 chips. The broadcast Write Blue Data Byte command is issued to write the blue color byte to all five XMAP2 chips. These three commands cause the appropriate bytes in the selected external color map word to be written. The broadcast INCR Address command could then be issued to point to the next external color map address. The red, green and blue data bytes would then be written in the same manner.

The external color map registers are read by writing the address of the register to be read into the address register and then issuing the R/W Red Data Byte, the R/W Green Data Byte and the R/W Blue Data Byte commands to the XMAP2 chip which controls the external color map to be read. The commands are issued by reading from the appropriate addresses as shown in Table 7.6. The format of the external color map words are defined in the Registers section of this chapter.

The next section describes how the XMAP2 chips perform the pixel formatting operations.

XMAP2 Pixel Formatting Operations

The XMAP2 chips are used to perform the pixel formatting operations specified by the mode registers. The enhanced configuration of the Display Subsystem contains five XMAP2 chips which form part of the five channel pixel pipeline. The pixels in the Raster Subsystem bitplanes are divided into five pixel groups and each of the five XMAP2 chips processes one of the five pixels. The first pixel is processed by the channel 0 XMAP2 chip. The second pixel is processed by the channel 1 XMAP2 chip. The third pixel is processed by the channel 2 XMAP2 chip. The fourth pixel is processed by the channel 3 XMAP2 chip. Finally the fifth pixel is processed by the channel 4 XMAP2 chip. This processing is continued for each five pixel group along the current scan line. Each XMAP2 chip receives input data from the Raster Subsystem bitplanes and generates outputs which are sent to the three RAMDAC chips.

As shown in Figure 7.6, the input data to the XMAP2 chips consists of 24 bits of Frame Buffer bitplane pixel data (FB_PIXEL), 2 bits of PUP bitplane data (PUP), 2 bits of UAUX data (UAUX) and 2 bits of Window ID (WID) bitplane data from the Raster Subsystem for each pixel it processes. The WID data is used as an index to select one of the sixteen mode registers. The mode word in the selected mode register contains the necessary information to allow the XMAP2 to format the input pixel data and generate as outputs three 8 bit color bytes called ROUT, GOUT and BOUT. The ROUT output byte of each of the five XMAP2 chips is connected to the five pixel byte inputs on the Red RAMDAC. The GOUT output byte of each of the five XMAP2 chips is connected to the five pixel byte inputs on the Green RAMDAC. The BOUT output byte of each of the five XMAP2 chips is connected to the five pixel byte inputs on the Blue RAMDAC. The three RAMDACs use this data to generate the analog RGB signals to drive the high resolution monitor.

The 4 bits of WID data allow the sixteen mode registers to be indexed. The WID bitplane data are used to divide the screen into sixteen windows. All pixels in a window will have the same Window ID written into the WID bitplanes for each pixel in the window. Since the pixels on each scan line are processed in groups of five by the five XMAP2 chips the corresponding mode registers in each XMAP2 chip are programmed with the same values. This allows all pixels with the same WID to be formatted the same. The following paragraphs describe the pixel formatting operations to form the ROUT, GOUT and BOUT outputs of the XMAP2 chips.

ROUT, GOUT and BOUT Output Formation

Each XMAP2 chip formats a single pixel at a time. The three color bytes ROUT, GOUT and BOUT can come from one of following three sources:

- overlay color map
- external color map (Color Index pixels)
- input FB_PIXEL value (RGB pixels)

For each pixel the XMAP2 processes it checks the various bits in the mode register selected by the WID bits to determine the source for the output color values. The XMAP2 first checks for an overlay or underlay pixel condition. If these conditions are not satisfied then the XMAP2 uses the mode register display mode bits to determine the Frame Buffer pixel formatting to be performed. The following sections describe the overlay, underlay or Frame Buffer pixel processing steps performed by the XMAP2 chips.

Overlay and Underlay Pixel Processing

The overlay and underlay processing is dependent on the 2 PUP bits, the 2 UAUX bits and the FB_PIXEL inputs. It is also dependent on the following mode register bits:

- 4 Overlay Enable bits (bits 7-4)
- Underlay Enable bit (bit 8)

The use of these mode register bits and the input data in performing the overlay or underlay processing is shown in Table 7.7.

Table 7.7 Overlay and Underlay Pixel Processing

UAUX[1] & OE[3]	UAUX[0] & OE[2]	PUP[1] & OE[1]	PUP[0] & OE[0]	Underlay Enable	FB_PIXEL	Description
1	X	X	X	X	X	Overlay Pixel
X	1	X	X	X	X	Overlay Pixel
X	X	1	X	X	X	Overlay Pixel
X	X	X	1	X	X	Overlay Pixel
0	0	0	0	1	0	Underlay Pixel
0	0	0	0	1	NZ	Normal Pixel
0	0	0	0	0	X	Normal Pixel

If any of the Overlay Enable bits (bits 7-4) in the mode register and the corresponding PUP or UAUX bit are both one then the overlay condition is true. If the overlay enable bit or the corresponding PUP or UAUX bit is 0 for all four overlay bits then the overlay condition is false. If the overlay condition is false then the XMAP2 checks for an underlay condition. If the Underlay Enable bit (bit 8) in the mode register is a one and the input PIXEL value is a zero then the underlay condition is true. If the Underlay Enable bit is not set or the Underlay Enable bit is set but the PIXEL value is nonzero then the underlay condition is false.

If an overlay or underlay condition is true then the three output color bytes ROUT, GOUT and BOUT come from the selected overlay color map word. The overlay color map selection depends on the setting of the WID/AUX Control Bit register. If this register is set to 0 (enhanced configuration) then the 4 bits formed by the 2 bits of PUP data and the 2 bits of UAUX data are used as an index into the overlay color map. The 2 PUP bits form the LSB 2 bits of the index and the 2 UAUX bits form the MSB 2 bits of the index. If the WID/AUX Control Bit register is set to 1 (base configuration) then the LSB 2 bits of the overlay color map index are the 2 WID bits and the MSB 2 bits of the index are 0. The base configuration of the MGR adapter uses XPC1 chips instead of XMAP2 chips so the WID/AUX control bit register will always be set for 0 on the enhanced configuration.

If the overlay and underlay conditions are both false then the overlay color map word will not be output. The XMAP2 will perform the Frame Buffer pixel processing to determine the output color bytes.

Frame Buffer Pixel Processing

The frame buffer pixels are either color index or RGB pixels. The Frame Buffer pixel processing is a function of the following mode register bits from the mode register selected by the WID data bits:

- Display Mode (Bit 2-0)
- Buffer Select (Bit 3)
- Multimap Mode Enable (Bit 9)
- Multimap constant (Bits 13-10)

The Display mode bits controls the pixel type and should be set to match the pixel type selected with the GE_PIXTYPE token. The XMAP2 chips supports the following types of pixels:

- 4 bit Color Index double buffer pixels
- 8 bit Color Index single buffer pixels
- 12 bit Color Index double buffer pixels
- 8 bit RGB single buffer pixels
- 12 bit RGB double buffer pixels
- 24 bit RGB single buffer pixels

The 4 bit Color Index double buffered, 8 bit Color Index single buffer and 8 bit RGB single buffer pixels are supported are normally not used on the enhanced configuration of the adapter. These pixel types are provided by the XMAP2 chip when it is used in a base adapter configuration.

For double buffer pixels the buffer select bit controls whether the front or back buffer is displayed and is correlated to the pixel write mask as set with the GE_PIXWRITEMASK token. Usually when a double buffer pixel type is specified the host software sets the pixel write mask so that one buffer is being written into while the buffer select bit in the mode register is set so that the opposite buffer is being displayed. When the host software has finished writing into the current buffer the buffer select bit is switched to display the newly written buffer and the pixel write mask is changed to allow the host software to update the opposite buffer. The Buffer select bit is ignored for single buffer pixels and should be set to 0 to select the front buffer. The host software should also set the pixel write mask to allow only the front buffer to be written for single buffer pixels.

The multimap enable bit is used to control whether the multimap constant is used to increase the number of available color indexes. When enabled the multimap constant nibble is placed in the upper nibble of the color index before it is used to perform the color map look up.. This increases the range of colors that the host software can access in the color map.

When an overlay or underlay condition is not true, the XMAP2 processes the FB_PIXEL input data which is the pixel data from the Frame Buffer. The XMAP2 output color bytes are either a 24 bit value selected from the color map or the 24 bit FB_PIXEL value. The Frame Buffer pixel processing is shown in Table 7.8.

Table 7.8 XMAP2 Frame Buffer Pixel Processing

Display Mode Bit 2-0	Buffer Select Bit 3	Multimap Mode Enable Bit 9	Color Map Address Bits 11-8	Color Map Address Bits 7-4	Color Map Address Bits 3-0	ROUT	GOUT	BOUT
000	X	0	0	FB_PIXEL Bits 7-4	FB_PIXEL Bits 3-0	Red CM Byte	Green CM Byte	Blue CM Byte
		1	Multimap Constant					
001	0	0	0	0	FB_PIXEL Bits 3-0	Red CM Byte	Green CM Byte	Blue CM Byte
		1	Multimap Constant					
	1	0	0	0	FB_PIXEL Bits 7-4			
		1	Multimap Constant					
010	0	0	FB_PIXEL Bits 11-8	FB_PIXEL Bits 7-4	FB_PIXEL Bits 3-0	Red CM Byte	Green CM Byte	Blue CM Byte
		1	Multimap Constant					
	1	0	FB_PIXEL Bits 23-20	FB_PIXEL Bits 19-16	FB_PIXEL Bits 15-12			
		1	Multimap Constant					
100	X	X	X	X	X	FB_PIXEL bits 7-0	FB_PIXEL bits 15-8	FB_PIXEL bits 23-16
101	0	X	X	X	X	FB_PIXEL bits 3-0 in both nibbles	FB_PIXEL bits 11-8 in both nibbles	FB_PIXEL bits 19-16 in both nibbles
	1	X	X	X	X	FB_PIXEL bits 7-4 in both nibbles	FB_PIXEL bits 15-12 in both nibbles	FB_PIXEL bits 23-20 in both nibbles

The XMAP2 chip supports display modes 0, 1, 2, 4 and 5. Modes 0, 1 and 2 use the color map to output a 24 bit value to the three RAMDACs while modes 4 and 5 use the 24 bit FB_PIXEL value as the output. The following paragraphs describe these display modes.

8 Bit CI Single Buffer Pixels

Mode 0 is an 8 bit single buffer mode which uses the color map to produce the output to the three RAMDACs. In this mode the buffer select bit in the mode register is ignored. This mode is generally used as an 8 bit Color Index mode on 8 bitplane configurations.

As an 8 bit Color Index mode the 8 bit FB_PIXEL value represents an 8 bit index into the color map which is programmed by the graphics application. In Color Index mode, if the multimap enable bit from the selected mode register is 1 then bits 11-8 of the color map address will be the multimap constant bits from the mode register. If the multimap enable bit is 0 then bits 11-8 of the color map address will be zero.

This mode would also be used for 8 bit RGB pixels which would be treated by the XMAP2 as an 8 bit Color Index value. In this usage the multimap enable bit should be set to 0 so that the multimap constant is not used. The host software would also have to load the special RGB color map into the external color map.

This mode is generally used only for 8 bitplane configurations and would not be used in the extended configuration.

4 Bit CI Double Buffer Pixels

Mode 1 is a 4 bit double buffer mode which uses the color map to produce the output. This mode only operates in Color Index mode and the color map address is dependent on the buffer select bit and the multimap enable bit from the selected mode register. In this mode Color Map address bits 7-4 will always be zero. If the buffer select bit is 0 then Color Map address bits 3-0 will be FB_PIXEL bits 3-0. If the buffer select bit is 1 then Color Map address bits 3-0 will be FB_PIXEL bits 7-4. If the multimap enable bit is 1 then Color Map address bits 11-8 will be the multimap constant bits from the selected mode register. If the multimap enable bit is 0 then bits 11-8 of the color map address will be zero. This mode is generally used only for 8 bitplane configurations and would not be used in the extended configuration.

12 Bit CI Double Buffer Pixels

Mode 2 is a 12 bit double buffer mode which uses the color map to produce the output to the three RAMDACs. In this mode the buffer select bit in the mode register is used to select the appropriate 12 bits from the PIXEL input value as the 12 bit address for the Color Map. If the buffer select bit is 0 then PIXEL bits 11-0 are used as the Color Map address. If buffer select is 1 then PIXEL bits 23-12 are used as the Color Map address.

As a 12 bit Color Index Double Buffer mode the multimap enable bit from the selected mode register determines the upper 4 bits of the Color Map address. If the multimap enable bit is 1 then bits 11-8 of the color map address will be the multimap constant bits from the mode register. If the multimap enable bit is 0 then bits 11-8 of the color map address will be the appropriate input FB_PIXEL bits as described above. This mode is only available for the enhanced configuration.

24 Bit RGB Single Buffer Pixels

Mode 4 is a 24 bit RGB single buffer mode. The 24 bits of FB_PIXEL input data are used as the output from the XMAP2 to the three RAMDACs. The buffer select and multimap bits are ignored. This mode is only available for the enhanced configuration.

12 Bit RGB Single Buffer Pixels

Mode 5 is a 12 bit RGB double buffer mode. The buffer select bit determines which of the 12 input bits are used to form the output from the XMAP2. If the buffer select bit is 0 then FB_PIXEL bits 3-0 are replicated in both nibbles of ROUT. FB_PIXEL bits 11-8 are replicated in both nibbles of GOUT. FB_PIXEL bits 19-16 are replicated in both nibbles of BOUT. If the buffer select bit is 1 then FB_PIXEL bits 7-4 are replicated in both nibbles of ROUT, FB_PIXEL bits 15-12 are replicated in both nibbles of GOUT and FB_PIXEL bits 23-19 are replicated in both nibbles of BOUT. The multimap enable bit is ignored in this mode.

RAMDAC

The three RAMDACs are Brooktree Bt457 chips which are configured as a Red RAMDAC, a Green RAMDAC and a Blue RAMDAC. Each RAMDAC contains an input multiplexer which can accept the 8-bit pixel color input and the 2-bit overlay input from five different sources. These five inputs are referred to as the A through E inputs. Each RAMDAC has the appropriate color byte from each of the five XMAP2 chips connected to its five input ports. The two bits output by the Cursor chips are connected to the overlay inputs on all five inputs of each RAMDAC. The color bytes are used as an index into a gamma correction color palette. The two cursor bits are an index into an overlay palette. The 8 bit output of the color palette or overlay palette is then converted by the DAC to an analog signal which drives one of the three RGB outputs to the high resolution color monitor. The Red DAC provides the R analog signal, the Green DAC provides the G analog signal and the Blue DAC provides the B analog signal. Figure 7.8 shows the RAMDAC block diagram for the Red color channel.

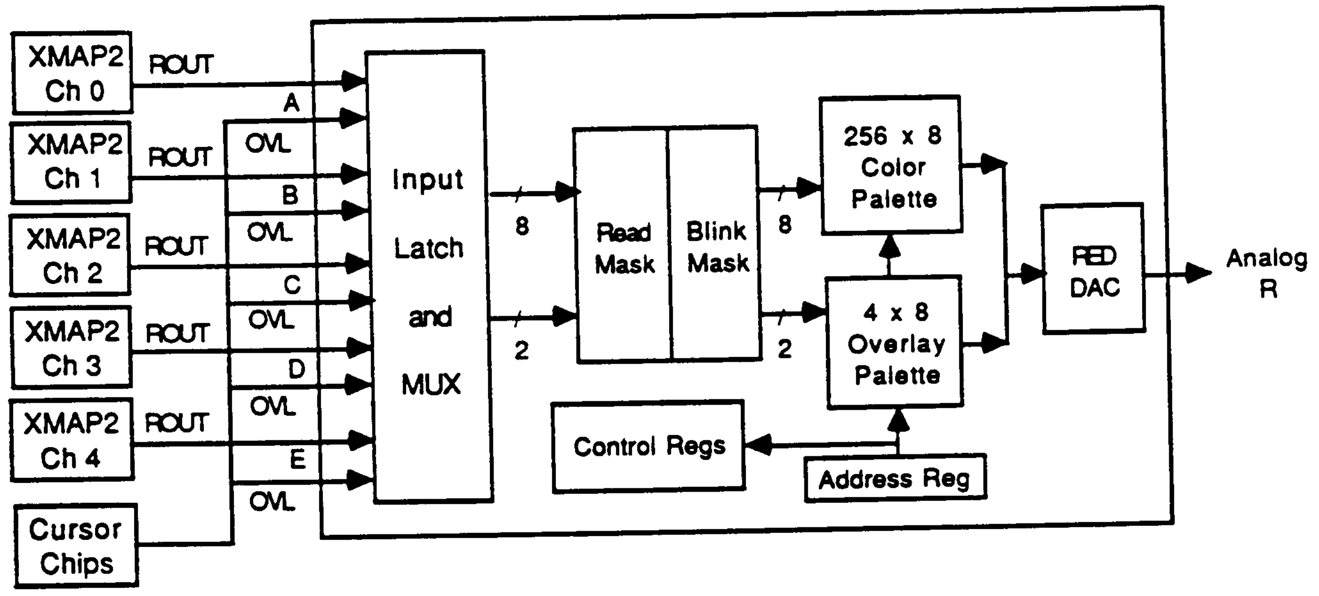

Figure 7.8 RAMDAC Block Diagram

The following sections describe the host access to the programmable registers and the basic operations of the RAMDAC.

Host Access to the RAMDAC Registers

Each RAMDAC contains an address register and four control registers. The control registers include a command register, a read mask register, a blink mask register and a test register. Each RAMDAC also contains a 256 x 8 bit color palette and a 4 x 8 bit overlay palette. Table 7.9 shows the address decoding to access the registers, color palette and overlay palette in the three RAMDACs. The HQMMSB Register must be set to 1 to read or write the three RAMDACs.

Address Register

The address register is used to specify a location in the color palette, the overlay palette or select one of the four control registers. The address register is loaded with the values shown in Table 7.9 and then the palette entry or register is accessed.

Table 7.9 Red, Green and Blue RAMDAC Address Map

Host Address	RAMDAC Address Register Value	Source/Destination	
500 Hex	XX	RAMDAC Address Register	Red RAMDAC
504 Hex	00 - FF Hex	Color Palette RAM	
508 Hex	04	Read Mask Register	
508 Hex	05	Blink Mask Register	
508 Hex	06	Command Register	
508 Hex	07	Control/Test Register	
50C Hex	00 - 03	Overlay Palette RAM	
520 Hex	XX	RAMDAC Address Register	Green RAMDAC
524 Hex	00 - FF Hex	Color Palette RAM	
528 Hex	04	Read Mask Register	
528 Hex	05	Blink Mask Register	
528 Hex	06	Command Register	
528 Hex	07	Control/Test Register	
52C Hex	00 - 03	Overlay Palette RAM	
540 Hex	XX	RAMDAC Address Register	Blue RAMDAC
544 Hex	00 - FF Hex	Color Palette RAM	
548 Hex	04	Read Mask Register	
548 Hex	05	Blink Mask Register	
548 Hex	06	Command Register	
548 Hex	07	Control/Test Register	
54C Hex	00 - 03	Overlay Palette RAM	

HQMMSB Register must be set to 1

Color Palette

Each RAMDAC contains a 256 x 8 bit color palette which is used to perform gamma correction on the input color byte. The color palette byte is then output to the DAC for conversion to analog. To read a color value from or write a color value to the color palette the desired entry in the color palette is written into the address register and then the color byte is read or written. After the color value is read or written the address register is incremented. If it is 0xFF it is wrapped to 0.

If the appropriate color bits are set in the control/test register then the three color bytes can be written the same as for the RGB RAMDAC. Each of the three RAMDACs will only respond to the color for which the control register color bit is set. When the color bits are set then the host software writes the three color bytes and each RAMDAC loads the appropriate byte into the color palette..

After the third color byte is read or written the address register is automatically incremented. This allows the host software to write the starting address and then read or write groups of three colors to the the color palette. If the address register contains a 0xFF it wraps to 0x00 after the third color byte is read from or written to the color palette.

Overlay Palette

The chip also contains a 4 x 8 bit overlay palette which is selected by the 2-bit overlay input from the two cursor chips. To read color values from or write color values to the overlay palette the desired entry in the overlay palette is written into the address register and then the color byte is read or written. After the color value is read or written the address register is incremented. If it is 0x3 it is incremented to 4.

If the appropriate color bits are set in the control/test register then the three color bytes can be written the same as for the RGB RAMDAC. Each of the three RAMDACs will only respond to the color for which the control register color bit is set. When the color bits are set then the host software writes the three color bytes are read from or written to the overlay palette address. The three color bytes are the red, green and blue color values. After the third color byte is read or written the address register is automatically incremented. This allows the host software to write the starting address and then read or write groups of three colors to set the colors to the overlay palette. If the address register contains a 3 it increments to 4 after the third color byte is read from or written to the overlay palette.

Command Register

The command register is used to control the operation of the input multiplexer, the overlay transparency, the blink rate, the overlay blink and the overlay display. The command register is read or written by writing the value 6 into the address register and then reading or writing the control register address.

The input multiplexer control bit can select either four or five input sources. The MGR always selects five since it has five XPC1s which are connected to the RGB RAMDACs inputs.

The overlay transparency control bit allows the overlay input equal to zero to be made transparent or to use the color value stored in overlay palette location 0. If it is made transparent then the color value in the color palette will be used instead of the value in the overlay palette. This bit is set for transparent mode in the MGR since a zero or nonzero value on the OVL inputs is used to determine if a word from the color palette or the overlay palette is used to get the RGB color values.

The blink rate control bits allow the number of vertical retrace periods the pixels will be on and off to be specified.

The overlay display control allows the overlay bits to be used or ignored. The MGR is programmed to utilize both overlay bits.

Read Mask Register

The read mask register allows bits in the input pixel bytes to be forced to zero. The MGR is programmed to not mask any of the eight bits. The read register is read or written by writing the value 4 into the address register and then reading or writing the control register address.

Blink Mask Register

The blink mask register controls which of the pixel bitplanes will blink. The MGR is programmed to not allow any of the bitplanes to blink. Blink is accomplished by varying the color palette values. The overlay blink control allows the overlay bit blinking capability. If enabled the overlay colors could be blinked between the overlay color and the transparent color. This capability is not used on the MGR. The blink mask register is read or written by writing the value 5 into the address register and then reading or writing the control register address.

Test Register

The control/test register allows a nibble of the output data to be read. It also allows the RAMDAC color or overlay palettes to be accessed in a manner which is compatible with the RGB RAMDAC. If the appropriate color enable bits are set in the control/test register in the three RAMDACs then the three color bytes can be accessed with three successive reads or writes to the same address as in the RGB RAMDAC. Each RAMDAC will only respond for the color byte enabled by the color enable bits in the control/test register. The format of this register is described in the Registers section of this chapter. The test register is read or written by writing the value 7 into the address register and then reading or writing the control register address.

RAMDAC Basic Operations

The input multiplexer can accept the 8-bit pixel input and the 2-bit overlay input from five different sources. The MGR takes the appropriate color byte from the five XMAP2 chips and routes them to the five different pixel input channels on the RAMDAC. These five inputs are referred to as the A through E inputs. The two bits from the cursor chips are also connected to the five 2 bit overlay inputs on each RAMDAC. Each time the Display State Machine generates a load signal to the RAMDACs the 10-bits on the five inputs are latched. The DSM then generates clock pulses which cause the each RAMDAC to process the latched input data. On the first clock the A input is processed then on the second clock the B input is processed and so on until finally the E input is processed. This processing is then repeated for each group of five pixels.

On each clock cycle generated by the Display State Machine, the selected eight bits of pixel data and two bits of overlay data are processed by the RAMDACs using the read mask, the blink mask and the various fields in the control register. The read mask is used to force some of the input bits to zero. The host software should program the read mask to not mask any of the input bits. The blink mask is used to cause some of the input bits to vary between zero and one at the blink rate programmed into the control register. The host software should program the blink register and the control register so that blinking is disabled. The control register determines how the overlay bits are used. The host software should program the host software so that the overlay bits are not masked and the overlay bits equal to zero are treated as a transparency. This allows the pixel bits to select a color palette entry. The 8 bits of pixel data and the 2 bits of overlay data are left unchanged by the read and blink mask registers and are applied directly as an index to the color palette and the overlay palette.

Each RAMDAC contains a 256 x 8 bit color palette and a 4 x 8 bit overlay palette. The 8 bit values in the color palette and the overlay palette consist of 8 bits of Red color data in the Red DAC, 8 bits of Green data in the Green DAC and 8 bits of Blue data in the Blue DAC. The color values stored in the color palette are used as a gamma correction factor for the color values. The overlay palette will contain the three available cursor colors.

If the 2 bit overlay input is nonzero the 2 bit overlay input will be used to select an entry in the overlay palette. If the overlay input is zero then the 8 bit pixel input data is used as an index into the color palette. The selected entry in the overlay or color palette outputs the 8 bit color value to the DAC. The red color value is output to the Red DAC, the green color value to the Green DAC and the blue color value to the Blue DAC in the three different RAMDACs. The three DACs convert the bytes to analog signals which conform to RS-343A and are capable of driving doubly-terminated 75-ohm coaxial cable directly, without requiring external buffering. Each RAMDAC also merges the SYNC and BLANK signals onto the analog RGB signals.

Registers

The following paragraphs describe the registers contained in the various components used in the base and enhanced configurations. The registers in the following components are described:

- XPC1 registers
- RGB RAMDAC registers
- Display registers
- XMAP2 registers
- Color Map words
- RAMDAC registers.

XPC1 Registers

The following paragraphs define the bit definitions for the following XPC1 registers:

- Address Register
- Mode Registers

XPC1 Address Register

The XPC1 Address Register contains the address of a mode register data byte. The four mode registers are addressed as 8 bytes which means that only bits 2-0 of the address register are valid. The address register format is shown in Figure 7.9.

7	6	5	4	3	2	1	0
0	0	0	0	0	Address		

Figure 7.9 XPC1 Address Register

Bits 7-3 : Unused

Bits 2-0 : Mode Register Byte Address. These bits are used to select one of the 8 mode register bytes for reading or writing.

XPC1 Mode Registers

The XPC1 has a table of four Mode Registers which each have the same format and functionality. The four registers are selected by the 2 bit WID input values.

The Mode Register controls the pixel output format and the overlay output format of the XPC1 for each window ID. The format of the mode register is shown in Figure 7.10.

13	12	11	10	9	8	7	6	5	4	3	2	1	0
Multimap Constant				ME	UE	0	0	OE		BS	0	0	DM

Figure 7.10 XPC1 Mode Register

The following bit definitions use the following input and output names:

FB_PIXEL - 8 bitplane input Frame Buffer pixel value
PUP - 2 bitplane input auxiliary value
CURSOR - 2 bitplane input cursor value

XOUT - 8 bit output pixel value
OVL - 2 bit overlay value

Bits 13-10 : Multimap Constant (MC) bits. These bits specify the upper 4 bits of the XOUT pixel data byte when bit 9 is 1 to enable multimap mode.

mmmm - contains the multimap nibble bits.

Bit 9 : Multimap Mode Enable (ME) bit. This bit is used to enable the multimap mode. In this mode the upper 4 bits of XOUT are formed from bits 10-13 of the mode register if the display mode is 4 bit CI double buffer mode.

0 - Multimap mode disabled
1 - Multimap mode enabled (XOUT bits 7-4 = mmmm)

Bit 8 : Underlay Enable (UE) bit. This bit is used to enable the use of the PUP input as an underlay value. If the bit is a 1 the OVL output is set to PUP in this case if the CURSOR and FB_PIXEL data are 0 and the overlay condition is not true.

if CURSOR == 0 and FB_PIXEL == 0 and not overlay then

0 - OVL == 00
1 - OVL == PUP

Bits 7-6 : Unused

Bits 5-4 : Overlay Enable (OE) Bits. These bits are used to select the PUP bitplanes as the OVL output if either PUP bit and its corresponding enable are both set to 1. This is dependent on the CURSOR input bits which must be 0. If either CURSOR bit is 1 then OVL = CURSOR.

if CURSOR = 00 then

00 - OVL bits = 00 if underlay mode not selected
01 - OVL bits = PUP bits if PUP bit 0 set
10 - OVL bits = PUP bits if PUP bit 1 set
11 - OVL bits = PUP bits if either PUP bit set

Bit 3 : Buffer Select (BS) bit. This bit is used to select the front or the back buffer for display for double buffer pixel types.

0 - Front Buffer
1 - Back Buffer

Bits 2-1 : Unused

Bit 0 : Display Mode (DM) bit. This bit is used to control the display mode of the XPC1.

0 - 8 bit CI or 8 bit RGB mode single buffer
1 - 4 bit CI double buffer mode

RGB RAMDAC Registers

The following paragraphs describe the following RGB RAMDAC registers:

- address register
- command register
- read mask register
- blink mask register
- test register

Address Register

The address register is used to specify a location in the color palette, a location in the overlay palette or select one of the other four control registers. The format of the address register is shown in Figure 7.11.

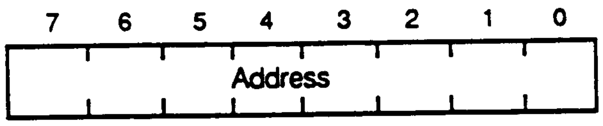

Figure 7.11 RGB RAMDAC Address Register

Bits 7-0 : Address. These 8 bits contain addresses.

00 - FF	color palette
00 - 03	overlay palette
04	read mask register
05	blink mask register
06	command register
07	test register

Command Register

The command register is shown in Figure 7.12 and is used to control the operation of the RGB RAMDAC. It controls the input multiplexing, the internal RAM enables, the blink rate selection and the overlay enables.

7	6	5	4	3	2	1	0
MS	RE	BR		BE1	BE0	DE1	DE0

Figure 7.12 RGB RAMDAC Command Register

Bit 7 : Multiplex Select (MS). This bit specifies whether 4:1 or 5:1 multiplexing is to be used for the pixel and overlay inputs (A-E).

0 - 4:1 multiplexing
1 - 5:1 multiplexing (Used on the MGR)

Bit 6 : RAM Enable (RE). When the overlay input bits are 00, this bit specifies whether to use the color palette RAM or overlay color 0 to provide color information.

0 - use overlay color 0
1 - use color palette RAM (transparent mode used on MGR)

Bits 5-4 : Blink rate selection (BR). These bits control the blink rate cycle time and duty cycle. They are specified as the number of vertical retrace intervals. The parentheses specify the duty cycle (%on/off).

00 - 16 on, 48 off (25/75)
01 - 16 on, 16 off (50/50)
10 - 32 on, 32 off (50/50)
11 - 64 on, 64 off (50/50)

Bit 3 : OL1 Blink Enable (BE1). This bit is used to enable or disable blinking on the overlay input bit OL1 (A-E) inputs. If enabled the OL1 inputs are toggled between the input value and logical 0 at the specified blink rate. If disabled the OL1 inputs are used unchanged. This bit is dependent on Bit 1 being set to logical 1.

0 - disable blinking (Selected on MGR)
1 - enable blinking

Bit 2 : OL0 Blink Enable (BE0). This bit is used to enable or disable blinking on the overlay input bit OL0 (A-E) inputs. If enabled the OL0 inputs are toggled between the input value and logical 0 at the specified blink rate. If disabled the OL0 inputs are used unchanged. This bit is dependent on Bit 0 being set to logical 1.

0 - disable blinking (Selected on MGR)
1 - enable blinking

Bit 1 : OL1 Display Enable (DE1). This bit is used to enable or disable the use of the OL1 (A-E) inputs. If enabled the OL1 inputs are used. If disabled the OL1 inputs are set to logical 0.

0 - disable OL1 input
1 - enable OL1 input (Selected on MGR)

Bit 0 : OL0 Display Enable (DE0). This bit is used to enable or disable the use of the OL0 (A-E) inputs. If enabled the OL0 inputs are used. If disabled the OL0 inputs are set to logical 0.

0 - disable OL0 input
1 - enable OL0 input (Selected on MGR)

Read Mask Register

The read mask register is used to enable or disable a bit plane from addressing the color palette RAM. The eight bits in the read mask register mask the corresponding bits P7-P0 in the input byte for each of the A-E input bytes. Bit 0 corresponds to P0 and bit 7 corresponds to P7. Each register bit is logically anded with the corresponding input bit. This register may be read or written at any time and has no initial value. The MGR sets all eight bits to not mask any bitplanes. The format of the register is shown in Figure 7.13.

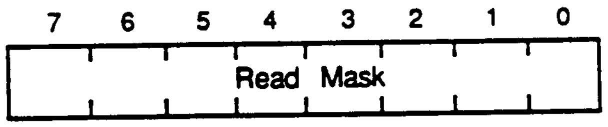

Figure 7.13 RGB RAMDAC Read Mask Register

Bits 7-0 : Read Mask. These bits allow any of the eight bits in the input bytes to be disabled or enabled.

0 - Disable input bit
1 - Enable input bit (Selected on MGR)

Blink Mask Register

The blink mask register is used to enable or disable a bit plane from blinking at the blink rate and duty cycle specified by the command register bits 5-4. The eight bits in the read mask register mask the corresponding bits P7-P0 in the input byte for each of the A-E input bytes. Bit 0 corresponds to P0 and bit 7 corresponds to P7. In order for a bit plane to blink, the corresponding bit in the read mask register must be a logical one. This register may be read or written at any time and has no initial value. The MGR disables blink on all bitplanes. The format of the register is shown in Figure 7.14.

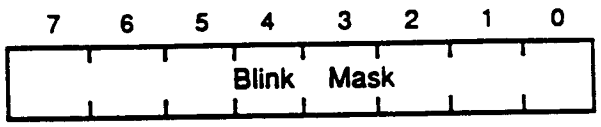

Figure 7.14 RGB RAMDAC Blink Mask Register

Bits 7-0 : Blink Mask. These bits allow any of the eight bits in the input bytes to be disabled or enabled for blinking.

0 - Disable Blink (Selected on MGR)
1 - Enable Blink

Test Register

The test register provides diagnostic capability by allowing the selected output nibble to a selected DAC to be read. It may be read or written at any time and has no initial value. Only one output nibble at a time can be read. The format of the register is shown in Figure 7.15.

7	6	5	4	3	2	1	0
Output Data Read				NS	BE	GE	RE

Figure 7.15 RGB RAMDAC Test Register

Bits 7-4 : Read Data Nibble. These 4 bits contain the data nibble selected by bits 3-0 when the register is read. These bits are ignored for a write.

Bits 3 : Nibble Select (NS). This bit is used to select either the high or low nibble of the output byte.

0 - Select high nibble
1 - Select low nibble

Bit 2 : Blue Enable (BE). This bit selects the Blue output as the data to be read.

0 - Don't read Blue output
1 - Read the selected nibble of the Blue output

Bit 1 : Green Enable (GE). This bit selects the Green output as the data to be read.

0 - Don't read Green output
1 - Read the selected nibble of the Green output

Bit 0 : Red Enable (RE). This bit selects the Red output as the data to be read.

0 - Don't read Red output
1 - Read the selected nibble of the Red output

Display Registers

The Display Subsystem has five Display Registers which are used to return status information and to control the Display State Machine operation. All bits which are defined are reset to 0 on power on or reset of the MGR adapter. The following display registers are described in the following paragraphs:

- Display Register 0
- Display Register 1
- Display Register 2
- Display Register 3
- Display Register 4

Display Register 0

Display Register 0 returns the Z buffer card installed status to the other MGR subsystems and to the host when read. The format of Display Register 0 is shown in Figure 7.16.

7	6	5	4	3	2	1	0
X	X	X	X	ZBIN	X	X	X

Figure 7.16 Display Register 0

Bits 7-4 : Undefined

Bit 3 : ZBIN (Read Only). Indicates if the Z buffer card (MZB1) is installed.

0 - MZB1 card is installed
1 - MZB1 card is not installed

Bits 2-0 : Undefined

Display Register 1

Display Register 1 allows the sync on the Green output signal to be enabled or disabled. The format of this register is shown in Figure 7.17.

7	6	5	4	3	2	1	0
X	X	X	X	0	VRLS	MTYPE	SE

Figure 7.17 Display Register 1

Bits 7-4 : Undefined

Bit 3 : Reserved

Bit 2 : Vertical Retrace Latency Select (VRLS) bit. This bit selects either a short or long time interval for the vertical retrace latency time. This bit determines the number of scan lines before the start of the vertical retrace when the vertical retrace interrupt is generated. This bit only affects the 60 Hz monitor.

0 - long latency time

scan lines	monitor type
8	60 Hz
2	30 Hz
1	RS170
1	EURO
8	STEREO

1 - short latency time

scan lines	monitor type
2	60 Hz
2	30 Hz
1	RS170
1	EURO
8	STEREO

Bit 1 : Monitor Type (MTYPE) bit (R/W). This bit is the most significant bit of the three bit monitor type selection field. The least significant two bits of the monitor type selection field are bits 1-0 of Display Register 4. Refer to Display Register 4 for the monitor types supported.

Bit 0 : Sync Enable (SE) bit (R/W). This bit allows the Green output signal to the monitor to contain the sync information or to have the sync information not included.

0 - Sync Output Enabled
1 - Sync Output Disabled

Display Register 2

Display Register 2 is used to place the MGR in a standby mode and to return the status of whether the enhanced display card (MEV2) is installed. In standby mode the screen is blanked and the display refresh is stopped. The format of this register is shown in Figure 7.18.

7	6	5	4	3	2	1	0
X	X	X	X	BPIN	0	0	SB

Figure 7.18 Display Register 2

Bits 7-4 : Undefined

Bit 3 : BPIN (Read only). This bit indicates if the enhanced video card is installed.

0 - Enhance Video card (MEV2) installed
1 - MEV2 card not installed (MDE1 card installed instead)

Bits 2-1 : Reserved

Bit 0 : Standby (R/W). This bit controls whether the MGR is in normal display operational mode or in standby mode.

0 - Standby mode
1 - Normal Operation mode

Display Register 3

Display Register 3 is used to reload the Xilinx chips logic configuration from the 8K PROM. It is also used to enable a newly selected monitor type. It returns the FIFO empty status to the other MGR subsystems. The format of this register is shown in Figure 7.19.

7	6	5	4	3	2	1	0
X	X	X	X	FE	MS	LR	X

Figure 7.19 Display Register 3

Bits 7-4 : Undefined

Bit 3 : FIFO Empty (Read only). This bit indicates if the FIFO in the Geometry Subsystem is empty.

0 - FIFO is empty
1 - FIFO is not empty

Bit 2 : Monitor Set (R/W). This bit is used to enable a newly selected monitor type. The bit is set to 0 and then back to 1 to take effect.

0 - Load new monitor timing table from PROM
1 - Normal operation

Bit 1 : LCARESET (R/W). This bit is used to reload the Xilinx logic configuration data from the PROM. The bit is set to 1 and then back to 0 to tack effect.

0 - Normal operation
1 - Load Xilinx logic data from PROM

Bit 0 : Undefined

Display Register 4

Display Register 4 is used to select the monitor type, the clock source and to enable the asynchronous and synchronous clocks. The format of this register is shown in Figure 7.20.

7	6	5	4	3	2	1	0
MS	CES	CEA	X	X	ECK	MTYPE	

Figure 7.20 Display Register 4

Bit 7 : MS (R/W). This bit is used to select either the lower 4K color map or the upper 4K color map.

0 - Selects the lower 4K color map (power on reset default)
1 - Selects the upper 4K color map

Bit 6 : CKENS (Write only). This bit is used to enable or disable the synchronous clock.

0 - Synchronous clock disabled
1 - Synchronous clock enabled

Bit 5 : CKENA (Write only). This bit is used to enable or disable the asynchronous clock.

0 - Asynchronous clock disabled
1 - Asynchronous clock enabled

Bits 4-3 : Undefined

Bit 2 : EXTCK (R/W). This bit is used to select the internal clock or the external Pixel Clock.

0 - Use Internal Display State Machine clock
1 - Use External Pixel Clock

Bits 1-0 : MTYPE (R/W). These two bits are the least significant two bits of the three bit monitor type selection field. The third bit of the monitor type selection field is bit 1 of Display Register 1.

D1	D4		
1	1	0	bits
0	0	0	- 1280 x 1024 60Hz noninterlaced
0	0	1	- 1280 x 1024 30Hz interlaced
0	1	0	- RS170 (NTSC)
0	1	1	- PAL
1	0	0	- STEREO
1	0	1	- undefined
1	1	0	- undefined
1	1	1	- undefined

XMAP2 Registers

The paragraphs in this section describe the format of the following XMAP2 registers:

- address register
- mode registers
- overlay color map registers
- WID/AUX control bit register
- color map words

XMAP2 Address Register

The XMAP2 Address Register contains the address of a mode register data byte, an auxiliary color map register, a word in the external color map or the WID/AUX control bit register. The address register is 13 bits wide and is programmed using the Write LSB Address byte command and the Write MSB Address byte command. Only the low five bits are valid for the Write MSB Address byte command. The address register format is shown in Figure 7.21.

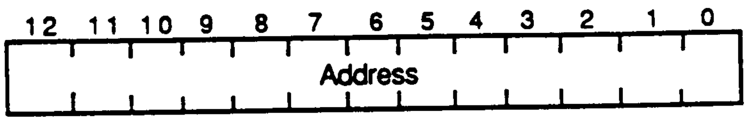

Figure 7.21 XMAP2 Address Register

Bits 12-0 : Address bits (R/W). These bits are used to select one of the 32 mode register bytes, one of the 16 auxiliary color map registers, one of the external 4K words of the color maps or the WID/AUX control bit register.

0 - 1F hex	Mode Register Byte or Auxiliary Color Map word
20 hex	WID/AUX control bit register
1000 - 1FFF hex	Color Map word

XMAP2 Mode Registers

The XMAP2 has a table of sixteen Mode Registers which each have the same format and functionality. The sixteen registers are selected by the WID input values. The Mode Register controls the XMAP2 display format and thus the 24 bit output value for each pixel. The format of the mode register is shown in Figure 7.22.

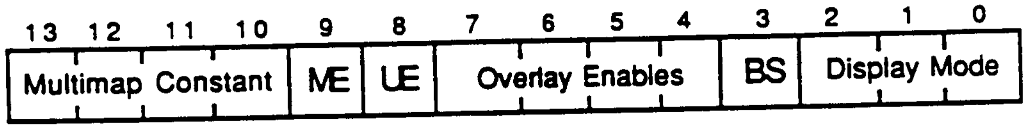

Figure 7.22 XMAP2 Mode Register

The following bit definitions use the following input and output names:

FB_PIXEL - 24 bit input pixel value
PUP - 2 bit PUP bitplane value
UAUX - 2 bit UAUX bitplane value

Bits 13-10 : Multimap Constant bits. These bits specify the upper 4 bits of the color map address in Color Index modes when bit 9 is set to 1 to enable multimap mode.

mmmm - contains the multimap nibble bits.

Bit 9 : Multimap Mode Enable (ME) bit. This bit is used to enable the multimap mode. In this mode the upper 4 bits of the color map address are formed from bits 10-13 of the mode register if the display mode is a Color Index mode.

0 - Multimap mode disabled
1 - Multimap mode enabled (Color Map Address bits 11-8 = mmmm)

Bit 8 : Underlay Enable (UE) bit. This bit is used to enable the use of the overlay color map as an underlay pixel color if the FB_PIXEL value is 0 and the overlay condition is not true.

if FB_PIXEL == 0 and not overlay then

0 - underlay not enabled
1 - underlay enabled (XMAP2 output comes from auxiliary color map word)

Bits 7-4 : Overlay Enable Bits. These bits are used to select the overlay Color Map as the XMAP2 output if any 1 of the PUP or UAUX bits and its corresponding enable bit are both set to 1.

0000	- Overlay condition not enabled
XXX1	- Overlay Enabled if PUP bit 0 set
XX1X	- Overlay Enabled if PUP bit 1 set
X1XX	- Overlay Enabled if UAUX bit 0 set
1XXX	- Overlay Enabled if UAUX bit 1 set
1111	- Overlay Enabled if any PUP or UAUX bit set

Bit 3 : Buffer Select (BS) bit. This bit is used to select the front or the back buffer for display.

0 - Front Buffer
1 - Back Buffer

Bits 2-0 : Display Mode (DM) bits. These bits are used to control the display mode of the XMAP2.

000	8 bit CI or 8 bit RGB single buffer mode
001	4 bit CI double buffer mode
010	12 bit CI double buffer mode
100	24 bit RGB single buffer mode
101	12 bit RGB double buffer mode

other combinations are invalid

Overlay Color Map Registers

The XMAP2 has sixteen overlay color map registers. The 2 bit PUP and the 2 bit UAUX inputs are used to select one of the sixteen registers during pixel processing operations if the WID/AUC Control Bit register is set to 0. If the WID/AUX control bit register is set to one then the 2 PUP bits and 2 zero bits are used to access the overlay color map during pixel processing operations. The address register is used to select one of the registers for host access. Each register is 24 bits wide and contains a Red, Green and Blue color byte as shown in Figure 7.23. The R/W Red byte command is used to access the red color byte. The R/W Green byte command is used to access the green color byte. The R/W Blue byte command is used to access the blue color byte.

23 16	15 8	7 0
Blue Color Byte	Green Color Byte	Red Color Byte

Figure 7.23 Auxiliary Color Map Register

Bits 23-16 : Blue color byte

Bits 15-8 : Green color byte

Bits 7-0 : Red color byte

WID/AUX Control Bit Register

The WID/AUX control bit is used to control the WID/AUX multiplexer. The output of the WID/AUX multiplexer is used to access the overlay color map register during pixel processing operations. The register has only a single bit as shown in Figure 7.24.

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	CB

Figure 7.24 XMAP2 WID/AUX Control Bit Register

Bits 7-1 : Unused

Bit 0 : WID/AUX Control Bit (CB). This bit is used to select either two PUP bits and 2 zero bits as the overlay register index or 2 PUP bits and 2 UAUX bits as the index. The output of the WID/AUX multiplexer is used to select one of the overlay color map registers.

0 - 2 PUP bits 2 2 UAUX bits selected
1 - 2 PUP bits + 2 bits of zero selected

Color Map Words

Each Color Map has 4K words of color data which are accessed via the XMAP2 chips. The XMAP2 address register is used to select one of the words for host access. Each word is 24 bits wide and contains a Red, Green and Blue color byte as shown in Figure 7.25. The R/W Red byte command is used to access the red color byte. The R/W Green byte command is used to access the green color byte. The R/W Blue byte command is used to access the blue color byte.

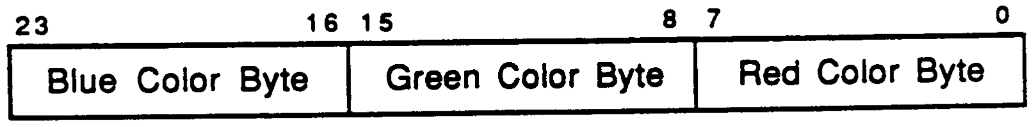

Figure 7.25 Color Map Word Format

Bits 23-16 : Blue color byte

Bits 15-8 : Green color byte

Bits 7-0 : Red color byte

RAMDAC Registers

The following paragraphs describe the following RAMDAC registers:

- address register
- command register
- read mask register
- blink mask register
- control/test register

Address Register

The address register is used to specify a location in the color palette, a location in the overlay palette or select one of the other four control registers. The format of the address register is shown in Figure 7.26.

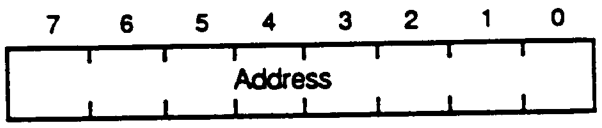

Figure 7.26 RAMDAC Address Register

Bits 7-0 : Address. These 8 bits contain addresses.

00 - FF	for color palette
00 - 03	for overlay palette
04	for read mask register
05	for blink mask register
06	for command register
07	for control/test register

Command Register

The command register is shown in Figure 7.27 and is used to control the operation of the RAMDAC. It controls the input multiplexing, the internal RAM enables, the blink rate selection and the overlay enables.

7	6	5	4	3	2	1	0
MS	RE	BR		BE1	BE0	DE1	DE0

Figure 7.27 RAMDAC Command Register

Bit 7 : Multiplex Select (MS). This bit specifies whether 4:1 or 5:1 multiplexing is to be used for the pixel and overlay inputs (A-E).

0 - 4:1 multiplexing
1 - 5:1 multiplexing (Used on the MGR)

Bit 6 : RAM Enable (RE). When the overlay input bits are 00, this bit specifies whether to use the color palette RAM or overlay color 0 to provide color information.

0 - use overlay color 0
1 - use color palette RAM (transparent mode used on MGR)

Bits 5-4 : Blink rate selection (BR). These bits control the blink rate cycle time and duty cycle. They are specified as the number of vertical retrace intervals. The parentheses specify the duty cycle (%on/off).

00 - 16 on, 48 off (25/75)
01 - 16 on, 16 off (50/50)
10 - 32 on, 32 off (50/50)
11 - 64 on, 64 off (50/50)

Bit 3 : OL1 Blink Enable (BE1). This bit is used to enable or disable blinking on the overlay input bit OL1 (A-E) inputs. If enabled the OL1 inputs are toggled between the input value and logical 0 at the specified blink rate. If disabled the OL1 inputs are used unchanged. This bit is dependent on Bit 1 being set to logical 1.

0 - disable blinking (Selected on MGR)
1 - enable blinking

Bit 2 : OL0 Blink Enable (BE0). This bit is used to enable or disable blinking on the overlay input bit OL0 (A-E) inputs. If enabled the OL0 inputs are toggled between the input value and logical 0 at the specified blink rate. If disabled the OL0 inputs are used unchanged. This bit is dependent on Bit 0 being set to logical 1.

0 - disable blinking (Selected on MGR)
1 - enable blinking

Bit 1 : OL1 Display Enable (DE1). This bit is used to enable or disable the use of the OL1 (A-E) inputs. If enabled the OL1 inputs are used. If disabled the OL1 inputs are set to logical 0.

0 - disable OL1 input
1 - enable OL1 input (Selected on MGR)

Bit 0 : OL0 Display Enable (DE0). This bit is used to enable or disable the use of the OL0 (A-E) inputs. If enabled the OL0 inputs are used. If disabled the OL0 inputs are set to logical 0.

0 - disable OL0 input
1 - enable OL0 input (Selected on MGR)

Read Mask Register

The read mask register is used to enable or disable a bit plane from addressing the color palette RAM. The eight bits in the read mask register mask the corresponding bits P7-P0 in the input byte for each of the A-E input bytes. Bit 0 corresponds to P0 and bit 7 corresponds to P7. Each register bit is logically anded with the corresponding input bit. This register may be read or written at any time and has no initial value. The MGR sets all eight bits to not mask any bitplanes. The format of the register is shown in Figure 7.28.

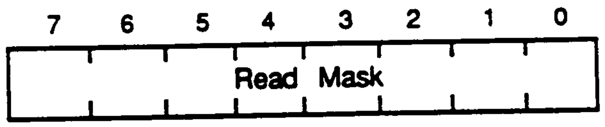

Figure 7.28 RAMDAC Read Mask Register

Bits 7-0 : Read Mask. These bits allow any of the eight bits in the input bytes to be disabled or enabled.

0 - Disable input bit
1 - Enable input bit (Selected on MGR)

Blink Mask Register

The blink mask register is used to enable or disable a bit plane from blinking at the blink rate and duty cycle specified by the command register bits 5-4. The eight bits in the read mask register mask the corresponding bits P7-P0 in the input byte for each of the A-E input bytes. Bit 0 corresponds to P0 and bit 7 corresponds to P7. In order for a bit plane to blink, the corresponding bit in the read mask register must be a logical one. This register may be read or written at any time and has no initial value. The MGR disables blink on all bitplanes. The format of the register is shown in Figure 7.29.

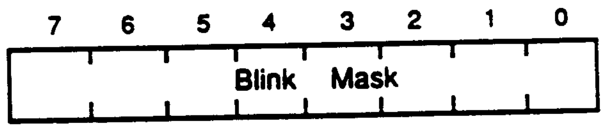

Figure 7.29 RAMDAC Blink Mask Register

Bits 7-0 : Blink Mask. These bits allow any of the eight bits in the input bytes to be disabled or enabled for blinking.

0 - Disable Blink (Selected on MGR)
1 - Enable Blink

Control/Test Register

The control/test register provides diagnostic capability by allowing the selected output nibble to a selected DAC to be read. It may be read or written at any time and has no initial value. Only one output nibble at a time can be read. The register also allows the RAMDAC color or overlay palette to be accessed in a manner that is compatible with the RGB RAMDAC. If any one of the color enable bits is set to logical one then the RAMDAC expects that three bytes of color data are being written to each RAMDAC address as in the RGB RAMDAC. Each RAMDAC only responds to a single color byte of the three depending on which color enable bit is set. The format of the register is shown in Figure 7.30.

7	6	5	4	3	2	1	0
Output Data Read				NS	BE	GE	RE

Figure 7.30 RAMDAC Control/Test Register

Bits 7-4 : Read Data Nibble. These 4 bits contain the data nibble selected by bits 3-0 when the register is read. These bits are ignored for a write.

Bits 3 : Nibble Select (NS). This bit is used to select either the high or low nibble of the output byte.

0 - Select high nibble
1 - Select low nibble

Bit 2 : Blue Enable (BE). This bit selects the Blue output as the data to be read. It also causes the RAMDAC to only respond to the blue color byte of the three color bytes being written or read.

0 - Don't read Blue output
1 - Read the selected nibble of the Blue output

Bit 1 : Green Enable (GE). This bit selects the Green output as the data to be read. It also causes the RAMDAC to only respond to the green color byte of the three color bytes being written or read.

0 - Don't read Green output
1 - Read the selected nibble of the Green output

Bit 0 : Red Enable (RE). This bit selects the Red output as the data to be read. It also causes the RAMDAC to only respond to the red color byte of the three color bytes being written or read.

0 - Don't read Red output
1 - Read the selected nibble of the Red output

Interrupts

The Display Subsystem generates a vertical retrace interrupt which is sent to the Host Interface Subsystem. Refer to the Host Interface Subsystem chapter for a description of how the Host Interface Subsystem generates host interrupts.

Vertical Retrace Interrupt

The vertical retrace interrupt is generated by the Display State Machine each time it blanks the screen and does a vertical retrace of the electron beam from the bottom of the screen to the top of the screen to begin the next raster sweep. This interrupt is used by the host to perform updates of the Display Subsystem components while the screen is blanked. The vertical retrace interrupt is generated before the actual start of the vertical retrace. This allows the host operating system time to transfer control to the interrupt service routine without taking up part of the vertical retrace period. The latency is shown in scan lines for the four monitor types.

scan lines	monitor type
8	60 Hz
2	30 Hz
1	RS170
1	EURO
8	STEREO

The Host Interface Subsystem uses the settings in the Interrupt Mask Register to determine which interrupts are passed on to the host. The host enables the vertical retrace interrupt by setting bit 0 in the interrupt mask register. When the interrupt occurs, bit 0 in the interrupt status register will be set to indicate that a vertical retrace is pending. The host can then use bit 7 of the Channel Zero DMA Control/Status register to determine when the vertical retrace is actually occurring.

Display Subsystem Basic Operations

The Display Subsystem is responsible for converting the frame buffer pixel data into analog RGB signals for display on the high resolution monitor. The operation of the Display Subsystem depends on the hardware configuration. The base configuration operates somewhat differently than the enhanced configuration. The Display State Machine and the Genlock Circuitry operate the same for both configurations and are described in the following paragraphs. The differences between the base and enhanced configurations are described in the sections following these paragraphs.

The Display Subsystem is organized as five pixel pipelines because of the timing considerations for a 1280 X 1024 pixel resolution. For each pixel on the screen, the Raster Subsystem maintains a total of 12 or 32 bits of pixel data which is used by the Display Subsystem to generate the colors which are displayed at each pixel location on the screen. For the base configuration the pixel data consists of 8 bits of Frame Buffer bitplane data, 2 bits of PUP bitplane data and 2 bits of Window ID bitplane data. For the enhanced configuration the pixel data consists of 24 bits of Frame Buffer bitplane data, 2 bits of PUP bitplane, 2 bits of UAUX bitplane data and 4 bits of Window ID bitplane data.

The VRAM chips which contain the bitplane data are organized in an interleaved configuration so that the pixels are grouped in five pixel groups. Each XPC1 or XMAP2 chips processes one of the five pixels to form the five pixel pipeline. Successive groups of five pixels are processed by the five XPC1 or XMAP2 chips with each XPC1 or XMAP2 processing every fifth pixel on the current scan line. The Display State Machine (DSM) generates control signals to the VRAM in the Raster Subsystem to cause the appropriate row of bits to be shifted out of the VRAM chips to the XPC1 or XMAP2 chips for the current scan line. The first row to be processed is specified in the Raster Engine TOPSCAN register.

The first pixel in the current row is shifted out to the first XPC1 or XMAP2 chip, the second pixel to the second chip and so forth until each of the five XPC1 or XMAP2 chips has processed a pixel. The sequence is then repeated across the scan line for the remaining groups of five pixels. After each pixel is processed by the appropriate XPC1 or XMAP2 chip the formatted pixel data is feed to one of the five inputs on the RGB RAMDAC or the red, green and blue RAMDACs. The pixel value is used to select an entry from a color palette or an overlay palette. The output color values from the selected palette entry are converted to analog RGB signals which are output to the high resolution monitor.

Each scan line is processed in the same manner until the number of scan lines specified in the Raster Engine TOPSCAN register have been processed. The DSM then generates the vertical retrace and repeats the processing again. This raster image generation continues as long as the DSM is running. The Display Registers can be used to put the DSM in a standby mode to cause the raster image display operations to be stopped. The sync signal which is output on the green analog line can be disabled to allow a genlock device to control the monitor synchronization. The monitor type bits in the display registers control the DSM timing operations. The pixel clock which is used to control the DSM timing can be from one of the onboard clocks or it can be from an external pixel clock provided by the genlock adapter.

The operation of the Genlock circuitry is dependent on the optional genlock adapter. The genlock circuitry is used to close a phase locked loop. The genlock adapter provides the pixel clock to drive the Display State Machine. The external pixel clock is divided by a constant and is output to the genlock adapter to provide a closed loop. This signal is used for horizontal synchronization. The genlock adapter also provides a gensync signal which is used for vertical synchronization.

Base Configuration Basic Operations

The base configuration uses five XPC1 chips and an RGB RAMDAC chip to perform the display operations. Each XPC1 chip uses 8 bits of frame buffer bitplane pixel data, 2 bits of PUP bitplane data, 2 bits of Window ID bitplane data and 2 bits of cursor data to produce an 8 bit index output XOUT and a 2 bit overlay output OVL. The XOUT and OVL outputs from the five XPC1s are used to drive the five input ports on the RGB RAMDAC. These input values are used to select an RGB color value from either the overlay palette or the color palette located inside the RGB RAMDAC. The selected 24 bit value is then converted into the analog RGB signals which drive the high resolution monitor.

The XPC1 chips contain four mode registers which are used to control the processing of the input data to produce the XOUT and OVL outputs. The two Window ID bits are used as an index to select one of the mode registers. The mode register determines whether the XPC1 is in single buffer mode or double buffer mode and whether the front or back buffer is used as the source if in double buffer mode. The mode register also controls the enabling or disabling of the overlay and underlay conditions and the use of the multimap constant bits. The mode register bits are defined in the Register section of this chapter.

When the mode register is set for single buffer mode the 8 bits of input pixel data are placed unchanged on the XOUT output. If double buffer mode is selected the buffer select bit determines which four of the input pixel bits are placed on the low nibble of XOUT. If the front buffer is selected then the low nibble of the input pixel data is placed on the low nibble of XOUT. If the back buffer is selected then the upper nibble of the input pixel data is placed on the low nibble of XOUT. The multimap enable bit determines the value of the upper nibble of XOUT in double buffer mode. If the multimap enable bit is zero then the multimap constant nibble is ignored and the upper nibble of XOUT is zero. If the multimap enable bit is one then the multimap constant nibble is output as the upper nibble of XOUT.

The OVL output is determined by the 2 bits of cursor data, the 2 bits of PUP bitplane data, the 8 bit pixel data and the overlay and underlay enable bits in the mode register. The OVL output can be either the cursor bits, the 2 PUP bits or zero. The PUP bits can represent either an overlay or an underlay condition. The highest precedence is the cursor input bits. If either of the cursor bits is a one then the OVL output is the cursor input data. If both cursor data bits are zero then the overlay condition has the next highest precedence. The overlay condition is true if either of the PUP input bits is a one and the corresponding overlay enable bit is set. If the overlay condition is true then the OVL output is set to the 2 PUP bits. If the overlay condition is not satisfied then the OVL output is either zero or the PUP bits as an underlay value. For the underlay condition to be true the cursor and overlay conditions must be false, the underlay enable bit in the mode register must be set and the input pixel value must be zero. If the pixel value is nonzero and the cursor and overlay conditions are false then the OVL output will be zero.

The XOUT and OVL outputs from each of the five XPC1 chips are connected to the A-E input ports of the RGB RAMDAC. Internally the RGB RAMDAC has a five to one multiplexer that selects one of the XOUT inputs and one of the OVL inputs. The OVL input has a higher precedence than the XOUT input. If either of the OVL input bits is nonzero then the 2 bit value is used as an index to select a 24 bit RGB color value from the overlay palette. The RGB RAMDAC is programmed so that if the OVL input bits are both zero then the XOUT input byte is used as an index to select a 24 bit RGB color value from the color palette. The selected 24 bit RGB color value from either the overlay palette or the color palette is then converted to the analog RGB signals to drive the high resolution monitor. The blanking and sync data is placed on the Green signal along with the green color data.

The overlay palette is programmed with the desired colors for the cursor, the overlays and the underlays. Only three different colors can be programmed at any one time. The color palette is programmed for a maximum of 256 different colors out of a possible 16 million colors. The values that are programmed into the color palette are dependent on whether color index mode or 8 bit RGB mode is selected. If color index mode is selected then arbitrary color values may be placed into each of the 256 entries as defined by a user application or a window manager application on the host system. If 8 bit RGB mode is selected then a special color table is loaded based on the 8 bit RGB color value. This value has the 2 most significant bits as Blue, the middle 3 bits as Green and the lower 3 bits as Red. These bits represent the most significant bits of the original three bytes of RGB color value. The color index table and the RGB table may need to be modified with gamma correction offsets before being loaded into the RGB RAMDAC.

Enhanced Configuration Basic Operations

The Display Subsystem uses five XMAP2 chips and three RAMDAC chips to perform the display operations. Each XMAP2 chip receives 24 bits of frame buffer bitplane pixel data, 2 bits of PUP bitplane data, 2 bits of UAUX bitplane data and 4 bits of Window ID bitplane data as input and produces three output bytes called ROUT, GOUT and BOUT. The ROUT bytes from the five XMAP2s are used to drive the five input ports on the Red RAMDAC. The five GOUT bytes are connected to the five input ports on the Green RAMDAC and the five BOUT bytes are connected to the five input ports on the Blue RAMDAC. The two cursor data bits are connected to the five OVL input ports on all three RAMDACS. The OVL inputs and the color byte inputs are used to select a color byte from either the overlay palette or the color palette located inside each of the three RAMDACs. The selected color bytes are then converted into the RGB analog signals used to drive the RGB monitor.

The XMAP2 chips contain sixteen mode registers which are used to control the processing of the input data to produce the three output data bytes. The four Window ID bits are used as an index to select one of the mode registers. The mode register determines the display mode and whether the front or back buffer is used as the source if in double buffer mode. The mode register also controls the enabling or disabling of the overlay and underlay conditions and the use of the multimap constant bits. The mode register bits are defined in the Register section of this chapter.

The XMAP2 chip contains 16 overlay color map registers which contain Red, Green and Blue color bytes. The WID/AUX control bit register is used to control a multiplexer which selects either the 2 PUP bits and 2 zero bits for the base configuration or 2 PUP bits and 2 UAUX bits for the enhanced configuration. For the MGR adapter the enhanced configuration setting is used. The selected 4 bits are an index into the overlay color map. The XMAP2 chip also contains the address and control lines to address an external 4K color map. The color map contains Red, Green and Blue color bytes at each of the 4K locations.

As described above, the XMAP2 chips produce three bytes of color data called ROUT, GOUT and BOUT. These three bytes can come from the 24 bit input Frame Buffer pixel data for RGB pixels, from the external 4K color map for Color Index pixels or from the overlay color map registers for overlay or underlay pixels. The selected source is determined by the mode register settings and the Frame Buffer, PUP, UAUX and WID bitplane data. The overlay condition has highest precedence for determining the source of the output data. If any of the mode register overlay enable bits and their corresponding PUP or UAUX bitplane bits are both one then the overlay condition is true and the three output bytes come from the selected overlay color map register. If the overlay condition is not true and the input Frame Buffer pixel data is zero then the underlay condition is true if the Underlay Enable bit is one. In this case the three output bytes come from the selected overlay color map register. If neither the overlay or underlay condition is true then the display mode bits in the mode register determines the output data.

The XMAP2 chip supports five different display modes. Mode 0 is an 8 bit Color Index or 8 bit RGB single buffer mode. If the multimap enable bit is set the upper nibble of the 12 bit color index is the multimap constant else it is 0. Mode 1 is a 4 bit Color Index double buffer mode. The buffer select bit selects which buffer is displayed. If the multimap enable bit is set the upper nibble of the 12 bit color index is the multimap constant else it is 0. Modes 0 and 1 are used in the 8 bitplane configuration.

Mode 2 is a 12 bit double buffer color index mode. In this mode the input Frame Buffer pixel data is used as an index into the external color map which produces the three color bytes for output. If the front buffer is selected then the low 12 bits of the Frame Buffer pixel data are used as the index into the external color map. If the back buffer is selected then the upper 12 bits of the Frame Buffer pixel are used as the index into the external color map. If the multimap enable bit is set then the upper 4 bits of the index will be the multimap constant nibble instead of the pixel bits. The low 8 bits of the index will still be from the pixel input bits.

Mode 4 is single buffer 24 bit RGB mode. In this mode the three output bytes come from the 24 bits of Frame Buffer pixel data. Mode 5 is double buffer 12 bit RGB pixel mode. In this mode the three output bytes come from the Frame Buffer pixel data. If the front buffer is selected then bits 3-0 of the Frame Buffer pixel are placed in both nibbles of ROUT, bits 11-8 of the Frame Buffer pixel are placed in both nibbles of GOUT and bits 19-16 of the Frame Buffer pixel are placed in both nibbles of BOUT. If the back buffer is selected then bits 7-4 of the Frame Buffer pixel are placed in both nibbles of ROUT, bits 15-12 of the Frame Buffer pixel are placed in both nibbles of GOUT and bits 23-20 of the Frame Buffer pixel are placed in both nibbles of BOUT. In modes 4 and 5 the multimap enable bit has no affect on the output.

The ROUT outputs from each of the five XMAP2 chips are connected to the A-E input ports of the Red RAMDAC. The GOUT outputs from each of the five XMAP2 chips are connected to the A-E input ports of the Green RAMDAC. The BOUT outputs from each of the five XMAP2 chips are connected to the A-E input ports of the Blue RAMDAC. The two cursor output bits are connected to the five OVL inputs on the three RAMDACs. Internally each RAMDAC has a five to one multiplexer that selects one of the color input bytes and one of the OVL inputs. The OVL input has a higher precedence than the color byte input. If either of the OVL input bits is nonzero then the 2 bit value is used as an index to select a color byte from the overlay palette. Each RAMDAC is programmed so that if the OVL input bits are both zero then the color input byte is used as an index to select a color byte from the color palette. The selected color byte from either the overlay palette or the color palette in the three RAMDACs are then converted to the analog RGB signals to drive the high resolution monitor. The blanking and sync data is placed on all three signals along with the color data.

The overlay palette is programmed with the desired colors for the cursor. Only three different colors can be programmed at any one time. The color palettes are programmed to contain gamma correction offsets. If no gamma correction is needed then each entry in the color palettes is programmed to the index value used to select it. This produces no change in the color value selected by the XMAP2 chips.

Programming Considerations

The following sections describe the programming considerations for the following Display Subsystem components:

- five XPC1s
- five XMAP2s
- one RGB RAMDAC
- three RAMDACs
- five Display Registers
- Display State Machine

Other paragraphs describe the Display Subsystem initialization. The example usages of the various macros do not necessarily represent an exact real world usage but are intended to show how they might be used.

XPC1 Programming Considerations

The XPC1 chips have four mode registers and an address register which is used to access the mode registers. Each mode register is accessed as two bytes. The low byte contains the display mode, the buffer select bit and the overlay enable bits. The macro XMAP_MKMODELO can be used to create the low mode byte. The high byte contains 6 valid bits for the underlay enable bit, the multimap enable bit and the multimap select nibble. The macro XMAP_MKMODEHI can be used to create the high mode byte. These two macros are the same as those for the XMAP2 chips since all the valid mode register bits in the XPC1 are the same as those in the XMAP2 mode register.

The XPC1 mode registers should only be updated during vertical retrace. Also the HQ MAR MSB register must be set to one before the XPC1 chips can be accessed by the host. The address of the desired mode byte must first be written into the address register before the mode byte can be accessed. The macro XPC_ADDRWR is provided to write the desired mode register address. This macro uses the XPC_ALL_OFF definition to cause the address to be broadcast to all five XPC1 chips. It is important that all five channels be programmed the same. Once the low byte address is set the macro XPC_MODEWR can be used to write the mode byte. The address would be incremented and the high mode byte would be written using the same two macros. The HQ MAR MSB should then be set to zero again. The following macros have been written by Silicon Graphics to access the XPC1 chips. The macro XPC_MODERD is provided to read a mode register byte. These macros are located in the file mgr.h.

```
/* XPC1 base address definitions for the five channels */

#define   XPC0_OFF          0x510       /* Channel 0 */
#define   XPC1_OFF          0x530       /* Channel 1 */
#define   XPC2_OFF          0x550       /* Channel 2 */
#define   XPC3_OFF          0x570       /* Channel 3 */
#define   XPC4_OFF          0x590       /* Channel 4 */
#define   XPC_ALL_OFF       0x5B0       /* Broadcast */

#define   XPC_NMODEREGS     4           /* mode regs */

/* XPC1 command address offsets from base address */

#define   XPC_ADDR          (0x0 << 2)
#define   XPC_MODE          (0x1 << 2)

/* macros to read and write the address and mode regs */

#define   XPC_ADDRWR(x)    \
    *(volatile long *)(GRP | XPC_ALL_OFF | XPC_ADDR) = (long)((x) & 0x7)

#define   XPC_MODEWR(x)    \
   *(volatile long *)(GRP | XPC_ALL_OFF | XPC_MODE) = (long)((x) & 0xff)

#define   XPC_MODERD((xpc_off, x)  \
   x = *(volatile long *)(GRP | (xpc_off) | XPC_MODE)  & 0xff

/* macro to form the mode register lo and hi bytes */

#define   XMAP_MKMODELO(colmode, bufsel, overlay)      \
          ((colmode) | ((bufsel) << 3) | ((overlay) <<4))
```

```
#define    XMAP_MKMODEHI(underlay, multimap, mapsel) \
           ((underlay) | ((multimap) << 1) | ((mapsel) << 2))
```

Example Usage:

```
#include "mgr.h"

unsigned char         modelo, modehi;
unsigned char         colmode = 1;          /* Double buffer */
unsigned char         bufsel = 0;           /* front buffer */
unsigned char         overlay = 0x3;        /* Enable overlay bits */
unsigned char         underlay = 1;         /* Enable underlay bit */
unsigned char         multimap = 0;         /* Disable multimap */
unsigned char         mapsel = 0x1;         /* multimap 1 */
unsigned char         addr = 0;             /* mode reg 0 */
GRPsetup;

/* determine mode register lo and hi values */

modelo = XMAP_MKMODELO(colmode, bufsel, overlay);

modehi = XMAP_MKMODEHI(underlay, multimap, mapsel);

/* At vertical retrace time, update the desired mode reg */

HQMMSB(1);              /* set HQMMSB reg */

XPC_ADDRWR(addr);       /* select mode reg lo byte */
XPC_MODEWR(modelo);     /* write lo mode byte */

++addr;                 /* mode reg high byte */

XPC_ADDRWR(addr);
XPC_MODEWR(modehi); /* write to hi mode 6 bits */

HQMMSB(0);              /* reset HQMMSB reg */

/* A mode register can be read at any time */

HQMMSB(1);              /* set HQMMSB reg */

addr = 4;               /* mode reg 2 */

XPC_ADDRWR(addr);       /* select mode reg lo byte */
XPC_MODERD(XPC2_OFF, modelo); /* read lo mode byte */

++addr;                 /* mode reg high byte */

XPC_ADDRWR(addr);       /* select mode reg lo byte */
XPC_MODERD(XPC2_OFF, modehi); /* read hi mode byte */

HQMMSB(0);              /* reset HQMMSB reg */
```

XMAP2 Programming Considerations

The XMAP2 chips have sixteen mode registers, sixteen auxiliary color map registers and a WID/AUX control register. There is also an address register which is used to access the other registers and the external 4K color map. Each mode register is accessed as two bytes. The low byte contains the display mode, the buffer select bit and the overlay enable bits. The macro XMAP_MKMODELO can be used to create the low mode byte. The high byte contains 6 bits which are for the underlay enable bit, the multimap enable bit and the multimap select nibble. The macro XMAP_MKMODEHI can be used to create the high mode byte.

The XMAP2 registers should only be updated during vertical retrace. Also the HQ MAR MSB register must be set to one before the XMAP2 chips can be accessed by the host. The HQ MAR MSB should then be set to zero again after the XMAP register or color map have been accessed. The address of the desired register or color map entry must first be written into the address register before it can be accessed. The macros XMAP_ALOWR and XMAP_AHIWR are provided to write the desired address. These macros use the XMAPALL_OFF definition to cause the address to be broadcast to all five XMAP2 chips. It is important that all five channels be programmed the same.

The mode registers are accessed as a low and a high byte. Once the low byte address is set the macro XMAP_OTHERWR can be used to write the low mode byte. The address would be incremented and the high mode byte would be written using the same macros. The macro XMAP_INCA is provided to increment the address. The macro XMAP_OTHERRD is provided to read a mode register byte.

The WID/AUX control bit register should be set to zero for the enhanced configuration. The XMAP_ALOWR and XMAP_AHIWR macros are used to write the address 0x20 into the address register. The macro XMAP_OTHERWR is used to write the WID/AUX control bit register.

The overlay color map registers and the external color map entries are programmed with the desired red, green and blue color bytes. The desired address of the register or color map entry is written into the address register using the XMAP_ALOWR and XMAP_AHIWR macros. Then the XMAP_COLWR macro is used to write each of the color bytes. The macro XMAP_COLRD can be used to read from the currently addressed register or color map entry.

The following macros have been written by Silicon Graphics to access the XMAP2 chips. These macros are located in the file mgr.h.

```
/* XMAP2 base address definitions for the five channels */

#define  XMAP0_OFF     0x400    /* Channel 0 */
#define  XMAP1_OFF     0x420    /* Channel 1 */
#define  XMAP2_OFF     0x440    /* Channel 2 */
#define  XMAP3_OFF     0x460    /* Channel 3 */
#define  XMAP4_OFF     0x480    /* Channel 4 */
#define  XMAPALL_OFF   0x4A0    /* Broadcast */

/* XMAP2 command macros */

#define  XMAP_NOOP     (0x0 << 2)   /* no operation */
#define  XMAP_BLUE     (0x1 << 2)   /* R/W blue */
#define  XMAP_GREEN    (0x2 << 2)   /* R/W green bytes */
#define  XMAP_RED      (0x3 << 2)   /* R/W red bytes */
#define  XMAP_AINC     (0x4 << 2)   /* increment addr */
#define  XMAP_OTHER    (0x5 << 2)   /* R/W other regs */
```

```
#define   XMAP_AHI          (0x6 << 2)   /* W addr hi byte */
#define   XMAP_ALO          (0x7 << 2)   /* W addr lo byte */

/* XMAP2 display mode definitions */

#define   XMAP_8CI          0x0   /* 8 bit sb color index */
#define   XMAP_8DCI         0x1   /* 4 bit db color index */
#define   XMAP_24DCI        0x2   /* 12 bit db color index */
#define   XMAP_24RGB        0x4   /* 24 bit sb RGB */
#define   XMAP_24DRGB       0x5   /* 12 bit db RGB */

/* other miscellaneous definitions */

#define   XMAP_BUFSEL              0x08
#define   XMAP_MAPSEL              0x3C

#define   XMAP_MODEREG_OFF         0
#define   XMAP_WIDAUX_OFF          0
#define   XMAP_NXMAP               5      /* number XMAP2s */
#define   XMAP_NCOLMAPENT          4096   /* col map entries */
#define   XMAP_NAUXMAPENT          16     /* aux map entries */
#define   XMAP_NMULTIMAP           16     /* multimaps */
#define   XMAP_NMODEREGS           16     /* mode regs */

/* macros to read and write the address register, the mode register, WID/AUX control register
and the RGB color bytes in the overlay color map registers and the external color map */

#define   XMAP_COLSEL        0x1000 /* color map base addr */

/* In the following macros, xmap_off refers to the base address of desired XMAP2 chip and
xmap_col refers to the red, green or blue command for the color to be written. */

#define   XMAP_COLRD(xmap_off, xmap_col, x) \
 x = *(volatile long *)(GRP |  (xmap_off) | (xmap_col)) & 0xff

#define   XMAP_COLWR(xmap_col, x) \
   *(volatile long *)(GRP |  XMAPALL_OFF | (xmap_col)) = (long)((x) & 0xff)

#define   XMAP_OTHERRD(xmap_off, x)       \
x = *(volatile long *)(GRP | (xmap_off) | XMAP_OTHER) &0xff

#define   XMAP_OTHERWR(x) \
  *(volatile long *)(GRP | XMAPALL_OFF | XMAP_OTHER) = (long)((x) & 0xff)

#define   XMAP_INCA()         \
   *(volatile long *)(GRP | XMAPALL_OFF | XMAP_AINC)  = 0

#define   XMAP_ALOWR(x)     \
   *(volatile long *)(GRP | XMAPALL_OFF | XMAP_ALO) = (long)((x) & 0xff)

#define   XMAP_AHIWR(x)     \
   *(volatile long *)(GRP | XMAPALL_OFF | XMAP_AHI) = (long)((x) >> 8)

/* macros to form the mode register lo and hi bytes */
```

```
#define    XMAP_MKMODELO(colmode, bufsel, overlay)        \
           ((colmode) | ((bufsel) << 3) | ((overlay) <<4))

#define    XMAP_MKMODEHI(underlay, multimap, mapsel) \
           ((underlay) | ((multimap) << 1) | ((mapsel) << 2))
```

Example Usage:

```
#include "mgr.h"

unsigned char          modelo, modehi;
unsigned char          colmode = 1;  /* Double buffer */
unsigned char          bufsel = 0;             /* front buffer */
unsigned char          overlay = 0xF; /* Enable overlay bits */
unsigned char          underlay = 1;  /* Enable underlay bit */
unsigned char          multimap = 0; /* Disable multimap */
unsigned char          mapsel = 0x1; /* multimap 1 */
unsigned char          addr = 0;       /* mode reg 0 */
unsigned char          red = 0, green = 0, blue = 0; /* Black */
GRPsetup;

/* determine mode register lo and hi values */

modelo = XMAP_MKMODELO(colmode, bufsel, overlay);

modehi = XMAP_MKMODEHI(underlay, multimap, mapsel);

/* At vertical retrace time, update the desired mode reg */

HQMMSB(1);              /* set HQMMSB reg */

XMAP_ALOWR(addr);    /* select mode reg lo byte */

XMAP_AHIWR(addr);    /* select mode reg lo byte */

XMAP_OTHERWR(modelo);        /* write lo mode byte */

XMAP_INCA;  /* increment address to mode high byte */

XMAP_OTHERWR(modehi);        /* write to hi mode 6 bits */

HQMMSB(0);              /* reset HQMMSB reg */

/* A mode register can be read at any time */

HQMMSB(1);              /* set HQMMSB reg */

addr = 4;          /* mode reg 2 */

XMAP_ALOWR(addr); /* select mode reg lo byte */

XMAP_AHIWR(addr); /* select mode reg lo byte */
```

```
XMAP_OTHERRD(XPC2_OFF, modelo); /* read lo byte */

XMAP_INCA;  /* increment address to mode high byte */

XMAP_OTHERRD(XPC2_OFF, modehi); /* read hi 6 bits */

HQMMSB(0);              /* reset HQMMSB reg */

/* During initialization, at vertical retrace, set the WID/AUX control bit to zero for the
enhanced configuration */

HQMMSB(1);              /* set HQMMSB reg */

XMAP_ALOWR(XMAP_WIDAUX_OFF); /* WID/AUX reg */

XMAP_AHIWR(XMAP_WIDAUX_OFF); /* WID/AUX reg */

XMAP_OTHERWR(0); /* set for AUX input selection */

HQMMSB(0);              /* reset HQMMSB reg */

/* at vertical retrace set the color map or aux color map */

HQMMSB(1);              /* set HQMMSB reg */

XMAP_ALOWR(XMAP_COLSEL); /* color map entry 0 */

XMAP_AHIWR(XMAP_COLSEL); /* color map entry 0 */

XMAP_COLWR(XMAP_RED, red);  /* set red byte */
XMAP_COLWR(XMAP_GREEN, green); /* set green byte */
XMAP_COLWR(XMAP_BLUE, blue);  /* set blue byte */

HQMMSB(0);              /* reset HQMMSB reg */
```

RGB RAMDAC Programming Considerations

The RGB RAMDAC contains a 256 entry RGB color palette and a 4 entry overlay color palette. It also contains a command register, a read mask register, a blink mask register and a test register. The following macros are provided in mgr.h to allow these registers and color palettes to be accessed.

```
/* RGB RAMDAC base address definition */

#define  DACRGB_OFF 0x5D0

/* address offset definitions */

#define  DAC_AREG           (0x0 << 2)  /* addr reg */
#define  DAC_COL            (0x1 << 2)  /* color pal */
#define  DAC_CR             (0x2 << 2)  /* cntl reg */
#define  DAC_OVRLAY         (0x3 << 2)  /* overlay pal */

/* Miscellaneous DAC definitions */

#define  DAC_CMDMASK        (0x3 << 2)  /* cmd mask */
#define  DAC_READMASK       4           /* read mask */
#define  DAC_BLINKMASK      5           /* blink mask */
#define  DAC_CMD            6           /* cmd reg */
#define  DAC_TEST           7           /* test reg */
#define  DAC_NGAMMAENT      256  /* gamma entries */

/* DAC command register bit settings */

#define  DAC_5TO1MUX        0x80 /* 5:1 multiplexing */
#define  DAC_RAMENBL        0x40 /* enable color pal */
#define  DAC_BLINK0         0x00 /* blink rate 0 */
#define  DAC_BLINK1         0x10 /* blink rate 1 */
#define  DAC_BLINK2         0x20 /* blink rate 2 */
#define  DAC_BLINK3         0x30 /* blink rate 3 */
#define  DAC_OL1BENBL       0x08 /* OL1 blink enable */
#define  DAC_OL0BENBL       0x04 /* OL0 blink enable */
#define  DAC_OL1ENBL        0x02 /* OL1 enable */
#define  DAC_OL0ENBL        0x01 /* OL0 enable */

/* RGB DAC macro definitions */

#define  DAC_ARD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_AREG) & 0xff

#define  DAC_AWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_AREG) = (long)((x) & 0xff)

#define  DAC_COLRD(col_off, x) \
   x = *(volatile long *)(GRP | (col_off) | DAC_COL) & 0xff

#define  DAC_COLWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_COL) = ((x) & 0xff)
```

```
#define   DAC_CRRD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_CR) & 0xff

#define   DAC_CRWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_CR) = (long)((x) & 0xff)

#define   DAC_OVERLAYRD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_OVERLAY) & 0xff

#define   DAC_OVERLAYWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_OVERLAY) = (long)((x) & 0xff)
```

Example Usage:

```
/* Initialize the read mask to all ones by writing the readmask address into the address register
and then writing the value 0xff to the Control Register address */

DAC_AWR(DACRGB_OFF, DAC_READMASK);
DAC_CRWR(DACRGB_OFF, 0xff);

/* Initialize the blink mask to all zeroes by writing the blinkmask address into the address
register and then writing the value 0x00 to the Control Register address */

DAC_AWR(DACRGB_OFF, DAC_BLINKMASK);
DAC_CRWR(DACRGB_OFF, 0x00);

/* Initialize the command register for 5 to 1 multiplexing, for transparent mode and to enable
the two overlay input bits by writing the command register address into the address register
and then writing the appropriate command value to the Control Register address */

DAC_AWR(DACRGB_OFF, DAC_CMD);
DAC_CRWR(DACRGB_OFF, DAC_5TO1MUX | DAC_RAMENBL | DAC_OL0ENBL | DAC_OL1ENBL);

/* Initialize the test register to all zeroes by writing the test register address into the address
register and then writing the value 0x00 to the Control Register address */

DAC_AWR(DACRGB_OFF, DAC_TEST);
DAC_CRWR(DACRGB_OFF, 0x00);

/* Initialize the overlay palette entry 1 for a red cursor color. */

DAC_AWR(DACRGB_OFF, 1);
DAC_OVRLAYWR(DACRGB_OFF, 0xff);
DAC_OVRLAYWR(DACRGB_OFF, 0x00);
DAC_OVRLAYWR(DACRGB_OFF, 0x00);

/* Initialize the color palette entry 7 for a white color. */

DAC_AWR(DACRGB_OFF, 7);
DAC_COLWR(DACRGB_OFF, 0xff);
DAC_COLWR(DACRGB_OFF, 0xff);
DAC_COLWR(DACRGB_OFF, 0xff);
```

RAMDAC Programming Considerations

Each of the three RAMDACs contain a 256 entry color palette and a 4 entry overlay color palette. They also contain a command register, a read mask register, a blink mask register and a test register. The only programming differences between the RGB RAMDAC and the three RAMDACs are that the three RAMDACs are addressed individually. When programming a color the red color is written to the DACR_OFF address, the green color is written to the DACG_OFF address and the blue color is written to the DACB_OFF address. The following macros are provided in mgr.h to allow these registers and color palettes to be accessed.

```
/* RAMDAC base address definitions */

#define   DACR_OFF     0x500
#define   DACG_OFF     0x520
#define   DACB_OFF     0x540

/* address offset definitions */

#define   DAC_AREG                  (0x0 << 2)   /* addr reg */
#define   DAC_COL                   (0x1 << 2)   /* color pal */
#define   DAC_CR                    (0x2 << 2)   /* cntl reg */
#define   DAC_OVRLAY                (0x3 << 2)   /* overlay pal */

/* Miscellaneous DAC definitions */

#define   DAC_CMDMASK               (0x3 << 2)   /* cmd mask */
#define   DAC_READMASK              4            /* read mask */
#define   DAC_BLINKMASK             5            /* blink mask */
#define   DAC_CMD                   6            /* cmd reg */
#define   DAC_TEST                  7            /* test reg */
#define   DAC_NGAMMAENT             256   /* gamma entries */

/* DAC command register bit settings */

#define   DAC_5TO1MUX               0x80  /* 5:1 multiplexing */
#define   DAC_RAMENBL               0x40  /* enable color pal */
#define   DAC_BLINK0                0x00  /* blink rate 0 */
#define   DAC_BLINK1                0x10  /* blink rate 1 */
#define   DAC_BLINK2                0x20  /* blink rate 2 */
#define   DAC_BLINK3                0x30  /* blink rate 3 */
#define   DAC_OL1BENBL              0x08  /* OL1 blink enable */
#define   DAC_OL0BENBL              0x04  /* OL0 blink enable */
#define   DAC_OL1ENBL               0x02  /* OL1 enable */
#define   DAC_OL0ENBL               0x01  /* OL0 enable */

/* RGB DAC macro definitions */

#define   DAC_ARD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_AREG) & 0xff

#define   DAC_AWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_AREG) = (long)((x) & 0xff)
```

```
#define    DAC_COLRD(col_off, x) \
   x = *(volatile long *)(GRP | (col_off) | DAC_COL) & 0xff

#define    DAC_COLWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_COL) = ((x) & 0xff)

#define    DAC_CRRD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_CR) & 0xff

#define    DAC_CRWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_CR) = (long)((x) & 0xff)

#define    DAC_OVERLAYRD(col_off, x) \
   x = *(volatile long *)(GRP | col_off | DAC_OVERLAY) & 0xff

#define    DAC_OVERLAYWR(col_off, x) \
   *(volatile long *)(GRP | (col_off) | DAC_OVERLAY) = (long)((x) & 0xff)
```

Example Usage:

```
/* Initialize the read mask in each RAMDAC to all ones by writing the readmask address into the
address register and then writing the value 0xff to the Control Register address in each of the
three RAMDACs*/

DAC_AWR(DACR_OFF, DAC_READMASK);
DAC_CRWR(DACR_OFF, 0xff);

DAC_AWR(DACG_OFF, DAC_READMASK);
DAC_CRWR(DACG_OFF, 0xff);

DAC_AWR(DACB_OFF, DAC_READMASK);
DAC_CRWR(DACB_OFF, 0xff);

/* Initialize the blink mask in the three RAMDACs to all zeroes by writing the blinkmask
address into the address register and then writing the value 0x00 to the Control Register
address in each of the three RAMDACs*/

DAC_AWR(DACR_OFF, DAC_BLINKMASK);
DAC_CRWR(DACR_OFF, 0x00);

DAC_AWR(DACG_OFF, DAC_BLINKMASK);
DAC_CRWR(DACG_OFF, 0x00);

DAC_AWR(DACB_OFF, DAC_BLINKMASK);
DAC_CRWR(DACB_OFF, 0x00);

/* Initialize the command register in each of the three RAMDACs for 5 to 1 multiplexing, for
transparent mode and to enable the two overlay input bits by writing the command register
address into the address register and then writing the appropriate command value to the Control
Register address in each of the RAMDACs*/

DAC_AWR(DACR_OFF, DAC_CMD);
DAC_CRWR(DACR_OFF, DAC_5TO1MUX | DAC_RAMENBL | DAC_OL0ENBL | DAC_OL1ENBL);
```

```
DAC_AWR(DACG_OFF, DAC_CMD);
DAC_CRWR(DACG_OFF, DAC_5TO1MUX | DAC_RAMENBL | DAC_OL0ENBL | DAC_OL1ENBL);

DAC_AWR(DACB_OFF, DAC_CMD);
DAC_CRWR(DACB_OFF, DAC_5TO1MUX | DAC_RAMENBL | DAC_OL0ENBL | DAC_OL1ENBL);

/* Initialize the test register in the three RAMDACs to all zeroes by writing the test register
address into the address register and then writing the value 0x00 to the Control Register
address in each RAMDAC */

DAC_AWR(DACR_OFF, DAC_TEST);
DAC_CRWR(DACR_OFF, 0x00);

DAC_AWR(DACG_OFF, DAC_TEST);
DAC_CRWR(DACG_OFF, 0x00);

DAC_AWR(DACB_OFF, DAC_TEST);
DAC_CRWR(DACB_OFF, 0x00);

/* Initialize the overlay palette entry 1 for a red cursor color. */

DAC_AWR(DACR_OFF, 1);
DAC_OVRLAYWR(DACR_OFF, 0xff);

DAC_AWR(DACG_OFF, 1);
DAC_OVRLAYWR(DACG_OFF, 0x00);

DAC_AWR(DACB_OFF, 1);
DAC_OVRLAYWR(DACB_OFF, 0x00);

/* Initialize the color palette entry 7 for a white color. */

DAC_AWR(DACR_OFF, 7);
DAC_COLWR(DACR_OFF, 0xff);

DAC_AWR(DACG_OFF, 7);
DAC_COLWR(DACG_OFF, 0xff);

DAC_AWR(DACB_OFF, 7);
DAC_COLWR(DACB_OFF, 0xff);
```

Display Register Programming Considerations

There are five display registers on the MGR board. The five registers control various Display Subsystem timing and monitor selection. They also provide status bits to the host for determining the hardware configuration. The following macros are provided in mgr.h to allow access to the five display registers. The register definitions are provided in the Registers section of this chapter. When reading a register the bits that are designated as don't care or write only can be either a logical one or logical zero and should not be used.

```
/* Display Register Address Definitions */

#define   DREG0_OFF   0x4E0
#define   DREG1_OFF   0x4C0
#define   DREG2_OFF   0x5E0
#define   DREG3_OFF   0x5C0
#define   DREG4_OFF   0x5A0

/* Display Register defines and operations */

#define   NDREG              5     /* number of display regs */

          /* DREG 0 defines */

#define   DREG_ZBUF0 0x08 /* Z buffer installed bit */

          /* DREG 1 defines */

#define   DREG_SYNCGRN      0x01  /* Sync enable bit */

          /* DREG 2 defines */

#define   DREG_SCREENON     0x01  /* bit to turn on screen */
#define   DREG_BITPLANES 0x08 /* bitplanes installed bit */

          /* DREG 3 defines */

#define   DREG_LCARESET     0x02  /* bit to reset LCA */
#define   DREG_MONITORRESET  0x04 /* reset mon type */
#define   DREG_FIFOEMPTY    0x08  /* fifo empty bit */

          /* DREG 4 defines */

#define   DREG_MONITORMASK       0x03  /* mon type mask */
#define   DREG_EXTCLKSEL         0x04  /* ext clock enable */
#define   DREG_ACLKEN            0x20  /* async clk enable */
#define   DREG_SCLKEN            0x40  /* sync clk enable */

          /* Monitor type defines */

#define   DREG_MONITOR60HZ       0x00  /* 60Hz  1280x1024  */
#define   DREG_MONITOR30HZ       0x01  /* 30Hz  1280x1024  */
#define   DREG_MONITORNTSC       0x02  /* NTSC  645x485 */
#define   DREG_MONITORPAL        0x03  /* PAL  780x575 */
```

```
#define   DREG_STEREO              0x04 /* 120 Hz monitor */

/* read/write macro defines */

#define   DREG_RD_XPC(dreg_off, x) \
    x = *(volatile long *)(GRP | (dreg_off)) & 0xFF

#define   DREG_WR(dreg_off, x)      \
    *(volatile long *)(GRP | (dreg_off)) = (long)((x) & 0xff)
```

Example Usage:

```
#include "mgr.h"

int       b, hwconfig;

HQMMSB_WR(1);        /* set HQ MAR MSB register */

/* Get the Zbuffer installed status bit and put it in hwconfig bit 4 */

DREG_RD_XPC(DREG0_OFF, b);
hwconfig = (b & DREG_ZBUF0) << 1;

/* Get the Bitplanes installed status bit and put it in hwconfig bit 3 */

DREG_RD_XPC(DREG2_OFF, b);
hwconfig |= (b & DREG_BITPLANES);

HQMMSB_WR(0);        /* reset HQ MAR MSB register */

/* Turn on the screen */

HQMMSB_WR(1);        /* set HQ MAR MSB register */

DREG_WR(DREG2_OFF, DREG_SCREENON);

HQMMSB_WR(0);        /* reset HQ MAR MSB register */
```

Display State Machine Programming Considerations

The Display State Machine (DSM) is controlled by the following bits in the five display registers:

- LCA Reset bit (display register 3 - bit 1)
- External Clock Enable bit (display register 4 - bit 2)
- Asynchronous Clock Enable bit (display register 4 - bit 5)
- Synchronous Clock Enable bit (display register 4 - bit 6)
- Monitor Set bit (display register 3 - bit 2)
- Monitor Type bits (display register 4 - bits 1 and 0)
- Sync Output Enable bit (display register 1 - bit 0)
- Standby bit (display register 2 - bit 0)

The DSM is a programmable Xilinx logic chip which must be programmed from a PROM after the MGR adapter has been reset. The LCA Reset bit is set to a logical one by a hardware reset condition. It must be reset to a logical zero to cause the Xilinx to be reprogrammed. This bit would normally not need to be programmed other than after a reset of the MGR adapter.

The DSM clocking is controlled by the external clock enable bit, the asynchronous clock enable bit and the synchronous clock enable bit. The External Clock Enable bit must be reset to logical zero to select the on board clock for the DSM. This bit would only be set to logical one for an external clock when a genlock adapter is providing the clock signal. The Asynchronous Clock Enable bit must be set to logical one before the Synchronous Clock Enable bit is set. After the Asynchronous clock Enable bit has been set then the Synchronous Clock Enable bit must be set to logical one.

Four different monitor types are supported by the Display State Machine. These are a non-interlaced 1280 x 1024 high resolution analog RGB monitor, an interlaced 1280 x 1024 high resolution analog RGB monitor, an NTSC compatible monitor and a PAL compatible monitor. The monitor type is selected by writing the monitor type to the Monitor Type bits. The monitor type is then made active by toggling the Monitor Set bit. This bit must be reset to a logical zero and after a delay set to logical one to cause the new monitor timing to become active.

The Standby bit must be set to a logical one for the Display State Machine to be activated. If this bit is logical zero then the screen will be blanked. The Sync Output Enable bit must be set to logical zero for the sync output to be enabled. This is the reset condition so no special programming is necessary to enable sync output. If this bit is set to a logical one then sync output is disabled. In the base configuration the sync signal will only be on the green output when the sync output is enabled. On the enhanced configuration the sync signal will be placed on all three outputs.

Display Subsystem Initialization Programming Considerations

After the MGR adapter has been reset the display subsystem must be initialized to allow the proper display of the pixel data from the bitplanes in the Raster Subsystem.

Example Code :

```
#include "mgr.h"

/* 8 initial colors */

unsigned char     initcolor[8][3]  = {
    0x00, 0x00, 0x00,          /* BLACK */
    0xFF, 0x00, 0x00,          /* RED */
    0x00, 0xFF, 0x00,          /* GREEN */
    0xFF, 0xFF, 0x00,          /* YELLOW */
    0x00, 0x00, 0xFF,          /* BLUE */
    0xFF, 0x00, 0xFF,          /* MAGENTA */
    0x00, 0xFF, 0xFF,          /* CYAN */
    0xFF, 0xFF, 0xFF,          /* WHITE */

unsigned char     initauxcolor[4][3]  = {
    0x00, 0x00, 0x00           /* Transparent entry */
    0xFF, 0x00, 0x00,          /* RED */
    0x00, 0x00, 0x00,          /* BLACK */
    0xFF, 0xFF, 0xFF,          /* WHITE */

long      dacaddrs[] = { DACR_OFF, DACG_OFF, DACB_OFF };

int       b, hwconfig;
char      monitor = 0;         /* for example select 60 Hz non-interlaced monitor */

/* Determine the hardware configuration */

HQMMSB_WR(1);                        /* must set HQMMSB to access display registers */

/* Get the Zbuffer installed status bit and put it in hwconfig bit 4.  If Zbuffer card is installed
then DREG_ZBUF0 will be zero. */

DREG_RD_XPC(DREG0_OFF, b);
hwconfig = (b & DREG_ZBUF0) << 1;

/* Get the Bitplanes installed status bit and put it in hwconfig bit 3.  If enhanced bitplanes card
is installed then DREG_ZBUF0 will be zero. */

DREG_RD_XPC(DREG2_OFF, b);
hwconfig |= (b & DREG_BITPLANES);

/* Initialize the display registers */

HQMMSB(1);                      /* must set HQMMSB to access display registers */

/* set monitor type */
```

```
DREG_WR(DREG4_OFF, monitor);      /* first select the monitor type low 2 bits */
DREG_WR(DREG1_OFF, (monitor & 0x4) >> 1);      /* set mon type upper bit */

/* enable asynchronous clock and then the synchronous clock */

DREG_WR(DREG4_OFF, monitor | DREG_ACLKEN);     /* Enable async clock */
DREG_WR(DREG4_OFF, monitor | DREG_ACLKEN | DREG_SCLKEN); /* Enable sync clock */

/* Load the current monitor timing selected by the monitor type */

DREG_WR(DREG3_OFF, 0);
DREG_WR(DREG3_OFF, DREG_MONITORRESET);

/* Turn on the screen by taking DSM out of standby mode */

DREG_WR(DREG2_OFF, DREG_SCREENON);

/* Initialize the XPC1 or XMAP2 chips */

/* Initialize all mode register for 8 bit CI */

if (hwconfig & DREG_BITPLANES) {     /* base configuration */
    for (i = 0 ; i < 8 ; ) {
        XPC_ADDRWR(i++);     /* select mode reg lo byte */
        XPC_MODERD(0); /* read lo mode byte */
        XPC_ADDRWR(i++);     /* select mode reg lo byte */
        XPC_MODERD(0); /* read hi mode byte */
    }
} else {     /* enhanced configuration */
    XMAP_AHIWR(0);                       /* set upper 5 bits of address */
    for (i = 0 ; i < XMAP_WIDAUX_OFF ; i += 2) {
        XMAP_ALOWR(i);          /* select mode reg lo byte */
        XMAP_OTHERWR(0);        /* write lo mode byte */
        XMAP_INCA;              /* increment address to mode high byte */
        XMAP_OTHERWR(0);        /* write to hi mode 6 bits */
    }

    /* set WID/AUX control bit to 0 to select 2 PUP bits and 2 UAUX bits for enhanced
    configuration and 1 for 2 PUP and 2 zero bits for the base configuration */

    XMAP_ALOWR(XMAP_WIDAUX_OFF);        /* set WID/AUX reg lo byte */
    XMAP_AHIWR(XMAP_WIDAUX_OFF);        /* set upper 5 bits of address */
    XMAP_OTHERWR(0);                    /* write   WID/AUX bit */

    /* setup first 8 entries in the external color map */

    XMAP_AHIWR(XMAP_COLSEL);

    for (i = 0 ; i < 8 ; ++i) {
        XMAP_ALOWR(i | XMAP_COLSEL);
        XMAP_COLWR(XMAP_RED, initcolor[i][0]);
        XMAP_COLWR(XMAP_GREEN, initcolor[i][1]);
        XMAP_COLWR(XMAP_BLUE, initcolor[i][2]);
```

```
}

/* setup rest of entries in the external color map to BLACK */

for (i = 8 ; i < XMAP_NCOLMAPENT ; ++i) {
    XMAP_ALOWR(i | XMAP_COLSEL);
    XMAP_AHIWR(i | XMAP_COLSEL);
    XMAP_COLWR(XMAP_RED, initcolor[0][0]);
    XMAP_COLWR(XMAP_GREEN, initcolor[0][1]);
    XMAP_COLWR(XMAP_BLUE, initcolor[0][2]);
}

/* setup first 3 entries in the overlay color map beginning with entry 1*/

XMAP_AHIWR(0);

for (i = 1 ; i < 4 ; ++i) {
    XMAP_ALOWR(i);
    XMAP_COLWR(XMAP_RED, initauxcolor[i][0]);
    XMAP_COLWR(XMAP_GREEN, initauxcolor[i][1]);
    XMAP_COLWR(XMAP_BLUE, initauxcolor[i][2]);
}

/* setup rest of entries in the overlay color map to BLACK*/

XMAP_AHIWR(0);

for (i = 4 ; i < 16; ++i) {
    XMAP_ALOWR(i);
    XMAP_COLWR(XMAP_RED, initauxcolor[2][0]);
    XMAP_COLWR(XMAP_GREEN, initauxcolor[2][1]);
    XMAP_COLWR(XMAP_BLUE, initauxcolor[2][2]);
}

/* initialize the RGB RAMDAC or the three RAMDACs */

if (hwconfig & DREG_BITPLANES) {      /* base configuration */
    DAC_AWR(DACRGB_OFF, DAC_READMASK);       /* select read mask reg */
    DAC_CRWR(DACRGB_OFF, 0xFF);              /* set read mask to all one's */

    DAC_AWR(DACRGB_OFF, DAC_BLINKMASK);      /* select blink mask reg */
    DAC_CRWR(DACRGB_OFF, 0x00);              /* disable blinking */

    DAC_AWR(DACRGB_OFF, DAC_CMD);            /* select command reg */

    /* set command reg for 5 inputs, transparency mode for color palette enable, overlay 0
    palette index bit enabled, overlay palette index bit 1 disabled */

    DAC_CRWR(DACRGB_OFF, DAC_5TO1MUX | DAC_RAMENBL |
                         DAC_OL0ENBL | DAC_OL1ENBL);

    DAC_AWR(DACRGB_OFF, DAC_DACTEST);  /* select test reg */
    DAC_CRWR(DACRGB_OFF, 0x00);        /* disable test mode */
```