Intel® Open Source HD Graphics, Intel Iris™ Graphics, and Intel Iris™ Pro Graphics

Programmer's Reference Manual

For the 2015 - 2016 Intel Core™ Processors, Celeron™ Processors, and Pentium™ Processors based on the "Skylake" Platform

Volume 12: Display

May 2016, Revision 1.0

Creative Commons License

Notices and Disclaimers

Table of Contents

VGA and Extended VGA Registers

This section describes the registers and the functional operation notations for the observable registers in the VGA section. This functionality is provided as a means for support of legacy applications and operating systems. It is important to note that these registers in general have the desired effects only when running VGA display modes.

The main exceptions to this are the palette interface which allows real mode DOS applications and full screen VGA applications under an OS control running in high resolution (non-VGA) modes to access the palette through the VGA register mechanisms and the use of the ST01 status bits that determine when the VGA enters display enable and sync periods. Other exceptions include the register bits that control the memory accesses through the A000:0000 and B000:0000 memory segments which are used during operating system emulation of VGA for "DOS box" applications. Some of the functions of the VGA are enabled or defeated through the programming of the VGA control register bits that are located in the MMIO register space.

Given the legacy nature of this function, it has been adapted to the changing environment that it must operate within. The three most notable changes are the addition of high resolution display mode support, new operating system support, and the use of fixed resolution display devices (such as LCD panels). Additional control bits in the PCI Config space will affect the ability to access the registers and memory aperture associated with VGA.

Mode of Operation	VGA Disable	VGA Display	VGA Registers	Palette (VGA)	VGA Memory	VGA Banking
VGA DOS	No	Yes	Yes	Yes	Yes	No
HiRes DOS	Yes	No	Yes	Yes	No	Yes
Fullscreen DOS	Yes/No	No/Yes	Yes	Yes	Yes	Yes
DOS Emulation	Yes	No	Yes	Yes	Yes	Yes

VGA Display Mode	Dot Clock Select	Dot Clock Range	132 Column Text Support	9-Dot Disable Support	Main Use
Native	VGA Clock Select	25/28 MHz	No	No	Analog CRT (VGA connector)
Centered	Fixed at display Requirements	Product Specific	No	Yes	Digital Display
Upper Left Corner	Fixed at display Requirements	Product Specific	No	Yes	Internal Panel

Native, Centered, and Upper Left Corner support varies from product to product.

Even in the native VGA display operational modes, not all combinations of bit settings result in functional operating modes. VGA display modes have the restriction that they can be used only when all other display planes are disabled.

These registers are accessed via I/O space. The I/O space resides in the PCI compatibility hole and uses only the addresses that were part of the original VGA I/O space (which includes EGA and MDA emulation). Accesses to the VGA I/O addresses are steered to the proper bus and rely on proper setup of bridge registers. Extended VGA registers such as GR10 and GR11 use additional indexes for the already defined I/O addresses. VGA register accesses are allowed as 8 or 16 bit naturally aligned transactions only. Word transactions must have the least significant bit of the address set to zero. DWORD I/O operations should not be performed on these registers.

Some products may support access to these registers through MMIO. The access method varies and is documented elsewhere.

General Control and Status Registers

The setup, enable, and general registers are all directly accessible by the CPU. A sub indexing scheme is not used to read from and write to these registers.

Name	Function	Read		Write	
		I/O	Memory Offset	I/O	Memory Offset
ST00	VGA Input Status Register 0	3C2h	3C2h	--	--
ST01	VGA Input Status Register 1	3BAh/3DAh[1]	3BAh/3DAh[1]	--	--
FCR	VGA Feature Control Register	3CAh	3CAh	3BAh/3DAh[1]	3BAh/3DAh[1]
MSR	VGA Miscellaneous Output Register	3CCh	3CCh	3C2h	3C2h

[1] The address selection for ST01 reads and FCR writes is dependent on CGA or MDA emulation mode as selected via the MSR register.

Various bits in these registers provide control over the real-time status of the horizontal sync signal, the horizontal retrace interval, the vertical sync signal, and the vertical retrace interval. The horizontal retrace interval is the period during the drawing of each scan line containing active video data, when the active video data is not being displayed. This period includes the horizontal front and back porches, and the horizontal sync pulse. The horizontal retrace interval is always longer than the horizontal sync pulse. The vertical retrace interval is the period during which the scan lines not containing active video data are drawn. This includes the vertical front porch, back porch, and the vertical sync pulse. The vertical retrace interval is normally longer than the vertical sync pulse.

ST00 - Input Status 0

I/O (and Memory Offset) Address: 3C2h

Default: 00h

Attributes: Read Only

Bit	Descriptions
7	**CRT Interrupt Pending.** This bit is here for EGA compatibility and **will always return zero**. The generation of interrupts was originally enabled, through bits [4,5] of the Vertical Retrace End Register (CR11). This ability to generate interrupts at the start of the vertical retrace interval is a feature that is typically unused by DOS software and therefore is only supported through other means for use under a operating system support. 0 = CRT (vertical retrace interval) interrupt is not pending. 1 = CRT (vertical retrace interval) interrupt is pending
6:5	**Reserved.** Read as 0s.
4	**RGB Comparator / Sense.** This bit is here for compatibility and **will always return one**. Monitor detection must be done be done through the programming of registers in the MMIO space. 0 = Below threshold 1 = Above threshold
3:0	**Reserved.** Read as 0s.

ST01 - Input Status 1

I/O (and Memory Offset) Address: 3BAh/3DAh

Default: 00h

Attributes: Read Only

The address selection is dependent on CGA or MDA emulation mode as selected via the MSR register.

Bit	Descriptions
7	**Reserved (as per VGA specification).** Read as 0s.
6	**Reserved.** Read as 0.
5:4	**Video Feedback 1, 0.** These bits are connected to 2 of the 8 color bits sent to the palette. Bits 4 and 5 of the Color Plane Enable Register (AR12) selects which two of the 8 possible color bits become connected to these 2 bits of this register. These bits exist for EGA compatibility.
3	**Vertical Retrace/Video.** 0 = VSYNC inactive (Indicates that a vertical retrace interval is not taking place). 1 = VSYNC active (Indicates that a vertical retrace interval is taking place). VGA pixel generation is not locked to the display output but is loosely coupled. A VSYNC indication may not occur during the actual VSYNC going to the display but during the VSYNC that is generated as part of the VGA pixel generation. The exact relationship will vary with the VGA display operational mode. This status bit will remain active when the VGA is disabled and the device is running in high resolution modes (non-VGA) to allow for applications that (now incorrectly) use these status registers bits. In this case, the status will come from the pipe that the VGA is assigned to. Bits 4 and 5 of the Vertical Retrace End Register (CR11) previously could program this bit to generate an interrupt at the start of the vertical retrace interval. This ability to generate interrupts at the start of the vertical retrace interval is a feature that is largely unused by legacy software. Interrupts are not supported through the VGA register bits.
2:1	**Reserved.** Read as 0s.

Bit	Descriptions
0	**Display Enable Output.** Display Enable is a status bit (bit 0) in VGA Input Status Register 1 that indicates when either a horizontal retrace interval or a vertical retrace interval is taking place. This bit was used with the EGA graphics system (and the ones that preceded it, including MDA and CGA). In those cases, it was important to check the status of this bit to ensure that one or the other retrace intervals was taking place before reading from or writing to the frame buffer. In these earlier systems, reading from or writing to the frame buffer at times outside the retrace intervals meant that the CRT controller would be denied access to the frame buffer. Those behaviors resulted in either "snow" or a flickering display. This bit provides compatibility with software designed for those early graphics controllers. This bit is currently used in DOS applications that access the palette to prevent the sparkle associated with read and write accesses to the palette RAM with the same address on the same clock cycle. **This status bit remains active when the VGA display is disabled and the device is running in high resolution modes (non-VGA) to allow for applications that (now considered incorrect) use these status registers bits. In this case, the status will come from the pipe that the VGA is assigned to. When in panel fitting VGA or centered VGA operation, the meaning of these bits will not be consistent with native VGA timings.** 0 = Active display data is being sent to the display. Neither a horizontal retrace interval or a vertical retrace interval is currently taking place. 1 = Either a horizontal retrace interval (horizontal blanking) or a vertical retrace interval (vertical blanking) is currently taking place.

FCR - Feature Control

I/O (and Memory Offset) Address: 3BAh/3DAh - Write; 3CAh - Read

Default: 00h

Attributes: See Address above

The I/O address used for writes is dependent on CGA or MDA emulation mode as selected via the MSR register. In the original EGA, bits 0 and 1 were used as part of the feature connector interface. Feature connector is not supported in these devices and those bits will always read as zero.

Bit	Descriptions
7:4	**Reserved.** Read as 0.
3	**VSYNC Control.** This bit is provided for compatibility only and has no other function. Reads and writes to this bit have no effect other than to change the value of this bit. The previous definition of this bit selected the output on the VSYNC pin. 0 = Was used to set VSYNC output on the VSYNC pin (default). 1 = Was used to set the logical 'OR' of VSYNC and Display Enable output on the VSYNC pin. This capability was not typically very useful.
2:0	**Reserved.** Read as 0.

MSR - Miscellaneous Output

I/O (and Memory Offset) Address: 3C2h - Write; 3CCh - Read

Default: 00h

Attributes: See Address above

Bit	Descriptions
7	**CRT VSYNC Polarity.** This is a legacy function that is used in native VGA modes. For most cases, sync polarity will be controlled by the port control bits. The VGA settings can be optionally selected for compatibility with the original VGA when used in the VGA native mode. Sync polarity was used in VGA to signal the monitor how many lines of active display are being generated. 0 = Positive Polarity (default). 1 = Negative Polarity.
6	**CRT HSYNC Polarity.** This is a legacy function that is used in native VGA modes. For most cases, sync polarity will be controlled by the port control bits. The VGA settings can be optionally selected for compatibility with the original VGA when used in the VGA native mode. 0 = Positive Polarity (default). 1 = Negative Polarity
5	**Page Select.** In Odd/Even Memory Map Mode 1 (GR6), this bit selects the upper or lower 64 KB page in display memory for CPU access: 0 = Upper page (default) 1 = Lower page. Selects between two 64KB pages of frame buffer memory during standard VGA odd/even modes (modes 0h through 5h). Bit 1 of register GR06 can also program this bit in other modes. This bit is would normally set to 1 by the software.
4	**Reserved.** Read as 0.
3:2	**Clock Select.** These bits can select the dot clock source for the CRT interface. The bits should be used to select the dot clock in standard native VGA modes only. When in the centering or upper left corner modes, these bits should be set to have no effect on the clock rate. The actual frequencies that these bits select, if they have any affect at all, is programmable through the DPLL MMIO registers. 00 = CLK0, 25.175 MHz (for standard VGA modes with 640 pixel (8-dot) horizontal resolution) (default) 01 = CLK1, 28.322 MHz. (for standard VGA modes with 720 pixel (9-dot) horizontal resolution) 10 = Was used to select an external clock (now unused) 11 = Reserved

Bit	Descriptions
1	**A0000-BFFFFh Memory Access Enable.** VGA Compatibility bit enables access to video memory (frame buffer) at A0000-BFFFFh. When disabled, accesses to VGA memory are blocked in this region. This bit is independent of and does not block CPU access to the video linear frame buffer at other addresses. 0 = Prevent CPU access to memory/registers/ROM through the A0000-BFFFF VGA memory aperture (default). 1 = Allow CPU access to memory/registers/ROM through the A0000-BFFFF VGA memory aperture. This memory must be mapped as UC by the CPU.
0	**I/O Address Select.** This bit selects 3Bxh or 3Dxh as the I/O address for the CRT Controller registers, the Feature Control Register (FCR), and Input Status Register 1 (ST01). Presently ignored (whole range is claimed), but will "ignore" 3Bx for color configuration or 3Dx for monochrome. It is typical in AGP chipsets to shadow this bit and properly steer I/O cycles to the proper bus for operation where a MDA exists on another bus such as ISA. 0 = Select 3Bxh I/O address (MDA emulation) (default). 1 = Select 3Dxh I/O address (CGA emulation).

In standard VGA modes using the analog VGA connector, bits 7 and 6 indicate which of the three standard VGA vertical resolutions the standard VGA display should use. Extended modes, including those with a vertical resolution of 480 scan lines, may use a setting of 0 for both of these bits. Different connector standards and timing standards specify the proper use of sync polarity. This setting was "reserved" in the VGA standard.

Analog CRT Display Sync Polarities

V	H	Display	Horizontal Frequency	Vertical Frequency
P	P	200 Line	15.7 KHz	60 Hz
N	P	350 Line	21.8 KHz	60 Hz
P	N	400 Line	31.5 KHz	70 Hz
N	N	480 Line	31.5 KHz	60 Hz

Sequencer Registers

The sequencer registers are accessed via either I/O space or Memory space. To access registers the VGA Sequencer Index register (SRX) at I/O address 3C4h (or memory address 3C4h) is written with the index of the desired register. Then the desired register is accessed through the data port for the sequencer registers at I/O address 3C5 (or memory address 3C5).

SRX - Sequencer Index

I/O (and Memory Offset) Address: 3C4h

Default: 00h

Attributes: Read/Write

Bit	Description
7:3	**Reserved.** Read as 0s.
2:0	**Sequencer Index.** This field contains a 3-bit Sequencer Index value used to access sequencer data registers at indices 0 through 7.

SR00 - Sequencer Reset

I/O (and Memory Offset) Address: 3C5h(Index=00h)

Default: 00h

Attributes: Read/Write

Bit	Descriptions
7:2	**Reserved.** Read as 0.
1	**Reserved.** Reserved for VGA compatibility (was reset).
0	**Reserved.** Reserved for VGA compatibility. (was reset)

SR01 - Clocking Mode

I/O (and Memory Offset) Address: 3C5h (Index=01h)

Default: 00h

Attributes: Read/Write

Bit	Descriptions
7:6	**Reserved.** Read as 0s.
5	**Screen Off.** 0 = Normal Operation (default). 1 = Disables video output (blanks the screen) and turns off display data fetches. Synchronization pulses to the display, however, are maintained. Setting this bit to 1 had been used as a way to more rapidly update and improve CPU access performance to the frame buffer during VGA modes. In non-VGA modes (VGA Disable=1), this bit has no effect. Before the VGA is disabled through the MMIO VGA control register, this bit should be set to stop the memory accesses from the display. The following sequence must be used when disabling the VGA plane. 1. Write SR01 to set bit 5 = 1 to disable video output. 2. Wait for 100us. 3. Disable the VGA plane via Bit 31 of the MMIO VGA control register (location found in the MMIO display register programming specification).
4	**Shift 4.** 0 = Load video shift registers every 1 or 2 character clocks (depending on bit 2 of this register) (default). 1 = Load shift registers every 4th character clock.
3	**Dot Clock Divide.** Setting this bit to 1 stretches doubles all horizontal timing periods that are specified in the VGA horizontal CRTC registers. This bit is used in standard VGA 40-column text modes to stretch timings to create horizontal resolutions of either 320 or 360 pixels (as opposed to 640 or 720 pixels, normally used in standard VGA 80-column text modes). The effect of this is that there will actually be twice the number of pixels sent to the display per line. 0 = Pixel clock is left unaltered (used for 640 (720) pixel modes); (default). 1 = Pixel clock divided by 2 (used for 320 (360) pixel modes).
2	**Shift Load.** Bit 4 of this register must be 0 for this bit to be effective. 0 = Load video data shift registers every character clock (default). 1 = Load video data shift registers every other character clock.
1	**Reserved.** Read as 0.
0	**8/9 Dot Clocks.** This bit determines whether a character clock is 8 or 9 dot clocks long if clock doubling is disabled and 16 or 18 clocks if it is. This also changes the interpretation of the pixel panning values (see chart). An additional control bit determines if this bit is to be ignored and 8-dot characters are to be used always. The 9-dot disable would be used when doubling the horizontal pixels on a 1280 wide display or non-doubling on a 640 wide display. Panning however will occur according to the expected outcome. 0 = 9 dot clocks (9 horizontal pixels) per character in text modes with a horizontal resolution of 720 pixels. 1 = 8 dot clocks (8 horizontal pixels) per character in text or graphics modes with a horizontal resolution of 640 pixels.

SR02 - Plane/Map Mask

I/O (and Memory Offset) Address: 3C5h (Index=02h)

Default: 00h

Attributes: Read/Write

Bit	Descriptions
7:4	**Reserved.** Read as 0s.
3:0	**Memory Planes [3:0] Processor Write Access Enable.** In both the Odd/Even Mode and the Chain 4 Mode, these bits still control access to the corresponding color plane. 0 = Disable. 1 = Enable. This register is referred to in the VGA standard as the Map Mask Register.

SR03 - Character Font

I/O (and Memory Offset) Address: 3C5h (index=03h)

Default: 00h

Attributes: Read/Write

In text modes, bit 3 of the video data's attribute byte normally controls the foreground intensity. This bit may be redefined to control switching between character sets. This latter function is enabled whenever there is a difference in the values of the Character Font Select A and the Character Font Select B bits. If the two values are the same, the character select function is disabled and attribute bit 3 controls the foreground intensity.

Bit 1 of the Memory Mode Register (SR04) must be set to 1 for the character font select function of this register to be active. Otherwise, only character maps 0 and 4 are available.

<table>
<tr><th>Bit</th><th>Descriptions</th></tr>
<tr><td>7:6</td><td>Reserved. Read as 0s.</td></tr>
<tr><td>3:2,5</td><td>Character Map Select Bits for Character Map B. These three bits are used to select the character map (character generator tables) to be used as the secondary character set (font). The numbering of the maps is not sequential.
<table>
<tr><th>Bit [3:2,5]</th><th>Map Number</th><th>Table Location</th></tr>
<tr><td>00,0</td><td>0</td><td>1st 8KB of plane 2 at offset 0 (default)</td></tr>
<tr><td>00,1</td><td>4</td><td>2nd 8KB of plane 2 at offset 8K</td></tr>
<tr><td>01,0</td><td>1</td><td>3rd 8KB of plane 2 at offset 16K</td></tr>
<tr><td>01,1</td><td>5</td><td>4th 8KB of plane 2 at offset 24K</td></tr>
<tr><td>10,0</td><td>2</td><td>5th 8KB of plane 2 at offset 32K</td></tr>
<tr><td>10,1</td><td>6</td><td>6th 8KB of plane 2 at offset 40K</td></tr>
<tr><td>11,0</td><td>3</td><td>7th 8KB of plane 2 at offset 48K</td></tr>
<tr><td>11,1</td><td>7</td><td>8th 8KB of plane 2 at offset 56K</td></tr>
</table></td></tr>
<tr><td>1:0,4</td><td>Character Map Select Bits for Character Map A. These three bits are used to select the character map (character generator tables) to be used as the primary character set (font). The numbering of the maps is not sequential.
<table>
<tr><th>Bit [1:0,4]</th><th>Map Number</th><th>Table Location</th></tr>
<tr><td>00,0</td><td>0</td><td>1st 8KB of plane 2 at offset 0 (default)</td></tr>
<tr><td>00,1</td><td>4</td><td>2nd 8KB of plane 2 at offset 8K</td></tr>
<tr><td>01,0</td><td>1</td><td>3rd 8KB of plane 2 at offset 16K</td></tr>
<tr><td>01,1</td><td>5</td><td>4th 8KB of plane 2 at offset 24K</td></tr>
<tr><td>10,0</td><td>2</td><td>5th 8KB of plane 2 at offset 32K</td></tr>
<tr><td>10,1</td><td>6</td><td>6th 8KB of plane 2 at offset 40K</td></tr>
<tr><td>11,0</td><td>3</td><td>7th 8KB of plane 2 at offset 48K</td></tr>
<tr><td>11,1</td><td>7</td><td>8th 8KB of plane 2 at offset 56K</td></tr>
</table></td></tr>
</table>

SR04 - Memory Mode Register

I/O (and Memory Offset) Address: 3C5h (index=04h)

Default: 00h

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0.
3	**Chain 4 Mode.** The selections made by this bit affect both CPU Read and write accesses to the frame buffer. 0 = The manner in which the frame buffer memory is mapped is determined by the setting of bit 2 of this register (default). 1 = The frame buffer memory is mapped in such a way that the function of address bits 0 and 1 are altered so that they select planes 0 through 3. This setting is used in mode x13 to allow all four planes to be accessed via sequential addresses.
2	**Odd/Even Mode.** Bit 3 of this register must be set to 0 for this bit to be effective. The selections made by this bit affect only non-paged CPU accesses to the frame buffer through the VGA aperture. 0 = The frame buffer memory is mapped in such a way that the function of address bit 0 such that even addresses select planes 0 and 2 and odd addresses select planes 1 and 3 (default). 1 = Addresses sequentially access data within a bit map, and the choice of which map is accessed is made according to the value of the Plane Mask Register (SR02).
1	**Extended Memory Enable.** This bit must be set to 1 to enable the selection and use of character maps in plane 2 via the Character Map Select Register (SR03). 0 = Disable CPU accesses to more than the first 64KB of VGA standard memory (default). 1 = Enable CPU accesses to the rest of the 256KB total VGA memory beyond the first 64KB.
0	**Reserved.** Read as 0.

SR07 - Horizontal Character Counter Reset

I/O (and Memory Offset) Address: 3C5h (index=07h)

Default: 00h

Attributes: Read/Write

For standard VGAs, writing this register (with any data) causes the horizontal character counter to be held in reset (the character counter output will remain 0). It remained in reset until a write occurred to any other sequencer register location with SRX set to an index of 0 through 6. In this implementation that sequence has no such special effect.

The vertical line counter is clocked by a signal derived from the horizontal display enable (which does not occur if the horizontal counter is held in reset). Therefore, if a write occurs to this register during the vertical retrace interval, both the horizontal and vertical counters will be set to 0. A write to any other sequencer register location (with SRX set to an index of 0 through 6) may then be used to start both counters with reasonable synchronization to an external event via software control. Although this was a standard VGA register, it was not documented.

Bit	Description
7:0	**Horizontal Character Counter.**

Graphics Controller Registers

The graphics controller registers are accessed via either I/O space or Memory space. Accesses to the registers of the VGA Graphics Controller are done through the use of address 3CEh (or memory address 3CEh) written with the index of the desired register. Then the desired register is accessed through the data port for the graphics controller registers at I/O address 3CFh (or memory address 3CFh). Indexes 10 and 11 should only be accessed through the I/O space only.

GRX - GRX Graphics Controller Index Register

I/O (and Memory Offset) Address: 3CEh

Default: 000UUUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7:5	**Reserved.** Read as 0.
4:0	**Graphics Controller Register Index.** This field selects any one of the graphics controller registers (GR00-GR18) to be accessed via the data port at I/O (or memory offset) location 3CFh.

GR00 - Set/Reset Register

I/O (and Memory Offset) Address: 3CFh (index=00h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0.
3:0	**Set/Reset Plane [3:0].** When the Write Mode bits (bits 0 and 1) of the Graphics Mode Register (GR05) are set to select Write Mode 0, all 8 bits of each byte of each memory plane are set to either 1 or 0 as specified in the corresponding bit in this register, if the corresponding bit in the Enable Set/Reset Register (GR01) is set to 1. When the Write Mode bits (bits 0 and 1) of the Graphics Mode Register (GR05) are set to select Write Mode 3, all CPU data written to the frame buffer is rotated, then logically ANDed with the contents of the Bit Mask Register (GR08), and then treated as the addressed data's bit mask, while value of these four bits of this register are treated as the color value.

GR01 - Enable Set/Reset Register

I/O (and Memory Offset) Address: 3CFh (Index=01h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0.
3:0	**Enable Set/Reset Plane [3:0].** This register works in conjunction with the Set/Reset Register (GR00). The Write Mode bits (bits 0 and 1) must be set for Write Mode 0 for this register to have any effect. 0 = The corresponding memory plane can be read from or written to by the CPU without any special bitwise operations taking place. 1 = The corresponding memory plane is set to 0 or 1 as specified in the Set/Reset Register (GR00).

GR02 - Color Compare Register

I/O (and Memory Offset) Address: 3CFh (Index=02h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0.
3:0	**Color Compare Plane [3:0].** When the Read Mode bit (bit 3) of the Graphics Mode Register (GR05) is set to select Read Mode 1, all 8 bits of each byte of each of the 4 memory planes of the frame buffer corresponding to the address from which a CPU read access is being performed are compared to the corresponding bits in this register (if the corresponding bit in the Color Don't Care Register (GR07) is set to 1). The value that the CPU receives from the read access is an 8-bit value that shows the result of this comparison, wherein value of 1 in a given bit position indicates that all of the corresponding bits in the bytes across all of the memory planes that were included in the comparison had the same value as their memory plane's respective bits in this register.

GR03 - Data Rotate Register

I/O (and Memory Offset) Address: 3CFh (Index=03h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:5	**Reserved.** Read as 0.
4:3	**Function Select.** These bits specify the logical function (if any) to be performed on data that is meant to be written to the frame buffer (using the contents of the memory read latch) just before it is actually stored in the frame buffer at the intended address location. 00 = Data being written to the frame buffer remains unchanged, and is simply stored in the frame buffer. 01 = Data being written to the frame buffer is logically ANDed with the data in the memory read latch before it is actually stored in the frame buffer. 10 = Data being written to the frame buffer is logically ORed with the data in the memory read latch before it is actually stored in the frame buffer. 11 = Data being written to the frame buffer is logically XORed with the data in the memory read latch before it is actually stored in the frame buffer.
2:0	**Rotate Count.** These bits specify the number of bits to the right to rotate any data that is meant to be written to the frame buffer just before it is actually stored in the frame buffer at the intended address location.

GR04 - Read Plane Select Register

I/O (and Memory Offset) Address: 3CFh (Index=04h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:2	**Reserved.** Read as 0.
1:0	**Read Plane Select.** These two bits select the memory plane from which the CPU reads data in Read Mode 0. In Odd/Even Mode, bit 0 of this register is ignored. In Chain 4 Mode, both bits 1 and 0 of this register are ignored. The four memory planes are selected as follows: 00 = Plane 0 01 = Plane 1 10 = Plane 2 11 = Plane 3 These two bits also select which of the four memory read latches may be read via the Memory read Latch Data Register (CR22). The choice of memory read latch corresponds to the choice of plane specified in the table above. The Memory Read Latch Data register and this additional function served by 2 bits are features of the VGA standard that were never documented.

GR05 - Graphics Mode Register

I/O (and Memory Offset) Address: 3CFh (Index=05h)

Default: 0UUU U0UUb (U=Undefined)

Attributes: Read/Write

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>7</td><td>Reserved. Read as 0.</td></tr>
<tr><td>6:5</td><td>
Shift Register Control. In standard VGA modes, pixel data is transferred from the 4 graphics memory planes to the palette via a set of 4 serial output bits. These 2 bits of this register control the format in which data in the 4 memory planes is serialized for these transfers to the palette.

Bits [6:5]=00

One bit of data at a time from parallel bytes in each of the 4 memory planes is transferred to the palette via the 4 serial output bits, with 1 of each of the serial output bits corresponding to a memory plane. This provides a 4-bit value on each transfer for 1 pixel, making possible a choice of 1 of 16 colors per pixel.

Serial
<table>
<tr><th>Out</th><th>1st Xfer</th><th>2nd Xfer</th><th>3rd Xfer</th><th>4th Xfer</th><th>5th Xfer</th><th>6th Xfer</th><th>7th Xfer</th><th>8th Xfer</th></tr>
<tr><td>Bit 3</td><td>plane3 bit7</td><td>plane3 bit6</td><td>plane3 bit5</td><td>plane3 bit4</td><td>plane3 bit3</td><td>plane3 bit2</td><td>plane3 bit1</td><td>plane3 bit0</td></tr>
<tr><td>Bit 2</td><td>plane2 bit7</td><td>plane2 bit6</td><td>plane2 bit5</td><td>plane2 bit4</td><td>plane2 bit3</td><td>plane2 bit2</td><td>plane2 bit1</td><td>plane2 bit0</td></tr>
<tr><td>Bit 1</td><td>plane1 bit7</td><td>plane1 bit6</td><td>plane1 bit5</td><td>plane1 bit4</td><td>plane1 bit3</td><td>plane1 bit2</td><td>plane1 bit1</td><td>plane1 bit0</td></tr>
<tr><td>Bit 0</td><td>plane0 bit7</td><td>plane0 bit6</td><td>plane0 bit5</td><td>plane0 bit4</td><td>plane0 bit3</td><td>plane0 bit2</td><td>plane0 bit1</td><td>plane0 bit0</td></tr>
</table>

Bits [6:5]=01

Two bits of data at a time from parallel bytes in each of the 4 memory planes are transferred to the palette in a pattern that alternates per byte between memory planes 0 and 2, and memory planes 1 and 3. First the even-numbered and odd-numbered bits of a byte in memory plane 0 are transferred via serial output bits 0 and 1, respectively, while the even-numbered and odd-numbered bits of a byte in memory plane 2 are transferred via serial output bits 2 and 3. Next, the even-numbered and odd-numbered bits of a byte in memory plane 1 are transferred via serial output bits 0 and 1, respectively, while the even-numbered and odd-numbered bits of memory plane 3 are transferred via serial out bits 1 and 3. This provides a pair of 2-bit values (one 2-bit value for each of 2 pixels) on each transfer, making possible a choice of 1 of 4 colors per pixel.

Serial
<table>
<tr><th>Out</th><th>1st Xfer</th><th>2nd</th><th>3rd Xfer</th><th>4th Xfer</th><th>5th Xfer</th><th>6th Xfer</th><th>7th Xfer</th><th>8th Xfer</th></tr>
</table>
</td></tr>
</table>

Bit	Description

	Xfer							
Bit 3	plane2 bit7	plane2 bit5	plane2 bit3	plane2 bit1	plane3 bi7t	plane3 bit5	plane3 bit3	plane3 bit1
Bit 2	plane2 bit6	plane2 bit4	plane2 bit2	plane2 bit0	plane3 bit6	plane3 bit4	plane3 bit2	plane3 bit0
Bit 1	plane0 bit7	plane0 bit5	plane0 bit3	plane0 bit1	plane1 bit7	plane1 bit5	plane1 bit3	plane1 bit1
Bit 0	plane0 bit6	plane0 bit4	plane0 bit2	plane0 bit0	plane1 bit6	plane1 bit4	plane1 bit2	plane1 bit0

This alternating pattern is meant to accommodate the use of the Odd/Even mode of organizing the 4 memory planes, which is used by standard VGA modes 2h and 3h.

Bits [6:5]=1x

Four bits of data at a time from parallel bytes in each of the 4 memory planes are transferred to the palette in a pattern that iterates per byte through memory planes 0 through 3. First the 4 most significant bits of a byte in memory plane 0 are transferred via the 4 serial output bits, followed by the 4 least significant bits of the same byte. Next, the same transfers occur from the parallel byte in memory planes 1, 2 and lastly, 3. Each transfer provides either the upper or lower half of an 8 bit value for the color for each pixel, making possible a choice of 1 of 256 colors per pixel. This is the setting used in mode x13.

Serial

Out	1st Xfer	2nd Xfer	3rd Xfer	4th Xfer	5th Xfer	6th Xfer	7th Xfer	8th Xfer
Bit 3	plane0 bit7	plane0 bit3	plane1 bit7	plane1 bit3	plane2 bit7	plane2 bit3	plane3 bit7	plane3 b3it
Bit 2	plane0 bit6	plane0 bit2	plane1 bit6	plane1 bit2	plane2 bit6	plane2 bit2	plane3 bit6	plane3 bit2
Bit 1	plane0 bit5	plane0 bit1	plane1 bit5	plane1 bit1	plane2 bit5	plane2 bit1	plane3 bit5	plane3 bit1
Bit 0	plane0 bit4	plane0 bit0	plane1 bit4	plane1 bit0	plane2 bit4	plane2 bit0	plane3 bit4	plane3 bit0

This pattern is meant to accommodate mode 13h, a standard VGA 256-color graphics mode.

Bit	Description
4	**Odd/Even Mode.** 0 = Addresses sequentially access data within a bit map, and the choice of which map is accessed is made according to the value of the Plane Mask Register (SR02). 1 = The frame buffer is mapped in such a way that the function of address bit 0 is such that even addresses select memory planes 0 and 2 and odd addresses select memory planes 1 and 3. This works in a way that is the inverse of (and is normally set to be the opposite of) bit 2 of the Memory Mode Register (SR02).
3	**Read Mode.** 0 = During a CPU read from the frame buffer, the value returned to the CPU is data from the memory plane selected by bits 1 and 0 of the Read Plane Select Register (GR04). 1 = During a CPU read from the frame buffer, all 8 bits of the byte in each of the 4 memory planes corresponding to the address from which a CPU read access is being performed are compared to the corresponding bits in this register (if the corresponding bit in the Color Don't Care Register (GR07) is set to 1). The value that the CPU receives from the read access is an 8-bit value that shows the result of this comparison. A value of 1 in a given bit position indicates that all of the corresponding bits in the bytes across all 4 of the memory planes that were included in the comparison had the same value as their memory plane's respective bits in this register.
2	**Reserved.** Read as 0.

Bit	Description
1:0	**Write Mode.** 00 = Write Mode 0 - During a CPU write to the frame buffer, the addressed byte in each of the 4 memory planes is written with the CPU write data after it has been rotated by the number of counts specified in the Data Rotate Register (GR03). If, however, the bit(s) in the Enable Set/Reset Register (GR01) corresponding to one or more of the memory planes is set to 1, then those memory planes will be written to with the data stored in the corresponding bits in the Set/Reset Register (GR00). 01 = Write Mode 1 - During a CPU write to the frame buffer, the addressed byte in each of the 4 memory planes is written to with the data stored in the memory read latches. (The memory read latches stores an unaltered copy of the data last read from any location in the frame buffer.) 10 = Write Mode 2 - During a CPU write to the frame buffer, the least significant 4 data bits of the CPU write data is treated as the color value for the pixels in the addressed byte in all 4 memory planes. The 8 bits of the Bit Mask Register (GR08) are used to selectively enable or disable the ability to write to the corresponding bit in each of the 4 memory planes that correspond to a given pixel. A setting of 0 in a bit in the Bit Mask Register at a given bit position causes the bits in the corresponding bit positions in the addressed byte in all 4 memory planes to be written with value of their counterparts in the memory read latches. A setting of 1 in a Bit Mask Register at a given bit position causes the bits in the corresponding bit positions in the addressed byte in all 4 memory planes to be written with the 4 bits taken from the CPU write data to thereby cause the pixel corresponding to these bits to be set to the color value. 11 = Write Mode 3 - During a CPU write to the frame buffer, the CPU write data is logically ANDed with the contents of the Bit Mask Register (GR08). The result of this ANDing is treated as the bit mask used in writing the contents of the Set/Reset Register (GR00) are written to addressed byte in all 4 memory planes.

GR06 - Miscellaneous Register

I/O (and Memory Offset) Address: 3CFh (Index=06h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0s.
3:2	**Memory Map Mode.** These 2 bits control the mapping of the VGA address range for frame buffer into the CPU address space as follows: 00 = A0000h - BFFFFh 01 = A0000h - AFFFFh 10 = B0000h - B7FFFh 11 = B8000h - BFFFFh This function is used in standard VGA modes, extended VGA modes (132 column text), and in non-VGA modes (hi-res). 132 column text modes are no longer supported. VGA aperture memory accesses are also controlled by the PCI configuration Memory Enable bit and MSR<1>. For accesses using GR10 and GR11 to paged VGA RAM or to device MMIO registers, set these bits to 01 to select the (A0000-AFFFF) range. The CPU must map this memory as uncacheable (UC).
1	**Chain Odd/Even.** This bit provides the ability to alter the interpretation of address bit A0, so that it may be used in selecting between the odd-numbered memory planes (planes 1 and 3) and the even-numbered memory planes (planes 0 and 2). 0 = A0 functions normally. 1 = A0 is switched with a high order address bit, in terms of how it is used in address decoding. The result is that A0 is used to determine which memory plane is being accessed (A0=0 for planes 0 and 2 and A0=1 for planes 1 and 3).
0	**Graphics/Text Mode.** This is one of two bits that are used to determine if the VGA is operating in text or graphics modes. The other bit is in AR10[0], these two bits need to be programmed in a consistent manner to achieve the proper results. 0 = Text mode. 1 = Graphics mode.

GR07 - Color Don't Care Register

I/O (and Memory Offset) Address: 3CFh (Index=07h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.** Read as 0.
3:0	**Ignore Color Plane [3:0].** These bits have effect only when bit 3 of the Graphics Mode Register (GR05) is set to 1 to select read mode 1. 0 = The corresponding bit in the Color Compare Register (GR02) will not be included in color comparisons. 1 = The corresponding bit in the Color Compare Register (GR02) is used in color comparisons.

GR08 - Bit Mask Register

I/O (and Memory Offset) Address: 3CFh (Index=08h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Bit Mask.** 0 = The corresponding bit in each of the 4 memory planes is written to with the corresponding bit in the memory read latches. 1 = Manipulation of the corresponding bit in each of the 4 memory planes via other mechanisms is enabled. This bit mask applies to any writes to the addressed byte of any or all of the 4 memory planes, simultaneously. This bit mask is applicable to any data written into the frame buffer by the CPU, including data that is also subject to rotation, logical functions (AND, OR, XOR), and Set/Reset. To perform a proper read-modify-write cycle into frame buffer, each byte must first be read from the frame buffer by the CPU (and this will cause it to be stored in the memory read latches), this Bit Mask Register must be set, and the new data then written into the frame buffer by the CPU.

GR10 - Address Mapping

I/O (avoid MMIO access) Address: 3CFh (Index=10h)

Default: 00h

Attributes: R/W

This register should only be accessed using I/O operations and never be accessed through the A/B segment addressing map, I/O space register map, or direct MMIO operations.

Bit	Description
7:4	**Page Select Extension - Unused** These bits form the upper bits of a 12-bit page selection value. When combined with the GR11 <7:0> bits they define the offset into stolen memory to the 64KB page that is accessible via the VGA Memory paging mechanism. These bits are ignored.
3	**Reserved**
2:1	**Paging Map Target.** When paging is enabled, these bits determine the target for data cycle accesses through the VGA memory aperture. VGA graphics memory starts from the base of graphics data stolen memory defined in the PCI configuration BDSM register. VGA display uses the first four 64KB pages of VGA graphics memory. 00 = VGA Graphics Memory 01 = Reserved 10 = Reserved 11 = Reserved
0	**Page Mapping Enable.** This mode allows the mapping of the VGA memory address space. Once this is enabled, no VGA memory address swizzle will be performed, addresses are directly mapped to memory. A single paging register is used to map the 64KB [A0000:AFFFF] window. An internal address is generated using GR11 as the address lines extension to the lower address lines of the access A[15:2]. When mapping is enabled, the B0000:BFFFF area must be disabled using GR06<3:2>=01. The use of addresses in the A0000-BFFFF range require that both the graphics device PCI configuration memory enable and MSR<1> be enabled. 0 = Disable (default) 1 = Enable

GR11 - Page Selector

I/O (avoid MMIO access) Address: 3CFh (Index=11h)

Default: 00h

Attributes: R/W

Bit	Description
7	**Reserved**
6:0	**Page Select.** When concatenated with the GR10<7:4> bits, selects a 64KB window within target area when Page Mapping is enabled (GR10[0]=1). This requires that the graphics device PCI configuration space memory enable, the GR06<3:2> bits to be 01 (select A0000-AFFFF only), and the MSR<1:1> bit to be set. This register provides the Address[22:16] bits for the access. VGA paging of frame buffer memory is for non-VGA packed modes only and should not be enabled when using basic VGA modes. This register should only be accessed using I/O operations.

GR18 - Software Flags

I/O (and Memory Offset) Address: 3CFh (Index=18h)

Default: 00

Attributes: R/W

Bit	Description
7:0	**Software Flags.** Used as scratch pad space in video BIOS. These bits are separate from the bits which appear in the memory mapped IO space. They are used specifically by the SMI BIOS which does not have access to memory mapped IO at the time they are required. These register bits have no effect on H/W operation.

Attribute Controller Registers

Unlike the other sets of indexed registers, the attribute controller registers are not accessed through a scheme employing entirely separate index and data ports. I/O address 3C0h (or memory address 3C0h) is used both as the read and write for the index register, and as the write address for the data port. I/O address 3C1h (or memory address 3C1h) is the read address for the data port.

To write to the attribute controller registers, the index of the desired register must be written to I/O address 3C0h (or memory address 3C0h), and then the data is written to the very same I/O (memory) address. A flip-flop alternates with each write to I/O address 3C0h (or memory address 3C0h) to change its function from writing the index to writing the actual data, and back again. This flip-flop may be deliberately set so that I/O address 3C0h (or memory address 3C0h) is set to write to the index (which provides a way to set it to a known state) by performing a read operation from Input Status Register 1 (ST01) at I/O address 3BAh (or memory address 3BAh) or 3DAh (or memory address 3DAh), depending on whether the graphics system has been set to emulate an MDA or a CGA as per MSR[0].

To read from the attribute controller registers, the index of the desired register must be written to I/O address 3C0h (or memory address 3C0h), and then the data is read from I/O address 3C1h (or memory address 3C1h). A read operation from I/O address 3C1h (or memory address 3C1h) does not reset the flip-flop to writing to the index. Only a write to 3C0h (or memory address 3C0h) or a read from 3BAh or 3DAh (or memory address 3BAh or 3DAh), as described above, will toggle the flip-flop back to writing to the index.

ARX - Attribute Controller Index Register

I/O (and Memory Offset) Address: 3C0h

Default: 00UU UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7:6	**Reserved.** Read as 0s.
5	**Video Enable.** In the VGA standard, this is called the "Palette Address Source" bit. Clearing this bit will cause the VGA display data to become all 00 index values. For the default palette, this will cause a black screen. The video timing signals continue. Another control bit will turn video off and stop the data fetches. 0 = Disable. Attribute controller color registers (AR[00:0F]) can be accessed by the CPU. 1 = Enable. Attribute controller color registers (AR[00:0F]) are inaccessible by the CPU.
4:0	**Attribute Controller Register Index.** These five bits are used to select any one of the attribute controller registers (AR[00:14]), to be accessed.

AR[00:0F] - Palette Registers [0:F]

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=00h-0Fh)

Default: 00UU UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7:6	**Reserved.** Read as 0.
5:0	**Palette Bits P[5:0].** In each of these 16 registers, these are the lower 6 of 8 bits that are used to map either text attributes or pixel color input values (for modes that use 16 colors) to the 256 possible colors available to be selected in the palette. Bits 3 and 2 of the Color Select Register (AR14) supply bits P7 and P6 for the values contained in all 16 of these registers. Bits 1 and 0 of the Color Select Register (AR14) can also replace bits P5 and P4 for the values contained in all 16 of these registers, if bit 7 of the Mode Control Register (AR10) is set to 1.

AR10 - Mode Control Register

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=10h)

Default: UUh (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**Palette Bits P5, P4 Select.** **0** = P5 and P4 for each of the 16 selected colors (for modes that use 16 colors) are individually provided by bits 5 and 4 of their corresponding Palette Registers (AR[00:0F]). **1** = P5 and P4 for all 16 of the selected colors (for modes that use 16 colors) are provided by bits 1 and 0 of Color Select Register (AR14).
6	**Pixel Width/Clock Select.** **0** = Six bits of video data (translated from 4 bits via the palette) are output every dot clock. **1** = Two sets of 4 bits of data are assembled to generate 8 bits of video data which is output every other dot clock, and the Palette Registers (AR[00:0F]) are bypassed. This bit is set to 0 for all of the standard VGA modes, except mode 13h.
5	**Pixel Panning Compatibility.** **0** = Scroll both the upper and lower screen regions horizontally as specified in the Pixel Panning Register (AR13). **1** = Scroll only the upper screen region horizontally as specified in the Pixel Panning Register (AR13). This bit has application only when split-screen mode is being used, where the display area is divided into distinct upper and lower regions which function somewhat like separate displays.
4	**Reserved.** Read as 0.
3	**Enable Blinking/Select Background Intensity.** **0** = Disables blinking in graphics modes, and for text modes, sets bit 7 of the character attribute bytes to control background intensity, instead of blinking. **1** = Enables blinking in graphics modes and for text modes, sets bit 7 of the character attribute bytes to control blinking, instead of background intensity. The blinking rate is derived by dividing the VSYNC signal. The Blink Rate Control field of the VGA control register defines the blinking rate.

Bit	Description
2	**Enable Line Graphics Character Code.** **0** = Every 9th pixel of a horizontal line (i.e., the last pixel of each horizontal line of each 9-pixel wide character box) is assigned the same attributes as the background of the character of which the given pixel is a part. **1** = Every 9th pixel of a horizontal line (i.e., the last pixel of each horizontal line of each 9-pixel wide character box) is assigned the same attributes as the 8th pixel if the character of which the given pixel is a part. This setting is intended to accommodate the line-drawing characters of the PC's extended ASCII character set -- characters with an extended ASCII code in the range of B0h to DFh. In some literature describing the VGA standard, the range of extended ASCII codes that are said to include the line-drawing characters is mistakenly specified as C0h to DFh, rather than the correct range of B0h to DFh.
1	**Select Display Type.** **0** = Attribute bytes in text modes are interpreted as they would be for a color display. **1** = Attribute bytes in text modes are interpreted as they would be for a monochrome display.
0	**Graphics/Alphanumeric Mode.** This bit (along with GR06[0]) select either graphics mode or text mode. These two bits must be programmed in a consistent manner to achieve the desired results. **0** = Alphanumeric (text) mode. **1** = Graphics mode.

AR11 - Overscan Color Register

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=11h)

Default: UUh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:0	**Overscan.** These 8 bits select the overscan (border) color index value. The actual border color will be determined by the contents of the palette at the selected index. The border color is displayed between the end of active and the beginning of blank or the end of blank and the beginning of active on CRT type devices driven from the DAC output port. For native VGA modes on digital display ports there is the option of including the border in the active region or not depending on a control bit in the port control register. For centered VGA modes, the VGA control register determines if the border is included in the centered region or not. For monochrome displays, this value should be set to 00h.

AR12 - Memory Plane Enable Register

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=12h)

Default: 00UU UUUUb (U=Undefined)

Attributes: Read/Write

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved. Read as 0.</td></tr>
<tr><td>5:4</td><td>Video Status Mux. These 2 bits are used to select 2 of the 8 possible palette bits (P7-P0) to be made available to be read via bits 5 and 4 of the Input Status Register 1 (ST01). The table below shows the possible choices.

<table>
<tr><th>Bit [5:4]</th><th>ST01 Bit 5</th><th>ST01 Bit 4</th></tr>
<tr><td>00</td><td>P2 (default)</td><td>P0 (default)</td></tr>
<tr><td>01</td><td>P5</td><td>P4</td></tr>
<tr><td>10</td><td>P3</td><td>P1</td></tr>
<tr><td>11</td><td>P7</td><td>P6</td></tr>
</table>
These bits are typically unused by current software; they are provided for EGA compatibility.</td></tr>
<tr><td>3:0</td><td>Enable Plane [3:0]. These 4 bits individually enable the use of each of the 4 memory planes in providing 1 of the 4 bits used in video output to select 1 of 16 possible colors from the palette to be displayed.

0 = Disable the use of the corresponding memory plane in video output to select colors, forcing the bit that the corresponding memory plane would have provided to a value of 0.
1 = Enable the use of the corresponding memory plane in video output to select colors.

AR12 is referred to in the VGA standard as the Color Plane Enable Register.</td></tr>
</table>

AR13 - Horizontal Pixel Panning Register

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=13h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.**
3:0	**Horizontal Pixel Shift 3-0.** This field holds a 4-bit value that selects the number of pixels by which the image is shifted horizontally to the left. This function is available in both text and graphics modes and allows for pixel panning. In text modes with a 9-pixel wide character box, the image can be shifted up to 9 pixels to the left. In text modes with an 8-pixel wide character box, and in graphics modes other than those with 256 colors, the image can be shifted up to 8 pixels to the left. A pseudo 9-bit mode is when the 9-dot character is selected but overridden by the VGA control bit. In standard VGA mode 13h (where bit 6 of the Mode Control Register, AR10, is set to 1 to support 256 colors), bit 0 of this register must remain set to 0, and the image may be shifted up to only 4 pixels to the left. In this mode, the number of pixels by which the image is shifted can be further controlled using bits 6 and 5 of the Preset Row Scan Register (CR08).

Number of Pixels Shifted				
Bits [3:0]	**9-dot**	**Pseudo 9-dot**	**8-dot**	**256-Color**
0	1	1	0	0
1	2	2	1	Undefined
2	3	3	2	1
3	4	4	3	Undefined
4	5	5	4	2
5	6	6	5	Undefined
6	7	7	6	3
7	8	7	7	Undefined
8	0	0	Undefined	Undefined

AR14 - Color Select Register

I/O (and Memory Offset) Address: Read at 3C1h and Write at 3C0h; (index=14h)

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7:4	**Reserved.**
3:2	**Palette Bits P[7:6].** These are the 2 upper-most of the 8 bits that are used to map either text attributes or pixel color input values (for modes that use 16 colors) to the 256 possible colors contained in the palette. These 2 bits are common to all 16 sets of bits P5 through P0 that are individually supplied by Palette Registers 0-F (AR[00:0F]).
1:0	**Alternate Palette Bits P[5:4].** These 2 bits can be used as an alternate version of palette bits P5 and P4. Unlike the P5 and P4 bits that are individually supplied by Palette Registers 0-F (AR[00:0F]), these 2 alternate palette bits are common to all 16 of Palette Registers. Bit 7 of the Mode Control Register (AR10) is used to select between the use of either the P5 and P4 bits that are individually supplied by the 16 Palette Registers or these 2 alternate palette bits.

VGA Color Palette Registers

In devices that have multiple display pipes, there is one palette for each display pipe. These palettes are the same for VGA modes and non-VGA modes. Accesses through VGA register methods can optionally read or write from either one.

For each palette, the color data stored in these 256 color data positions can be accessed only through a complex sub-addressing scheme, using a data register and two index registers. The Palette Data Register at I/O address 3C9h (or memory address offset 3C1h) is the data port. The Palette Read Index Register at I/O address 3C7h (or memory address offset 3C7h) and the Palette Write Index Register at I/O address 3C8h (or memory address offset 3C8h) are the two index registers. The Palette Read Index Register is the index register that is used to choose the color data position that is to be read from via the data port, while the Palette Write Index Register is the index register that is used to choose the color data position that is to be written to through the same data port. This arrangement allows the same data port to be used for reading from and writing to two different color data positions. Reading and writing the color data at a color data position involves three successive reads or writes since the color data stored at each color data position consists of three bytes.

To read a palette color data position, the index of the desired color data position must first be written to the Palette Read Index Register. Then all three bytes of data in a given color data position may be read at the Palette Data Register. The first byte read from the Palette Data Register retrieves the 8-bit value specifying the intensity of the red color component. The second and third bytes read are the corresponding 8-bit values for the green and blue color components respectively. After completing the third read operation, the Palette Read Index Register is automatically incremented so that the data of the next color data position becomes accessible for being read. This allows the contents of all of the 256 color data positions of the palette to be read in sequence. This is done by specifying only the index of the 0th color data position in the Palette Read Index Register, and then simply performing 768 successive reads from the Palette Data Register.

Writing a color data position, entails a very similar procedure. The index of the desired color data position must first be written to the Palette Write Index Register. Then all three bytes of data to specify a given color may be written to the Palette Data Register. The first byte written to the Palette Data Register specifies the intensity of the red color component, the second byte specifies the intensity for the green color component, and the third byte specifies the same for the blue color component. One important detail is that all three of these bytes must be written before the hardware will actually update these three values in the given color data position. When all three bytes have been written, the Palette Write Index Register is automatically incremented so that the data of the next color data position becomes accessible for being written. This allows the contents of all of the 256 color data positions of the palette to be written in sequence. This is done by specifying only the index of the 0th color data position in the Palette Write Index Register, and then simply performing 768 successive writes to the Palette Data Register.

DACMASK - Pixel Data Mask Register

I/O (and Memory Offset) Address: 3C6h

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Pixel Data Mask.** In indexed-color mode, the 8 bits of this register are logically ANDed with the 8 bits of pixel data received from the frame buffer for each pixel. The result of this ANDing process becomes the actual index used to select color data positions within the palette. This has the effect of limiting the choice of color data positions that may be specified by the incoming 8-bit data. 0 = Corresponding bit in the resulting 8-bit index being forced to 0. 1 = Allows the corresponding bit in the resulting index to reflect the actual value of the corresponding bit in the incoming 8-bit pixel data.

DACSTATE - DAC State Register

I/O (and Memory Offset) Address: 3C7h

Default: 00h

Attributes: Read Only

Bit	Description
7:2	**Reserved.** Read as 0.
1:0	**DACState.** This field indicates which of the two index registers was most recently written. **Bits [1:0]** Index Register Indicated 00 = Palette Write Index Register at I/O Address 3C7h (default) 01 = Reserved 10 = Reserved 11 = Palette Read Index Register at I/O Address 3C8h

DACRX - Palette Read Index Register

I/O (and Memory Offset) Address: 3C7h

Default: 00h

Attributes: Write Only

Bit	Description
7:0	**Palette Read Index.** The 8-bit index value programmed into this register chooses which of 256 standard color data positions within the palette are to be made accessible for being read from via the Palette Data Register (DACDATA). The index value held in this register is automatically incremented when all three bytes of the color data position selected by the current index have been read. A write to this register will abort a uncompleted palette write sequence. This register allows access to the palette even when running non-VGA display modes.

DACWX - Palette Write Index Register

I/O (and Memory Offset) Address: 3C8h

Default: 00h

Attributes: Write Only

Bit	Description
7:0	**Palette Write Index.** The 8-bit index value programmed into this register chooses which of 256 standard color data positions within the palette are to be made accessible for being written via the Palette Data Register (DACDATA). The index value held in this register is automatically incremented when all three bytes of the color data position selected by the current index have been written. This register allows access to the palette even when running non-VGA display modes.

DACDATA - Palette Data Register

I/O (and Memory Offset) Address: 3C9h

Default: Undefined

Attributes: Read/Write

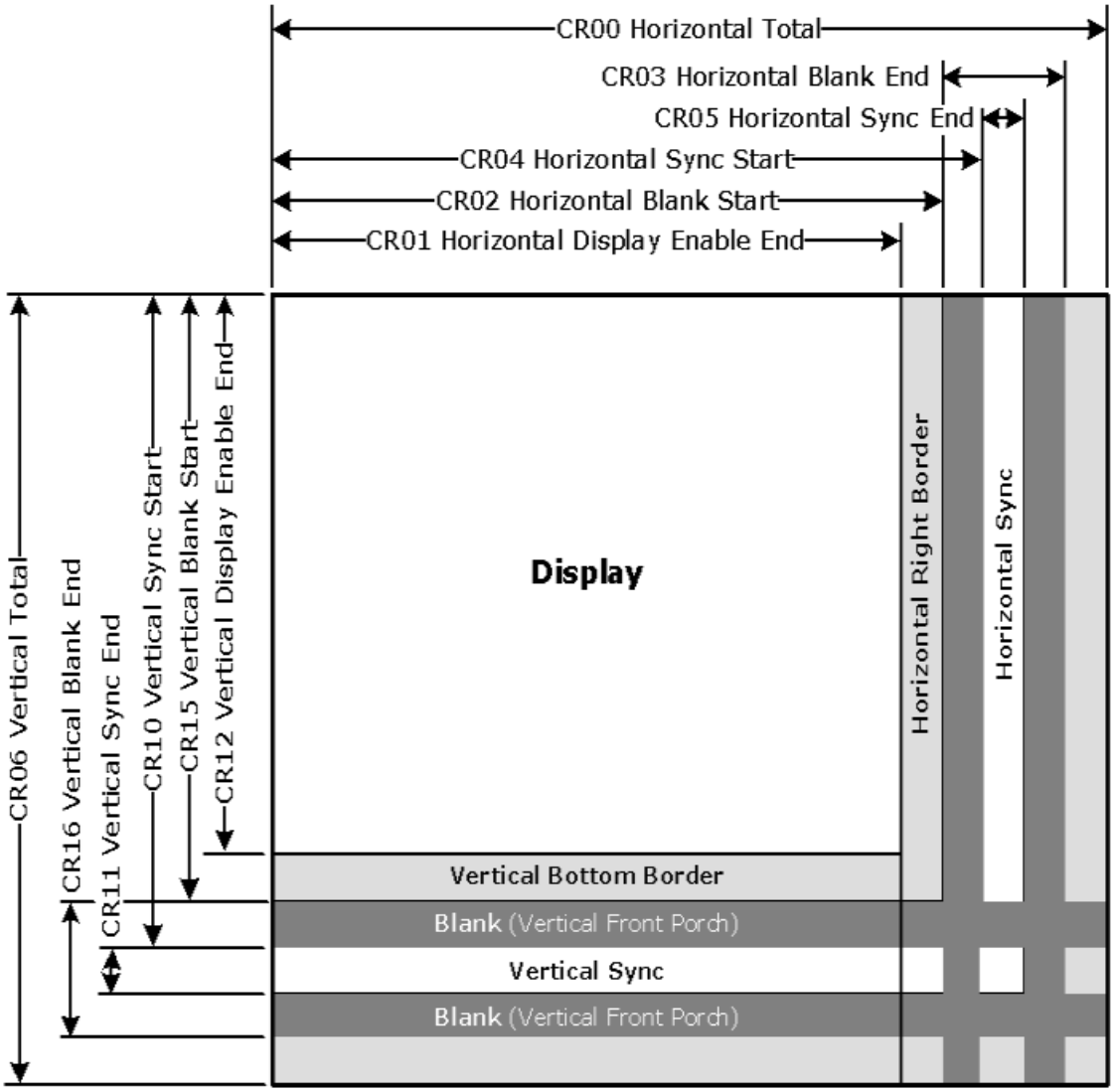
Bit	Description
7:0	**Palette Data.** This byte-wide data port provides read or write access to the three bytes of data of each color data position selected using the Palette Read Index Register (DACRX) or the Palette Write Index Register (DACWX). The three bytes in each color data position are read or written in three successive read or write operations. The first byte read or written specifies the intensity of the red component of the color specified in the selected color data position. The second byte is for the green component, and the third byte is for the blue component. When writing data to a color data position, all three bytes must be written before the hardware will actually update the three bytes of the selected color data position. When reading or writing to a color data position, ensure that neither the Palette Read Index Register (DACRX) or the Palette Write Index Register (DACWX) are written to before all three bytes are read or written. A write to either of these two registers causes the circuitry that automatically cycles through providing access to the bytes for red, green and blue components to be reset such that the byte for the red component is the one that will be accessed by the next read or write operation via this register. This register allows access to the palette even when running non-VGA display modes. Writes to the palette can cause sparkle if not done during inactive video periods. This sparkle is caused by an attempt to write and read the same address on the same cycle. Anti-sparkle circuits will substitute the previous pixel value for the read output.

CRT Controller Register

For native VGA modes, the CRTC registers determine the display timing that is to be used. In centered VGA modes, these registers determine the size of the VGA image that is to be centered in the larger timing generator defined rectangle.

The CRT controller registers are accessed by writing the index of the desired register into the CRT Controller Index Register at I/O address 3B4h or 3D4h, depending on whether the graphics system is configured for MDA or CGA emulation. The desired register is then accessed through the data port for the CRT controller registers located at I/O address 3B5h or 3D5h, again depending upon the choice of MDA or CGA emulation as per MSR[0]. For memory mapped accesses, the Index register is at 3B4h (MDA mode) or 3D3h (CGA mode) and the data port is accessed at 3B5h (MDA mode) or 3D5h (CGA mode).

The following figure shows display fields and dimensions and the particular CRxx register that provides the control.



B6781-01

Group 0 Protection: In the original VGA, CR[0:7] could be made write-protected by CR11[7]. In BIOS code, this write protection is set following each mode change. Other protection groups have no current use, and would not be used going forward by the BIOS or by drivers. They are the result of an industry fad some years ago to attempt to write protect other groups of registers; however, all such schemes were chip specific. Only the write protection (Group 0 Protection) is supported.

CRX - CRT Controller Index Register

I/O (and Memory Offset) Address: 3B4h/3D4h

Default: 0Uh (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**Reserved.** Read as 0.
6:0	**CRT Controller Index.** These 7 bits are used to select any one of the CRT controller registers to be accessed via the data port at I/O location 3B5h or 3D5h, depending upon whether the graphics system is configured for MDA or CGA emulation. The data port memory address offsets are 3B5h/3D5h.

CR00 - Horizontal Total Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=00h)

Default: 00h

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7:0	**Horizontal Total.** This register is used to specify the total length of each scan line. This encompasses both the part of the scan line that is within the active display area and the part that is outside of it. Programming this register to a zero has the effect of stopping the fetching of display data. This field should be programmed with a value equal to the total number of character clocks within the entire length of a scan line, minus 5.

CR01 - Horizontal Display Enable End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=01h)

Default: Undefined

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7:0	**Horizontal Display Enable End.** This register is used to specify the end of the part of the scan line that is within the active display area relative to its beginning. In other words, this is the horizontal width of the active display area. This field should be programmed with a value equal to the number of character clocks that occur within the horizontal active display area, minus 1. Horizontal display enable will go active at the beginning of each line during vertical active area, it will go inactive based on the programming of this register or the programming of the horizontal total (CR00) register. When this register value is programmed to a number that is larger than the total number of characters on a line, display enable will be active for all but the last character of the horizontal display line.

CR02 - Horizontal Blanking Start Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=02h)

Default: Undefined

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7:0	**Horizontal Blanking Start.** This register is used to specify the beginning of the horizontal blanking period relative to the beginning of the active display area of a scan line. Horizontal blanking should always be set to start no sooner than after the end of horizontal active. This field should be programmed with a value equal to the number of character clocks that occur on a scan line from the beginning of the active display area to the beginning of the horizontal blanking.

CR03 - Horizontal Blanking End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=03h)

Default: 1UUU UUUUb (U=Undefined)

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7	**Reserved.** Values written to this bit are ignored, and to maintain consistency with the VGA standard, a value of 1 is returned when this bit is read. At one time, this bit was used to enable access to certain light pen registers. At that time, setting this bit to 0 provided this access, but setting this bit to 1 was necessary for normal operation.
6:5	**Display Enable Skew Control.** Defines the degree to which the start and end of the active display area are delayed along the length of a scan line to compensate for internal pipeline delays. These 2 bits describe the delay in terms of a number character clocks. **Bit [6:5]** Amount of Delay 00 = no delay 01 = delayed by 1 character clock 10 = delayed by 2 character clocks 11 = delayed by 3 character clocks
4:0	**Horizontal Blanking End Bits [4:0].** This field provides the 5 least significant bits of a 6-bit value that specifies the end of the blanking period relative to its beginning on a single scan line. Bit 7 of the Horizontal Sync End Register (CR05) supplies the most significant bit. This 6-bit value should be programmed to be equal to the least significant 6 bits of the result of adding the length of the blanking period in terms of character clocks to the value specified in the Horizontal Blanking Start Register (CR02). End of blanking should occur before horizontal total.

CR04 - Horizontal Sync Start Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=04h)

Default: Undefined

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7:0	**Horizontal Sync Start** This register is used to specify the position of the beginning of the horizontal sync pulse relative to the start of the active display area on a scan line. This field should be set equal to the number of character clocks that occur from beginning of the active display area to the beginning of the horizontal sync pulse on a single scan line. Horizontal sync should always occur at least 2 clocks after the start of horizontal blank and 2 clocks before the end of horizontal blank. The actual start of sync will also be affected by both the horizontal sync skew register field and whether it is a text or graphics mode.

CR05 - Horizontal Sync End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=05h)

Default: 00h

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7	**Horizontal Blanking End Bit 5.** This bit provides the most significant bit of a 6-bit value that specifies the end of the horizontal blanking period relative to its beginning. Bits [4:0] of Horizontal Blanking End Register (CR03) supplies the 5 least significant bits. See CR03[4:0] for further details. This 6-bit value should be set to the least significant 6 bits of the result of adding the length of the blanking period in terms of character clocks to the value specified in the Horizontal Blanking Start Register (CR02).
6:5	**Horizontal Sync Delay.** This field defines the degree to which the start and end of the horizontal sync pulse are delayed to compensate for internal pipeline delays. This capability is supplied to implement VGA compatibility. These field describes the delay in terms of a number character clocks. **Bit [6:5] — Amount of Delay** 00 — no delay 01 — delayed by 1 character clock 10 — delayed by 2 character clocks 11 — delayed by 3 character clocks
4:0	**Horizontal Sync End.** This field provides the 5 least significant bits of a 5-bit value that specifies the end of the horizontal sync pulse relative to its beginning. A value equal to the 5 least significant bits of the horizontal character counter value at which time the horizontal retrace signal becomes inactive (logical 0). Thus, this 5-bit value specifies the width of the horizontal sync pulse. To obtain a retrace signal of W, the following algorithm is used: Value of Horizontal Sync start Register (CR04) + width of horizontal retrace signal in character clock units = 5 bit result to be programmed in this field

CR06 - Vertical Total Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=06h)

Default: 00h

Attributes: Read/Write (Group 0 Protection)

Bit	Description
7:0	**Vertical Total Bits [7:0].** This field provides the 8 least significant bits of either a 10-bit or 12-bit value that specifies the total number of scan lines. This includes the scan lines both inside and outside of the active display area. In standard VGA modes, the vertical total is specified with a 10-bit value. The 8 least significant bits of this value are supplied by these 8 bits of this register, and the 2 most significant bits are supplied by bits 5 and 0 of the Overflow Register (CR07).

CR07 - Overflow Register (Vertical)

I/O (and Memory Offset) Address: 3B5h/3D5h (index=07h)

Default: UU0U UUU0b (U=Undefined)

Attributes: Read/Write (Group 0 Protection on bits [7:5, 3:0])

Bit	Description
7	**Vertical Sync Start Bit 9.** The vertical sync start is a 10-bit that specifies the beginning of the vertical sync pulse relative to the beginning of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Sync Start Register (CR10), and the most and second-most significant bits are supplied by this bit and bit 2, respectively, of this register. This 10-bit value should be programmed to be equal to the number of scan lines from the beginning of the active display area to the start of the vertical sync pulse. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which the vertical sync pulse begins.
6	**Vertical Display Enable End Bit 9.** The vertical display enable end is a 10-bit that specifies the number of the last scan line within the active display area. In standard VGA modes, the vertical display enable end is specified with a 10-bit value. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Display Enable End Register (CR12), and the most and second-most significant bits are supplied by this bit and bit 1, respectively, of this register. This 10-bit value should be programmed to be equal to the number of the last scan line within in the active display area. Since the active display area always starts on the 0th scan line, this number should be equal to the total number of scan lines within the active display area, minus 1.
5	**Vertical Total Bit 9.** The vertical total is a 10-bit value that specifies the total number of scan lines. This includes the scan lines both inside and outside of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Total Register (CR06), and the most and second-most significant bits are supplied by this bit and bit 0, respectively, of this register. This 10-bit value should be programmed equal to the total number of scan lines, minus 2.
4	**Line Compare Bit 8.** This bit provides the second most significant bit of a 10-bit value that specifies the scan line at which the memory address counter restarts at the value of 0. Bit 6 of the Maximum Scan Line Register (CR09) supplies the most significant bit, and bits 7-0 of the Line Compare Register (CR18) supply the 8 least significant bits. Normally, this 10-bit value is set to specify a scan line after the last scan line of the active display area. When this 10-bit value is set to specify a scan line within the active display area, it causes that scan line and all subsequent scan lines in the active display area to display video data starting at the very first byte of the frame buffer. The result is what appears to be a screen split into a top and bottom part, with the image in the top part being repeated in the bottom part. When used in cooperation with the Start Address High Register (CR0C) and the Start Address Low Register (CR0D), it is possible to create a split display, as described earlier, but with the top and bottom parts displaying different data. The top part will display what data exists in the frame buffer starting at the address specified in the two aforementioned start address registers, while the bottom part will display what data exists in the frame buffer starting at the first byte of the frame buffer.

Bit	Description
3	**Vertical Blanking Start Bit 8.** The vertical blanking start is a 10-bit that specifies the beginning of the vertical blanking period relative to the beginning of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Blanking Start Register (CR15), and the most and second-most significant bits are supplied by bit 5 of the Maximum Scan Line Register (CR09) and this bit of this register, respectively. This 10-bit value should be programmed to be equal to the number of scan lines from the beginning of the active display area to the beginning of the blanking period. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which the vertical blanking period begins.
2	**Vertical Sync Start Bit 8.** The vertical sync start is a 10-bit value that specifies the beginning of the vertical sync pulse relative to the beginning of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Sync Start Register (CR10), and the most and second-most significant bits are supplied by bit 7 and this bit, respectively, of this register. This 10-bit value should be programmed to be equal to the number of scan lines from the beginning of the active display area to the start of the vertical sync pulse. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which the vertical sync pulse begins.
1	**Vertical Display Enable End Bit 8.** The vertical display enable end is a 10-bit value that specifies the number of the last scan line within the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Display Enable End Register (CR12), and the two most significant bits are supplied by bit 6 and this bit, respectively, of this register. This 10-bit or value should be programmed to be equal to the number of the last scan line within in the active display area. Since the active display area always starts on the 0th scan line, this number should be equal to the total number of scan lines within the active display area, minus 1.
0	**Vertical Total Bit 8.** The vertical total is a 10-bit value that specifies the total number of scan lines. This includes the scan lines both inside and outside of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Total Register (CR06), and the most and second-most significant bits are supplied by bit 5 and this bit, respectively, of this register. This 10-bit value should be programmed to be equal to the total number of scan lines, minus 2.

CR08 - Preset Row Scan Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=08h)

Default: 0UUU UUUUb (U=Undefined)

Attributes: Read/Write

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>7</td><td>Reserved. Read as 0s.</td></tr>
<tr><td>6:5</td><td>Byte Panning. This field holds a 2-bit value that selects number of bytes (up to 3) by which the image is shifted horizontally to the left on the screen. This function is available in both text and graphics modes.

In text modes with a 9-pixel wide character box, the image can be shifted up to 27 pixels to the left, in increments of 9 pixels. In text modes with an 8-pixel wide character box, and in all standard VGA graphics modes, the image can be shifted up to 24 pixels to the left, in increments of 8 pixels. When the Nine dot disable bit of the VGA control register is set, the pixel shift will be equivalent to the 8-dot mode.

The image can be shifted still further, in increments of individual pixels, through the use of bits [3:0] of the Horizontal Pixel Panning Register (AR13).
<table>
<tr><th colspan="3">Number of Pixels Shifted</th></tr>
<tr><th>Bit [6:5]</th><th>9-Pixel Text</th><th>8-Pixel Text & Graphics</th></tr>
<tr><td>00</td><td>0</td><td>0</td></tr>
<tr><td>01</td><td>9</td><td>8</td></tr>
<tr><td>10</td><td>18</td><td>16</td></tr>
<tr><td>11</td><td>27</td><td>24</td></tr>
</table>
</td></tr>
<tr><td>4:0</td><td>Starting Row Scan Count. This field specifies which horizontal line of pixels within the character boxes of the characters used on the top-most row of text on the display will be used as the top-most scan line. The horizontal lines of pixels of a character box are numbered from top to bottom, with the top-most line of pixels being number 0. If a horizontal line of the these character boxes other than the top-most line is specified, then the horizontal lines of the character box above the specified line of the character box will not be displayed as part of the top-most row of text characters on the display. Normally, the value specified by these 5 bits should be 0, so that all of the horizontal lines of pixels within these character boxes will be displayed in the top-most row of text, ensuring that the characters in the top-most row of text do not look as though they have been cut off at the top.</td></tr>
</table>

CR09 - Maximum Scan Line Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=09h)

Default: 00h

Attributes: Read/Write

Bit	Description
7	**Double Scanning Enable.** 0 = Disable. When disabled, the clock to the row scan counter is equal to the horizontal scan rate. This is the normal setting for many of the standard VGA modes. 1 = Enable. When enabled, the clock to the row scan counter is divided by 2. This is normally used to allow CGA-compatible modes that have only 200 scan lines of active video data to be displayed as 400 scan lines (each scan line is displayed twice).
6	**Line Compare Bit 9.** This bit provides the most significant bit of a 10-bit value that specifies the scan line at which the memory address counter restarts at the value of 0. Bit 4 of the Overflow Register (CR07) supplies the second most significant bit, and bits 7-0 of the Line Compare Register (CR18) supply the 8 least significant bits. Normally, this 10-bit value is set to specify a scan line after the last scan line of the active display area. When this 10-bit value is set to specify a scan line within the active display area, it causes that scan line and all subsequent scan lines in the active display area to display video data starting at the very first byte of the frame buffer. The result is what appears to be a screen split into a top and bottom part, with the image in the top part being repeated in the bottom part. When used in cooperation with the Start Address High Register (CR0C) and the Start Address Low Register (CR0D), it is possible to create a split display, as described earlier, but with the top and bottom parts displaying different data. The top part will display whatever data exists in the frame buffer starting at the address specified in the two aforementioned start address registers, while the bottom part will display whatever data exists in the frame buffer starting at the first byte of the frame buffer.
5	**Vertical Blanking Start Bit 9.** The vertical blanking start is a 10-bit value that specifies the beginning of the vertical blanking period relative to the beginning of the active display area. The 8 least significant bits of this value are supplied by bits [7:0] of the Vertical Blanking Start Register (CR15), and the most and second-most significant bits are supplied by this bit and bit 3 of the Overflow Register (CR07), respectively. This 10-bit value should be programmed to be equal to the number of scan line from the beginning of the active display area to the beginning of the blanking period. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which the vertical blanking period begins.
4:0	**Starting Row Scan Count.** This field provides all 5 bits of a 5-bit value that specifies the number of scan lines in a horizontal row of text. This value should be programmed to be equal to the number of scan lines in a horizontal row of text, minus 1.

CR0A - Text Cursor Start Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Ah)

Default: 00UU UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7:6	**Reserved.** Read as 0.
5	**Text Cursor Off.** This text cursor exists only in text modes, so this register is entirely ignored in graphics modes. 0 = Enables the text cursor. 1 = Disables the text cursor.
4:0	**Text Cursor Start.** This field specifies which horizontal line of pixels in a character box is to be used to display the first horizontal line of the cursor in text mode. The horizontal lines of pixels in a character box are numbered from top to bottom, with the top-most line being number 0. The value specified by these 5 bits should be the number of the first horizontal line of pixels on which the cursor is to be shown.

CR0B - Text Cursor End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Bh)

Default: 0UUU UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**Reserved.** Read as 0.
6:5	**Text Cursor Skew.** This field specifies the degree to which the start and end of each horizontal line of pixels making up the cursor is delayed to compensate for internal pipeline delays. These 2 bits describe the delay in terms of a number character clocks. **Bit [6:5]** Amount of Delay 00 = No delay 01 = Delayed by 1 character clock 10 = Delayed by 2 character clocks 11 = Delayed by 3 character clocks
4:0	**Text Cursor End.** This field specifies which horizontal line of pixels in a character box is to be used to display the last horizontal line of the cursor in text mode. The horizontal lines of pixels in a character box are numbered from top to bottom, with the top-most line being number 0. The value specified by these 5 bits should be the number of the last horizontal line of pixels on which the cursor is to be shown.

CR0C - Start Address High Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Ch)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Start Address Bits [15:8].**This register provides either bits 15 through 8 of a 16-bit value that specifies the memory address offset from the beginning of the frame buffer at which the data to be shown in the active display area begins. (default is 0) In standard VGA modes, the start address is specified with a 16-bit value. The eight bits of this register provide the eight most significant bits of this value, while the eight bits of the Start Address Low Register (CR0D) provide the eight least significant bits.

CR0D - Start Address Low Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Dh)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Start Address Bits [7:0]** This register provides either bits 7 through 0 of a 16 bit value that specifies the memory address offset from the beginning of the frame buffer at which the data to be shown in the active display area begins. (default is 0) In standard VGA modes the start address is specified with a 16-bit value. The eight bits of the Start Address High Register (CR0C) provide the eight most significant bits of this value, while the eight bits of this register provide the eight least significant bits.

CR0E - Text Cursor Location High Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Eh)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Text Cursor Location Bits [15:8].** This field provides the 8 most significant bits of a 16-bit value that specifies the address offset from the beginning of the frame buffer at which the text cursor is located. Bit 7:0 of the Text Cursor Location Low Register (CR0F) provide the 8 least significant bits.

CR0F - Text Cursor Location Low Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=0Fh)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Text Cursor Location Bits [7:0].** This field provides the 8 least significant bits of a 16-bit value that specifies the address offset from the beginning of the frame buffer at which the text cursor is located. Bits 7:0 of the Text Cursor Location High Register (CR0E) provide the 8 most significant bits.

CR10 - Vertical Sync Start Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=10h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Vertical Sync Start Bits [7:0].** This register provides the 8 least significant bits of a 10-bit that specifies the beginning of the vertical sync pulse relative to the beginning of the active display area of a screen. In standard VGA modes, this value is described in 10 bits with bits [7,2] of the Overflow Register (CR07) supplying the 2 most significant bits. This 10-bit value should equal the vertical sync start in terms of the number of scan lines from the beginning of the active display area to the beginning of the vertical sync pulse. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which the vertical sync pulse begins.

CR11 - Vertical Sync End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=11h)

Default: 0U00 UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**Protect Registers [0:7].** The ability to write to Bit 4 of the Overflow Register (CR07) is not affected by this bit (i.e., bit 4 of the Overflow Register is always writeable). 0 = Enable writes to registers CR[00:07]. (default) 1 = Disable writes to registers CR[00:07].
6	**Reserved.** In the VGA standard, this bit was used to switch between 3 and 5 frame buffer refresh cycles during the time required to draw each horizontal line.
5	**Vertical Interrupt Enable.** This bit is reserved for compatibility only. While this bit may be written or read, it's value will have no effect. VGA does not provide an interrupt signal which would be connected to an input of the system's interrupt controller. Bit 7 of Input Status Register 0 (ST00) originally indicated the status of the vertical retrace interrupt. 0 = Enable the generation of an interrupt at the beginning of each vertical retrace period. 1 = Disable the generation of an interrupt at the beginning of each vertical retrace period.
4	**Vertical Interrupt Clear.** This is reserved for compatibility only. VGA does not provide an interrupt signal which would be connected to an input of the system's interrupt controller. 0 = Setting this bit to 0 clears a pending vertical retrace interrupt. This bit must be set back to 1 to enable the generation of another vertical retrace interrupt.
3:0	**Vertical Sync End.** This 4-bit field provides a 4-bit value that specifies the end of the vertical sync pulse relative to its beginning. This 4-bit value should be set to the least significant 4 bits of the result of adding the length of the vertical sync pulse in terms of the number of scan lines that occur within the length of the vertical sync pulse to the value that specifies the beginning of the vertical sync pulse (see the description of the Vertical Sync Start Register for more details).

CR12 - Vertical Display Enable End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=12h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Vertical Display Enable End Bits [7:0].** This register provides the 8 least significant bits of a 10-bit value that specifies the number of the last scan line within the active display area. In standard VGA modes, this value is described in 10 bits with bits [6,1] of the Overflow Register (CR07) supplying the two most significant bits. This 10-bit value should be programmed to be equal to the number of the last scan line within in the active display area. Since the active display area always starts on the 0th scan line, this number should be equal to the total number of scan lines within the active display area,minus 1.

CR13 - Offset Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=13h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Offset Bits [7:0].** This register provides either all 8 bits of an 8-bit value that specifies the number of words or DWords of frame buffer memory occupied by each horizontal row of characters. Whether this value is interpreted as the number of words or DWords is determined by the settings of the bits in the Clocking Mode Register (SR01). In standard VGA modes, the offset is described with an 8-bit value, all the bits of which are provided by this register. This 8-bit value should be programmed to be equal to either the number of words or DWords (depending on the setting of the bits in the Clocking Mode Register, SR01) of frame buffer memory that is occupied by each horizontal row of characters.

CR14 - Underline Location Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=14h)

Default: 0UUU UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**Reserved.** Read as 0.
6	**DWord Mode.** 0 = Frame buffer addresses are interpreted by the frame buffer address decoder as being either byte addresses or word addresses, depending on the setting of bit 6 of the CRT Mode Control Register (CR17). 1 = Frame buffer addresses are interpreted by the frame buffer address decoder as being DWord addresses, regardless of the setting of bit 6 of the CRT Mode Control Register (CR17). This bit is used in conjunction with bits 6 and 5 of the CRT Mode Control Register (CR17) to select how frame buffer addresses from the CPU are interpreted by the frame buffer address decoder as shown below: **CR14[6] / CR17[6] / Addressing Mode** 0 / 0 / Word Mode 0 / 1 / Byte Mode 1 / 0 / DWord Mode 1 / 1 / DWord Mode
5	**Count By 4.** 0 = The memory address counter is incremented either every character clock or every other character clock, depending upon the setting of bit 3 of the CRT Mode Control Register. 1 = The memory address counter is incremented either every 4 character clocks or every 2 character clocks, depending upon the setting of bit 3 of the CRT Mode Control Register. . This is used in mode x13 to allow for using all four planes. This bit is used in conjunction with bit 3 of the CRT Mode Control Register (CR17) to select the number of character clocks are required to cause the memory address counter to be incremented as shown, below: **CR14[5] / CR17[3] / Addressing Incrementing Interval** 0 / 0 / every character clock 0 / 1 / every 2 character clocks 1 / 0 / every 4 character clocks 1 / 1 / every 2 character clocks

Bit	Description
4:0	**Underline Location.** This field specifies which horizontal line of pixels in a character box is to be used to display a character underline in text mode. The horizontal lines of pixels in a character box are numbered from top to bottom, with the top-most line being number 0. The value specified by these 5 bits should be the number of the horizontal line on which the character underline mark is to be shown.

CR15 - Vertical Blanking Start Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=15h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Vertical Blanking Start Bits [7:0].** This register provides the 8 least significant bits of a 10-bit value that specifies the beginning of the vertical blanking period relative to the beginning of the active display area of the screen. In standard VGA modes, the vertical blanking start is specified with a 10-bit value. The most and second-most significant bits of this value are supplied by bit 5 of the Maximum Scan Line Register (CR09) and bit 3 of the Overflow Register (CR07), respectively. This 10-bit value should be programmed to be equal the number of scan lines from the beginning of the active display area to the beginning of the vertical blanking period. Since the active display area always starts on the 0th scan line, this number should be equal to the number of the scan line on which vertical blanking begins.

CR16 - Vertical Blanking End Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=16h)

Default: Undefined

Attributes: Read/Write

This register provides a 8-bit value that specifies the end of the vertical blanking period relative to its beginning.

Bit	Description
7:0	**Vertical Blanking End Bits [7:0].** This 8-bit value should be set equal to the least significant 8 bits of the result of adding the length of the vertical blanking period in terms of the number of scan lines that occur within the length of the vertical blanking period to the value that specifies the beginning of the vertical blanking period (see the description of the Vertical Blanking Start Register for details).

CR17 - CRT Mode Control

I/O (and Memory Offset) Address: 3B5h/3D5h (index=17h)

Default: 0UU0 UUUUb (U=Undefined)

Attributes: Read/Write

Bit	Description
7	**CRT Controller Reset.** This bit has no effect except in native VGA modes (non-centered). 0 = Forces horizontal and vertical sync signals to be inactive. No other registers or outputs are affected. 1 = Permits normal operation.
6	**Word Mode or Byte Mode.** 0 = The memory address counter's output bits are shifted by 1 bit position before being passed on to the frame buffer address decoder such that they are made into word-aligned addresses when bit 6 of the Underline Location Register (CR17) is set to 0. 1 = The memory address counter's output bits remain unshifted before being passed on to the frame buffer address decoder such that they remain byte-aligned addresses when bit 6 of the Underline Location Register (CR17) is set to 0. This bit is used in conjunction with bits 6 and 5 of the CRT Mode Control Register (CR17) to control how frame buffer addresses from the memory address counter are interpreted by the frame buffer address decoder as shown below: CR14[6] CR17[6] **Address Mode** 0 0 Word Mode - Addresses from the memory address counter are shifted once to become word-aligned 0 1 Byte Mode - Addresses from the memory address counter are not shifted 1 0 DWord Mode - Addresses from the memory address counter are shifted twice to become DWord-aligned 1 1 DWord Mode - Addresses from the memory address counter are shifted twice to become DWord-aligned
5	**Address Wrap.** This bit is only effective when word mode is made active by setting bit 6 in both the Underline Location Register and this register to 0. 0 = Wrap frame buffer address at 16 KB. This is used in CGA-compatible modes. 1 = No wrapping of frame buffer addresses.
4	**Reserved.** Read as 0.

Bit	Description
3	**Count By 2.** This bit is used in conjunction with bit 5 of the Underline Location Register (CR14) to select the number of character clocks are required to cause the memory address counter to be incremented. 0 = The memory address counter is incremented either every character clock or every 4 character clocks, depending upon the setting of bit 5 of the Underline Location Register. 1 = The memory address counter is incremented either every other clock. CR14[5] CR17[3] Address Incrementing interval 0 0 every character clock 0 1 every 2 character clocks 1 0 every 4 character clocks 1 1 every 2 character clocks
2	**Horizontal Retrace Select.** This bit provides a way of effectively doubling the vertical resolution by allowing the vertical timing counter to be clocked by the horizontal retrace clock divided by 2 (usually, it would be undivided). 0 = The vertical timing counter is clocked by the horizontal retrace clock. 1 = The vertical timing counter is clocked by the horizontal retrace clock divided by 2.
1	**Select Row Scan Counter.** 0 = A substitution takes place, where bit 14 of the 16-bit memory address generated of the memory address counter (after the stage at which these 16 bits may have already been shifted to accommodate word or DWord addressing) is replaced with bit 1 of the row scan counter at a stage just before this address is presented to the frame buffer address decoder. 1 = No substitution takes place. See following tables.
0	**Compatibility Mode Support.** 0 = A substitution takes place, where bit 13 of the 16-bit memory address generated of the memory address counter (after the stage at which these 16 bits may have already been shifted to accommodate word or DWord addressing) is replaced with bit 0 of the row scan counter at a stage just before this address is presented to the frame buffer address decoder. 1 = No substitution takes place. See following tables.

The following tables show the possible ways in which the address bits from the memory address counter can be shifted and/or reorganized before being presented to the frame buffer address decoder. First, the address bits generated by the memory address counter are reorganized, if need be, to accommodate byte, word or DWord modes. The resulting reorganized outputs (MAOut15-MAOut0) from the memory address counter may also be further manipulated with the substitution of bits from the row scan counter (RSOut1 and RSOut0) before finally being presented to the input bits of the frame buffer address decoder (FBIn15-FBIn0).

Memory Address Counter Address Bits [15:0]

	Byte Mode CR14 bit 6=0 CR17 bit 6=1 CR17 bit 5=X	Word Mode CR14 bit 6=0 CR17 bit 6=0 CR17 bit 5=1	Word Mode CR14 bit 6=0 CR17 bit 6=0 CR17 bit 5=0	DWord Mode CR14 bit 6=1 CR17 bit 6=X CR17 bit 5=X
MAOut0	0	15	13	12
MAOut1	1	0	0	13
MAOut2	2	1	1	0
MAOut3	3	2	2	1
MAOut4	4	3	3	2
MAOut5	5	4	4	3
MAOut6	6	5	5	4
MAOut7	7	6	6	5
MAOut8	8	7	7	6
MAOut9	9	8	8	7
MAOut10	10	9	9	8
MAOut11	11	10	10	9
MAOut12	12	11	11	10
MAOut13	13	12	12	11
MAOut14	14	13	13	12
MAOut15	15	14	14	13

X = Don't Care

Frame Buffer Address Decoder

	CR17 bit 1=1	CR17 bit 1=1	CR17 bit 1=0	CR17 bit 1=0
	CR17 bit 0=1	CR17 bit 0=0	CR17 bit 0=1	CR17 bit 0=0
FBIn0	MAOut0	MAOut0	MAOut0	MAOut0
FBIn1	MAOut1	MAOut1	MAOut1	MAOut1
FBIn2	MAOut2	MAOut2	MAOut2	MAOut2
FBIn3	MAOut3	MAOut3	MAOut3	MAOut3
FBIn4	MAOut4	MAOut4	MAOut4	MAOut4
FBIn5	MAOut5	MAOut5	MAOut5	MAOut5
FBIn6	MAOut6	MAOut6	MAOut6	MAOut6
FBIn7	MAOut7	MAOut7	MAOut7	MAOut7
FBIn8	MAOut8	MAOut8	MAOut8	MAOut8
FBIn9	MAOut9	MAOut9	MAOut9	MAOut9
FBIn10	MAOut10	MAOut10	MAOut10	MAOut10
FBIn11	MAOut11	MAOut11	MAOut11	MAOut11

	CR17 bit 1=1	CR17 bit 1=1	CR17 bit 1=0	CR17 bit 1=0
	CR17 bit 0=1	CR17 bit 0=0	CR17 bit 0=1	CR17 bit 0=0
FBIn12	MAOut12	MAOut12	MAOut12	MAOut12
FBIn13	MAOut13	MAOut13	RSOut0	RSOut0
FBIn14	MAOut14	RSOut1	MAOut14	RSOut1
FBIn15	MAOut15	MAOut15	MAOut15	MAOut15

CR18 - Line Compare Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=18h)

Default: Undefined

Attributes: Read/Write

Bit	Description
7:0	**Line Compare Bits [7:0].** This register provides the 8 least significant bits of a 10-bit value that specifies the scan line at which the memory address counter restarts at the value of 0. Bit 6 of the Maximum Scan Line Register (CR09) supplies the most significant bit, and bit 4 of the Overflow Register (CR07) supplies the second most significant bit. Normally, this 10-bit value is set to specify a scan line after the last scan line of the active display area. When this 10-bit value is set to specify a scan line within the active display area, it causes that scan line and all subsequent scan lines in the active display area to display video data starting at the very first byte of the frame buffer. The result is what appears to be a screen split into a top and bottom part, with the image in the top part being repeated in the bottom part. (This register is only used in split screening modes, and this is not a problem because split screening is not actually used for extended modes. As a result, there is no benefit to extending the existing overflow bits for higher resolutions.) When used in cooperation with the Start Address High Register (CR0C) and the Start Address Low Register (CR0D), it is possible to create a split display, as described earlier, but with the top and bottom parts displaying different data. The top part will display whatever data exists in the frame buffer starting at the address specified in the two aforementioned start address registers, while the bottom part will display whatever data exists in the frame buffer starting at the first byte of the frame buffer.

CR22 - Memory Read Latch Data Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=22h)

Default: 00h

Attributes: Read Only

Bit	Description
7:0	**Memory Read Latch Data.** This field provides the value currently stored in 1 of the four memory read latches. Bits 1 and 0 of the Read Map Select Register (GR04) select which of the four memory read latches may be read via this register.

CR24 - Toggle State of Attribute Controller Register

I/O (and Memory Offset) Address: 3B5h/3D5h (index=24h)

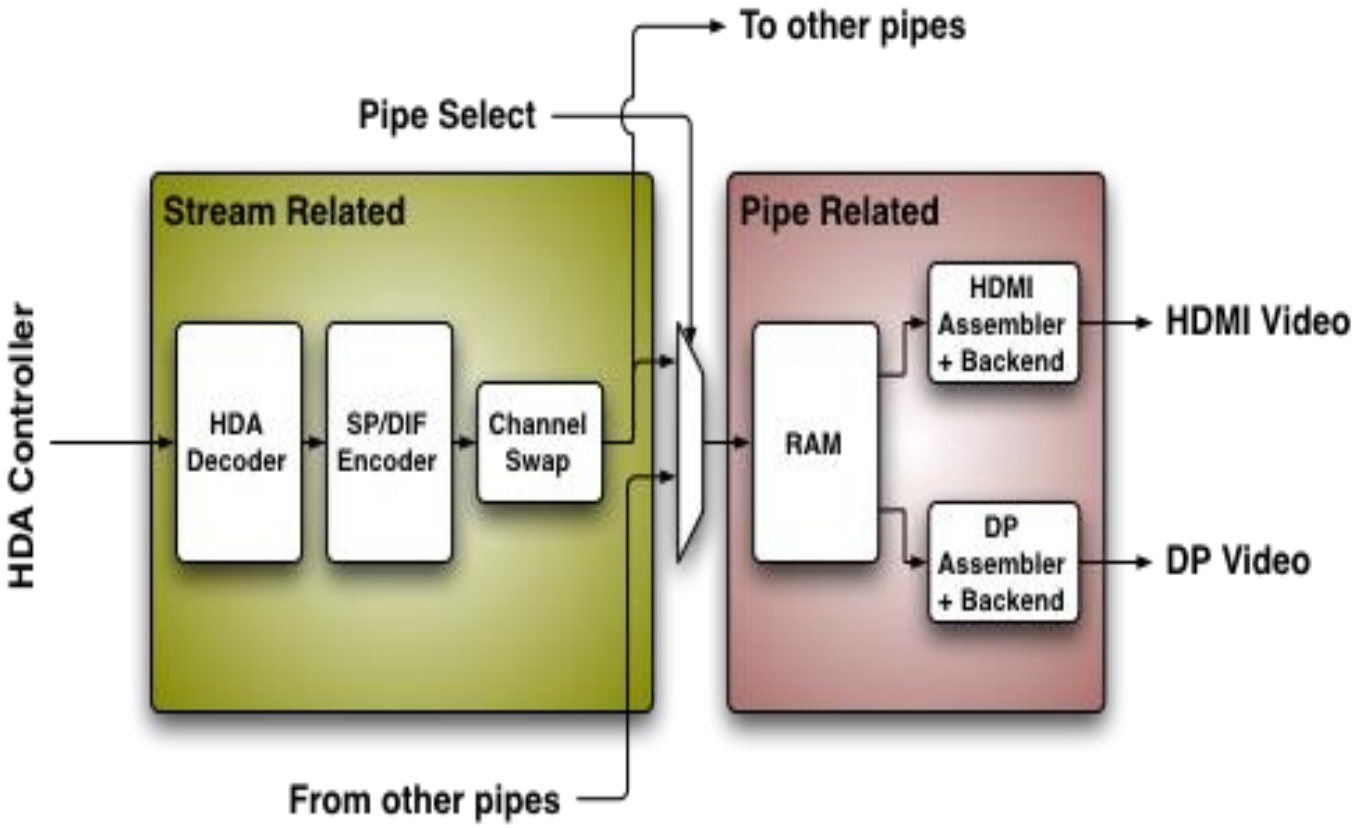
Default: 00h

Attributes: Read Only

Bit	Description
7	**Toggle Status.** Indicates where the last write to attribute register was to: 0 = index port 1 = data port
6:0	**Reserved.** Read as 0.

Display Audio Codec Verbs

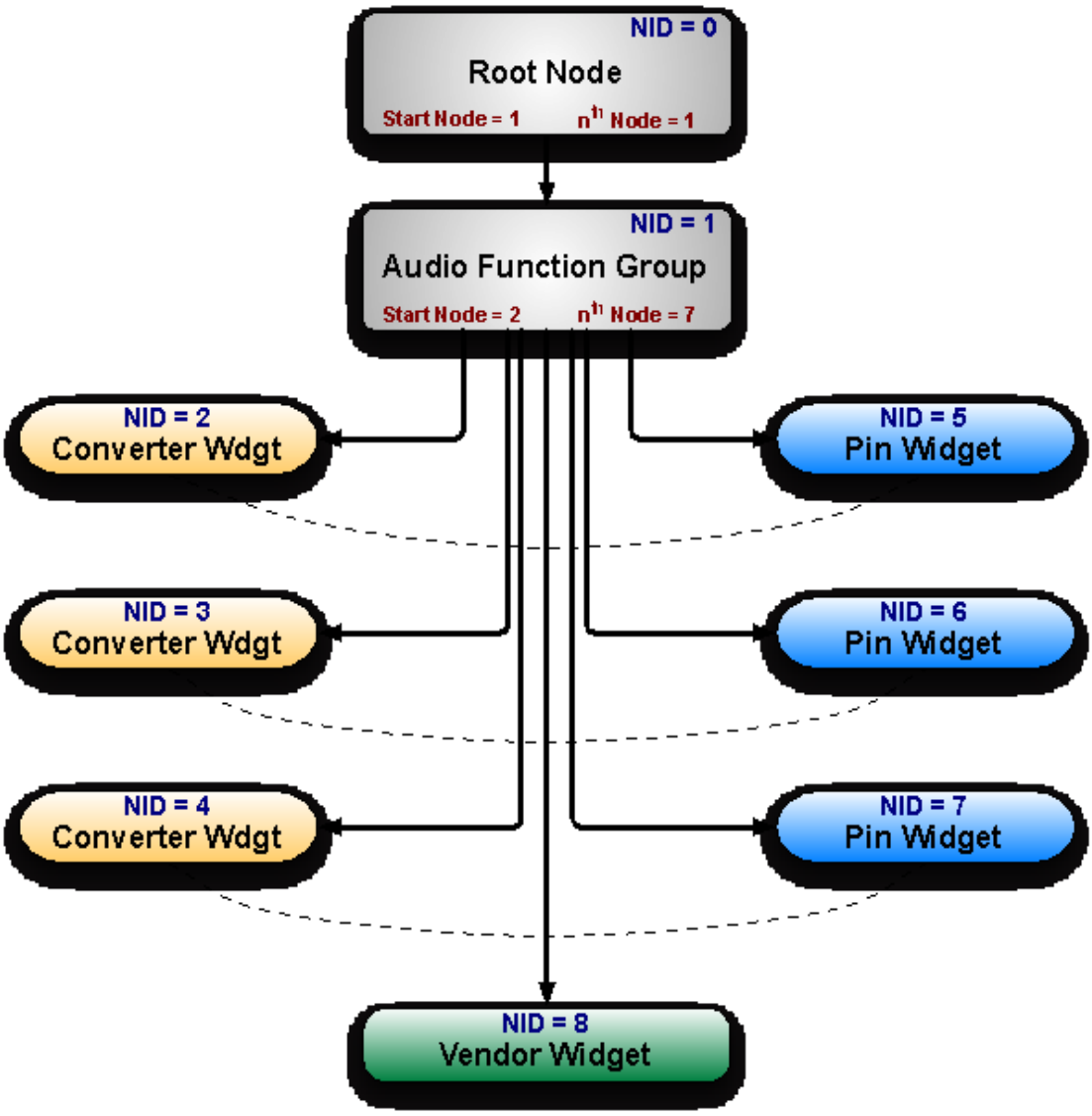
Block Diagram



Codec Node Hierarchy

The diagram below shows the hierarchy of the internal codec. The codec is presented as a single codec with multiple endpoints. By operating as a single codec, only one driver needs to be loaded on the system.

Inside the codec are three "converter widgets" and three "pin widgets", responsible for taking data from HD Audio DMA engines and placing into an HDMI/DP stream. Each pin widget has a 1-1 connection to a converter widget (as indicated by the dotted lines in the diagram).



Programming

Programming of the codec is performed by "verbs" as described in the HD Audio specification. These verbs travel over the internal HD Audio link at a rate of 1 verb per frame. A verb can either come from the CORB, with responses using the RIRB, or using an immediate command and response mechanism (ICR). Device 2 contains its own copy of an ICR mechanism as a back-door into the audio codec.

Verb Support

Verb ID		Verb Name/Description	Node ID							
Set	Get		01h	02h	03h	04h	05h	06h	07h	08h
2h	Ah	Stream Descriptor Format		Y	Y	Y				
3h	Bh	Set Amplifier Mute					Y	Y	Y	
-	F00h	Get Parameters	Y	Y	Y	Y	Y	Y	Y	Y
701h	F01h	Connection Select Control					Y	Y	Y	
-	F02h	Connection List Entry					Y	Y	Y	
705h	F05h	Power State		Y	Y	Y	Y	Y	Y	
706h	F06h	Channel and Stream ID		Y	Y	Y				
707h	F07h	Pin Widget Control					Y	Y	Y	
708h	F08h	Unsolicited Response Enable					Y	Y	Y	Y
-	F09h	Pin Sense					Y	Y	Y	
-	F0Dh	Digital Converter		Y	Y	Y				
70Dh	-	Digital Converter 1		Y	Y	Y				
70Eh	-	Digital Converter 2		Y	Y	Y				
-	F1Ch	Configuration Default					Y	Y	Y	
71Ch	-	Configuration Default Byte 0					Y	Y	Y	
71Dh	-	Configuration Default Byte 1					Y	Y	Y	
71Eh	-	Configuration Default Byte 2					Y	Y	Y	
71Fh	-	Configuration Default Byte 3					Y	Y	Y	
-	F20h	Subsystem ID	Y							
-	F21h	Subsystem ID	Y							
-	F22h	Subsystem ID	Y							
-	F23h	Subsystem ID	Y							
720h	-	Subsystem ID[7: 0]	Y							
721h	-	Subsystem ID[15: 8]	Y							
722h	-	Subsystem ID[23:16]	Y							
723h	-	Subsystem ID[31:24]	Y							
72Dh	F2Dh	Converter Channel Count		Y	Y	Y				
-	F2Eh	HDMI/DP Info Size					Y	Y	Y	
730h	F30h	HDMI Info Index					Y	Y	Y	

Verb ID		Verb Name/Description	Node ID							
Set	Get		01h	02h	03h	04h	05h	06h	07h	08h
731h	F31h	HDMI Info Data					Y	Y	Y	
732h	F32h	HDMI Info Transmit Control					Y	Y	Y	
734h	F34h	Converter Channel Map					Y	Y	Y	
735h	F35h	Device Select					Y	Y	Y	
-	F36h	Display Device List Entry					Y	Y	Y	
73Ch	73Ch	DisplayPort Stream ID					Y	Y	Y	
73Eh	-	Digital Converter 3		Y	Y	Y				
73Fh	-	Digital Converter 4		Y	Y	Y				
-	F80h	HDMI / DP Status								Y
781h	F81h	HDMI Vendor Verb								Y
782h	-	GTC Capture Trigger								Y
-	F83h	Captured Wall Clock Value								Y
-	F84h	Captured GTC Value								Y
-	F85h	Get GTC Offset Value								Y
785h	-	Set GTC Offset Value[7: 0]								Y
786h	-	Set GTC Offset Value[15: 8]								Y
787h	-	Set GTC Offset Value[23:16]								Y
788h	-	Set GTC Offset Value[31:24]								Y
789h	F89h	Converter Channel Count								Y

Parameter Support

Param ID	Parameter Name	Node ID								
		00h	01h	02h	03h	04h	05h	06h	07h	08h
00h	Vendor ID	Y								
02h	Revision ID	Y								
04h	Subordinate Node Count	Y	Y							
05h	Function Group Type		Y							
08h	Audio Function Group Capabilities									
09h	Audio Widget Capabilities			Y	Y	Y	Y	Y	Y	Y
0Ah	Sample Size, Rate CAPs			Y	Y	Y				
0Bh	Stream Formats			Y	Y	Y				
0Ch	Pin Capabilities						Y	Y	Y	
0Dh	Input Amp Capabilities									
0Eh	Connection List Length						Y	Y	Y	
0Fh	Supported Power States		Y							
10h	Processing Capabilities									
11h	GPIO Count									
12h	Output Amp Capabilities						Y	Y	Y	
13h	Volume Knob Capabilities									
15h	Device List Length						Y	Y	Y	

Node ID 00h Root Node Verbs

The root node only contains a single verb - the "Get Parameters" verb at F00h.

F00h - Get Parameters

Parameter	Symbol	Register Name
00h	PARAM_VID	Vendor ID
02h	PARAM_RID	Revision ID
04h	PARAM_SNC	Subordinate Node Count

Parameter 00h: VID - Vendor ID

Bit	Reset	Description
31:16	8086h	**Vendor ID (VID):** Indicates the 16-bit Vendor ID values used to identify the codec to the PnP subsystem.

Parameter 02h: RID - Revision ID

Bit	Reset	Description
31:24	*0*	*Reserved*
23:20	1h	**Major Revision (MJR):** Indicates the major revision number (left of the decimal) of the High Definition Audio Specification to which the codec is fully compliant.
19:16	0h	**Minor Revision (MNR):** Indicates the minor revision number (right of the decimal) or "dot number" of the High Definition Audio Specification to which the codec is fully compliant.
15:08	00h	**Revision ID (RID):** Indicates the vendor's revision number for this given Device ID.
07:00	00h	**Stepping ID (SID):** Indicates optional vendor stepping number within the revision.

Parameter 04h: PARAM_SNC - Subordinate Node Count

Bit	Reset	Description
31:24	*0*	*Reserved*
23:16	0h	**Starting Node Number (SNN):** Indicates the first sub-node's ID is 01h.
15:08	00h	Reserved
07:00	01h	**Total Number of Nodes (TNN):** Indicates one sub-node

F37h GET CCF - Get Current Clock Frequency

Bits	Default	Description
31:6	0	*Reserved*
05	0	**Current Clock 192 MHz (C192)**: Indicates the current clock is 192 MHz. Reserved for Display Codec
04	1	**Current Clock 96 MHz (C96)**: Indicates the current clock is 96 MHz.

Bits	Default	Description
03	0	**Current Clock 48 MHz (C48)**: Indicates the current clock is 48 MHz.
02	0	**Current Clock 24 MHz (C24)**: Indicates the current clock is 24 MHz. Reserved for Display Codec
01	0	**Current Clock 12 MHz (C12)**: Indicates the current clock is 12 MHz. Reserved for Display Codec
00	0	**Current Clock 6 MHz (C6)**: Indicates the current clock is 6 MHz. Reserved for Display Codec

Node ID 01h Audio Function Group Verbs

Set Verb	Get Verb	Symbol	Name
-	F00h	GET PARAM	Get Parameters
705h	F05h	SET_PS / GET_PS	Set Power State
-	F20h	GET_SSID	Get Subsystem ID
720h	F20h	SET_SSID0	Set/Get Subsystem ID
721h	F21h	SET_SSID1	Set/Get Subsystem ID
722h	F22h	SET_SSID2	Set//Get Subsystem ID
723h	F23h	SET_SSID3	Set/Get Subsystem ID
724h	F24h	SET CCF	Set/Get Current Clock Frequency

F00h Get Parameters

Parameter	Symbol	Register Name
04h	PARAM_SNC	Subordinate Node Count
05h	PARAM_FGT	Function Group Type
08h	PARAM_FGC	Function Group Capabilities
0Fh	PARAM SPS	Supported Power States

Parameter 04h: PARAM_SNC - Subordinate Node Count

Bit	Reset	Description
31:24	*0*	*Reserved*
23:16	02h	**Start Node Number (SNN):** Indicates the start node number of widget or functional nodes in the Functional Group.
15:08	*0*	*Reserved*
07:00	07h	**Total Number of Nodes (TNN):** Indicates 7 widgets in the Functional Group. (HDMI/DP converters (3) + HDMI/DP pins (3) + Vendor Defined Widget (1)).

Parameter 05h: PARAM_FGT - Function Group Type

Bit	Reset	Description
31:09	*0*	*Reserved*
08	0	**Unsolicited Capable (UC):** Not capable of generating an unsolicited response.
07:00	01h	**Node Type (NT):** Indicates Audio Function Group.

Parameter 08h: PARAM_FGC - Function Group Capability

Bit	Reset	Description
31:04	*0*	*Reserved*
03:00	00h	**Output Delay (OD)**Output Delay.

Parameter 0Fh: PARAM_SPS - Supported Power States

Bit	Reset	Description
31	1	**Extended Power State Supported (EPSS):** Indicates support for low power states
30	1	**Clock Stop (CS):** Indicates support for D3 when clock is stopped.
29:04	*0*	*Reserved*
03	1	**D3 Supported (D3S):** Indicates support for D3.
02	0	**D2 Supported (D2S):** Indicates no support for D2.
01	0	**D1 Supported (D1S):** Indicates no support for D1.
00	1	**D0 Supported (D0S):** Indicates support for D0.

Parameter 16h: PARAM_A2CAP - Azalia 2 Capabilities

Bits	Reset	Description
31:17	*0*	*Reserved*
16	0	**Independent Codec Clock (ICC):** When set, this indicates that the codec generates its own clock, which may drift from the link clock. When cleared, the codec's clock is locked to the link clock.
15:06	*0*	*Reserved*
05	*0*	*Reserved for 192 MHz support.*
04	1	**96MHz Supported (S96):** Indicates 96 MHz clock is supported.
03	1	**48MHz Supported (S48):** Indicates 48 MHz clock is supported.This bit must always be set.
02	0	**24MHz Supported (S24):** Indicates 24 MHz clock is supported. Reserved for Display Codec
01	0	**12MHz Supported (S12):** Indicates 12 MHz clock is supported. Reserved for Display Codec
00	0	**6MHz Supported (S6)**Indicates 6 MHz clock is supported. Reserved for Display Codec

705h SET_PS - Set Power State

Bits	Description
07:02	Reserved
01:00	**Requested Power State (RPS):** Only D0 (00) and D3 (11) may be requested

F05h GET_PS - Get Power State

Bits	Reset	Description
31:11	0	Reserved
10	0	**Settings Reset (SR):** Haswell does not change the default values.
09	1	**Clock Stop OK (CSOK):** Clock stopping in D3 is OK
08	0	**Error (ERR):** No error will ever be reported.
07:06	0	Reserved
05:04	11	**Actual Power State (APS):** Indicates the current power state of the node.
03:02	0	Reserved
01:00	11	**Requested Power State (CPS):** Reflects value written with SET_PS verb.

F20h GET SSID - Get Subsystem ID0

Bits	Reset	Description
31:00	80860101h	**Subsystem ID (SSID):** Reports the sub-system ID set via SET_SSIDx verbs.

720h SET SSID0 - Set Subsystem ID0

Bits	Description
07:00	Subsystem ID Bits [7:0]

721h SET SSID1 - Set Subsystem ID1

Bits	Description
07:00	Subsystem ID Bits [15:8]

722h SET SSID2 - Set Subsystem ID2

Bits	Description
07:00	Subsystem ID Bits [23:16]

723h SET SSID3 - Set Subsystem ID3

Bits	Description
07:00	Subsystem ID Bits [31:24]

724h SET CCF - Set Current Clock Frequency

Bits	Description
07:06	Reserved
05	**Set Clock 192 MHz (S192)**: Set clock to 192 MHz.
04	**Set Clock 96 MHz (S96)**: Set clock to 96 MHz
03	**Set Clock 48 MHz (S48)**: Set clock to 48 MHz
02	**Set Clock 24 MHz (S24)**: Set clock to 24 MHz
01	**Set Clock 12 MHz (S12)**: Set clock to 12 MHz
00	**Set Clock 6 MHz (S6)**: Set clock to 6 MHz

F24h GET CCF - Get Current Clock Frequency

Bits	Bits	Description
31:06	0	*Reserved*
05	0	**Current Clock 192 MHz (C192)**: Indicates the current clock is 192 MHz.
04	0	**Current Clock 96 MHz (C96)**: Indicates the current clock is 96 MHz.
03	0	**Current Clock 48 MHz (C48)**: Indicates the current clock is 48 MHz.
02	0	**Current Clock 24 MHz (C24)**: Indicates the current clock is 24 MHz.
01	0	**Current Clock 12 MHz (C12)**: Indicates the current clock is 12 MHz.
00	0	**Current Clock 6 MHz (C6)**: Indicates the current clock is 6 MHz.

7FFh SET Function Group Reset

Bits	Reset	Description
07:00	00h	The Function Reset command causes the functional unit, and all widgets associated with the functional unit, to return to their power-on reset values. Note that some controls such as the Configuration Default controls should not be reset with this command. It is also possible that certain other controls, such as Caller-ID, should not be reset. This command does not affect the Link interface logic, which must be reset with the link RST# signal. Therefore, a codec must not initiate a Status Change request on the link.

Node ID 02h 03h 04h Audio Output Convertor Widget Verbs

Verb	Symbol	Verb Name
2h	SET_SDF	Set Stream Descriptor Format
Ah	GET_SDF	Get Stream Descriptor Format
F00h	GET_PARAM	Get Parameters
705h	SET_PS	Set Power State
F05h	GET_PS	Get Power State
706h	SET_CSID	Set Channel and Stream ID
F06h	GET_CSID	Get Channel and Stream ID
F0Dh	SET_DC1	Get Digital Converter
70Dh	SET_DC1	Set Digital Converter 1
70Eh	SET_DC2	Set Digital Converter 2
73Eh	SET_DC3	Set Digital Converter 3
73Fh	SET_DC4	Set Digital Converter 4
72Dh	SET_CCC	Set Converter Channel Count
F2Dh	GET_CCC	Get Converter Channel Count

2hAh SETGET_SDF - SetGET Stream Descriptor Format

Bits	Reset	Description
31:15	0	Reserved
14	0	**Sample Base Rate (SBR):**
13:11	000	**Sample Base Rate Multiplier (SBRM):**
10:08	000	**Sample Base Rate Divisor (SBRD):**
07	0	Reserved

Bits	Reset	Description
06:04	011	**Bits / Sample (BPS):** • 001b: Data is packed in memory in 16 bit containers on 16 bit boundaries • 010b: Data is packed in memory in 20 bit containers on 32 bit boundaries • 011b: Data is packed in memory in 24 bit containers on 24 bit boundaries • 100b: Data is packed in memory in 32 bit containers on 32 bit boundaries All other bit combinations reserved
03:00	1h	**# Channels in Stream (NCS):** 2 channels in each frame

F00h Get Parameters

Parameter	Symbol	Register Name
09h	PARAM_AWC	Audio Widget Capabilities
0Ah	PARAM_PSB	Parameter Sizes and Bit Rates
0Bh	PARAM_SF	Stream Formats
0Fh	PARAM_SPS	Power Supported States

Parameter 09h: AWC - Audio Widget Capabilities

Bits	Reset	Description
31:24	0	Reserved
23:20	0h	**Widget Type (TYPE):** Indicates this is an audio output widget
19:16		Sample Delay in Widget (DELAY):
15:13	011	**Channel Count Extension (CCE):** These three bits, combined with STRO, indicate that there are 8 channels supported.
11	0	**L-R Swap (LRS):** Indicates no left/right channel swap.
10	1	**Power Control (PC):** Indicates power state control
09	1	**Digital (DIG):** Indicates support for digital streams.
08	0	**Connection List (CL):** Indicates no connection list
07	0	**Unsolicited Capable (UC):** Indicates support for unsolicited responses.

Bits	Reset	Description
06	0	**Processing Widget (PW):** Indicates no support for processing
05	0	**Stripe (STRP):** Indicates striping not supported.
04	1	**Format Override (FO):** Indicates support for formatting
03	1	**Amp Parameter Override (APO):** Indicates no amplifier support.
02	0	**Out Amp Present (OAP):** Indicates no output amplifier present.
01	0	**In Amp Present (IAP):** Indicates no input amplifier present.
00	1	**Stereo (STRO):** Indicates a stereo widget

Parameter 0Ah: PSB - PCM Sizes and Bit Rates

Bits	Reset	Description
31:21	0	*Reserved*
20	1	**32-bit Support (B32):** Indicates 32-bit samples supported
19	1	**24-bit Support (B24):** Indicates 24-bit samples supported
18	0	**20-bit Support (B20):** Indicates 20-bit samples supported
17	1	**16-bit Support (B16):** Indicates 16-bit samples supported
16	0	**8-bit Support (B8):** Indicates 8-bit samples not supported
15:12	0	*Reserved*
11	0	**384 kHz Support (R12):** Indicates 384 kHz not supported
10	1	**192 kHz Support (R11):** Indicates 192 kHz supported
09	1	**176.4 kHz Support (R10):** Indicates 176.4 kHz supported
08	1	**96 kHz Support (R9):** Indicates 96 kHz supported
07	1	**88.2 kHz Support (R8):** Indicates 88.2 kHz supported

Bits	Reset	Description
06	1	**48 kHz Support (R7):** Indicates 48 kHz supported
05	1	**44.1 kHz Support (R6):** Indicates 44.1 kHz supported
04	1	**32 kHz Support (R5):** Indicates 32 kHz supported
03	0	**22.05 kHz Support (R4):** Indicates 22.05 kHz not supported
02	0	**16 kHz Support (R3):** Indicates 16 kHz not supported
01	0	**11.025 kHz Support (R2):** Indicates 11.025 kHz not supported
00	0	**8 kHz Support (R1):** Indicates 8 kHz not supported

Parameter 0Bh: SF - Stream Formats

Bits	Reset	Description
31:03	0	*Reserved*
02	1	**AC3 Support (AC3):** Indicates AC3 stream format is supported
01	0	**Float32 Support (F32):** Indicates float32 stream format not supported
00	1	**PCM Support (PCM):** Indicates PCM format is supported.

Parameter 0Fh: PARAM_SPS - Supported Power States

Bit	Reset	Description
31	1	**Extended Power State Supported (EPSS):** Indicates support for low power states
30:04	0	*Reserved*
03	1	**D3 Supported (D3S):** Indicates support for D3.
02	0	**D2 Supported (D2S):** Indicates no support for D2.
01	0	**D1 Supported (D1S):** Indicates no support for D1.
00	1	**D0 Supported (D0S):** Indicates support for D0.

705h SET_PS - Set Power State

Bits	Description
07:02	Reserved
01:00	**Requested Power State (RPS):** Only D0 (00) and D3 (11) may be requested

F05h GET_PS - Get Power State

Bits	Reset	Description
31:11	0	Reserved
10	0	**Settings Reset (SR):** ???
09	0	**Clock Stop OK (CSOK):** Clock stopping in D3 is not OK
08	0	**Error (ERR):** No error will ever be reported.
07:06	0	Reserved
05:04	11	**Actual Power State (APS):** Indicates the current power state of the node.
03:02	0	Reserved
01:00	11	**Requested Power State (CPS):** Reflects value written with SET_PS verb.

706hF06h GETSET_CSID - GetSet Channel and Stream ID

Bits	Reset	Description
07:04	0h	**Stream ID (SID):** Link stream used by the converter for data output.
03:00	0h	**Lowest Channel Number (LCN):** Lowest channel used by the converter.

Digital Converter Verbs

F0Dh: GET_DC - Get Digital Converter

Bits	Reset	Description
31:24	0	Reserved
23	1	**Keep Alive (KA):** See SET_DC3.KA
22:20	0	Reserved
19:16	0h	**IEC Coding Type (ICT):** See SET_DC3.ICT

Bits	Reset	Description
15	0	Reserved
14:08	00h	**Category Code (CC):** See SET_DC1.CC
07	0	**Level (LVL):** See SET_DC1.LVL
06	0	**Professional (PRO):** See SET_DC1.PRO
05	0	**Audio is not PCM (AUDIO):** See SET_DC1.AUDIO
04	0	**Copyright (COPY):** See SET_DC1.COPY
03	0	**Pre-emphasis (PRE):** See SET_DC1.PRE
02	0	**Validity Configuration (VCFG):** See SET_DC1.VCFG
01	0	**Validity (V):** See SET_DC1.V
00	1	**Digital Enable (DIGEN):** See SET_DC1.DIGEN

70Dh: SET_DC1 - Set Digital Converter 1

Bits	Description
07	**Level (LVL):** S/PDIF IEC Generation Level.
06	**Professional (PRO):** When set, indicates professional use of channel.
05	**Audio is not PCM (AUDIO):** When set, data is non-PCM format.
04	**Copyright (COPY):** When set, copyright asserted.
03	**Pre-emphasis (PRE):** When set, enables filter pre-emphasis.
02	**Validity Configuration (VCFG):** Determines S/PDIF transmitter behavior when data is not being transmitted.
01	**Validity (V):** Affects the validity flag transmitted in each sub-frame, and enables S/PDIF transmitter to maintain connection during error or mute conditions.
00	**Digital Enable (DIGEN):** When set, enables digital content

70Eh: Digital Converter 2

Bits	Description
07	Reserved
06:00	**Category Code (CC):** S/PDIF IEC Category Code.

73Eh: Digital Converter 3

Bits	Description
07	Keep Alive
06:04	Reserved
03:00	IEC Coding Type

73Fh: Digital Converter 4

Bits	Description
07:00	Reserved

72DhF2Dh GETSET_CCC - GetSet Converter Channel Count

Bits	Reset	Description
07:04	0	Reserved
03:00	0000	Converter Channel Count 1 (0th order)

Node ID 05h 06h 07h Pin Widget Verbs

Set Verb	Get Verb	Symbol	Verb Name
3h	-	SET_AM	Set Amplifier Mute
-	Bh	GET_AM	Get Amplifier Mute
-	F00h	-	Get Parameters
701h	F01h	SET_CSC / GET_CSC	Set/Get Connection Select Control
-	F02h	-	Get Connection List Entry
705h	F05h	SET_PS / GET_PS	Set/Get Power State
707h	F07h	SET_PWC / GET_PWC	Set/Get Pin Widget Control
708h	F08h	SET_UE / GET_UE	Set/Get Unsolicited Response Enable
-	F09h	-	Get Pin Sense
71Ch	-	SET_CD0	Set Configuration Default Byte 0
71Dh	-	SET_CD1	Set Configuration Default Byte 1
71Eh	-	SET_CD2	Set Configuration Default Byte 2
71Fh	-	SET_CD3	Set Configuration Default Byte 3
-	F1Ch	GET_CD	Get Configuration Default
-	F2Eh	GET_HDIS	Get HDMI/DP Info Size
730h	F30h	SET_HII / GET_HII	Set/Get HDMI Info Index
731h	F31h	SET_HID / GET_HID	Set/Get HDMI Info Data
732h	F32h	SET_HITC / GET_HITC	Set/Get HDMI Info Transmit Control
733h	F33h	SET_PC / GET_PC	Set/Get Protection Control
734h	F34h	SET_CCM / GET_CCM	Set/Get Converter Channel Map
735h	F35h	SET_DS / GET_DS	Set/Get Device Select
-	F36h	GET_DDLE	Get Display Device List Entry
73Ch	F3Ch	SET_DPID / GET_DPID	Set/Get DisplayPort Stream ID

3h SET_AM - Set Amplifier Mute

Bits	Bits	Description
15	0	**Set Output Amp (SOA):**.
14	0	**Set Input Amp (SIA):**.
13	0	**Set Left Amp (SLA):**.
12	0	**Set Right Amp (SRA):**.

Bits	Bits	Description
11:08	0h	**Index (IDX):**
07	1	**Mute (MUTE):** When set, amp muted.
06:00	0	*Reserved*

B8h GET_AM - Get Amplifier Mute

Bits	Bits	Description
31:08	0	*Reserved*
07	1	**Mute (MUTE):** When set, amp muted.
06:00	0	*Reserved*

F00h Get Parameters

Parameter	Symbol	Register Name
09h	PARAM_AWC	Audio Widget Capabilities
0Ch	PARAM_PC	Pin Capabilities
0Eh	PARAM_CLL	Connection List Length
12h	PARAM OAC	Output Amplifier Capabilities
15h	PARAM_DLL	Device List Length
0Fh	PARAM_SPS	Supported Power States

Parameter 09h: AWC - Audio Widget Capabilities

Bits	Reset	Description
31:24	0	Reserved
23:20	4h	**Widget Type (TYPE):** Indicates this is a pin complex widget
19:16	0	**Sample Delay in Widget (DELAY):** No delay through the pin widget.
15:13	011	**Channel Count Extension (CCE):** This field, combined with STRO, indicate 8 channels supported.
11	0	**L-R Swap (LRS):** Indicates no left/right channel swap.

Bits	Reset	Description
10	1	**Power Control (PC):** Indicates power state control
09	1	**Digital (DIG):** Indicates support for digital streams.
08	1	**Connection List (CL):** Indicates a connection list
07	1	**Unsolicited Capable (UC):** Indicates support for unsolicited responses.
06	0	**Processing Widget (PW):** Indicates no support for processing
05	0	**Stripe (STRP):** Indicates striping not supported.
04	0	**Format Override (FO):** Indicates no support for formatting
03	1	**Amp Parameter Override (APO):** Indicates no amplifier override support.
02	1	**Out Amp Present (OAP):** Indicates no output amplifier present.
01	0	**In Amp Present (IAP):** Indicates no input amplifier present.
00	1	**Stereo (STRO):** Indicates a stereo widget

Parameter 0Ch: PC - Pin Capabilities

Bits	Reset	Description
31:28	0	*Reserved*
27	1	**High Bit Rate (HBR):** Indicates support for high bit-rate audio
26	0	*Reserved*
25	1	**Multi Stream Capable (MSC):** Indicates support for DisplayPort Multistream. Will be zero in vanilla mode.
24	1	**DisplayPort (DP):** Indicates support for DisplayPort
23:08	0	*Reserved*
07	1	**HDMI (HDMI):** Indicates support for HDMI
06:05	0	*Reserved*

Bits	Reset	Description
04	1	**Output Capable (OC):** Pin is output capable
03	0	*Reserved*
02	1	**Presence Detect Capable (PDC):** Indicates capability for presence detection
01:00	0	*Reserved*

Parameter 0Eh: CLL - Connection List Length

Bits	Reset	Description
31:08	0	*Reserved*
07	0	**Long Form (LF):** Indicates connection list is short form
06:00	03h	**Length (LEN):** Indicates there is one item in the connection list.

Parameter 12h: OAC - Output Amplifier Capabilities

Bits	Reset	Description
31	1	**Mute Capable (MC):** Muting is capable on this pin
30:00	0	Reserved

Parameter 15h: DLL - Device List Length

Bits	Reset	Description
31:06	0	*Reserved*
05:00	00h	**Length (LEN):** Indicates no devices

Parameter 0Fh: PARAM_SPS - Supported Power States

Bit	Reset	Description
31	1	**Extended Power State Supported (EPSS):** Indicates support for low power states
30:04	0	*Reserved*
03	1	**D3 Supported (D3S):** Indicates support for D3.

Bit	Reset	Description
02	0	**D2 Supported (D2S):** Indicates no support for D2.
01	0	**D1 Supported (D1S):** Indicates no support for D1.
00	1	**D0 Supported (D0S):** Indicates support for D0.

701hF01h SETGET_CSC - SetGet Connection Select Control

Bits	Reset	Description
07:00	00h	**Connection Select Control (CSC):**

F02h GET_CLE - Get Connection List Entry

Bits	Reset	Description
31:08	0	Reserved
07:00	Varies	**Connection List Entry (CLE):** 02h for NodeID 05h, 03h for NodeID 06h, and 04h for NodeID 07h.

705h SET_PS - Set Power State

Bits	Description
07:02	Reserved
01:00	**Requested Power State (RPS):** Only D0 (00) and D3 (11) may be requested

F05h GET_PS - Get Power State

Bits	Reset	Description
31:11	0	Reserved
10	0	**Settings Reset (SR):** Haswell does not change the default values.
09	0	**Clock Stop OK (CSOK):** Clock stopping in D3 is not OK
08	0	**Error (ERR):** No error will ever be reported.
07:06	0	Reserved
05:04	11	**Actual Power State (APS):** Indicates the current power state of the node.
03:02	0	Reserved

Bits	Reset	Description
01:00	11	**Requested Power State (CPS):** Reflects value written with SET_PS verb.

707hF07h SETGET_PWC - SetGet Pin Widget Control

Bits	Reset	Description
07	0	Reserved
06	1	**Out Enable (OE):** When set, the audio is enabled
05:02	0	Reserved
01:00	00	**Encoded Packet Type (EPT):**

708hF08h SETGET_UE - SetGet Unsolicited Enable

Bits	Description
07	**Unsolicited Enable (UE):** When set, unsolicited responses are allowed
06	Reserved
05:00	**Tag (TAG):**

F09h GET_PS - Get Pin Sense

Bits	Reset	Description
31	0	**Presence Detect (PD):** When set presence is detected on this pin.
30	0	**ELD Value (ELDV):**
29	0	**Inactive (INA):**
28:00	28:00	Reserved

71Ch SET_CD0 - Set Configuration Default Byte 0

Bits	Description
07:04	Default Association (DA):
03:00	Sequence (SEQ):

71Dh SET_CD1 - Set Configuration Default Byte 1

Bits	Description
07:04	Color (COL):
03:00	Miscellaneous (MISC):

71Eh SET_CD2 - Set Configuration Default Byte 2

Bits	Description
07:04	Default Device (DD):
03:00	Connection Type (CT):

71Fh SET_CD3 - Set Configuration Default Byte 3

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>07:06</td><td>Port Connectivity (PC): External connectivity of the pin complex.
00 = Connected to jack
01 = No physical connection
10 = Fixed function device (integrated speaker, mic, etc.)
11 = Both a jack and internal connection</td></tr>
<tr><td>05:00</td><td>Location (LOC):
<table>
<tr><td></td><td colspan="10">Bits 3:0</td></tr>
<tr><td>Bits 5:4</td><td>0h: N/A</td><td>1h: Rear</td><td>2h: Front</td><td>3h: Left</td><td>4h: Right</td><td>5h: Top</td><td>6h: Bottom</td><td>7h: Special</td><td>8h: Special</td><td>9h: Special</td><td>Ah-Fh Reserved</td></tr>
<tr><td>00: External</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td></tr>
<tr><td>01: Internal</td><td>Y</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Y</td><td>Y</td><td>Y</td><td></td></tr>
<tr><td>10: Separate Chassis</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td><td></td><td></td></tr>
<tr><td>11: Other</td><td>Y</td><td></td><td></td><td></td><td></td><td></td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td></tr>
</table></td></tr>
</table>

F1Ch GET_CD - Get Configuration Default

Bits	Description
31:30	**Port Connectivity (PC):** See SET_CD3.PC
29:24	**Location (L):** See SET_CD3.L
23:20	**Default Device (DD):** See Set_CD2.DD
19:16	**Connection Type (CT):** See Set_CD2.CT
15:12	**Color (COL):** See SET_CD1.COL
11:08	**Miscellaneous (MISC):** See SET_CD1.MISC
07:04	**Default Association (DA):** See SET_CD0.DA
03:00	**Sequence (SEQ):** See SET_CD0.SEQ

F2Eh HDMIDP Info Size

Bits	Reset	Description
31:08	0	Reset
07:00	Varies	**Size (SZ):** Indexes 0 - 3 return 1Eh, index 1000 returns 53h, others reserved.

F2Fh Get ELD Data

Parameter	Symbol	Register Name
07:0h	PARAM_INDX	ELD DATA Index

Parameter nn: ELD Data

Bits	Reset	Description
31:00	0	ELD Data

730hF30h SETGET_HII - SetGet HDMI Info Index

Bits	Reset	Description
07:05	000	**Infoframe Packet Index (IPI):** Value / Name / Value / Name 000 Audio 011 GP3 001 GP 100 GP4 010 GP2 Others Reserved
04:00	00h	**Byte Offset Index Pointer (BOI):**

731hF31h SETGET_HID - SetGet HDMI Info Data

Bits	Reset	Description
07:00	00h	**Data (DATA):** Data at current index pointed to from SET_HII verb.

732hF32h SETGET_HITC - SetGet HDMI Info Transmit Control

Bits	Reset	Description
07:06	00	**InfoFrame Control Current Indexed Frame (IFCCIF):** • 00 = Disable Transmit • 01 = Reserved • 10 = Transmit Once • 11 = Best Effort
05:00	0	*Reserved*

733h SET_PC - Set Protection Control

Bits	Description
07:03	**Unsolicited Response Sub Tag (URST):** Subtag to use for unsolicited responsed.
02	Reserved

734hF34h SETGET_CCM - GetSet Converter Channel Map

Bits	Reset	Description
07:04	0h	Converter Channel (CC):
03:00	0h	Slot (SN):

735h SET_DS - Set Device Select

Bits	Reset	Description
07:06	0	*Reserved*
05:00	00h	**Device (D):** 000001, 000010 (based upon number of devices present)

F35h: GET_DS - Get Device Select

Bits	Description
31:12	**Reserved:** Set to 0
11:06	**SinK Device ID:** Sink Device ID in the multi stream topology of the DP hierarchy. Device attached to Pipe A will have ID of "00000", PipeB will have "00001" and Pipe C will have "00010".
05:00	**Device (D):**Device Entry index currently set

F36h GET_DDLE - Get Display Device List Entry

Bits	Bits	Description
31:12	0	*Reserved*
11	0	**Reserved?**
10	0	**IA of Entry 2**
09	0	**ELDV of Entry 2**
08	0	**PD of Entry 2**
07	0	**Reserved?**
06	0	**IA of Entry 1**
05	0	**ELDV of Entry 1**

Bits	Bits	Description
04	0	**PD of Entry 1**
03	0	**Reserved?**
02	0	**IA of Entry 0**
01	0	**ELDV of Entry 0**
00	0	**PD of Entry 0**

73ChF3Ch SETGET_DPID - SetGet DisplayPort Stream ID

Bits	Reset	Description
07:03	00h	**Tag (TAG):** Represents the SSID that will go in the lower 5 bits of the SSID
02:00	000	**Index (IDX):** Pointer to program multiple SSID

Node ID 08h Intel Vendor Widget Verbs

Set Verb	Get Verb	Symbol	Verb Name
-	F00h	GET_PARAM	Get Parameters
-	F80h	GET_HDPS	Get HDMI/DP Status
781h	F81h	SET_HVV / GET_HVV	Set/Get iDisp Codec Vendor Verb
782h	-	SET_GTCT	Set GTC Trigger
-	F83h	GET_CWC	Get Captured Wall Clock
	F84h	GET_CGTC	Get Captured GTC Value
	F85h	GET_GOF	Get GTC Offset Value
785h	-	SET_GOF0	Set GTC Offset Value Byte 0
786h	-	SET_GOF1	Set GTC Offset Value Byte 1
787h	-	SET_GOF2	Set GTC Offset Value Byte 2
788h	-	SET_GOF3	Set GTC Offset Value Byte 3
789h	F89h	SET_GDI / GET_GDI	Set/Get GTC Offset Device Index

F00h Get Parameters

Parameter	Symbol	Register Name
09h	PARAM_AWC	Audio Widget Capabilities

Parameter 09h: AWC - Audio Widget Capabilities

Bits	Reset	Description
31:24	0	Reserved
23:20	Fh	**Widget Type (TYPE):** Indicates this is a vendor defined widget
19:00	0	Reserved

71Eh SET_GET_GFXMAILBOX - Set Get GFX MAILBOX Byte 2

71Eh: SET GFX MAILBOXM

Bits	Default	Description
07:00	00	Contents to be defined by GFX driver and audio driver

F1Eh: GET GFX MAILBOX

Bits	Default	Description
31:24	00h	Other values of 71F, verbs
23:16	00h	Contents to be defined by GFX driver and audio driver
15:00	0000h	Other values of 71D, 71C verb

728h SET CLOCK OFF - Set Clock Off Command

Bits	Description
07:00	Data is Irrelevant

708hF08h SETGET_UE - SetGet Unsolicited Enable

Bits	Description
07	**Unsolicited Enable (UE):** When set, unsolicited responses from GFX MAIL BOX register writes are allowed.
06	Reserved
05:00	**Tag (TAG):** Tag for GFX Mail box register unsol responses.

781hF81h GETSET_VV - GetSet iDisp Codec Vendor Verb

Bits	Bits	Description
07:06	0	**Port Select:** This filed is programmed to select the port to be exposed to the inbox driver in the vanilla mode. Will not reset on double function group reset. 00 -> Port B

Bits	Bits	Description
		01 -> Port C 10 -> Port D 11 -> Reserved
05:04	0	**Reserved**
03	0	**Enable WFA:** When Set, WFA mode is enabled**.** Will not reset on double function group reset. Reserved for SKL.
02	0	**Reserved**
01	0	**Enable DP1.2 Features (EDP12):** When set, DP1.2 features are enabled.Will not reset on double function group reset.
00	0	**Enable 3rd Pin and Converter Widget (E3P):** When set, the second and third pin and converter widget is enabled and can respond to HD Audio Verbs. When cleared, the second and third pin and converter widget is disabled and cannot respond to HD Audio verbs.Will not reset on double function group reset.

782h SET_GTCT - Set GTC Trigger

Bits	Bits	Description
07:00	0	**Any data:** The value of this field is irrelevant. The access of the SET causes a capture to occur.

F83h GET_CGTC - Get Captured GTC Value

Bits	Bits	Description
07	0	**GTC Value:** 32-bit GTC value captured on the SET_GTCT verb

F84h GET_CWC - Get Captured Wall Clock Value

Bits	Bits	Description
31:00	0	**Wall Clock Value:** 32-bit wall clock value captured on the SET_GTCT verb

F85h GET GOF - Get GTC Offset Value

Bits	Reset	Description
31:00	0h	**Value (VAL):** Reports the GTC Offset Value.

785h SET GOF0 - Set GTC Offset Value Byte 0

Bits	Description
07:00	GTC Offset Value Bits [7:0]

786h SET GOF1 - Set GTC Offset Value Byte 1

Bits	Description
07:00	GTC Offset Value Bits [15:8]

787h SET GOF2 - Set GTC Offset Value Byte 2

Bits	Description
07:00	GTC Offset Value Bits [23:16]

788h SET GOF3 - GTC Offset Value Byte 3

Bits	Description
07:00	GTC Offset Value Bits [31:24]

789hF89h SETGET_GDI - SetGet GTC Device Index

Bits	Reset	Description
07:06	0	*Reserved*
05:00	00h	**Device (D):** 000001, 000010 (based upon number of devices present) This is the pipe based. 000000 i s pipeA and so on and so forth.

3h SET_AM - Set Amplifier Mute

Bits	Bits	Description
15	0	**Set Output Amp (SOA):**.
14	0	**Set Input Amp (SIA):**.
13	0	**Set Left Amp (SLA):**.
12	0	**Set Right Amp (SRA):**.
11:08	0h	**Index (IDX):**

Bits	Bits	Description
07	1	**Mute (MUTE):** When set, amp muted.
06:00	0	*Reserved*

B8h GET_AM - Get Amplifier Mute

Bits	Bits	Description
31:08	0	*Reserved*
07	1	**Mute (MUTE):** When set, amp muted.
06:00	0	*Reserved*

F00h Get Parameters

Parameter	Symbol	Register Name
09h	PARAM_AWC	Audio Widget Capabilities
0Ch	PARAM_PC	Pin Capabilities
0Eh	PARAM_CLL	Connection List Length
12h	PARAM OAC	Output Amplifier Capabilities
0Fh	PARAM_PS	Suppoted Power States

Parameter 09h: AWC - Audio Widget Capabilities

Bits	Reset	Description
31:24	0	Reserved
23:20	4h	**Widget Type (TYPE):** Indicates this is a pin complex widget
19:16	0	**Sample Delay in Widget (DELAY):** No delay through the pin widget.
15:13	011	**Channel Count Extension (CCE):** This field, combined with STRO, indicate 8 channels supported.
11	0	**L-R Swap (LRS):** Indicates no left/right channel swap.
10	1	**Power Control (PC):** Indicates power state control
09	1	**Digital (DIG):** Indicates support for digital streams.

Bits	Reset	Description
08	1	**Connection List (CL):** Indicates a connection list
07	1	**Unsolicited Capable (UC):** Indicates support for unsolicited responses.
06	0	**Processing Widget (PW):** Indicates no support for processing
05	0	**Stripe (STRP):** Indicates striping not supported.
04	0	**Format Override (FO):** Indicates no support for formatting
03	1	**Amp Parameter Override (APO):** Indicates no amplifier override support.
02	1	**Out Amp Present (OAP):** Indicates no output amplifier present.
01	0	**In Amp Present (IAP):** Indicates no input amplifier present.
00	1	**Stereo (STRO):** Indicates a stereo widget

Parameter 0Ch: PC - Pin Capabilities

Bits	Reset	Description
31:28	0	*Reserved*
27	1	**High Bit Rate (HBR):** Indicates support for high bit-rate audio
26	0	*Reserved*
25	0	*MST capable (DP1.2)*
24	1	**DisplayPort (DP):** Indicates support for DisplayPort
23:08	0	*Reserved*
07	1	**HDMI (HDMI):** Indicates support for HDMI
06:05	0	*Reserved*
04	1	**Output Capable (OC):** Pin is output capable
03	0	*Reserved*

Bits	Reset	Description
02	1	**Presence Detect Capable (PDC):** Indicates capability for presence detection
01:00	0	*Reserved*

Parameter 0Eh: CLL - Connection List Length

Bits	Reset	Description
31:08	0	*Reserved*
07	0	**Long Form (LF):** Indicates connection list is short form
06:00	03h	**Length (LEN):** Indicates there is one item in the connection list.

Parameter 12h: OAC - Output Amplifier Capabilities

Bits	Reset	Description
31	1	**Mute Capable (MC):** Muting is capable on this pin
30:00	0	Reserved

Parameter 0Fh: PARAM_SPS - Supported Power States

Bit	Reset	Description
31	1	**Extended Power State Supported (EPSS):** Indicates support for low power states
30:04	0	*Reserved*
03	1	**D3 Supported (D3S):** Indicates support for D3.
02	0	**D2 Supported (D2S):** Indicates no support for D2.
01	0	**D1 Supported (D1S):** Indicates no support for D1.
00	1	**D0 Supported (D0S):** Indicates support for D0.

701hF01h SETGET_CSC - SetGet Connection Select Control

Bits	Reset	Description
07:00	00h	**Connection Select Control (CSC)**

Bits	Reset	Description

F02h GET_CLE - Get Connection List Entry

Bits	Reset	Description
31:24	0	Reserved
23:16	100	**Connection List Entry (CLE):** Hardwired to Converter ID 4
15:08	011	**Connection List Entry (CLE)**: Hardwired to Converter ID 3
07:00	010	**Connection List Entry (CLE)**: Hardwired to Converter ID 2

705h SET_PS - Set Power State

Bits	Description
07:02	Reserved
01:00	**Requested Power State (RPS):** Only D0 (00) and D3 (11) may be requested

F05h GET_PS - Get Power State

Bits	Reset	Description
31:11	0	Reserved
10	0	**Settings Reset (SR):** Haswell does not change the default values.
09	0	**Clock Stop OK (CSOK):** Clock stopping in D3 is not OK
08	0	**Error (ERR):** No error will ever be reported.
07:06	0	Reserved
05:04	11	**Actual Power State (APS):** Indicates the current power state of the node.
03:02	0	Reserved
01:00	11	**Requested Power State (CPS):** Reflects value written with SET_PS verb.

707hF07h SETGET_PWC - SetGet Pin Widget Control

Bits	Reset	Description

Bits	Reset	Description
07	0	Reserved
06	1	**Out Enable (OE):** When set, the audio is enabled
05:02	0	Reserved
01:00	00	**Encoded Packet Type (EPT)**

708hF08h SETGET_UE - SetGet Unsolicited Enable

Bits	Description
07	**Unsolicited Enable (UE):** When set, unsolicited responses are allowed
06	Reserved
05:00	**Tag (TAG)**

F09h GET_PS - Get Pin Sense

Bits	Reset	Description
31	0	**Presence Detect (PD):** When set presence is detected on this pin.
30	0	**ELD Value (ELDV)**
29	0	**Inactive (INA)**
28:00	28:00	Reserved

71Ch SET_CD0 - Set Configuration Default Byte 0

Bits	Description
07:04	**Default Association (DA)**
03:00	**Sequence (SEQ)**

71Dh SET_CD1 - Set Configuration Default Byte 1

Bits	Description
07:04	**Color (COL)**
03:00	**Miscellaneous (MISC)**

71Eh SET_CD2 - Set Configuration Default Byte 2

Bits	Description
07:04	**Default Device (DD)**
03:00	**Connection Type (CT)**

71Fh SET_CD3 - Set Configuration Default Byte 3

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>07:06</td><td>Port Connectivity (PC): External connectivity of the pin complex.
00 = Connected to jack
01 = No physical connection
10 = Fixed function device (integrated speaker, mic, etc.)
11 = Both a jack and internal connection</td></tr>
<tr><td>05:00</td><td>Location (LOC):
<table>
<tr><td></td><td colspan="10">Bits 3:0</td></tr>
<tr><td>Bits 5:4</td><td>0h: N/A</td><td>1h: Rear</td><td>2h: Front</td><td>3h: Left</td><td>4h: Right</td><td>5h: Top</td><td>6h: Bottom</td><td>7h: Special</td><td>8h: Special</td><td>9h: Special</td><td>Ah-Fh Reserved</td></tr>
<tr><td>00: External</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td></tr>
<tr><td>01: Internal</td><td>Y</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Y</td><td>Y</td><td>Y</td><td></td></tr>
<tr><td>10: Separate Chassis</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td><td></td><td></td></tr>
<tr><td>11: Other</td><td>Y</td><td></td><td></td><td></td><td></td><td></td><td>Y</td><td>Y</td><td>Y</td><td></td><td></td></tr>
</table></td></tr>
<tr><td></td><td></td></tr>
</table>

F1Ch GET_CD - Get Configuration Default

Bits	Description
31:30	**Port Connectivity (PC):** See SET_CD3.PC

Bits	Description
29:24	**Location (L):** See SET_CD3.L
23:20	**Default Device (DD):** See Set_CD2.DD
19:16	**Connection Type (CT):** See Set_CD2.CT
15:12	**Color (COL):** See SET_CD1.COL
11:08	**Miscellaneous (MISC):** See SET_CD1.MISC
07:04	**Default Association (DA):** See SET_CD0.DA
03:00	**Sequence (SEQ):** See SET_CD0.SEQ

F2Fh Get ELD Data

Parameter	Symbol	Register Name
07:0h	PARAM_INDX	ELD DATA Index

Parameter nn: ELD Data

Bits	Reset	Description
31:00	0	ELD Data

733h SET_PC - Set Protection Control

Bits	Description
07:03	**Unsolicited Response Sub Tag (URST):** Subtag to use for unsolicited responsed.
02	Reserved

734hF34h SETGET_CCM - GetSet Converter Channel Map

Bits	Reset	Description
07:04	0h	**Converter Channel (CC)**
03:00	0h	**Slot (SN)**

740hF40h SETGET PTS Offset Byte0

Bits	Reset	Description
07:00	00h	**PTS B0(PTSB0):** Bits 7:0 of the 32 bit 2's complement offset value. Should be added to the PTS value before sending to the circular buffer in the PES packet. Get verb will give the 32 bits value

741hF41h SETGET PTS Offset Byte1

Bits	Reset	Description
07:00	00h	**PTS B1(PTSB1):** Bits 15:8 of the 32 bit 2's complement offset value. Should be added to the PTS value before sending to the circular buffer in the PES packet. Get verb will give the 32 bits value

742hF42h SETGET PTS Offset Byte2

Bits	Reset	Description
07:00	00h	**PTS B2(PTSB2):** Bits 23:16 of the 32 bit 2's complement offset value. Should be added to the PTS value before sending to the circular buffer in the PES packet. Get verb will give the 32 bits value

743hF43h SETGET PTS Offset Byte3

Bits	Reset	Description
07:00	00h	**PTS B3(PTSB3):** Bits 31:24 of the 32 bit 2's complement offset value. Should be added to the PTS value before sending to the circular buffer in the PES packet. Get verb will give the 32 bits value

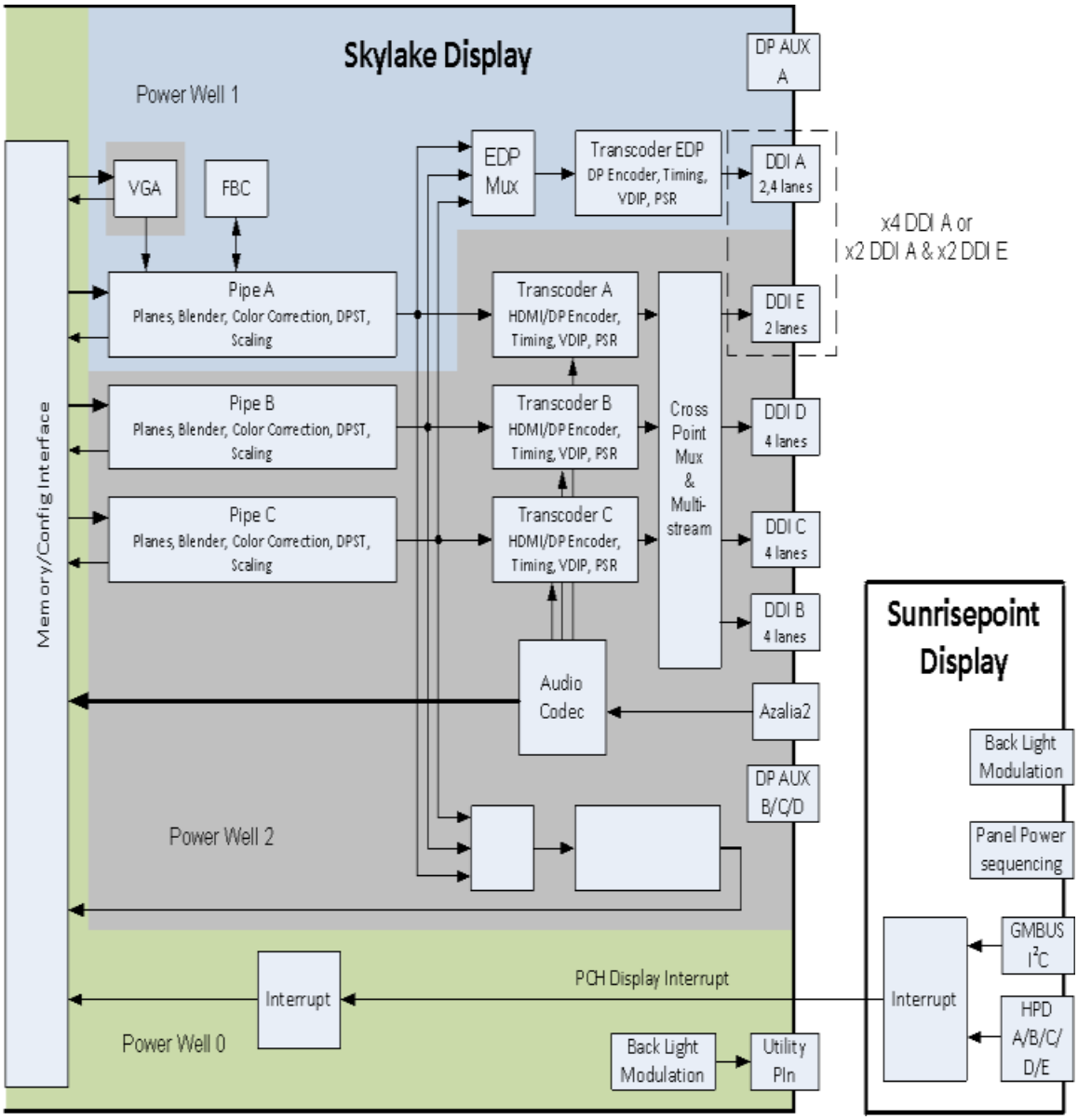
North Display Engine Registers

This chapter contains the register descriptions for the display portion of a family of graphics devices.

These registers vary by devices within the family of devices, so special attention needs to be paid to which devices use which registers and register fields.

Different devices within the family may add, modify, or delete registers or register fields relative to another device in the same family based on the supported functions of that device.

Display Connections

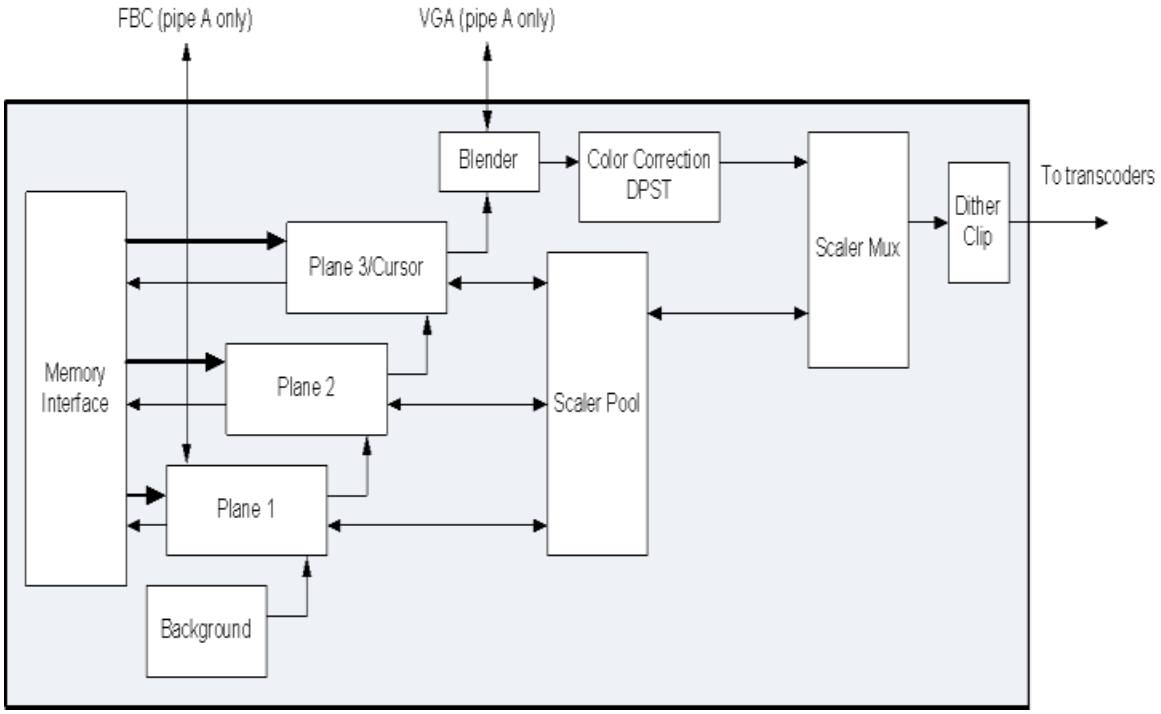


The front end of the display contains the pipes. There are three instances which are referred to as Pipe A, Pipe B, and Pipe C.

The pipes connect to the transcoders. There are five instances which are referred to as Transcoder A, Transcoder B, Transcoder C, Transcoder EDP, and Transcoder WD0.

The transcoders connect to the DDIs. There are five instances which are referred to as DDI A, DDI B, DDI C, DDI D, and DDI E.

Display Pipes



The display pipes contain the planes, blending, color correction, DPST, scaling, dithering, and clipping.

Each display pipe has three planes and a cursor. Each plane can be used as a sprite, primary, or overlay.

The background color that is seen under the bottom most plane is programmable.

The plane blending follows a fixed Z-order. Plane 1 is the bottom most plane and higher numbered planes stack on top of it.

The cursor and top most plane are mutually exclusive and cannot be both enabled at the same time.

Pipe A and Pipe B each have two pipe scalers. Pipe C has one pipe scaler.

Each pipe scaler can be assigned to scale an individual plane or scale the blended output.

Display Transcoders

The display transcoders contain the timing generators, port encoders, Audio/Video mixers, Video Data Island Packet mixers, and Panel Self Refresh controllers.

Transcoder EDP does not support HDMI, DVI, or Audio.

Transcoder WD0 only supports display capture and write back to memory.

Audio

The Azalia interface provides data to the audio codec.

The audio codec connects to the Audio/Video mixers in the transcoders.

DDIs

The DDIs contain the DisplayPort transport control and other port logic to interface to the DDI physical pins.

DDI A, DDI B, DDI C, and DDI D support lane reversal where the internal lane to package lane mapping is swapped.

DDI E does not support lane reversal.

DDI A and DDI E do not support DisplayPort multistream.

DDI A and DDI E share 2 lanes. DDI A is capable of supporting up to 4 lanes when DDI E is not connected, but only 2 lanes when DDI E is connected. DDI E is capable of supporting up to 2 lanes when connected. Dynamic switching between the two configurations is not supported.

DisplayPort A and E Lane Mapping:

Package Pin	Non-Reversed when DDI A is x4 capable	Reversed when DDI A is x4 capable	Non-Reversed when DDI A is x2 capable	Reversed when DDI A is x2 capable
DDIA/DDIE 3	Port A Main Link Lane 3	Port A Main Link Lane 0	Port E Main Link Lane 1	Port E Main Link Lane 1
DDIA/DDIE 2	Port A Main Link Lane 2	Port A Main Link Lane 1	Port E Main Link Lane 0	Port E Main Link Lane 0
DDIA/DDIE 1	Port A Main Link Lane 1	Port A Main Link Lane 2	Port A Main Link Lane 1	Port A Main Link Lane 0
DDIA/DDIE 0	Port A Main Link Lane 0	Port A Main Link Lane 3	Port A Main Link Lane 0	Port A Main Link Lane 1

DisplayPort B, C, and D Lane Mapping:

Package Pin	Non-Reversed	Reversed
DDI 3	Main Link Lane 3	Main Link Lane 0
DDI 2	Main Link Lane 2	Main Link Lane 1
DDI 1	Main Link Lane 1	Main Link Lane 2
DDI 0	Main Link Lane 0	Main Link Lane 3

HDMI/DVI TMDS Lane Mapping:

Package Pin	Non-Reversed	Reversed
DDI 3	TMDS Clock	TMDS Data2
DDI 2	TMDS Data0	TMDS Data1
DDI 1	TMDS Data1	TMDS Data0
DDI 0	TMDS Data2	TMDS Clock

DDI Equivalent Names:

DDI Name	Equivalent Names
DDI A	Port 0, DDI-A, DDIA, Port A, DDI0, eDP
DDI B	Port 1, DDI-B, DDIB, Port B, DDI1
DDI C	Port 2, DDI-C, DDIC, Port C, DDI2
DDI D	Port 3, DDI-D, DDID, Port D, DDI3
DDI E	Port 4, DDI-E, DDIE, Port E, DDI4

Pipe to Transcoder to DDI Mappings

Twin modes are not supported.

Any pipe can drive any single DDI.

With DisplayPort multistream it is possible to have multiple pipes driving a single DDI. DDI B, DDI C, and DDI D support multistream. DDI A and DDI E do not support multistream.

Pipe A can connect to either Transcoder A, Transcoder EDP, or Transcoder WD0, but not more than one simultaneously.

Pipe B can connect to either Transcoder B, Transcoder EDP, or Transcoder WD0, but not more than one simultaneously.

Pipe C can connect to either Transcoder C, Transcoder EDP, or Transcoder WD0, but not more than one simultaneously.

Transcoder A is tied to Pipe A.

Transcoder B is tied to Pipe B.

Transcoder C is tied to Pipe C.

Transcoder EDP can connect to Pipe A, Pipe B, or Pipe C, but only one at a time.

Transcoder WD0 can connect to Pipe A, Pipe B, or Pipe C, but only one at a time.

Transcoder A can connect to DDI B, DDI C, DDI D, or DDI E, but only one at a time.

Transcoder B can connect to DDI B, DDI C, DDI D, or DDI E, but only one at a time.

Transcoder C can connect to DDI B, DDI C, DDI D, or DDI E, but only one at a time.

Transcoder EDP can connect only to DDI A.

Transcoder WD0 does not connect to any DDI. Transcoder WD0 output only goes to memory write back.

DDI A can connect only to Transcoder EDP. DDI A does not support DisplayPort multistream.

DDI B can connect to Transcoder A, Transcoder B, or Transcoder C, individually or simultaneously if DisplayPort multistream is used.

DDI C can connect to Transcoder A, Transcoder B, or Transcoder C, individually or simultaneously if DisplayPort multistream is used.

DDI D can connect to Transcoder A, Transcoder B, or Transcoder C, individually or simultaneously if DisplayPort multistream is used.

DDI E can connect to Transcoder A, Transcoder B, or Transcoder C, individually.

Terminology

Term	Description
DP	DisplayPort
SST, DP SST	DisplayPort Single Stream Transport
MST, DP MST	DisplayPort Multi Stream Transport

Register Access Field	Description	Implementation
R/W (Read/Write)	The value written into this register will control hardware and is the same value that will be read.	Write data is stored. Read is from the stored data. Stored value is used to control hardware.
Reserved	Unused register bit. Don't assume a value for these bits. Writes have no effect.	Write data is ignored. Read is zero.
MBZ (Must Be Zero)	Always write a zero to this register.	May be implemented as Reserved or as R/W.
PBC (Preserve Bit Contents)	Software must write the original value back to this bit. This allows new features to be added using these bits.	May be implemented as Reserved or as R/W.
Read Only	The read value is determined by hardware. Writes to this bit have no effect.	Write data is ignored. Read is from a status signal or some other internal source.
Write Only	The value written into this register will control hardware. Reads return zero.	Write data is stored. Read is zero. Stored value is used to control hardware.
R/W Clear (Read/Write Clear)	Sticky status bit. Hardware will set the bit, software can clear it with a write of 1b.	Internal hardware events set a sticky bit. Read is from the sticky bit. A write of 1b clears the sticky bit.
R/W Set (Read/Write Set)	Sticky status bit. Software can set the bit with a write of 1b. Hardware will clear the bit.	A write of 1b sets a sticky bit. Internal hardware events clear a sticky bit. Read is from the sticky bit.

Register Access Field	Description	Implementation
Double Buffered	Write when desired and the written value will take effect at the time of the update point specified in the 'Double Buffer Update Point' parameter. Reads will return the written value, which is not necessarily the value being currently used to control hardware. Some have a specific arming sequence where a write to another register, specified in the 'Double Buffer Armed By' parameter, is required before the update can take place. Once the armed by register is written to, the written values of all registers controlled by that arming will take effect at the time of the double buffer update point. This is used to ensure atomic updates of several registers. Note: Once armed, by write to the armed by register, the registers controlled by this arming should not be changed until the double buffer update point is reached. If changed, this will disarm the sequence and will require another write to the armed by register to get it to the armed status again.	Two stages of registers used. Write data is stored into first stage. Read is from the first stage stored data. First stage stored value is transferred to second stage storage at the double buffer update point. Second stage stored value is used to control hardware. Arm/disarm logic may be used for some registers to control the double buffer update point.
Write/Read Status	The value written into this register will control hardware. The read value is determined by hardware.	Write data is stored. Stored value is used to control hardware. Read is from a status signal or some other internal source.

Mode Set

A mode set sequence is the programming sequence that must be followed when enabling or disabling output to a display. There are several different mode set sequences documented in the following sections. The sequence to use depends on which type of port is being enabled or disabled.

Skylake Sequences to Initialize Display

These sequences are used to initialize the display engine before any display engine functions can be enabled.

Most display engine functions will not operate while display is not initialized. Only basic PCI, I/O, and MMIO register read/write operations are supported when display is not initialized.

Initialize Sequence

1. Enable PCH Reset Handshake
    a. Set NDE_RSTWRN_OPT RST PCH Handshake En to 1b.
2. Enable Power Well 1 (PG1) and Misc IO Power
    a. Poll for FUSE_STATUS Fuse PG0 Distribution Status = 1b.
        - Timeout and fail after 5 us.
    b. Set PWR_WELL_CTL Power Well 1 Request and Misc IO Power Request to 1b.
    c. Poll for PWR_WELL_CTL Power Well 1 State and Misc IO Power State = 1b.
        - Timeout and fail after 10 us.
    d. Poll for FUSE_STATUS Fuse PG1 Distribution Status = 1b.
        - Timeout and fail after 5 us.
3. Enable CDCLK PLL
    a. Set CDCLK_CTL CD Frequency Select to the minimum.
    b. Configure DPLL0 link rate to the rate required for the eDP panel, or 810 MHz (DP 1.62 GHz) if there is no eDP panel.
    c. Set LCPLL1_CTL PLL enable to 1b.
    d. Poll for LCPLL1_CTL PLL lock = 1b.
        - Timeout and fail after 5 ms.
4. Follow Sequences for Changing CD Clock Frequency
5. Enable DBUF
    a. Set DBUF_CTL DBUF Power Request to 1b.
    b. Poll for DBUF_CTL DBUF Power State = 1b.
        - Timeout and fail after 10 us.

Un-initialize Sequence

1. Disable all display engine functions using the full mode set disable sequence on all pipes, transcoders, ports, planes, and power well 2 (PG2).
2. Disable DBUF
    a. Clear DBUF_CTL DBUF Power Request to 0b.
    b. Poll for DBUF_CTL DBUF Power State = 0b.
        - Timeout and fail after 10 us.
3. Disable CDCLK PLL
    a. Clear LCPLL1_CTL PLL enable to 0b.
    b. Poll for LCPLL1_CTL PLL lock = 0b.
        - Timeout and fail after 1 ms.
4. Disable Power Well 1 (PG1) and Misc IO Power
    a. Clear PWR_WELL_CTL Power Well 1 Request and Misc IO Power Request to 0b.
    b. Wait for 10us. Do not poll for the power well to disable. Other clients may be keeping it enabled.

Sequences for DisplayPort

This topic describes how to enable and disable DisplayPort.

Enable Sequence

Display must already be initialized

DDIA Lane Capability Control must be configured prior to enabling any ports or port clocks

1. **Enable Power Wells**

If not PipeA+DDIA - Enable Power Well 2

a. Enable PWR_WELL_CTL Power Well 2 Request
b. Wait for PWR_WELL_CTL Power Well 2 State = Enabled, timeout after 20 us
c. Wait for FUSE_STATUS FUSE PG2 Distribution Status = Done, timeout after 1 us

2. **If panel power sequencing is required - Enable Panel Power**
    a. Enable panel power sequencing
    b. Wait for panel power sequencing to reach the enabled state
3. **Enable Port PLL**
    a. If PLL is not already enabled, follow port clock programming sequence from Clocks section
    b. If PLL to port mapping is flexible, configure PLL to port mapping to direct the PLL output to the DDI
4. **If IO power is controlled through PWR_WELL_CTL - Enable IO Power**
    a. Enable PWR_WELL_CTL DDI IO Power Request for the DDI that will be used

   b. Wait for PWR_WELL_CTL DDI IO Power Request = Enabled, timeout after 20 us

5. **Enable and Train DisplayPort**
   a. Configure and enable DP_TP_CTL with link training pattern 1 selected
   b. Configure voltage swing and related IO settings. Refer to the DDI Buffer section.
   c. Configure and enable DDI_BUF_CTL
   d. Gen9: Wait >518 us for buffers to enable before starting training or allow for longer time in TP1 before software timeout.
   e. Follow DisplayPort specification training sequence (see notes for failure handling)
   f. If DisplayPort multi-stream - Set DP_TP_CTL link training to Idle Pattern, wait for 5 idle patterns (DP_TP_STATUS Min_Idles_Sent) (timeout after 800 us)
   g. Set DP_TP_CTL link training to Normal.
6. **Enable Planes, Pipe, and Transcoder (repeat to add multiple pipes on a single port for multi-streaming)**
   a. If DisplayPort multi-stream - use AUX to program receiver VC Payload ID table to add stream
   b. Configure Transcoder Clock Select to direct the Port clock to the Transcoder
   c. Configure and enable planes (VGA or hires). This can be done later if desired.
   d. If VGA - Clear VGA I/O register SR01 bit 5
   e. Enable panel fitter if needed (must be enabled for VGA)
   f. Configure transcoder timings, M/N/TU/VC payload size, and other pipe and transcoder settings
   g. Configure and enable TRANS_DDI_FUNC_CTL
   h. If DisplayPort multistream - Enable pipe VC payload allocation in TRANS_DDI_FUNC_CTL
   i. If DisplayPort multistream - Wait for ACT sent status in DP_TP_STATUS and receiver DPCD (timeout after >410us)
   j. Configure and enable TRANS_CONF
   k. If panel power sequencing is required - Enable panel backlight

SRD and/or Audio can be enabled after everything is complete. Follow the audio enable sequence in the audio registers section.

Notes

Changing voltage swing during link training:

- Change the swing setting following the DDI Buffer section. The port does not need to be disabled.

Changing port width (lane count) or frequency during link training:

1. Follow Disable Sequence for DisplayPort to Disable Port
2. If PLL frequency needs to change, Follow Disable Sequence for DisplayPort to Disable PLL, then follow Enable Sequence for DisplayPort to Enable PLL, using the new frequency settings

3. Follow Enable Sequence for DisplayPort to Enable and Train DisplayPort, using the new port width settings

If the mode set fails, follow the disable sequence to disable everything that had been enabled up to the failing point.

Enabling DisplayPort Sync Mode

See TRANS_DDI_FUNC_CTL Port Sync Mode Enable for restrictions.

1. Follow the enable sequence for the DisplayPort slave, but skip the step that sets DP_TP_CTL link training to Normal (stay in Idle Pattern).
    - Set TRANS_DDI_FUNC_CTL Port Sync Mode Master Select and Port Sync Mode Enable when configuring and enabling TRANS_DDI_FUNC_CTL.
2. Wait 200 uS.
3. Follow the enable sequence for the DisplayPort master, but skip the step that sets DP_TP_CTL link training to Normal (stay in Idle Pattern).
4. Set DisplayPort slave DP_TP_CTL link training to Normal.
5. Set DisplayPort master DP_TP_CTL link training to Normal.

Software may need to use DOUBLE_BUFFER_CTL to ensure updates to plane and pipe registers will take place in the same frame.

For example: If pipe A and pipe B are synchronized together and software needs the surface addresses for two planes to update at the same time, software should use DOUBLE_BUFFER_CTL when writing the surface address registers for both planes, otherwise there is a possibility that the updates could be split across a vertical blank such that one plane would update on the current vertical blank and the other plane would update on the next vertical blank.

Disable Sequence

SRD and Audio must be disabled first. Follow the audio disable sequence in the audio registers section.

1. **If panel power sequencing is required - Disable panel backlight**
2. **Disable Planes, Pipe, and Transcoder (repeat to remove multiple pipes from a single port for multi-streaming)**
    a. If VGA
        i. Set VGA I/O register SR01 bit 5 for screen off
        ii. Wait for 100 us
    b. Disable planes (VGA or hires)
    c. Disable TRANS_CONF
    d. Wait for off status in TRANS_CONF, timeout after two frame times
    e. If DisplayPort multistream - use AUX to program receiver VC Payload ID table to delete stream

f. If done with this VC payload
    i. Disable VC payload allocation in TRANS_DDI_FUNC_CTL
    ii. Wait for ACT sent status in DP_TP_STATUS and receiver DPCD
g. Disable TRANS_DDI_FUNC_CTL with DDI_Select set to None
h. Disable panel fitter
i. Configure Transcoder Clock Select to direct no clock to the transcoder

3. **Disable Port (all pipes and VC payloads on this port must already be disabled)**
    a. Disable DDI_BUF_CTL
    b. Disable DP_TP_CTL (do not set port to idle when disabling)
    c. Gen9+,EXCLUDE(BXT): Wait 8 us or poll on DDI_BUF_CTL Idle Status for buffers to return to idle
4. **If panel power sequencing is required - Disable Panel Power**
    a. Disable panel power sequencing
5. **If IO power is controlled through PWR_WELL_CTL - Disable IO Power**
    a. Disable PWR_WELL_CTL DDI IO Power Request for the DDI that was used
6. **Disable Port PLL**
    a. If PLL to port mapping is flexible, configure PLL to port mapping to direct no clock to the DDI
    b. If this PLL is no longer needed, follow PLL disable sequence from Clocks section
7. **Disable Power Wells**

a. If no required resource is in the power well - Disable PWR_WELL_CTL Power Well 2 Request

Disabling DisplayPort Sync Mode

1. Follow the disable sequence for the DisplayPort slave.
2. Follow the disable sequence for the DisplayPort master.

Sequences for HDMI and DVI

This topic describes how to enable and disable HDMI and DVI.

Enable Sequence

Display must already be initialized

1. **Enable Power Wells**

a. Enable PWR_WELL_CTL Power Well 2 Request
b. Wait for PWR_WELL_CTL Power Well 2 State = Enabled, timeout after 20 us
c. Wait for FUSE_STATUS FUSE PG2 Distribution Status = Done, timeout after 1 us

2. **Enable Port PLL**
    a. If PLL is not already enabled, follow port clock programming sequence from Clocks section
    b. If PLL to port mapping is flexible, configure PLL to port mapping to direct the PLL output to the DDI
3. **If IO power is controlled through PWR_WELL_CTL - Enable IO Power**
    a. Enable PWR_WELL_CTL DDI IO Power Request for the DDI that will be used
    b. Wait for PWR_WELL_CTL DDI IO Power Request = Enabled, timeout after 20 us
4. **Enable Planes, Pipe, and Transcoder**
    a. Configure Transcoder Clock Select to direct the Port clock to the Transcoder
    b. Configure and enable planes (VGA or hires). This can be done later if desired.
    c. If VGA - Clear VGA I/O register SR01 bit 5
    d. Enable panel fitter if needed (must be enabled for VGA)
    e. Configure transcoder timings and other pipe and transcoder settings
    f. Configure and enable TRANS_DDI_FUNC_CTL
    g. Configure and enable TRANS_CONF
5. **Enable Port**
    a. Configure voltage swing and related IO settings. Refer to the DDI Buffer section.
    b. Configure and enable DDI_BUF_CTL

Audio can be enabled after everything is complete. Follow the audio enable sequence in the audio registers section.

Notes

If the mode set fails, follow the disable sequence to disable everything that had been enabled up to the failing point.

Disable Sequence

Audio must be disabled first. Follow the audio disable sequence in the audio registers section.

1. **Disable Planes, Pipe, and Transcoder**
    a. If VGA
        i. Set VGA I/O register SR01 bit 5 for screen off
        ii. Wait for 100 us
    b. Disable planes (VGA or hires)
    c. Disable TRANS_CONF
    d. Wait for off status in TRANS_CONF, timeout after two frame times
    e. Disable TRANS_DDI_FUNC_CTL with DDI_Select set to None
    f. Disable panel fitter

g. Configure Transcoder Clock Select to direct no clock to the transcoder

2. **Disable Port**
   a. Disable DDI_BUF_CTL
3. **If IO power is controlled through PWR_WELL_CTL - Disable IO Power**
   a. Disable PWR_WELL_CTL DDI IO Power Request for the DDI that was used used
4. **Disable Port PLL**
   a. If PLL to port mapping is flexible, configure PLL to port mapping to direct no clock to the DDI
   b. If this PLL is no longer needed, disable it
5. **Disable Power Well 2**
   a. If no required resource is in the power well - Disable PWR_WELL_CTL Power Well 2 Request
6. **Disable Power Wells**

a. If no required resource is in the power well - Disable PWR_WELL_CTL Power Well 2 Request

Sequences for Display C5 and C6

Display C5 (DC5) is a power saving state where hardware dynamically disables power well 1 and the CDCLK PLL and saves the associated registers.

DC5 can be entered when software allows it, power well 2 is disabled, and hardware detects that all pipes are disabled or pipe A is enabled with PSR active.

Display C6 (DC6) is a deeper power saving state where hardware dynamically disables power well 0 and saves the associated registers.

DC6 can be entered when software allows it, the conditions for DC5 are met, and the PCU allows DC6.

DC6 cannot be used if the backlight is being driven from the display utility pin.

SKL supports DC5 and DC6.

The context save and restore program is reset on cold boot, warm reset, PCI function level reset, and hibernate/suspend.

Sequence to Allow DC5 or DC6

1. Load the correct stepping specific Display Context Save and Restore (CSR) program from the binary package.
   a. Read the package header and extract the correct individual firmware. Binary package format details can be found in sections below.
   b. Skip the header section at the start of the program binary.
   c. Copy the payload into Display CSR Program Storage.
   d. Perform the MMIO writes specified in the header section.

2. Configure display engine to have power well 2 disabled, following the appropriate mode set disable sequences for any ports using power well 2. This can be done earlier if desired.
3. Set display register 0x45520 bit 1 to 1b. It does not need to be cleared at any time.
4. Set DC_STATE_EN Dynamic DC State Enable = "Enable up to DC5" for DC5 or "Enable up to DC6" for DC6.

Workaround for DC5/DC6

Sequence to Disallow DC5 and DC6

1. Set DC_STATE_EN Dynamic DC State Enable = "Disable".

DMC Firmware Package

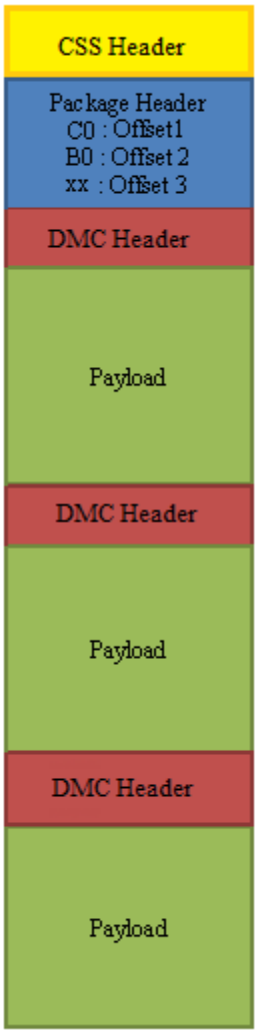
Display Micro-Controller firmware package includes all the firmwares that are required for different steppings of the product. The stepping dependent firmwares are all packaged and released as a single binary package. The package contains the CSS header, followed by the package header and the actual DMC firmwares.

Packaged firmware uses the following naming convention - <project>_dmc_ver<major>_<minor>.bin. The major version will get incremented whenever there is a change in the header layout and would require an update to the driver firmware loading module.

Major version 1

CSS Header	1.0
Package Header	1
DMC Header	1

Package Layout



CSS Header

```
typedef struct _CssHeader {
      uint32_t     moduleType;                        // 0x09 for DMC
      uint32_t     headerLen;                         // CSS header length in dwords
      uint32_t     headerVer;                         // 0x10000
      uint32_t     moduleID;                          // Not used
      uint32_t     moduleVendor;                      // Not used
      uint32_t     date;                              // YYYYMMDD(YYYY « 16 + MM « 8
+ DD)
      uint32_t     size;                              // Total dmc fw binary size in
dwords - (CSS_Headerlen + PackageHeaderLen + dmc FWsLen)/4
      uint32_t     keySize;                           // Not used
```

```
      uint32_t      modulusSize;                    // Not used
      uint32_t      exponentSize;                   // Not used
      uint32_t      reserved1[12];                  // Not used
      uint32_t      version;                        // Major Minor
      uint32_t      reserved2[8];                   // Not used
      uint32_t      uKernelHeaderInfo;              // Not used
} CssHeader;
```

Package Header

Package header contains the firmware/stepping mapping table and the corresponding firmware offsets to the individual binaries, within the package. Mapping table will list the exceptions first, followed by the default entries. An Offset value of "0xFFFFFFFF" in the mapping table indicates that there is no firmware available/supported for that stepping. The offsets to the individual binary are DWord aligned. The first individual binary starts at an offset value of "0x00000000" after the CSS Header and the Package Header.

Stepping/Version mapping example

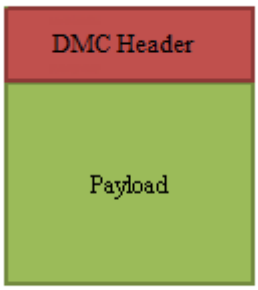
Stepping	FW Version
A1	1.1
B*	1.6
**	2.3

```
typedef struct _PackageHeader {
      uint8_t       headerLen;                      // DMC package header length
in dwords
      uint8_t       headerVer;                      // 0x01
      uint8_t       reserved[10];                   // Reserved
      uint32_t      numEntries;                     // Number of valid entries in
the FWInfo array below
      struct _FWInfo_ {
            uint16_t      reserved1;                // Reserved
            char          stepping;                 // Stepping (A, B, C, ..., *).
* is a wildcard
            char          substepping;              // Sub-stepping (0, 1, ...,
*). * is a wildcard
            uint32_t      offset;                   // FW offset within the
package
            uint32_t      reserved2;                // Reserved
      } FWInfo[20];
} PackageHeader;
```

DMC firmware binary

Each individual DMC firmware binary has a header followed by a payload whose size is specified in the header section. Along with the version, length, firmware size etc. the header section also specifies a list of MMIO addresses and data. These MMIO write cycles, in the 0x80000 - 0x8FFFF address range, should be executed as part of the initial CSR program setup.

DMC firmware Layout




```
for i = 1 to <mmioCount>
      Perform MMIO write to address <mmioaddr[i]> with data <mmiodata[i]>
typedef struct _DMCHeader {
     uint32_t    reserved;              // 0x40403E3E
     uint8_t     headerLen;             // DMC binary header length in bytes
     uint8_t     headerVer;             // 0x01
     uint16_t    dmccVer;               // Reserved
     uint32_t    project;               // Major, Minor
     uint32_t    fwSize;                // Firmware program size (excluding header)
     in dwords
     uint32_t    fwVersion;             // Major Minor version
     uint32_t    mmioCount;             // Number of valid MMIO cycles present in
     the MMIO address and data arrays below.
     uint32_t    mmioaddr[8];           // MMIO address
     uint32_t    mmiodata[8];           // MMIO data
     unint8_t    dfile[32];             // Reserved
     uint32_t    reserved1[2];          // Reserved
} DMCHeader;
```

Gen9 Display Resolution Support

A display resolution is only supported if it meets all the restrictions below for Maximum Pipe Pixel Rate, Maximum Port Link Rate, Maximum Memory Read Bandwidth, and Maximum Watermark.

Maximum Pipe Pixel Rate

The display resolution must fit within the maximum pixel rate output from the pipe. Make sure that the display pipe is able to feed pixels at a rate required to support the desired resolution.

For each enabled plane on the pipe {

    If plane scaling enabled {

        Horizontal down scale amount = Maximum[1, plane horizontal size / scaler horizontal window size]

        Vertical down scale amount = Maximum[1, plane vertical size / scaler vertical window size]

        Plane down scale amount = Horizontal down scale amount * Vertical down scale amount

        Plane Ratio = 1 / Plane down scale amount

    }

    Else {

        Plane Ratio = 1

    }

    If plane source pixel format is 64 bits per pixel {

        Plane Ratio = Plane Ratio * 8/9

    }

}

Pipe Ratio = Minimum Plane Ratio of all enabled planes on the pipe

If pipe scaling is enabled {

    Horizontal down scale amount = Maximum[1, pipe horizontal source size / scaler horizontal window size]

    Vertical down scale amount = Maximum[1, pipe vertical source size / scaler vertical window size]

        Note: The progressive fetch - interlace display mode is equivalent to a 2.0 vertical down scale

    Pipe down scale amount = Horizontal down scale amount * Vertical down scale amount

    Pipe Ratio = Pipe Ratio / Pipe down scale amount

}

Pipe maximum pixel rate = CDCLK frequency * Pipe Ratio // See the display clocks section for the supported CDCLK frequencies.

Maximum Port Link Rate

The display resolution must fit within the maximum link rate for each port type.

Port Type	Maximum Link Bit Rate
eDP/DP	HBR2 5.4 GHz
HDMI	3 GHz
DVI	1.65 GHz

Maximum Memory Read Bandwidth

The display resolution must not exceed the available system memory bandwidth, considering factors like thermal throttling and bandwidth available for other memory clients.

For each pipe {

For each plane enabled on the pipe { // cursor can be ignored

Plane bandwidth MB/s = pixel rate MHz * source pixel format in bytes * plane down scale amount * pipe down scale amount

Total display bandwidth MB/s = Total display bandwidth + Plane bandwidth

}

}

Raw system memory bandwidth = (# of memory channels * memory frequency * 8 bytes)

If Total display bandwidth > system memory bandwidth available for display {Bandwidth is exceeded}

Programming Note	
Context:	Gen9 Display Resolution Support
Do not use more thatn 60% of raw system memory bandwidth for display.	

Maximum Watermark

The display resolution must not exceed the level 0 maximum watermark value. See the volume on Watermark Programming.

Display Resolution Capabilities

These are examples of common resolutions that meet all the resolution restrictions for up to 3 simultaneous displays, 4 primary or sprite planes with 32bpp pixel format, and 1 cursor, with no panel fitter down scaling.

Attribute	CD 675 MHz	CD 540 MHz	CD 450 MHz	CD 337.5 MHz
eDP/DP x4 single stream	3840x2560 60Hz 24Bpp[4] 4096x2304 60Hz 24Bpp[4]	3840x2160 60Hz 30Bpp[4]	3200x2000 60Hz 30Bpp[3]	2880x1620 60Hz 30Bpp[2] 4096x2160 30Hz 30Bpp[2]
HDMI	Same as entry to the right	Same as entry to the right	Same as entry to the right	4Kx2K 24-30Hz 24Bpp[2]
DVI	Same as entry to the right	Same as entry to the right	Same as entry to the right	1920x1200 60Hz 24Bpp[1]

Each entry is showing the highest common resolutions at that CD clock frequency step. Lower resolutions are also supported. Higher, less common, resolutions can also work, but need to be calculated individually.

eDP, DP, and DVI are calculated using CVT 1.2 RB1 blanking and pixel rate.

HDMI is calculated using HDMI specification blanking and pixel rate.

Bpp is referring to the port output bits per pixel.

[1]Requires at least single channel DDR3 1333 for 3 simultaneous displays

[2]Requires at least single channel DDR3 1600 for 3 simultaneous displays

[3]Requires at least dual channel DDR3 1333 for 3 simultaneous displays

[4]Requires at least dual channel DDR3 1600 for 3 simultaneous displays

Examples

Example pipe pixel rate:

Plane 1 enabled at 32bpp, plane 2 enabled at 16bpp, pipe scaling enabled and down scale amount 1.12, and CDCLK 450 MHz:

Plane 1 ratio = 1

Plane 2 ratio = 1

Pipe ratio = Minimum[1, 1] = 1

Pipe ratio = 1/1.12 = 0.89

Pipe maximum pixel rate = 450 MHz * 0.89 = 400.5 MHz

Example pipe pixel rate:

Plane 1 enabled at 64bpp and plane down scale amount 1.25, plane 2 enabled at 32bpp, no panel fitting enabled, and CDCLK 540 MHz:

Plane 1 ratio = 1/1.25 * 8/9 = 0.71

Plane 2 ratio = 1

Pipe ratio = Minimum[1, 0.71] = 0.71

Pipe maximum pixel rate = 540 MHz * 0.71 = 383.4 MHz

Example memory bandwidth:

System memory bandwidth available for display = 4000 MB/s

Pipe A - Plane 1 enabled at 32bpp, plane 2 enabled at 16bpp, scaling disabled, pixel rate 148.5 MHz

Pipe B - Plane 1 enabled at 32bpp, scaling disabled, pixel rate 148.5 MHz

Pipe C - Plane 1 enabled at 32bpp, scaling disabled, pixel rate 148.5 MHz

Pipe A - Plane 1 bandwidth = 148.5 * 4 bytes = 594 MB/s

Pipe A - Plane 2 bandwidth = 148.5 * 2 bytes = 297 MB/s

Pipe B - Plane 1 bandwidth = 148.5 * 4 bytes = 594 MB/s

Pipe C - Plane 1 bandwidth = 148.5 * 4 bytes = 594 MB/s

Total display bandwidth = 594 + 297 + 594 + 594 = 2079 MB/s

System memory bandwidth available for display not exceeded

Example memory bandwidth:

System memory bandwidth available for display = 4000 MB/s

Pipe A - Plane 1 enabled at 32bpp, plane 2 plane enabled at 32bpp, pipe scaling enabled and down scale amount 1.12, pixel rate 414.5 MHz

Pipe B - Plane 1 enabled at 32bpp, scaling disabled, pixel rate 414.5 MHz

Pipe C - Plane 1 enabled at 32bpp, scaling disabled, pixel rate 414.5 MHz

Pipe A - Plane 1 bandwidth = 414.5 * 4 bytes * 1.12 = 1863 MB/s

Pipe A - Plane 2 bandwidth = 414.5 * 4 bytes * 1.12 = 1863 MB/s

Pipe B - Plane 1 bandwidth = 414.5 * 4 bytes = 1658 MB/s

Pipe C - Plane 1 bandwidth = 414.5 * 4 bytes = 1658 MB/s

Total display bandwidth = 1863 + 1863 + 1658 + 1658 = 7042 MB/s

System memory bandwidth available for display **exceeded**

Clocks

CDCLK_CTL

LCPLL1_CTL

LCPLL2_CTL

WRPLL_CTL

PORT_CLK_SEL

TRANS_CLK_SEL

DPLL_CTRL1

DPLL_CTRL2

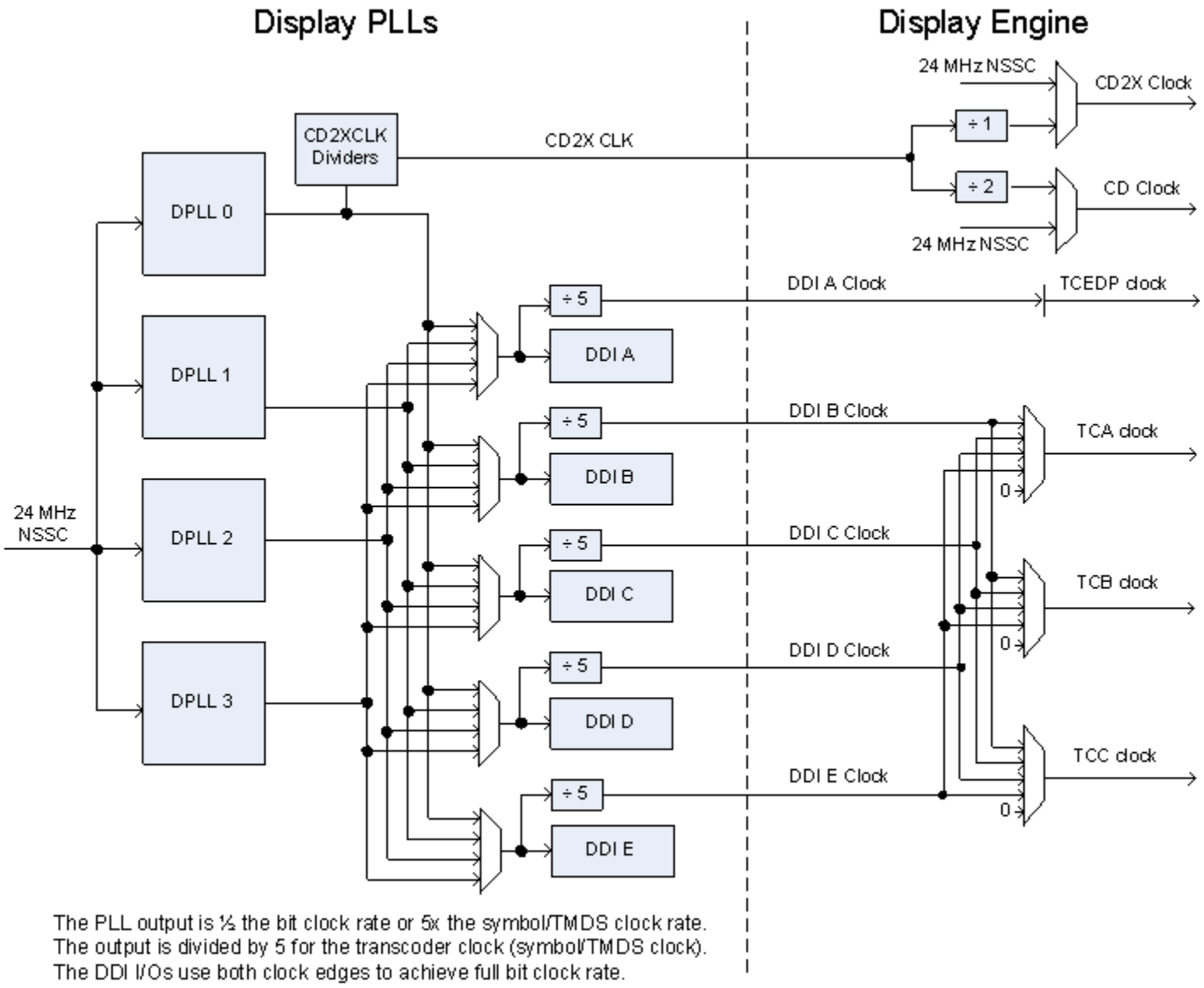
DPLL_STATUS

DPLL_CFGCR1

DPLL_CFGCR2

TIMESTAMP_CTR

Overview of Supported Display Clock Paths



The display engine clocking structure has multiple PLLs and clocks. The flow is from reference to PLL to DDI (port) clock to transcoder clock.

Display Engine Clock Reference

There is one display engine clock reference.

	Non-SSC (NSSC) Reference
Usage	Non-spread spectrum reference for DPLL and alternate source for CD and CD2X clocks
Frequency	24 MHz
Default After Reset	Enabled
Programming	Not programmable by display software.

Display Engine PLLs

There are four display engine PLLs (DPLLs).

The PLL output frequencies are 5x the symbol/TMDS rate, which is 1/2 the bit rate.

The PLL output is divided by 5 to become the symbol/TMDS clock frequency used in the display engine.

Both edges of the PLL output are used in the DDI I/Os to double the frequency to bit clock rate.

A single PLL output may be used by multiple DDI ports simultaneously if those DDI ports all require the same frequency and spread characteristics.

	DPLL0 (LCPLL 1)	DPLL1 (LCPLL 2), DPLL2 (WRPLL 1), DPLL3 (WRPLL 2)
Usage	Sources for DDI clocks and CD clock	Source for DDI clocks
Input	Non-SSC reference	Non-SSC reference
Frequency	Programmable selection in the range 0.1 to 2.7 GHz (0.2 to 5.4 GHz bit rate). CD clock usage does not allow spread spectrum to be used.	Programmable selection in the range 0.1 to 2.7 GHz (0.2 to 5.4 GHz bit rate). Spread spectrum can be added if needed.
Default After Reset	Disabled	Disabled
Programming	DPLL enable and frequency must be programmed by software when enabling and disabling a display. Frequency programming is done through the DPLL_CTRL1 register. Enable is programmed through the LCPLL1_CTL register. See the section on Port Clock Programming.	DPLL enable, SSC enable, and frequency must be programmed by software when enabling and disabling a display. Frequency and SSC programming is done through the DPLL_CTRL1 and DPLL*_CFGCR* registers. Enables are programmed through the LCPLL2_CTL and WRPLL_CTL registers. See the section on Port Clock Programming.

Recommended PLL Selection

Avoid changing frequencies on DPLL 0 since that requires DPLL 0 to be disabled, which impacts CDCLK and requires display features to be disabled.

Use DPLL 2 as a last choice because that PLL can be taken by HDPORT in some cases. See register HDPORT_STATE.

- For eDP Non-SSC, use DPLL 0 with frequency set to match the eDP panel requirement.
- For eDP SSC or external displays, use DPLLs 1, 3, 2 in that order, depending on the availability.
- If DPLL 0 is not used for eDP, run DPLL 0 in DisplayPort mode with link rate 810 MHz (DP 1.62 GHz). This will select VCO 8100 for generating CDCLK.

DDI Clocks

There is one DDI clock tied to each DDI port.

A single DDI clock output may be used by multiple transcoders simultaneously for DisplayPort Multi-streaming.

	DDI clocks A, B, C, D, and E
Usage	DDI ports I/O bit clock and symbol/TMDS clock, source for transcoder clocks
Input	Programmable selection between the DPLLs
Frequency	DPLL frequency
Default After Reset	Disabled
Programming	Mapping of DPLL to DDI must be programmed by software when enabling and disabling a display. Programming is done through the DPLL_CTRL2 registers. See the section on Port Clock Programming.

Transcoder Clocks

There is one Transcoder clock tied to each display transcoder, except Transcoder WD which uses only CD clock.

	Transcoder clock EDP	Transcoder clocks A, B, C
Usage	Transcoder EDP symbol clock	Transcoders A, B, and C symbol/TMDS clocks
Input	Always uses DDI A clock	Programmable selection between DDI clocks B, C, D, and E.
Frequency	PLL output frequency divided by 5	PLL output frequency divided by 5
Default After Reset	Connected to DDI A Clock	Disabled

Programming	Not programmable.	Mapping of DDI to Transcoder must be programmed by software when enabling and disabling a display. Programming is done through the TRANS_CLK_SEL registers.

CD Clock

CD clock refers to the Core Display clock which includes the Core Display 1X Clock (CD clock, CDclk, cdclk, CDCLK) and the Core Display 2X Clock (CD2X clock, cd2xclk, CD2XCLK).

	CD clock
Usage	Clocking for most display engine functions
Input	DPLL0 CD2X output (with several frequency options) and Non-SSC reference.
Frequency	Non-SSC reference - 24 MHz CD, 24 MHz CD2X DPLL0 with VCO 8100: • 450 MHz CD, 900 MHz CD2X • 540 MHz CD, 1080 MHz CD2X • 337.5 MHz CD, 675 MHz CD2X • 675 MHz CD, 1350 MHz CD2X DPLL0 with VCO 8640: • 432 MHz CD, 864 MHz CD2X • 540 MHz CD, 1080 MHz CD2X • 308.57 MHz CD, 617.14 MHz CD2X • 617.14 MHz CD, 1234.28 MHz CD2X
Default After Reset	Enabled with the Non-SSC reference - 24 MHz CD, 24 MHz CD2X
Programming	CD clock cannot be disabled. CD clock frequency must be programmed by software when enabling and disabling a display. Programming is done through the CDCLK_CTL register.

Port Clock Programming

DisplayPort Programming

DisplayPort PLL Enable Sequence

1. Configure DPLL_CTRL1 to enable programming, set SSC enable/disable (no SSC for DPLL0), select DP mode, and set link rate.
2. Read back DPLL_CTRL1 to ensure writes completed before the next step.
3. Enable DPLL through LCPLL1_CTL (DPLL0), LCPLL2_CTL (DPLL1), WRPLL_CTL1 (DPLL2), or WRPL_CTL2 (DPLL3).
4. Wait for PLL lock status in DPLL_STATUS.
5. For each DDI that will use this DPLL. Configure DPLL_CTRL2 to turn on the clock for the DDI and map the DPLL to the DDI.

DisplayPort PLL Disable Sequence

1. For each DDI that was using this DPLL. Configure DPLL_CTRL2 to turn off the clock for the DDI.
2. Disable DPLL through LCPLL1_CTL (DPLL0), LCPLL2_CTL (DPLL1), WRPLL_CTL1 (DPLL2), or WRPL_CTL2 (DPLL3).

Example of DisplayPort on DDIA using HBR 2.7 GHz link rate with SSC

DPLL0 does not support SSC, so this case must use one of the other DPLLs. This example assumes DPLL1 is available.

1. Configure DPLL1 in the DPLL_CTRL1 register
   - DPLL1 Override = 1b (Enable)
   - DPLL1 Link Rate = 001b (1350 MHz - DP 2.7 GHz)
   - DPLL1 SSC = 1b (Enable)
   - DPLL1 HDMI Mode = 0b (DP mode)
2. Read back DPLL_CTRL1 to ensure writes completed
3. Enable DPLL1 in the LCPLL2_CTL register
   - PLL Enable = 1b (Enabled)
4. Wait for PLL lock status in DPLL_STATUS
5. Configure DPLL1 mapping to DDIA in the DPLL_CTRL2 register
   - DDIA Select Override = 1b (Enable)
   - DDIA Clock Select = 01b (DPLL1)
   - DDIA Clock Off = 0b (On)

HDMI and DVI Programming

HDMI and DVI PLL Enable Sequence

1. Configure DPLL_CTRL1 to enable programming, disable SSC, and use HDMI mode.
2. Configure DPLL_CFGCR1 to enable programming and set DCO frequency.
3. Configure DPLL_CFGCR2 to set the dividers and DCO central frequency.
4. Read back DPLL_CTRL1, DPLL_CFGCR1, and DPLL_CFGCR2 ensure writes completed before the next step.
5. Enable DPLL through LCPLL2_CTL (DPLL1), WRPLL_CTL1 (DPLL2), or WRPL_CTL2 (DPLL3).
6. Wait for PLL lock status in DPLL_STATUS.
7. For each DDI that will use this DPLL. Configure DPLL_CTRL2 to turn on the clock for the DDI and map the DPLL to the DDI.

HDMI and DVI PLL Disable Sequence

1. For each DDI that was using this DPLL. Configure DPLL_CTRL2 to turn off the clock for the DDI.
2. Disable DPLL through LCPLL2_CTL (DPLL1), WRPLL_CTL1 (DPLL2), or WRPL_CTL2 (DPLL3).

Formula for HDMI and DVI DPLL Programming

Reference frequency = 24 MHz.

Symbol clock frequency MHz = DCO Frequency / (P0 * P1 * P2) / 5.

AFE clock = Symbol clock frequency MHz * 5 = DCO Frequency / (P0 * P1 * P2).

P2 can be 1, 2, 3, or 5.

If P2 != 2, then P1 must be 1. Else P1 can be 1 to 255.

If P1 == 1, then P0 can be 1, 2, 3, or 7. Else P0 can be 2, 3, or 7.

An even value for the divider (P0 * P1 * P2) is preferred.

DCO central frequency must be 8400, 9000, or 9600 MHz.

DCO frequency must be within +1% or -6% of DCO central frequency. It is preferred to get as close to the DCO central frequency as possible, but using an even divider value takes precedence.

Note: If the desired symbol clock frequency cannot be achieved with the valid values of P0, P1, P2, and DCO frequencies, use a different screen resolution with a different symbol clock frequency.

Algorithm to Find HDMI and DVI DPLL Programming

1. Find AFE clock
2. For each legal divider (P*Q*K), find the DCO. Try even dividers first. Only use odd dividers if a good even is not found.
3. Try the DCO with each central frequency.

4. Select the divider and central frequency that gives the DCO with minimum deviation from DCO central frequency and fits within the +1% and -6% requirement.
5. Find the P, Q, K values to create that divider.

Pseudo-code for HDMI and DVI DPLL Programming

```
 Minimum Positive Deviation = 1%
 Minimum Negative Deviation = 6%
 Chosen Central Frequency = NONE
 AFE Clock = 5 * Symbol Clock MHz
 Even Candidate Dividers[] = { 4, 6, 8, 10, 12, 14, 16, 18, 20, 24, 28, 30, 32, 36, 40, 42,
44, 48, 52, 54, 56, 60, 64, 66, 68, 70, 72, 76, 78, 80, 84, 88, 90, 92, 96, 98 }
 Odd Candidate Dividers[] = { 3, 5, 7, 9, 15, 21, 35 }

 For each Divider Parity (Even, Odd) // Prefer even values for divider
    If Divider Parity == Even
       Candidate Dividers[] = Even Candidate Dividers[]
    Else
       Candidate Dividers[] = Odd Candidate Dividers[]
    For each DCO Central Frequency // Try each central frequency
       For each Candidate Dividers // Try each divider
          DCO Frequency = Candidate Divider * AFE Clock
          If DCO Frequency > DCO Central Frequency  // Positive Deviation
             DCO Central Frequency Deviation = 100 * (DCO Frequency - DCO Central Frequency)
/ DCO Central Frequency
             If (DCO Central Frequency Deviation < Minimum Positive Deviation) // Check if it
meets the minimum requirement
                Minimum Positive Deviation = DCO Central Frequency Deviation
                Chosen Central Frequency = DCO Central Frequency
                Chosen DCO Frequency = DCO Frequency
                Chosen Divider = Candidate Divider
          Else // Negative Deviation
             DCO Central Frequency Deviation = 100 * ABS(DCO Frequency - DCO Central
Frequency)  / DCO Central Frequency
             If (DCO Central Frequency Deviation < Minimum Negative Deviation)
                Minimum Negative Deviation = DCO Central Frequency Deviation
                Chosen Central Frequency = DCO Central Frequency
                Chosen DCO Frequency = DCO Frequency
                Chosen Divider = Candidate Divider
       Next Candidate Divider
    Next DCO Central Frequency
    If Chosen Central Frequency != NONE // Break out early if a good even divider is found
       break
 Next Divider Parity
 getMultiplier(num)
 {
     if (num % 2 == 0) // Even
     {
         num1 = num / 2;
         if (num1 == 1 || num1 == 2 || num1 == 3 || num1 == 5)
         {
             P0 = 2;
             P1 = 1;
             P2 = num1;
         }
         else if (num1 % 2 == 0) // Div by 4
         {
             P0 = 2;
             P1 = num1 / 2;
             P2 = 2;
         }
```

```
        else if (num1 % 3 == 0) // Div by 6
        {
            P0 = 3;
            P1 = num1 / 3;
            P2 = 2;
        }
        else if (num1 % 7 == 0) // Div by 14
        {
            P0 = 7;
            P1 = num1 / 7;
            P2 = 2;
        }
    }
    // 3, 5, 7, 9, 15, 21, 35,
    else if (num == 3 || num == 9)
    {
        P0 = 3;
        P1 = 1;
        P2 = num / 3;
    }
    else if (num == 5 || num == 7)
    {
        P0 = num;
        P1 = 1;
        P2 = 1;
    }
    else if (num == 15)
    {
        P0 = 3;
        P1 = 1;
        P2 = 5;
    }
    else if (num == 21)
    {
        P0 = 7;
        P1 = 1;
        P2 = 3;
    }
    else if (num == 35)
    {
        P0 = 7;
        P1 = 1;
        P2 = 5;
    }
}
If (Chosen Central Frequency != NONE)
   getMultiplier(Chosen Divider)
   Program DPLL_CFGCR1 with Chosen DCO Frequency/24
   Program DPLL_CFGCR2 with Chosen DCO Central Frequency, P0 (P), P1 (Q), and P2 (K)
   Return Success // Valid result found
Return Fail  // No valid result found
```

Example of DVI on DDIB using 113.309 MHz symbol clock

This example assumes DPLL3 is available.

Frequency programming algorithm finds DCO central frequency = 9000 MHz, P0=2, P1=4, P2=2, DCO Frequency = 9064.72

1. Configure DPLL3 in the DPLL_CTRL1 register
    - DPLL3 Override = 1b (Enable)

- DPLL3 SSC = 0b (Disable)
- DPLL3 HDMI Mode = 1b (HDMI mode)

2. Configure DPLL3 in the DPLL3_CFGCR1 register

   DCO Integer = INT(9064.72/24) = 377

   DCO Fraction = INT(((9064.72/24) - DCO Integer) * 2^15) = 22,828

   - Frequency Enable = 1b (Enable)
   - DCO Fraction = 592Ch (22828)
   - DCO Integer = 179h (377)
3. Configure DPLL3 in the DPLL3_CFGCR2 register
   - Qdiv Ratio = P1 = 04h (4)
   - Qdiv Mode = 1b (Enable)
   - Kdiv = P2 = 01b (2)
   - Pdiv = P0 = 001b (2)
   - Central Frequency = 01b (9000 MHz)
4. Read back DPLL_CTL1, DPLL3_CFGCR1, and DPLL3_CFGCR2 to ensure writes completed
5. Enable DPLL3 in the WRPLL_CTL2 register
   - PLL Enable = 1b (Enabled)
6. Wait for PLL lock in DPLL_STATUS
7. Configure DPLL3 mapping to DDIB in the DPLL_CTRL2 register
   - DDIB Select Override = 1b (Enable)
   - DDIB Clock Select = 11b (DPLL3)
   - DDIB Clock Off = 0b (On)

Example of HDMI on DDIC using 296.703 MHz symbol clock

This example assumes DPLL2 is available.

Frequency programming algorithm finds DCO central frequency = 9000 MHz, P0=1, P1=3, P2=2, DCO Frequency = 8901.09

1. Configure DPLL2 in the DPLL_CTRL1 register
   - DPLL2 Override = 1b (Enable)
   - DPLL2 SSC = 0b (Disable)
   - DPLL2 HDMI Mode = 1b (HDMI mode)
2. Configure DPLL2 in the DPLL2_CFGCR1 register

   DCO Integer = INT(8901.09/24) = 370

   DCO Fraction = INT(((8901.09/24) - DCO Integer) * 2^15) = 28,794

   - Frequency Enable = 1b (Enable)

- DCO Fraction = 707Ah (28794)
- DCO Integer = 172h (370)

3. Configure DPLL2 in the DPLL2_CFGCR2 register
   - Qdiv Mode = 0b (Disable) // P1=1
   - Kdiv = P2 = 01b (2)
   - Pdiv = P0 = 010b (3)
   - Central Frequency = 01b (9000 MHz)
4. Read back DPLL_CTL1, DPLL2_CFGCR1, and DPLL2_CFGCR2 to ensure writes completed
5. Enable DPLL2 in the WRPLL_CTL1 register
   - PLL Enable = 1b (Enabled)
6. Wait for PLL lock status in DPLL_STATUS
7. Configure DPLL2 mapping to DDIC in the DPLL_CTRL2 register
   - DDIC Select Override = 1b (Enable)
   - DDIC Clock Select = 10b (DPLL2)
   - DDIC Clock Off = 0b (On)

Skylake Sequences for Changing CD Clock Frequency

The CD clock frequency can be programmed to select between several frequencies.

Restrictions
The CD clock frequency impacts the maximum supported pixel rate and display watermark programming. The CD clock frequency must be at least twice the frequency of the Azalia BCLK. Do not select the CD clock frequencies that have been restricted by the Display CDCLK Limit fuse (register DFSM).

Sequence for Changing CD Clock Frequency

1. Disable all display engine functions using the full mode set disable sequence on all pipes, ports, and planes.
   - Includes Global Time Code
2. Inform power controller of upcoming frequency change
   a. Ensure any previous GT Driver Mailbox transaction is complete.
   b. Write GT Driver Mailbox Data0 (GTTMMADDR offset 0x138128) = 0x00000003.
   c. Write GT Driver Mailbox Data1 (GTTMMADDR offset 0x13812C) = 0x00000000.
   d. Write GT Driver Mailbox Interface (GTTMMADDR offset 0x138124) = 0x80000007.
   e. Poll GT Driver Mailbox Interface for Run/Busy indication cleared (bit 31 = 0).
      - Timeout after 150 us. Do not change CD clock frequency if there is a timeout.
   f. Read GT Driver Mailbox Data0, if bit 0 is 0x1, continue, else go to step b.
      - If the condition in step f is not satisfied after cycling through steps b-f for 3 ms (typically <200 us), timeout and do not change CD clock frequency.
3. Change CDCLK_CTL register CD Frequency Select and CD Frequency Decimal to the desired frequency
   - The frequency change will complete within a few clock cycles.

Restrictions
4. Inform power controller of the selected frequency a. Write GT Driver Mailbox Data0 with the frequency selection. • If selecting 337.5 or 308.57 MHz CD clock, or disabling CD clock PLL, write 0x00000000. • If selecting 450 or 432 MHz CD clock, write 0x00000001. • If selecting 540 MHz CD clock, write 0x00000002. • If selecting 675 or 617.14 MHz CD clock, write 0x00000003. b. Write GT Driver Mailbox Data1 = 0x00000000. c. Write GT Driver Mailbox Interface = 0x80000007. • The power controller should complete within 100 us, but there is no need for display software to wait for that. 5. Update programming of register fields that are based on CD clock frequency Programming can be delayed to when the features are enabled. • Utility pin backlight frequency and duty cycle in the BLC_PWM_DATA register.

Resets

NDE_RSTWRN_OPT

The north and south display engines are reset by PCI Function Level Resets (FLR) and the chip level resets.

An FLR for Bus:Device:Function 0:2:0 resets the north and south display engines and audio codec and most of the related MMIO, PCI, and I/O configuration registers.

Display configuration registers that are reset by both the chip level reset and by FLR are marked as using the "soft" reset in the programming specification.

Display configuration registers that are reset only by the chip level reset and *not* by FLR are marked as using the "global" reset in the programming specification.

The south display engine runs panel power down sequencing (if configured to do so) before resetting.

Shared Functions

Fuses and Straps

FUSE_STATUS

DFSM

DSSM

DFSDONE

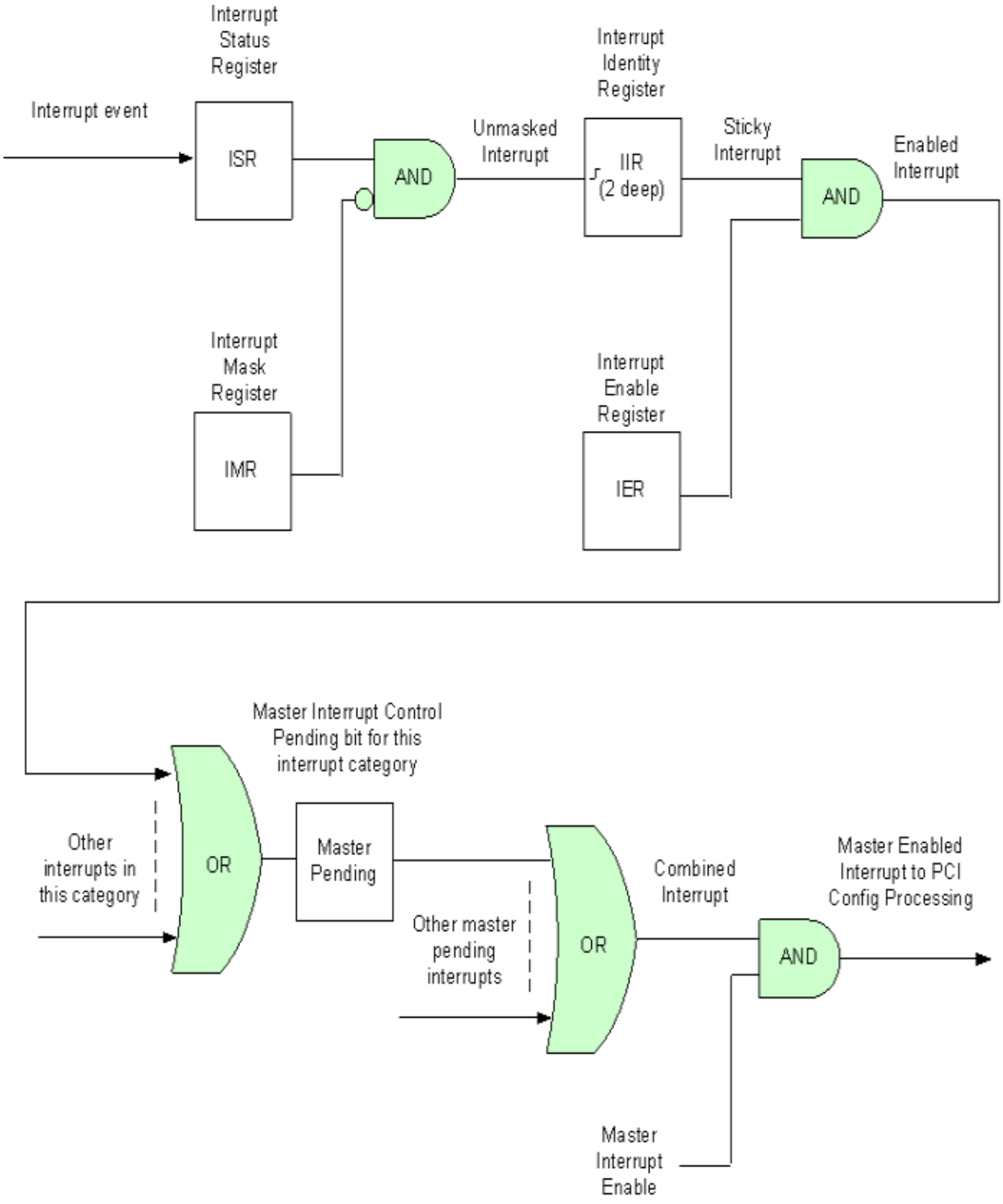
SFUSE_STRAP

Interrupts

MASTER_INT_CTL

Interrupt Flow

First Level Interrupts in Display



1. For every first level interrupt bit
    a. The interrupt event comes in to the interrupt handling logic.
        - There may be more levels of interrupt handling behind each event. For example, the PCH Display interrupt event is the result of the SDE interrupt registers.
    b. The interrupt event goes to the Interrupt Status Register (ISR) where live status can be read back.
        - The live status is mainly useful for hotplug interrupts where it gives the live state of the hotplug line.
        - The live status is not useful for pulse interrupt events due to the short period that the status will be present.
    c. The interrupt event is ANDed with the inverted Interrupt Mask Register (IMR) to create the unmasked interrupt.
        - Only unmasked interrupts will proceed.
    d. The unmasked interrupt rising edge sets the sticky bit in the Interrupt Indentity Register (IIR).
        - The IIR can be cleared by writing a 1 to it.
        - The IIR can queue up to two interrupt events. When the IIR is cleared, it will set itself again if a second event was stored.
    e. The sticky interrupt is ANDed with the Interrupt Enable Register (IER) to create the enabled interrupt.
        - Only enabled interrupts will proceed.
2. All enabled interrupts are then ORed by category (Pipe, Audio Codec, Render, etc.) to create a combined interrupt which is then visible in one of the Master Interrupt Control Register (MASTER_INT_CTL) pending category bits.
3. All pending interrupts are then ORed to create the combined interrupt.
4. The combined interrupt is ANDed with the Master Interrupt Enable (MASTER_INT_CTL Bit 31) to create the master enabled interrupt.
    - Only a master enabled interrupt will proceed.
5. The master enabled interrupt then goes to PCI device 2 configuration registers PCISTS2, PCICMD2, and MC which control the MSI and line interrupt.

A Function Level Reset (FLR) or Warm Reset will reset all graphics interrupt logic, causing the master enabled interrupt to de-assert which can cause the MSI or line interrupt to de-assert.

Interrupt Service Routine

1. Disable Master Interrupt Control
    - Clear bit 31 of MASTER_INT_CTL (0x44200)
    - This is required to prevent missing any interrupts occurring back to back or during the service routine
2. Find the category of interrupt that is pending

- Read MASTER_INT_CTL (0x44200) and record which interrupt pending category bits are set

3. Find the source(s) of the interrupt and clear the Interrupt Identity bits (IIR)
    - Read the IIR associated with each pending interrupt category, record which bits are set, then write back 1s to clear the bits that are set.
4. Process the interrupt(s) that had bits set in the IIRs
5. Re-enable Master Interrupt Control
    - Set bit 31 of MASTER_INT_CTL (0x44200)

Render Response

Display Engine Render Response Message Definition

DE_RRMR

DE_RR_DEST

Arbiter

ARB_CTL

ARB_CTL2

GSA

GTSP0

GTSP1 - Multiple Force Wake

GTSP2

GTSP3

GTSP4

GTSP5

GTSP6

GTSP7

GTFORCEAWAKE

GSA_AUDIO_BDF

GSA_TOUCH_BDF

Data Buffer

DBUF_CTL

DBUF_ECC_STAT

Backlight

This section refers to the CPU display backlight control. For PCH display backlight control, see South Display Engine Registers.

The backlight PWM output frequency is determined by the PWM clock frequency, increment, and frequency divider.

PWM output frequency = PWM clock frequency / PWM increment / PWM frequency divider

The frequency divider must be greater than or equal to the number of brightness levels required by software; typically 100 or 256.

Description
PWM clock frequency = CD clock frequency = 308.57 to 675 MHz PWM increment = 128 or 8, selectable by software PWM frequency divider maximum = 2^16 PWM output frequency range with PWM clock frequency 308.57 MHz and 100 brightness levels and increment 128 = 37 to 24,107 Hz PWM output frequency range with PWM clock frequency 308.57 MHz and 100 brightness levels and increment 8 = 589 to 385,713 Hz PWM output frequency range with PWM clock frequency 308.57 MHz and 256 brightness levels and increment 128 = 37 to 9,417 Hz PWM output frequency range with PWM clock frequency 308.57 MHz and 256 brightness levels and increment 8 = 589 to 150,669 Hz

Backlight Enabling Sequence

Description
1. Enable utility pin, select PWM mode, and set polarity in UTIL_PIN_CTL Util Pin Enable, Util Pin Mode, and Util Pin Output Polarity. 2. Set frequency and duty cycle in BLC_PWM_DATA Backlight Frequency and Backlight Duty Cycle. 3. Enable PWM output and set granularity in BLC_PWM_CTL PWM Enable and PWM granularity. ... 4. Change duty cycle as needed in BLC_PWM_DATA Backlight Duty Cycle.

If needed, granularity and polarity can be programmed earlier than shown.

Backlight Registers

BLC_PWM_CTL

BLC_PWM_DATA

Miscellaneous Shared Functions

UTIL_PIN_CTL - Utility Pin Control

HDPORT_STATE

TOUCH_MSG_ADDR

DOUBLE_BUFFER_CTL

SWF

GTSCRATCH

Central Power

Frame Buffer Compression

FBC Registers

FBC_CFB_BASE

FBC_CTL

DPFC_CONTROL_SA

DPFC_CPU_FENCE_OFFSET

Workaround	
Context:	FBC
To prevent blitter hangs, FBC blitter tracking must use the "Blitter Tracking With Nuke" method.	

Workaround	
Context:	FBC
To prevent missed invalidations around the time FBC is being enabled, FBC render tracking must use the "Render Tracking With Nuke" method.	

FBC Overview

Frame Buffer Compression (FBC) gives a lossless compression of the display frame buffer to save power by reducing system memory read bandwidth and increasing the time between display engine reads to system memory.

FBC is only available on specific plane(s), depending on project. See FBC_CTL for details. FBC compresses pixels for the plane(s) it is attached to as they are displayed. The compressed data is written into the Compressed Frame Buffer (CFB) in graphics data stolen memory. The compressed data is then read the next time the same line needs to be displayed. Changes to the display front buffer (currently displayed

memory surface) through host aperture, render (RCS), and blitter (BCS) are tracked and cause the compressed lines to be invalidated and recompressed. Flips or changes to plane size and panning cause the entire buffer to be recompressed (nuke).

FBC Compression Limit

The FBC compression limit reduces the size of the Compressed Frame Buffer (CFB) by limiting which lines will be compressed. This is used when the graphics stolen memory available for the FBC CFB is smaller than the size of the original uncompressed frame buffer.

When the compression limit is 1:1, every line is written to the CFB, so the CFB width is the same as the original uncompressed frame buffer.

When the compression limit is 2:1, only lines that compress to 1/2 their original size will be written to the CFB, so the CFB width can be 1/2 the original uncompressed frame buffer.

When the compression limit is 4:1, only lines that compress to 1/4 their original size will be written to the CFB, so the CFB width can be 1/4 the original uncompressed frame buffer.

CFB size = ((Stride of plane uncompressed surface / FBC compression limit) * plane vertical source size)

FBC Programming Overview

1. Set up the compressed frame buffer.
    - The compressed buffer resides in graphics data stolen memory.
    - The stolen memory must be contiguous and un-cached.
    - The stolen memory needed for compressed frame buffer must be greater or equal to CFB size (calculation above).
    - Manage the compressed buffer size at run-time by balancing other graphics memory needs with the FBC allocation, and implement appropriate memory needs prioritization schemes.
2. Tracking for CPU host front buffer modifications
    - Setup System Agent (SA) registers to track CPU modifications on the display front buffer.
3. Tracking for display front buffer rendering
    - Setup Target Base Address, Front Buffer Target, and Address Valid for FBC
    - Enable Address Valid before rendering to the front buffer.
    - If needed, send nuke LRIs after each render submission to the display front buffer
4. Tracking from display front buffer BLTs
    - Setup Target Base Address and Address Valid for FBC
    - Enable Address Valid when BLTs target the display front buffer and disable when BLTs do not target the display front buffer
    - Send nuke LRI or cache clean LRI after each blitter submission to the display front buffer

LRI commands to display address 0x50380 are used as part of the render and blitter tracking. Those LRIs must be followed SRM commands to the same address.

LRI to address 0x50380 with data 0x00000004 tells FBC to nuke and invalidate the entire compressed buffer.

LRI to address 0x50380 with data 0x00000002 tells FBC that previous blitter submissions have been flushed to memory and the invalidate lines can now be recompressed.

Render and blitter tracking can invalidate individual lines or nuke the entire buffer. The nuke method is used for workarounds.

Render Tracking With Nuke

- Software must send the nuke LRI after each render to the display front buffer

1. Render commands that touch the display front buffer
    a. Render submission
    b. PIPE_CONTROL
    c. LRI to address 0x50380 with data 0x00000004 (nuke)
    d. SRM to read address 0x50380 and store to a scratch page
2. More render commands that touch the display front buffer
    a. Render submission
    b. PIPE_CONTROL
    c. LRI to address 0x50380 with data 0x00000004 (nuke)
    d. SRM to read address 0x50380 and store to a scratch page
3. Render commands that do not touch the display front buffer
    a. Render submission
    b. PIPE_CONTROL

Render Tracking Without Nuke

- Render tracking must be setup by software, then hardware will track FBC line invalidations automatically.

1. Render commands that potentially touch the display front buffer
    a. LRIs to FBC_RT_BASE_ADDR_REGISTER_<UPPER,LOWER> with display front buffer address, front buffer target=1, base address valid for FBC=1
    b. PIPE_CONTROL to ensure LRI takes effect
    c. Render submission
    d. PIPE_CONTROL
2. More render commands that touch the display front buffer
    a. Render submission
    b. PIPE_CONTROL
3. Render commands that do not touch the display front buffer

a. Optional: LRIs to FBC_RT_BASE_ADDR_REGISTER_<UPPER,LOWER> with base address valid for FBC=0. Optional because hardware will detect that the render is not to the display front buffer address.
b. Optional: PIPE_CONTROL to ensure LRI takes effect
c. Render submission
d. PIPE_CONTROL

Blitter Tracking With Nuke

- Software must send the nuke LRI after each BLT to the display front buffer. Never set 221d0h bit 3 (Address Valid for FBC).

1. BLT commands that touch the display front buffer
   a. BLT submission
   b. MI_FLUSH_DW
   c. LRI to address 0x50380 with data 0x00000004 (nuke)
   d. SRM to read address 0x50380 and store to a scratch page
2. More BLT commands that touch the display front buffer (can leave Address Valid set from previous)
   a. BLT submission
   b. MI_FLUSH_DW
   c. LRI to address 0x50380 with data 0x00000004 (nuke)
   d. SRM to read address 0x50380 and store to a scratch page
3. BLT commands that do not touch the display front buffer
   a. BLT submission
   b. MI_FLUSH_DW

Blitter Tracking Without Nuke

- Blitter tracking must be enabled by software only when BLTs touch the display front buffer. Software must send the cache clean LRI after each BLT to the display front buffer.

1. BLT commands that definitely touch the front buffer
   a. LRI to address 221d0h to set bit 3 (Address Valid for FBC)
   b. MI_FLUSH_DW to ensure LRI takes effect
   c. BLT submission
   d. MI_FLUSH_DW
   e. LRI to address 0x50380 with data 0x00000002 (cache clean)
   f. SRM to read address 0x50380 and store to a scratch page
2. More BLT commands that touch the front buffer (can leave Address Valid for FBC set from previous)

a. BLT submission
b. MI_FLUSH_DW
c. LRI to address 0x50380 with data 0x00000002 (cache clean)
d. SRM to read address 0x50380 and store to a scratch page

3. BLT commands that do not touch the front buffer (must clear Address Valid for FBC)
   a. LRI to address 221d0h to set bit 3 (Address Valid for FBC)
   b. MI_FLUSH_DW to ensure LRI takes effect
   c. BLT submission
   d. MI_FLUSH_DW

CPU Host Aperture Tracking

- Host tracking must be setup by software, then hardware will track FBC line invalidations to the specified fence.

1. Program Y offset from the CPU fence to the Display Buffer base in DPFC_CPU_FENCE_OFFSET.
2. Program CPU fence ID and enable CPU fence tracking in DPFC_CONTROL_SA.
3. Disable CPU Fence tracking in DPFC_CONTROL_SA if no tiling fence is mapped to display buffer.

Display Plane Enabling with FBC

- FBC has to be enabled after the attached display plane is enabled.
- This is the general sequence. See FBC_CTL for any workarounds.

1. Enable display plane
2. Enable FBC as described in FBC_CTL register

Display Plane Disabling with FBC

- FBC has to be disabled prior to FBC attached display plane disabling.
- This is the general sequence. See FBC_CTL for any workarounds.

1. Disable FBC as described in FBC_CTL register
2. Disable display plane.

Watermarks

The watermark registers are used to control the display to memory request timing. The watermarks must be programmed according to the Display Watermark Programming section.

Description or Link
WM_PIPE, WM_LP, and WM_LP_SPR have been replaced by PLANE_WM and moved into the planes section.
WM_MISC **WM_LINETIME**

DC States

DC_STATE_EN

Power Wells

When a power well is disabled (powered down), access to any registers in the power well will complete, but write data will be dropped and read data will be all zeroes.

The power well enable requests from all sources are logically ORd together to enable a power well, so the power well will only disable after all sources have requested the power well to disable.

PWR_WELL_CTL

Pipe

Color Space Conversion

CSC_COEFF

CSC COEFFICIENT FORMAT

CSC_PREOFF

CSC_POSTOFF

CSC_MODE

The high color channel is the most significant bits of the color. The low color channel is the least significant bits of the color. The medium color channel is the bits between high and low. For example: In RGB modes Red is in the High channel, Green in Medium, and Blue in Low. In YUV modes, V is in the High channel, Y in Medium, and U in Low.

The color space conversion registers are double buffered and are updated on the start of vertical blank following a write to the CSC Mode register for the respective pipe.

The matrix equations are as follows:

OutputHigh = (CoefficientRY * InputHigh) + (CoefficientGY * InputMedium) + (CoefficientBY * InputLow)

OutputMedium = (CoefficientRU * InputHigh) + (CoefficientGU * InputMedium) + (CoefficientBU * InputLow)

OutputLow = (CoefficientRV * InputHigh) + (CoefficientGV * InputMedium) + (CoefficientBV * InputLow)

Example programming for RGB to YUV is in the following table:

The input is RGB on high, medium, and low channels respectively and the desired YUV output is VYU on high, medium, and low channels respectively.

Program CSC_MODE to put gamma before CSC.

Program the CSC Post-Offsets to +1/2, +1/16, and +1/2 for high, medium, and low channels respectively.

The coefficients and pre and post offsets can be scaled if desired.

	Bt.601		Bt.709	
	Value	Program	Value	Program
RU	0.2990	0x1990	0.21260	0x2D98
GU	0.5870	0x0968	0.71520	0x0B70
BU	0.1140	0x3E98	0.07220	0x3940
RV	-0.1687	0xAAC8	-0.11460	0xBEA8
GV	-0.3313	0x9A98	-0.38540	0x9C58
BV	0.5000	0x0800	0.50000	0x0800
RY	0.5000	0x0800	0.50000	0x0800
GY	-0.4187	0x9D68	-0.45420	0x9E88
BY	-0.0813	0xBA68	-0.04580	0xB5E0

Example programming for YUV to RGB is in the following table:

The input is VYU on high, medium, and low channels respectively.

The output is RGB on high, medium, and low channels respectively.

Program CSC_MODE to put gamma after CSC.

Program the CSC Pre-Offsets to -1/2, -1/16, and -1/2 for high, medium, and low channels respectively.

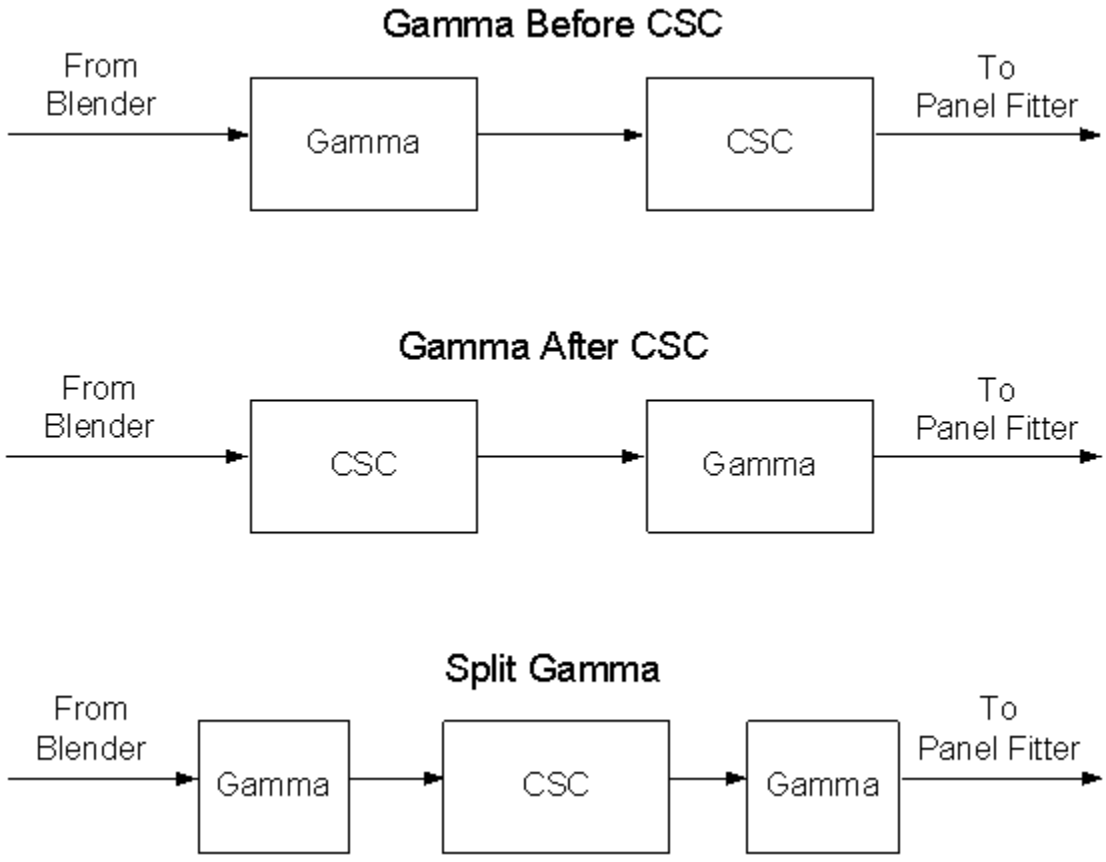
The coefficients and pre and post offsets can be scaled if desired.

	Bt.601 Reverse		Bt.709 Reverse	
	Value	Program	Value	Program
GY	1.000	0x7800	1.000	0x7800
BY	0.000	0x0000	0.000	0x0000
RY	1.371	0x7AF8	1.574	0x7C98
GU	1.000	0x7800	1.000	0x7800
BU	-0.336	0x9AC0	-0.187	0xABF8
RU	-0.698	0x8B28	-0.468	0x9EF8
GV	1.000	0x7800	1.000	0x7800
BV	1.732	0x7DD8	1.855	0x7ED8
RV	0.000	0x0000	0.000	0x0000

The pipe gamma and color space conversion blocks can be placed in three different arrangements:

- Gamma before CSC: Selected through the CSC Mode register. Mostly used for RGB to YUV conversion.
- Gamma after CSC: Selected through the CSC Mode register. Mostly used for YUV to RGB conversion or linear RGB to RGB conversion. Can be used with pipe color gamut enhancement.
- Split gamma: Selected through the Gamma Mode register. Mostly used for RGB to RGB conversion. Can be used with pipe color gamut enhancement. The pipe gamma enable per plane will control whether a plane will go through both gamma blocks. It is not possible to send a plane through one gamma block and not the other.

In either arrangement, the final output of the pipe gamma and CSC and gamut enhancement logic is clamped to fit in the 0 to 1.0 range before going to the ports.



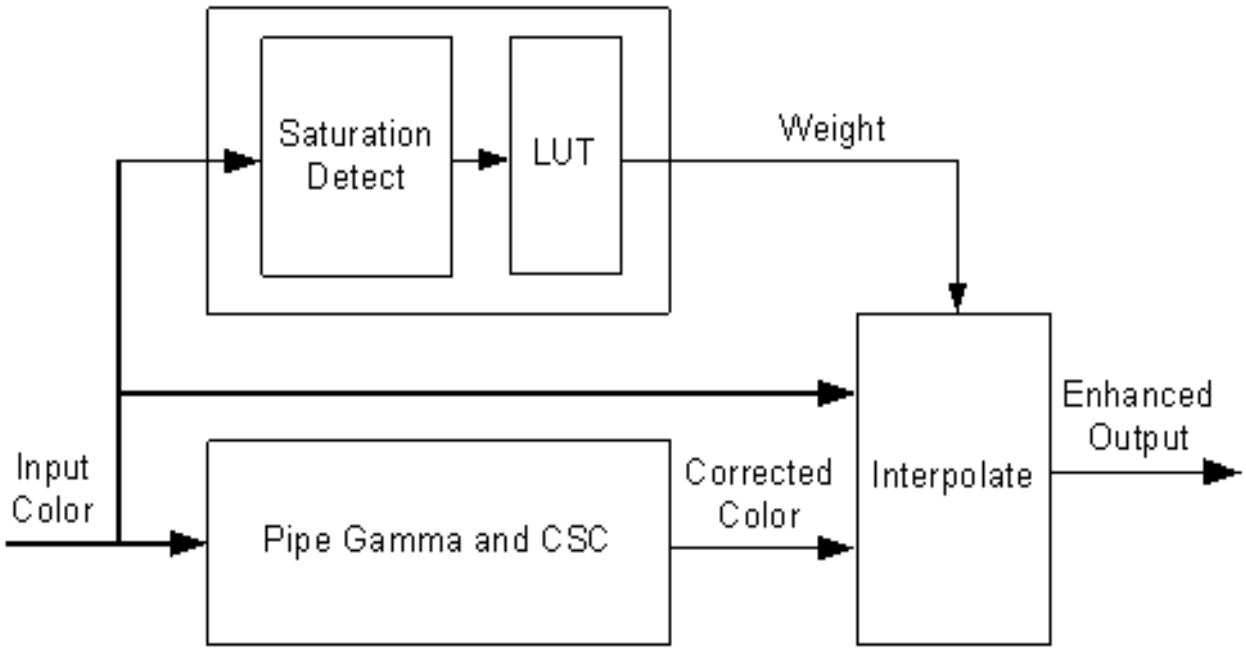
Pipe Color Gamut Enhancement

Pipe color gamut enhancement is used to enhance display of standard gamut content on wide gamut displays. It processes the color value from before and after the pipe gamma and color space correction blocks to create the color gamut enhanced output. The typical usage is to output the pipe gamma and CSC corrected color for areas of low saturated content and the input (not gamma or CSC corrected) color for areas of high saturated content. It is not recommended to use color gamut enhancement with wide gamut inputs.

CGE_CTRL

CGE_WEIGHT

The pipe Gamma and CSC must be programmed to either the split gamma mode or gamma after CSC mode when using pipe color gamut enhancement.



The saturation level of the pipe gamma and CSC input color is detected and used to index into a look up table (LUT) containing programmable weights. The saturation values are linearly distributed across the LUT indexes from the lowest index for lowest saturation to the highest index for highest saturation.

The enhanced output color is created by using the weight value to interpolate between the input color and corrected color. See the following table of weights to amount of input or corrected color used to create the enhanced output color.

Weighting of input and corrected colors

Weight from LUT	Amount of Input Color in Enhanced Output	Amount of Corrected Color in Enhanced Output
00 0000b (minimum)	0%	100%
...	...	...
00 1000b	25%	75%
...	...	...
01 0000b	50%	50%
...	...	...
01 1000b	75%	25%
...	...	...
10 0000b (maximum)	100%	0%

Example weight programming

CGE LUT Index	CGE Weight Value Decimal	CGE Weight Value Binary	CGE Weight Percent Input Color	CGE Weight Percent Corrected Color
0 (lowest saturation)	0	00 0000b	0%	100%
1	0	00 0000b	0%	100%
2	0	00 0000b	0%	100%
3	0	00 0000b	0%	100%
4	0	00 0000b	0%	100%
5	0	00 0000b	0%	100%
6	1.6	00 0010b	5%	95%
7	3.2	00 0011b	10%	90%
8	4.8	00 0101b	15%	85%
9	6.4	00 0110b	20%	80%
10	8.64	00 1001b	27%	73%
11	12.8	00 1101b	40%	60%
12	19.2	01 0011b	60%	40%
13	25.6	01 1010b	80%	20%
14	28.8	01 1101b	90%	10%
15	32	10 0000b	100%	0%
16 (highest saturation)	32	10 0000b	100%	0%

Pipe DPST

DPST_CTL

DPST_BIN

DPST_GUARD

Pipe Palette and Gamma

The display palette provides a means to correct the gamma of an image stored in a frame buffer to match the gamma of the monitor or presentation device. Additionally, the display palette provide a method for converting indexed data values to color values for VGA and 8-bpp indexed display modes. The display palette is located after the plane blender. Using the individual plane gamma enables, the blended pixels can go through or bypass the palette on a pixel by pixel basis.

PAL_LGC

PAL_PREC_INDEX

PAL_PREC_DATA

PAL_GC_MAX

PAL_EXT_GC_MAX

GAMMA_MODE

If any gamma value to be programmed exceeds the maximum allowable value in the associated gamma register, then the programmed value must be clamped to the maximum allowable value.

Programming Modes

The display palette can be accessed through multiple methods and operate in one of four different modes as follows.

8 bit legacy palette/gamma mode:

This provides a palette mode for indexed pixel data formats (VGA and primary plane 8 bpp) and gamma correction for legacy programming requirements.

All input values are clamped to the 0.0 to 1.0 range before the palette/gamma calculation. It is not recommended to use legacy palette mode with extended range formats.

For input values greater than or equal to 0 and less than 1.0, the input value is used to directly lookup the result value from one of the 256 palette/gamma entries. The 256 entries are stored in the legacy palette with 8 bits per color in a 0.8 format with 0 integer and 8 fractional bits.

The legacy palette is programmable through both MMIO and VGA I/O registers. Through VGA I/O, the palette can look as though there are only 6 bits per color component, depending on programming of other VGA I/O registers.

10 bit gamma mode:

This provides the highest quality gamma for pixel data formats of 30 bits per pixel or less.

All input values are clamped to the greater than -3.0 and less than 3.0 range before the gamma calculation.

For input values greater than or equal to 0 and less than 1.0, the input value is used to directly lookup the result value from one of the first 1024 gamma entries. The first 1024 entries are stored in the precision palette with 10 bits per color in a 0.10 format with 0 integer and 10 fractional bits.

For input values greater than or equal to 1.0 and less than 3.0, the input value is used to linearly interpolate between the 1024th and 1025th gamma entries to create the result value. The 1025th entry is stored in the PAL_EXT_GC_MAX register with 19 bits per color in a 3.16 format with 3 integer and 16 fractional bits.

For negative input values, gamma is mirrored along the X-axis, giving the same result as positive input values, except for a negative sign. When gamma input may be negative, the first gamma point should be programmed to a value of 0.0 in order to have a symmetric mirroring.

Split gamma mode:

Split gamma mode is composed of two gamma functions. The first gamma is before pipe color space conversion (CSC) and the second is after CSC. This split gamma mode permits mapping to linear gamma, then color space conversion, then mapping to monitor gamma. This provides the highest quality pipe color space conversion and gamma correction for inputs with non-linear gamma.

First gamma (before CSC):

> All input values are clamped to the greater than -3.0 and less than 3.0 range before the gamma calculation.
>
> For input values greater than or equal to 0 and less than 1.0, the input value is used to directly lookup the result value from one of the first 512 gamma entries. The first 512 entries are stored in the precision palette indexes 0 to 511 with 10 bits per color in a 0.10 format with 0 integer and 10 fractional bits.
>
> For input values greater than or equal to 1.0 and less than 3.0, the input value is used to linearly interpolate between the 512th and 513th gamma entries to create the result value. The 513th entry is stored in the PAL_EXT_GC_MAX register with 19 bits per color in a 3.16 format with 3 integer and 16 fractional bits.
>
> For negative input values, gamma is mirrored along the X-axis, giving the same result as positive input values, except for a negative sign. When gamma input may be negative, the first gamma point should be programmed to a value of 0.0 in order to have a symmetric mirroring.

Second gamma (after CSC):

> All input values are clamped to the 0.0 to 1.0 range before the gamma calculation.

For input values greater than or equal to 0 and less than 1.0, the input value is used to directly lookup the result value from one of the first 512 gamma entries. The first 512 entries are stored in the precision palette indexes 512 to 1023 with 10 bits per color in a 0.10 format with 0 integer and 10 fractional bits.

12 bit interpolated gamma mode:

This provides the highest quality gamma for pixel data formats greater than 30 bits per pixel.

The gamma correction curve is represented by specifying a set of gamma entry reference points spaced equally along the curve for values between -1 and 1. For extended values there is an extended gamma entry reference point at the maximum allowed input value.

All input values are clamped to the greater than -3.0 and less than 3.0 range before the gamma calculation.

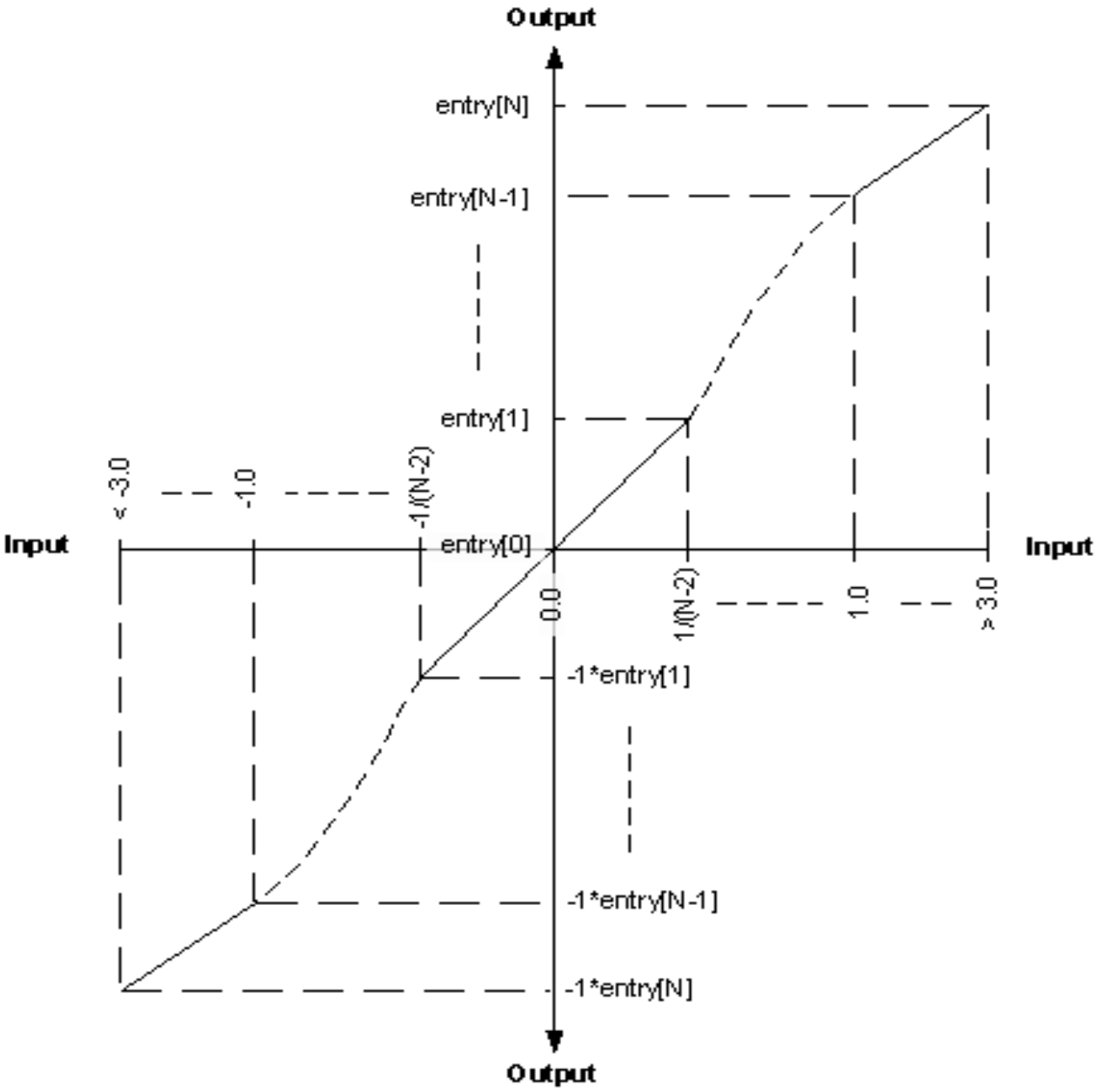
For input values greater than or equal to 0 and less than 1.0, the input value is used to linearly interpolate between two adjacent points of the first 513 gamma entries to create the result value. The first 512 entries are stored in the precision palette with 16 bits per color in a 0.16 format with 0 integer and 16 fractional bits (upper 10 bits in odd indexes, lower 6 bits in even indexes). The 513th entry is stored in the PAL_GC_MAX register with 17 bits per color in a 1.16 format with 1 integer and 16 fractional bits.

For input values greater than or equal to 1.0 and less than 3.0, the input value is used to linearly interpolate between the 513th and 514th gamma entries to create the result value. The 514th entry is stored in the PAL_EXT_GC_MAX register with 19 bits per color in a 3.16 format with 3 integer and 16 fractional bits.

For negative input values, gamma is mirrored along the X-axis, giving the same result as positive input values, except for a negative sign. When gamma input may be negative, the first gamma point should be programmed to a value of 0.0 in order to have a symmetric mirroring.

To program the gamma correction entries, calculate the desired gamma curve for inputs from 0 to 3.0. The curve must be flat or increasing, never decreasing. For inputs of 0 to 1.0, multiply the input value by 512 to find the gamma entry number, then store the desired gamma result in that entry. For inputs greater than 1.0 and less than or equal to 3.0, store the result for an input of 3.0 in the 514th gamma entry.

Example Pipe Gamma Correction Curve



Pipe Control

PIPE_SRCSZ

PIPE_SCANLINE

PIPE_SCANLINECOMP

PIPE_MISC

PIPE_BOTTOM_COLOR

PIPE_FRMCNT

PIPE_FLIPCNT

PIPE_FRMTMSTMP

PIPE_FLIPTMSTMP

Pipe Scaler

Each scaler has its own set of registers.

PS_WIN_POS

PS_WIN_SZ

PS_CTRL

PS_VSCALE

PS_HSCALE

PS_VPHASE

PS_HPHASE

PS_ECC_STAT

Planes

Plane Capability and Interoperability

Plane Assignments and Capabilities

Plane Capabilities:

	Pipe A	Pipe B	Pipe C
Plane 3	Plane or Cursor	Plane or Cursor	Plane or Cursor
Plane 2	Plane + Render Decompression	Plane + Render Decompression	Plane
Plane 1	Plane + Render Decompression, Frame Buffer compression	Plane + Render Decompression	Plane

Mapping to Command Streamer Plane Number

Display Plane Name	Command Streamer Plane Number
Plane 1 A	Plane 1
Plane 1 B	Plane 2
Plane 1 C	Plane 3
Plane 2 A	Plane 4
Plane 2 B	Plane 5
Plane 2 C	Plane 6
Plane 3 A	Plane 7
Plane 3 B	Plane 8
Plane 3 C	Plane 9
N/A	Plane 10
N/A	Plane 11
N/A	Plane 12

Plane Feature Interoperability

Display Features / Surface Formats:

	RGB32 2:10:10:10	RGB32 8:8:8:8	RGB 32-bit XR_BIAS 10:10:10	RGB64 16:16:16:16	RGB 16-bit 5:6:5	Indexed 8-bit	YUV 32-biwt 4:4:4	YUV 16-bit 4:2:2
Linear - 0/180 rotation	✓	✓	✓	✓	✓	✓	✓	✓
Linear - 90/270 rotation								
X Tiling - 0/180 rotation	✓	✓	✓	✓	✓	✓	✓	✓
X Tiling - 90/270 rotation								
Y Tiling (legacy) - 0/180 rotation	✓	✓	✓	✓	✓	✓	✓	✓
Y Tiling (legacy) - 90/270 rotation	✓	✓	✓				✓	✓
Yf Tiling - 0/180 rotation	✓	✓	✓		✓		✓	✓
Yf Tiling - 90/270 rotation	✓	✓	✓				✓	✓
Frame Buffer Compression		✓			✓			
Render Compression		✓						
Plane Scaling	✓	✓			✓		✓	✓
Per-pixel Alpha		✓						
Color Keying	✓	✓	✓	✓	✓	✓	✓	✓

Tiling Modes / Display Features:

	Rotation 0/180	Rotation 90/270	Render Decompression	Frame Buffer Compression	Interlacing (Interlaced Fetch)
Linear	✓			✓	✓
X Tiling	✓			✓	✓
Y Tiling Legacy	✓	✓ (some surface formats not supported)	✓	✓	
Yf Tiling	✓ (some surface formats not supported)	✓ (some surface formats not supported)	✓	✓	

Rotation Modes / Display Features:

	Interlacing	Frame Buffer Compression	Render Decompression
0/180 rotation	✓	✓	✓
90/270 rotation		✓	

Render/Display Decompression

GT Render engines uses a lossless scheme to compress the color Render targets. The goal of the compression is to reduce the memory bandwidth. The memory foot print increases slightly due to the need of the control surface.

- Decompression is supported on plane 1 and plane 2 of pipe A and pipe B with RGB8888 surface formats.
- Decompression is supported only in legacy tile Y and tile Yf surfaces.
- Decompression support is limited to left-right cache line pair mode. Top-bottom mode is not supported.
- When render decompression is enabled, display engine internally converts the Async flips to Sync flips.
- Decompression is not supported with 90/270 degree rotation.

Color Control Surface

The Color Control Surface (CCS) contains the compression status of the cache-line pairs**.** The compression state of the cache-line pair is specified by 2 bits in the CCS. Each CCS cache-line represents an area on the main surface of 16 x16 sets of 128 byte Y-tiled cache-line-pairs. CCS is always Y tiled. The

address of CCS surface is specified as an offset from the start of the Render Target main surface. CCS stride is programmed separately independent of the main surface stride.

Decompression Programming

When compressed Render targets are presented to Display, the display decompression must be enabled. Along with main surface programming, the following additional programming is required to enable the decompression

- **Decompression Enable**

  Decompression is enabled by programming the 'Render Decomp' bit in the PLANE_CTL register.

- **Color Control Surface Distance**

  The start of the CCS surface is programmed as the distance in bytes from the start of the main surface in the PLANE_AUX_DIST register. The CCS is always placed after the main surface and is 4K page aligned.

- **Color Control Surface Stride**

  The CCS stride is programmed in the PLANE_AUX_DIST register.

Plane Rotation Programming

This topic provides programming information for plane rotation.

The 180 rotation mode is unchanged and will continue to use the same programming. In the 180 rotation mode display hardware is responsible for walking the pages in the reverse order. The cacheline walk within the page is also reversed. The 90 and 270 rotation modes require more complicated page walk mechanism. The page walk is made transparent to the hardware by providing a different set of page translations (remapped GTT) for the same rendered surface. The remapping completely abstracts the page walk away from the hardware and the hardware walks the pages as if there is no rotation. Hardware is still responsible for handling the walk within the page appropriately. Also, the 90, 270 Rotation requires a new parameter - surface height. The changes needed in the driver programming is discussed below.

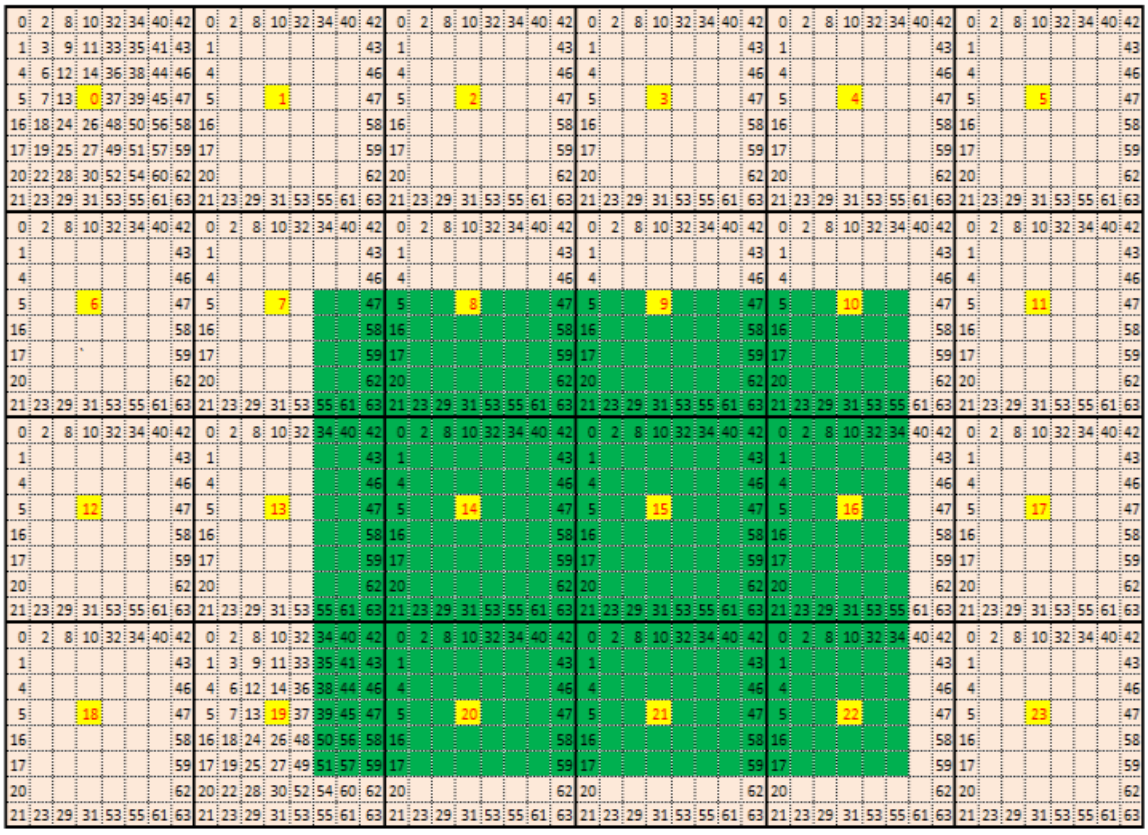
90 Rotation

For the 90 rotation programming, the plane parameters must use the following mapping

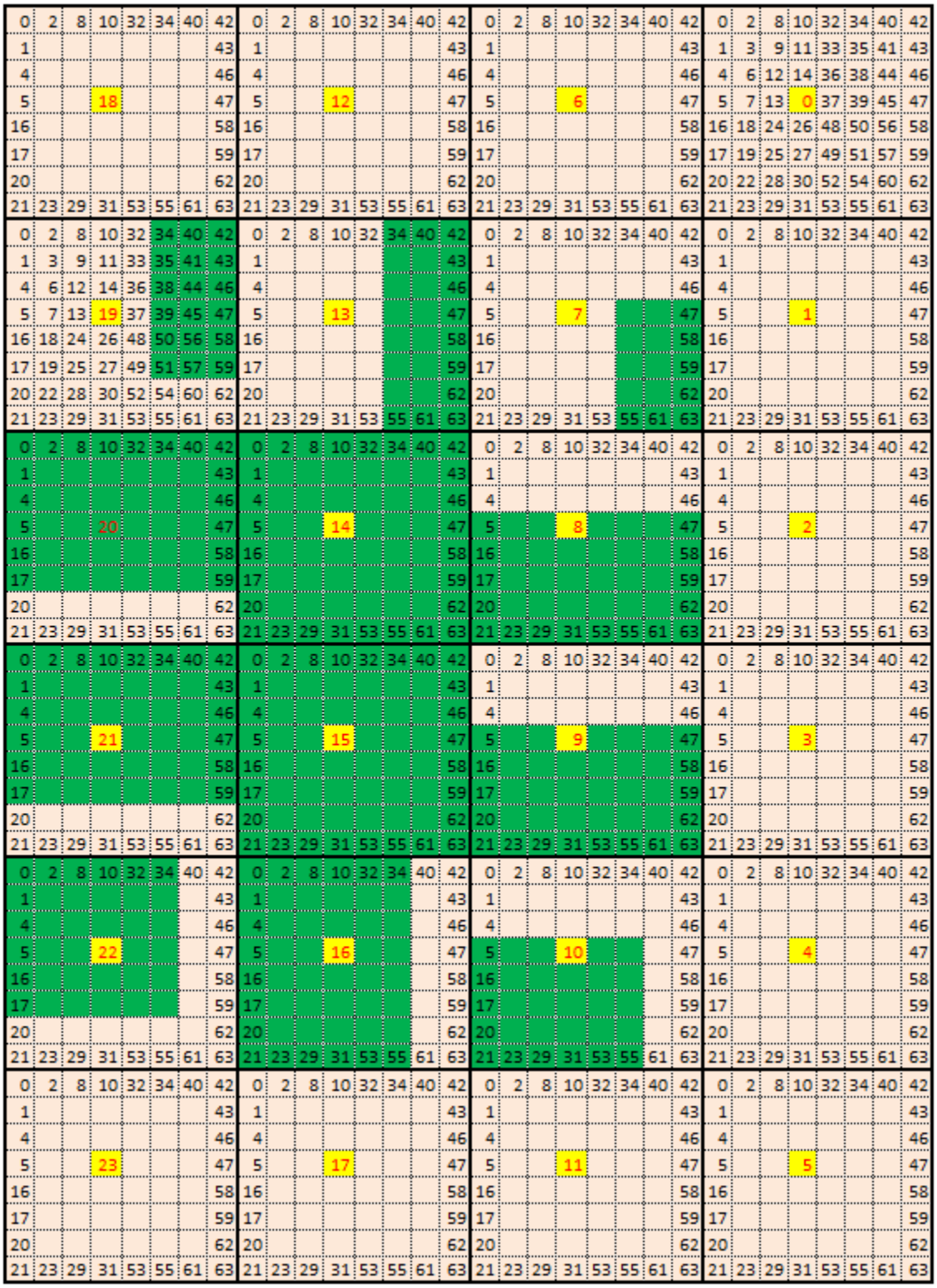
- Base address = New address (remapped GTT)
- Stride = Surface height in tiles
- Plane Width = Plane Height
- Plane Height = Plane Width
- X offset = (Surface height in tiles * tile height) - Plane Y offset - Plane Height *[Note: the calculated X offset will always be >= 0 since Plane Y Offset + Plane Height <= Surface Height in lines]*

- Y offset = X offset
- Driver can enable the Scaler, as needed, to fit the rotated content in to the pipe active display area.

Sample surface in frame buffer (Yf tiled)



Page access sequence in the rotated 90 mode



Example

Let us assume the following display programming for a single pipe – single plane, Yf tiled, non-rotated, 1920 x 1200, 4Bpp with the plane panned (100, 150), covering full active area and scaler not enabled.

GTT mapping

Here is a sample GTT mapping for 90 rotation mode.

Original GTT

Assumed Surface base = 0x200000

0x200000 = page 0

0x201000 = page 1

0x202000 = page 2

0x203000 = page 3

…

Remapped Display GTT – 90 rotation

Assumed new Surface base = 0x400000

0x400000 = page 18

0x401000 = page 12

0x402000 = page 6

0x403000 = page 0

Register programming for non-rotated scenario

- *PLANE_SURF->Surface Base Address = 0x200000*
- *PLANE_STRIDE->Stride = 60* [(1920 * 4)/128] [(width *bpp)/tile width]
- *PLANE_SIZE->Width = 1920*
- *PLANE_SIZE->Height = 1200*
- *PLANE_OFFSET->Start X Position = 100*
- *PLANE_OFFSET->Start Y Position = 150*
- *Surface Height in tiles (assumed) = 50 (allocated surface height in number of scan lines/tile height. For plane height = 1200, the surface height should be a minimum of 38 tiles (ceiling (1200/32). When panning is used, the rendered frame buffer surface will be larger than the plane size. Here, let us assume that the rendered surface height in tiles = 50).*

The programming changes to following for a 90 rotation scenario

- *PLANE_SURF->Surface Base Address = 0x400000* [uses remapped GTT]
- *PLANE_STRIDE->Stride = 50* [Surface height in tiles (assumed earlier)]
- *PLANE_SIZE->Width = 1200* [non-rotated Height]
- *PLANE_SIZE->Height = 1920* [non-rotated Width]

- *PLANE_OFFSET->Start X Position* = *250* [(50*32)-150-1200] [(Surface height * tile height) – non rotated Y position – non rotated Height]
- *PLANE_OFFSET->Start Y Position* = *100* [non-rotated X position]

The scaler should be programmed appropriately to fit the rotated plane in the pipe active area and the window position should be adjusted if it is desired to maintain the same apparent position on a physically rotated display.

270 rotation

Uses the same GTT remapping and register programming as 90 rotation mode with the Plane control register rotation mode set as 270.

Display Buffer Programming

This topic describes display buffer allocation and shows a basic allocation method for single and multi-pipe modes. The display driver can choose to use more advanced allocation techniques as desired.

Display Buffer Allocation

Allocation of the display buffer is programmable for each display plane, using the buffer start and buffer end values in PLANE_BUF_CFG.

Proper display buffer allocation is important for Display hardware to function correctly. Optimal allocation provides better display residency in memory low power modes. Display Buffer allocation must be recalculated and programmed when pipes/planes get enabled or disabled.

Display Buffer Size

Display Buffer Size	Total Display Buffer Blocks	Fixed Bypass Path Allocation in Blocks	Blocks Available for Driver Programming
448 KB	896	4	0 - 891

Each display buffer block is 8 cache lines.

Allocation Requirements

Allocation must not overlap between any enabled planes.

A minimum allocation is required for any enabled plane. See Minimum Allocation Requirements below.

A gap between allocation for enabled planes is allowed.

The allocation for enabled planes should be as large as possible to allow for higher watermarks and better residency in memory power saving modes.

Minimum Allocation Requirements

Allocation for each enabled plane must meet these minimum requirements.

Planes using Linear or X tiled memory formats must allocate a minimum of 8 blocks.

Planes using Y tiled memory formats must allocate blocks for a minimum number of scanlines worth of data. The formula and table of minimum scanlines is below.

Y tiled minimum allocation = Ceiling [(4 * Plane source width * Plane Bpp)/512] * MinScanLines/4 + 3

Plane Bpp	Minimum Scanlines for Y Tile	
	0/180 Rotation	90/270 Rotation
1	8	32
2	8	16
4	8	8
8	8	4

Basic Allocation Method

These are basic methods that can be used for single and multi-pipe modes. For optimal power usage, the display driver can choose to use more advanced allocation techniques as desired.

Single Pipe

Allocate a fixed number of blocks to cursors and then allocate the remaining blocks among planes, based on each plane's data rate.

BlocksAvailable = TotalBlocksAvailable

1. Allocate 32 blocks for cursor

   The driver frequently enables and disables the cursor or changes the cursor pixel format. Fixed allocation is preferred for cursor to minimize the buffer re-allocation. More allocation might be required to support deeper low power states (based on the results of watermark calculations).

   CursorBufAlloc = 32

   BlocksAvailable = BlocksAvailable – 32

2. Check for minimum buffer requirement

   For each enabled plane

   If Y tiled

   MinScanLines = Look up minimum scanlines needed from the table

   *PlaneMinAlloc = Ceiling [(4*Plane width*Bpp)/512] * MinScanLines/4 + 3*

   Else

   PlaneMinAlloc = 8

   If sum of PlaneMinAlloc > BlocksAvailable

   Error - Display Mode can't be supported.

   The driver can change the number of enabled planes or the plane configuration and rerun the algorithm.

3. Calculate Relative Data Rate for planes

   In this step the driver may want to use the expected maximum plane source sizes so it does not have to reallocate for a plane that is changing size.

   For each enabled plane

   If PlaneScalerEnabled

   *PlaneScaleFactor = (Plane width/Scaler window X size) * (Plane height/Scaler window Y size)*

   Else

   PlaneScaleFactor = 1

   *PlaneRelativeDataRate = Plane height * Plane width * plane source bytes per pixel * PlaneScaleFactor*

4. Allocate blocks for enabled planes as per the Data rate ratio.

   For each plane that needs allocation (PlaneBlockAllocFinal == false)

   *PlaneBufAlloc = floor (BlocksAvailable * PlaneRelativeDataRate/Sum of PlaneRelativeDataRate of all planes that need allocation).*

   *floor - rounds down to an integer value dropping the fractional part.

5. Adjust for minimum allocation requirement

   AdjustmentRequired = false

   For each plane needs allocation (PlaneBlockAllocFinal == false)

   If PlaneBufAlloc < PlaneMinAlloc

   AdjustmentRequired = true

   PlaneBufAlloc = PlaneMinAlloc

   PlaneBlockAllocFinal = true

   BlocksAvailable = BlocksAvailable - PlaneMinAlloc

   If AdjustmentRequired = true

   Go back to step 4

Multi-Pipe

NumPipes = Total number of display pipes in the hardware

Allocate a fixed number of blocks to cursors, allocate *1/NumPipes* of the remaining blocks to each pipe, then calculate each pipe individually as in the Single Pipe case.

1. Allocate 8 blocks for cursor per pipe

   The driver frequently enables and disables the cursor or changes the cursor pixel format. Fixed allocation is preferred for cursor to minimize the buffer re-allocation. More allocation might be required to support deeper low power states (based on the results of watermark calculations).

   For each enabled cursor

   CursorBufAlloc = 8

*BlocksAvailable = TotalBlocksAvailable - (8 * NumPipes)*

2. Distribute the blocks equally among the pipes

   BlocksAvailable = BlocksAvailable/NumPipes

3. Assign blocks to the planes

   For each pipe

Perform Single Pipe sequence, starting from step 2.

Buffer allocation re-distribution

When an additional pipe is getting enabled, or an existing pipe requires more buffer to support a new mode or is disabled, buffer reallocation may be necessary for proper display functionality. Whenever a portion of the allocated buffer is taken away from one pipe and allocated to a different pipe, the following sequence should be followed to make sure that there are no buffer allocation overlaps at any point of time.

1. *For each pipe whose allocation is reduced*
   a. *Program the new buffer allocation.*
   b. *Wait for VBlank of that pipe for new allocation to update.*
2. *For each pipe whose allocation is increased*
   a. *Program the new buffer allocation.*
   b. *Wait for VBlank of that pipe for new allocation to update.*

Display Buffer Allocation and Watermark programming prior to OS boot

Basic programming of the display buffer and watermarks to allow limited display usage prior to OS boot:

This will prevent package power saving states from enabling.

Supported usages:

- Up to 3 pipes enabled at once
- Up to one universal plane enabled per pipe. No cursor.
- Linear or Xtile memory
- Any RGB frame buffer pixel format 32bpp or less, without render compression
- Any screen resolution
- Downscaling less than or equal to 12.5%

Allocate 160 blocks per pipe.

Pipe A: 0-159, Pipe B: 160-319, Pipe C: 320-479

PLANE_BUF_CFG_<plane number>_A = 0x009F0000

PLANE_BUF_CFG_<plane number>_B = 0x013F00A0

PLANE_BUF_CFG_<plane number>_C = 0x01DF0140

Set level 0 watermarks for any enabled plane to 160 blocks and 2 lines.

PLANE_WM_<plane number>_<pipe>_0 = 0x800080A0

The higher level watermarks for any enabled plane must have bit 31=0 to keep the low power watermarks disabled.

VGA

The VGA Control register is located here. The VGA I/O registers are located in the VGA Registers document.

VGA_CONTROL

Cursor Plane

CUR_CTL

CUR_BASE

CUR_POS

CUR_PAL

CUR_FBC_CTL

PLANE_SURFLIVE

CUR_SURFLIVE

PLANE_BUF_CFG

PLANE_WM

The CUR_CTL and CUR_FBC_CTL active registers will be updated on the vertical blank or when pipe is disabled, after the CUR_BASE register is written, or when cursor is not yet enabled, providing an atomic update of those registers together with the CUR_BASE register.

Universal Plane

PLANE_CTL

PLANE_STRIDE

PLANE_POS

PLANE_SIZE

PLANE_SURF

PLANE_LEFT_SURF

PLANE_AUX_DIST

PLANE_OFFSET

PLANE_KEYVAL

PLANE_KEYMSK

PLANE_KEYMAX

PLANE_SURFLIVE

PLANE_WM

PLANE_BUF_CFG

PLANE_GAMC

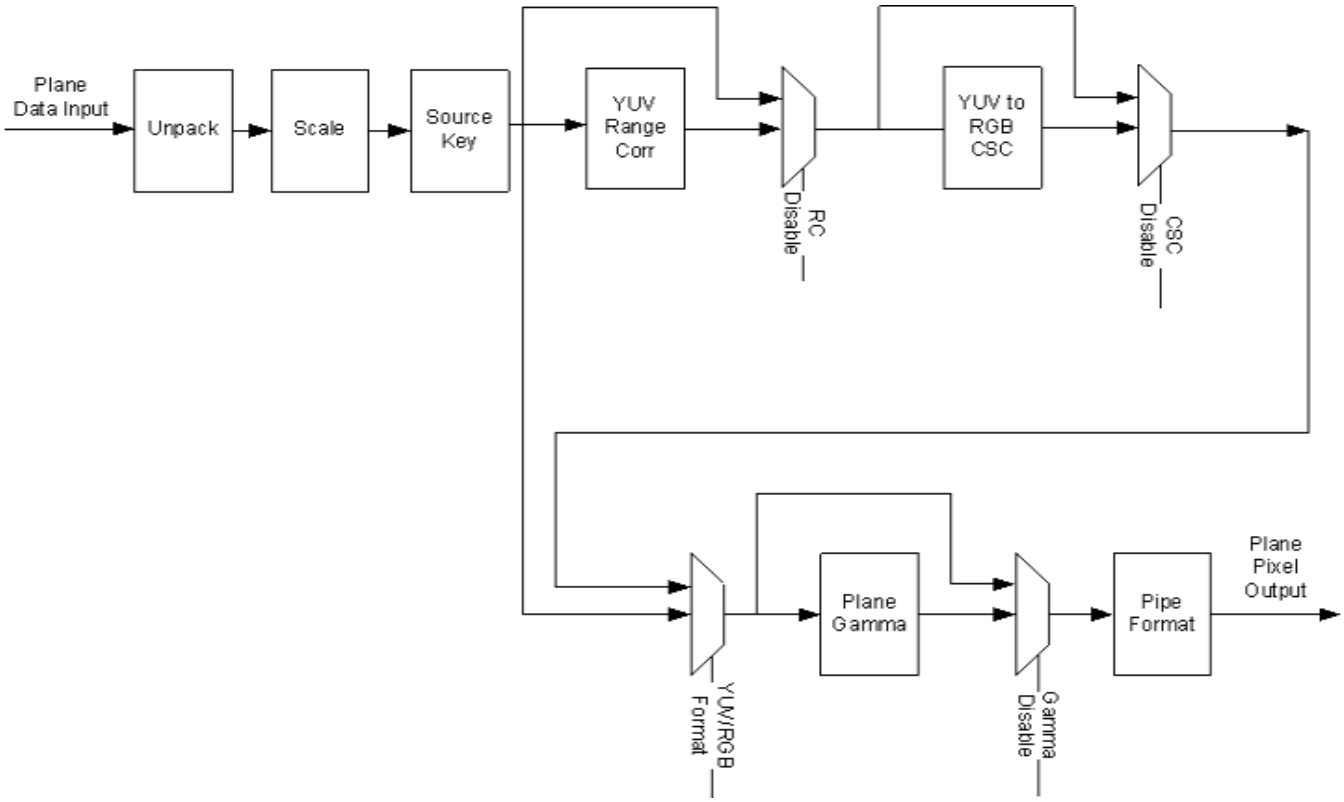
PLANE_GAMC16

PLANE_GAMC17

Many of the plane control active registers will be updated on the vertical blank or when pipe is disabled, after the surface base address register is written, or when the plane is not yet enabled, providing an atomic update of those registers together with the surface base address register.

Data flow through the plane (Steps 2-6 may be enabled or disabled by programming control bits):

1. Unpack data into pixels
2. Plane Scaling
3. Source Key
4. YUV Range Correction (can only be used by YUV source pixel formats)
5. YUV to RGB Color Space Conversion (can only be used by YUV source pixel formats)
6. Gamma Correction
7. Conversion to pipe data format



Plane Pixel Formats

Plane Source Pixel Format Mapping of Bits to Colors:

YUV 4:2:2 Packed Formats	Y1	U	Y2	V
YUV 4:2:2 YUYV 8 bpc	7:0	15:8	23:16	31:24
YUV 4:2:2 UYVY 8 bpc	15:8	7:0	31:24	23:16
YUV 4:2:2 YVYU 8 bpc	7:0	31:24	23:16	15:8
YUV 4:2:2 VYUY 8 bpc	15:8	23:16	31:24	7:0

YUV 4:4:4 Packed Formats	Ignored	Y	U	V
YUV 4:4:4 8 bpc	31:24	23:16	15:8	7:0

XRGB Formats	Ignored	Red	Green	Blue
RGB 32-bit 2:10:10:10 BGRX	31:30	29:20	19:10	9:0
RGB 32-bit 2:10:10:10 RGBX	31:30	9:0	19:10	29:20
RGB 32-bit 8:8:8:8 BGRX	31:24	23:16	15:8	7:0
RGB 32-bit 8:8:8:8 RGBX	31:24	7:0	15:8	23:16
RGB 64-bit 16:16:16:16 Float BGRX Each component is 1:5:10 MSb-sign:exponent:fraction	63:48	47:32	31:16	15:0
RGB 64-bit 16:16:16:16 Float RGBX Each component is 1:5:10 MSb-sign:exponent:fraction	63:48	15:0	31:16	47:32
RGB 32-bit XR_BIAS 10:10:10	31:30	9:0	19:10	29:20
16-bit BGR 5:6:5	N/A	15:11	10:5	4:0

ARGB Formats	Alpha	Red	Green	Blue
RGB 32-bit 8:8:8:8 BGRA	31:24	23:16	15:8	7:0
RGB 32-bit 8:8:8:8 RGBA	31:24	7:0	15:8	23:16

Transcoder

Transcoder Control

TRANS_CONF

TRANS_STEREO3D_CTL

Transcoder Timing

TRANS_HTOTAL

TRANS_HBLANK

TRANS_HSYNC

TRANS_VTOTAL

TRANS_VBLANK

TRANS_VSYNC

TRANS_FRM_TIME

TRANS_SPACE

TRANS_VSYNCSHIFT

TRANS_MULT

Transcoder MN Values

Description
These values are used for DisplayPort.

There is one instance of these registers per each transcoder.

For dynamic switching between multiple refresh rates, M/N values may be reprogrammed on the fly. The link N should be programmed last to trigger the update of all the data and link M and N registers and then the new M/N values will be used in the next frame that is output.

DATAM

DATAN

LINKM

LINKN

Clocks:

ls_clk is the link symbol clock. i.e. 270 MHz for HBR2.

strm_clk is the stream clock, which is the video pixel rate or dot clock.

cdclk is the core display clock.

The link only operates in Synchronous Clock mode. The link clock and stream clock are synchronous and the link M and N values stay constant for a given pixel rate.

Calculation of TU, Data M, and Data N:

TU is the Transfer Unit. It is recommended to program a TU Size of 64 link symbols per lane.

Active/TU Size = Payload/Capacity = Data M/N

Compression Ratio (CR) = DisplayPort Compression enabled ? min(ratio1,ratio2) : 1

- ratio1 = Compressor BW / Link BW = (cdclk * bytes per pixel) / (ls_clk * number of lanes)
- ratio2 = (Horizontal active in pixels * bytes per pixel / 4) / ((Horizontal active in pixels * bytes per pixel / 8) +2)

Data M/N = (strm_clk * bytes per pixel) / (CR * ls_clk * number of lanes)

Calculation of Link M and Link N:

Link M/N = strm_clk / ls_clk

Restriction on clocks and number of lanes:

- Number of lanes >= INT(strm_clk * bytes per pixel / ls_clk)

Restrictions on the Virtual Channel (VC) payload size in DisplayPort MST mode

- In a x1 lane config, each pipe stream on the link must use a VC payload size that is a multiple of 4.
- In a x2 lane config, each pipe stream on the link must use a VC payload size that is a multiple of 2.
- In a x4 lane config, each pipe stream on the link must use a VC payload size that is a multiple of 1.

Transcoder Video Data Island Packet

Data Island Packet (DIP) is a mechanism that allows data to be sent over a digital port during blanking, according to the HDMI and DisplayPort specifications. This includes header, payload, checksum, and ECC information.

Each type of Video DIP will be sent once each frame while it is enabled.

VIDEO_DIP_CTL

VIDEO_DIP_DATA

VIDEO_DIP_GCP

VIDEO_DIP_ECC

Construction of DIP for AVI, VS, or SPD (HDMI only):

Dword	Byte3	Byte2	Byte1	Byte0
0	Reserved	HB2	HB1	HB0
1	DB3	DB2	DB1	DB0
2	DB7	DB6	DB5	DB4
3	DB11	DB10	DB9	DB8
4	DB15	DB14	DB13	DB12
5	DB19	DB18	DB17	DB16
6	DB23	DB22	DB21	DB20
7	DB27	DB26	DB25	DB24
8 (RO)	Reserved	Reserved	Reserved	HB ECC
9 (RO)	DB ECC 3	DB ECC 2	DB ECC 1	DB ECC 0

HB = Header Byte, DB = Data Byte, RO = Read Only

Construction of DIP for GMP (HDMI or DisplayPort):

Dword	Byte3	Byte2	Byte1	Byte0
0	DP: HB3 HDMI: Reserved	HB2	HB1	HB0
1	DB3	DB2	DB1	DB0

Dword	Byte3	Byte2	Byte1	Byte0
2	DB7	DB6	DB5	DB4
3	DB11	DB10	DB9	DB8
4	DB15	DB14	DB13	DB12
5	DB19	DB18	DB17	DB16
6	DB23	DB22	DB21	DB20
7	DB27	DB26	DB25	DB24
8 (RO)	Reserved	Reserved	Reserved	DP: Reserved HDMI: HB ECC
9 (RO)	DP: Reserved HDMI: DB ECC 3	DP: Reserved HDMI: DB ECC 2	DP: Reserved HDMI: DB ECC 1	DP: Reserved HDMI: DB ECC 0
10 (RO)	DP: HB ECC 3 HDMI: Reserved	DP: HB ECC 2 HDMI: Reserved	DP: HB ECC 1 HDMI: Reserved	DP: HB ECC 0 HDMI: Reserved
11 (RO)	DP: DB ECC 3 HDMI: Reserved	DP: DB ECC 2 HDMI: Reserved	DP: DB ECC 1 HDMI: Reserved	DP: DB ECC 0 HDMI: Reserved
12 (RO)	DP: DB ECC 7 HDMI: Reserved	DP: DB ECC 6 HDMI: Reserved	DP: DB ECC 5 HDMI: Reserved	DP: DB ECC 4 HDMI: Reserved

HB = Header Byte, DB = Data Byte, DP = DisplayPort, RO = Read Only

Construction of DIP for VSC (DisplayPort only):

Dword	Byte3	Byte2	Byte1	Byte0
0	HB3	HB2	HB1	HB0
1	DB3	DB2	DB1	DB0
2	DB7	DB6	DB5	DB4
3	DB11	DB10	DB9	DB8
4	DB15	DB14	DB13	DB12
5	DB19	DB18	DB17	DB16
6	DB23	DB22	DB21	DB20
7	DB27	DB26	DB25	DB24
8	DB31	DB30	DB29	DB28
9 (RO)	HB ECC 3	HB ECC 2	HB ECC 1	HB ECC 0
10 (RO)	DB ECC 3	DB ECC 2	DB ECC 1	DB ECC 0
11 (RO)	DB ECC 7	DB ECC 6	DB ECC 5	DB ECC 4

HB = Header Byte, DB = Data Byte, RO = Read Only

The audio subsystem is also capable of sending Data Island Packets. These packets are programmed by the audio driver and can be read by in MMIO space via the audio control state and audio HDMI widget data island registers.

Video DIP data write sequence:

1. Wait for 1 VSync to ensure completion of any pending video DIP transmissions
2. Disable the video DIP being updated (disable VDSC before updating PPS DIP)
3. Program video DIP data buffer registers for DIP being updated
4. Enable the video DIP

The video DIP data and ECC buffers may be read at any time.

DIP data buffer registers must be programmed with valid data before enabling the DIP.

Partial DIPs are never sent out while the port is enabled. Disabling the DIP at the same time it is being transferred will result in the DIP being completed before the function is disabled.

Shutting off the port on which DIP is being transmitted will result in partial transfer of DIP data. There is no need to switch off the DIP enable bit if the port transmitting DIP is disabled.

When disabling both the DIP port and DIP transmission, first disable the port and then disable DIP.

Enabling a DIP function at the same time that the DIP would have been sent out (had it already been enabled) will result in the DIP being sent on the following frame.

For HDMI, even if no DIP is enabled, a single Null DIP will be sent at the same point in the stream that DIP packets would have been sent.

Transcoder DDI Function

TRANS_DDI_FUNC_CTL

TRANS_MSA_MISC

SRD

SRD enable sequence:

- Prerequisite: The associated transcoder and port are running.

1. Configure FBC host and render tracking. The FBC function does not need to be enabled in FBC_CTL.
2. Program Transcoder EDP VSC DIP data with a valid setting for SRD/PSR.
3. Configure and enable SRD_CTL.

SRD disable sequence:

- Prerequisite: The associated transcoder and port are running.

1. Disable SRD_CTL.
2. Wait for SRD_STATUS to show SRD is Idle. This will take up to one full frame time (1/refresh rate), plus SRD exit training time (max of 6ms), plus SRD aux channel handshake (max of 1.5ms).

PSR2 disable sequence:

1. Program PSR2_CTL reset Psr2Enable, SelectiveUpdateTrackingEnable bits.
2. Disable GTC if required.
3. Wait for PSR2_STATUS to show PSR2 is Idle. This will take up to one full frame time (1/refresh rate), plus exit training time (max of 6ms), plus aux channel handshake (max of 1.5ms).

Registers
SRD_CTL
SRD_STATUS
SRD_PERF_CNT
SRD Interrupt Bit Definition
SRD_IMR
SRD_IIR
PSR_MASK
PSR_EVENT
PSR2_CTL
PSR2_MAN_TRK_CTL
PSR2_SU_STATUS
PSR2_STATUS

Transcoder Port Sync

Feature Description

PORT SYNC is a transcoder level feature supported for DP/eDP and DSI protocols. This mode forces two or more transcoders to be in sync with one transcoder master and one or more transcoder slaves. In the case of DP/eDP, the master is unaware that it is operating in Port Sync mode. Only the slave is aware that it is operating in this mode. Hence, port sync mode is only enabled in the slave transcoder.

DP/eDP Port Sync Restrictions

1. The slave and master transcoders and associated ports must have identical parameters and properties.
2. They must be connected to the same PLL, have the same color format, link width (number of lanes enabled), resolution, refresh rate, dot clock, TU size, M and N programming, etc.
3. Port Sync Mode must only be enabled with DisplayPort SST.
4. PSR would need to be disabled when port sync mode is enabled.
5. Port Sync Mode Master Select must be programmed with a valid value when Port sync Mode is enabled.

Audio

This section describes Audio features.

Audio Bios Programming Sequence

Codec Verb Table

For each codec present on the High Definition Audio codec link, a corresponding pre-defined "Codec Verb Table" must be available to System BIOS. The Codec Verb Tables are based on codec specific information (coded datasheet) and platform design specific information (schematics) and are built by System BIOS writers and platform designers. The table contains a list of 32-bit "Verb"s (command and data payload) to be sent to the corresponding codec over the High Definition Audio codec link.

Below is a sample High Definition Audio Codec Verb Table for a platform with 1 codec at codec address 01h.

```
;Sample HIGH DEFINITION AUDIO Codec Verb Table
;Codec Address (CAd) = 02h
;Codec Vendor: XYZ Company
;VenID DevID:
    dd 12345678h
;-----------------------------------------------------
; FrontPanel_Supported?   ; 1=Supported ,0=Not supported
    db  01h
; # of Rear Panel Pin Complexes
   dw 000Ch
; # of Front Panel Pin Complexes
dw 0002h
;-----------------------------------------------------
```

Note: Set the bit 15 of register offset 0x65F10h of the Display Audio offset. Wait for the Codec to generate the wake event to the controller.

Follwing verbs should be send to the codec using the PIO method described in the below sections 9.1.3.

```
VerbTable0:
    ;Enable the third converter and Pin first (NID 08h)
     dd 20878101h
       //
  // Audio Verb Table - 0x80862805
```

//

// Pin Widget 5 - PORT B

0x20571C10,

0x20571D00,

0x20571E56,

0x20571F18,

// Pin Widget 6 - PORT C

0x20671C20,

0x20671D00,

0x20671E56,

0x20671F18,

// Pin Widget 7 - PORT D

0x20771C30,

0x20771D00,

0x20771E56,

0x20771F18

;disable the third converter and third Pin (NID 08h)

dd 20878100h

Codec Initialization Programming Sequence

After System BIOS has determined the presence of High Definition Audio codecs, it must follow the programming sequence below to update the codec with the correct jack information specific to the platform for the High Definition Audio driver to retrieve and use later.

There are two ways to send verbs to and receive response data from codecs over the High Definition Audio codec link: using CORB/RIRB (Command Output Ring Buffer / Response Input Ring Buffer) or using the Immediate Command/Immediate Response register pair. The sequence below uses the latter which does not require the availability of a memory buffer.

System BIOS should ensure that the High Definition Audio HDBAR D27:F0:10-17h contains a valid address value and is enabled by setting D27:F0:04h[1]. System BIOS must ensure program as mentioned in section 9.6, and then the Controller Reset# bit of Global Control register in memory-mapped space (HDBAR+08h[0]) is set to 1b and read back as 1b. Additional delay might be required to allow codec coming out of reset prior to subsequent operations, please contact your codec vendor for detail. When clearing this bit and setting it afterward, BIOS must ensure that minimum link timing requirements (minimum RESET# assertion time, etc.) are met.

Note: To initialize codec Bios should set the bit 15 of the register 0X65F10h of the SKL Display MMIO to 1. This bit needs to be after the controller is brought out of reset. BIOS should wait for Controller to detect the wake event and recognize the Codec.

For each High Definition Audio codec present as indicated by HDBAR + 0Eh[3:0], System BIOS should perform the codec initialization as described below:

1. Read the VendorID/DeviceID pair from the attached codec.

- Verify that the ICB bit, HDBAR + 68h[0], is 0.
- Write verb 200F0000h (dword) to the IC register, HDBAR + 60h, where: '2' (bits 31:28) represents the codec address (CAd).
- Program HDBAR + 68h[1:0] to 11b to send the verb to the codec.
- Poll the ICB bit, HDBAR+68h[0] until it returns 0 indicating the verb has been sent to the codec. BIOS may write HDBAR + 68h[0] to a 0 if the bit fails to return to 0 after a reasonable timeout period.
- If HDBAR + 68h[1] = 1b indicating the response data from the codec is now valid, read HDBAR + 64h; the data is the VID/DID value returned by the codec.

2. Check against internal list to determine if there is a stored verb table which matches the CAd/VID/DID information.

Steps 1 and 2 are System BIOS implementation-specific steps and can be done in different ways. If a System BIOS has prior knowledge of a fixed platform/codec combination (e.g., for a System BIOS having 3 stored verb tables for 3 known codecs at known codec addresses on a known platform), a simple pre-defined codec-to-table matching can be used and steps 1 and 2 can be eliminated. For a System BIOS to support multiple codec/platform combinations, an internal match-list might be needed to match a platform/codec combination to a codec verb table.

3. If there is a match, send the entire list of verbs in the matching verb table one by one to the codec.

- Verify the ICB bit, HDBAR + 68h[0] is 0.
- Write the next verb (dword) in the table to HDBAR + 60h.
- Program HDBAR + 68h[1:0] to 11b to send the verb to codec.
- Poll the ICB bit, HDBAR + 68h[0] until it returns 0 indicating the verb has been sent to the codec. BIOS may write HDBAR + 68h[0] to a 0 if the bit fails to return to 0 after a reasonable timeout period.
- Repeat the steps until all the verbs in the table have been sent.

Some verbs in the table may be dependent on certain platform-specific conditions. For example, for the sample table above, the verbs for Pin Complex 7 and 8 (NID=14,16 respectively) should be sent only if the Front Panel Jacks are present and connected on the platform, which may be indicated by a software flag that is controlled by a certain GPIO pin.

Audio Programming Sequence

The following HDMI and DisplayPort audio programming sequences are for use when enabling or disabling audio or temporarily disabling audio during a display mode set.

The audio codec and audio controller disable sequences must be followed prior to disabling the transcoder or port in a display mode set.

The audio codec and controller enable sequences can be followed after the transcoder is enabled and the port is enabled and completed link training (not sending training or idle patterns if DisplayPort).

The audio controller and audio codec sequences may be done in parallel or serial. In general, the change in ELDV/PD in the codec sequence will generate an unsolicited response to the audio controller driver to indicate that the controller sequence should start, but other mechanisms may be used. SW should make sure to set the Inactive (IA) bit to 0 before setting PD to 1.

Audio codec disable sequence:

- Disable sample fabrication
  - Set AUD_MISC_CTRL Sample_Fabrication_EN (bit 2) to "0".
- Disable timestamps
  - Set AUD_CONFIG N_value_index (bit 29) to "0" for HDMI or "1" for DisplayPort.
  - Set N_programming_enable (bit 28) to "1"
  - Set Upper_N_value and Lower_N_value (bits 27:20, 15:4) to all "0"s.
- Disable ELDV and ELD buffer
  - Set AUD_PIN_ELD_CP_VLD ELD_valid (bit 0, 4, or 8 based on which port is used) to "0"
- Wait for 2 vertical blanks
- Optional: Disable audio PD (Presence Detect)
  - Software may choose to skip this in order to keep PD enabled during a resolution switch.
  - Set AUD_PIN_ELD_CP_VLD Audio_Inactive (bit 3, 7, or 11) to "1". SW does not need to set this bit to enable Inactive bit.
  - Set AUD_PIN_ELD_CP_VLD Audio_Output_Enable (bit 2, 6, or 10) to "0".

Audio controller disable sequence:

- Program Stream ID to 0 - Verb ID 706
- Disable audio info frames transmission - Verb ID 732
- Disable Digen - Verb ID 70D
- Program the codec to D3 state if needed.
- Audio driver may stop the audio controller DMA engine at this point if needed, but not required.

Audio codec enable sequence:

- Enable audio Presence Detect
  - Set AUD_PIN_ELD_CP_VLD Audio_Inactive (bit 3, 7, or 11) to "0".

    - Set AUD_PIN_ELD_CP_VLD Audio_Output_Enable (bit 2, 6, or 10) to "1".
- Wait for 1 vertical blank
- Load ELD buffer and Enable ELDV
    - Set AUD_PIN_ELD_CP_VLD ELD_valid (bit 0, 4, or 8 based on which port is used) to "1".
- Enable timestamps
    - Set AUD_CONFIG N_value_index (bit 29) to "0" for HDMI or "1" for DisplayPort.
    - Set N_programming_enable (bit 28) to "0".
    - Program Upper_N_value and Lower_N_value (bits 27:20, 15:4) if a non-default N value is needed.
- Enable sample fabrication if this feature is needed
    - Set AUD_MISC_CTRL Sample_Fabrication_EN (bit 2) to "1".

Audio controller enable sequence:

- Program the codec to D0 state if in D3 state.
- Program Stream ID to non zero - Verb ID 706
- Enable audio info frames transmission - Verb ID 732
- Enable Digen - Verb ID 70D
- If audio controller DMA engine is stopped, audio driver can start the DMA engine at this point.

Audio Configuration

AUD_CONFIG

AUD_MISC_CTRL

AUD_VID_DID

AUD_RID

AUD_M_CTS_ENABLE

Audio Power State Format

AUD_PWRST

AUD_EDID_DATA

AUD_FREQ_CNTRL

AUD_INFOFR

AUD_PIN_PIPE_CONN_ENTRY_LNGTH

AUD_PIPE_CONN_SEL_CTRL

AUD_DIP_ELD_CTRL_ST

AUD_PIN_ELD_CP_VLD

Digital Display Interface

DDI Buffer

There is one instance of these registers per each DDI.

DDI_BUF_CTL

DDI_BUF_TRANS

DISPIO_CR_TX_BMU_CR0

DDI_BUF_TRANS and DISPIO_CR_TX_BMU_CR0 are programmed with the voltage swing values for each port. They can be configured in advance of the display mode set. The recommended values are listed below.

For DisplayPort, DDI_BUF_CTL is programmed during the mode set to select between the voltage swings pre-programmed in DDI_BUF_TRANS.

HDMI always uses the voltage swing programmed in DDI_BUF_TRANS entry 9.

I_boost

I_boost increases the I/O current, boosting the swing level. There are three I_boost values, in order of increasing boost; 0x1, 0x3, and 0x7.

I_boost can be configured separately for each port and can be enabled or disabled separately for each voltage swing level.

I_boost is configured by programming DISPIO_CR_TX_BMU_CR0 tx_blnclegdisbl=0x00 and tx_blnclegsctl_<selected DDI>=<I_boost value; 0x1, 0x3, or 0x7>. It is then enabled when DDI_BUF_CTL selects a voltage swing entry that has DDI_BUF_TRANS_<entry> Balance Leg Enable=0x1. For DDIA with x4 capability (DDI_BUF_CTL DDIA Lane Capability Control = DDIA x4), the I_boost value has to be programmed in both tx_blnclegsctl_0 and tx_blnclegsctl_4.

The recommended buffer translation programming only uses I_boost for a few entries. A customer may request increased I_boost beyond the recommended values by specifying an I_boost value of 0x1, 0x3, or 0x7 to be applied to all swing entries for a port, then software needs to configure I_boost to that specified value and set DDI_BUF_TRANS_<all entries> Balance Leg Enable=0x1 for that port. This selection overrides the recommended I_boost settings.

Recommended Buffer Translation Programming

Recommended buffer translation programming for DisplayPort										
DDI_BUF_TRANS Entry Number	Voltage Swing Level[1]	Pre-emphasis Level[1]	Non-Transition mV diff p-p	Transition mV diff p-p	Pre-emphasis dB	SKL H and S		SKL Y		SKL U	
						Dword 1 [31:0]	Dword 0 [31:0]	Dword 1 [31:0]	Dword 0 [31:0]	Dword 1 [31:0]	Dword 0 [31:0]
0	0	0	400	400	0	0000 00A0h	0000 2016h	0000 00A2h	0000 0018h	0000 00A2h	0000 201Bh
1	0	1	400	600	3.5	0000	0000	0000	0000	0000	0000

Recommended buffer translation programming for DisplayPort											
						009Bh	5012h	0088h	5012h	0088h	5012h
2	0	2	400	800	6	0000 0088h	0000 7011h	0000 00**CD**h	**8**000 7011h	0000 00**CD**h	**8**000 7011h
3	0	3	400	1000	9.5	0000 00C0h	**8**000 9010h[2]	0000 00C0h	**8**000 9010h[3]	0000 00C0h	**8**000 9010h[2]
4	1	0	600	600	0	0000 009Bh	0000 2016h	0000 009Dh	0000 0018h	0000 009Dh	0000 201Bh
5	1	1	600	900	3.5	0000 0088h	0000 5012h	0000 00C0h	**8**000 5012h[3]	0000 00C0h	**8**000 5012h[2]
6	1	2	600	1000	6	0000 00C0h	**8**000 7011h[2]	0000 00C0h	**8**000 7011h[3]	0000 00C0h	**8**000 7011h[2]
7	2	0	800	800	0	0000 00DFh	0000 2016h	0000 0088h	0000 0018h	0000 0088h	0000 2016h
8	2	1	800	1000	3.5	0000 00C0h	**8**000 5012h[2]	0000 00C0h	**8**000 5012h[3]	0000 00C0h	**8**000 5012h[2]

[1]The voltage swing level and pre-emphasis level values follow the naming used in the DisplayPort standard.

[2]I_boost needs to be configured to recommended level 0x1. DISPIO_CR_TX_BMU_CR0 tx_blnclegdisbl=0x00 and tx_blnclegsctl_<selected DDI>=0x1.

[3]I_boost needs to be configured to recommended level 0x3. DISPIO_CR_TX_BMU_CR0 tx_blnclegdisbl=0x00 and tx_blnclegsctl_<selected DDI>=0x3.

Recommended buffer translation programming for embedded DisplayPorts that support low voltage swings											
DDI_BUF_TRANS Entry Number	**Voltage Swing Level[1]**	**Pre-emphasis Level[1]**	**Non-Transition mV diff p-p**	**Transition mV diff p-p**	**Pre-emphasis dB**	**SKL H and S**		**SKL Y**		**SKL U**	
						Dword 1 [31:0]	**Dword 0 [31:0]**	**Dword 1 [31:0]**	**Dword 0 [31:0]**	**Dword 1 [31:0]**	**Dword 0 [31:0]**
0	0	0	200	200	0	0000 00A8h	0000 0018h	0000 00A8h	0000 0018h	0000 00A8h	0000 0018h
1	0	1	200	250	1.5	0000 00A9h	0000 4013h	0000 00ABh	0000 4013h	0000 00A9h	0000 4013h
2	0	2	200	350	6	0000 00A2h	0000 7011h	0000 00A4h	0000 7011h	0000 00A2h	0000 7011h
3	0	3	200	600	9.5	0000 009Ch	0000 9010h	0000 00DFh	0000 9010h	0000 009Ch	0000 9010h
4	1	0	250	250	0	0000 00A9h	0000 0018h	0000 00AAh	0000 0018h	0000 00A9h	0000 0018h
5	1	1	250	350	3.5	0000 00A2h	0000 6013h	0000 00A4h	0000 6013h	0000 00A2h	0000 6013h
6	1	2	250	500	6	0000 00A6h	0000 7011h	0000 009Dh	0000 7011h	0000 00A6h	0000 7011h
7	2	0	350	350	0	0000 00ABh	0000 0018h	0000 00A0h	0000 0018h	0000 00ABh	0000 2016h

Recommended buffer translation programming for embedded DisplayPorts that support low voltage swings											
8	2	1	350	500	4.5	0000 009Fh	0000 7013h	0000 00DFh	0000 6012h	0000 009Fh	0000 5013h
9	Fail Safe	Fail Safe	800	800	0	0000 00DFh	0000 0018h	0000 008Ah	0000 0018h	0000 00DFh	0000 0018h

[1]The voltage swing level and pre-emphasis level values follow the naming used in the DisplayPort standard.

Recommended buffer translation programming for HDMI and DVI						
Non-Transition mV diff p-p	**Transition mV diff p-p**	**Pre-emphasis dB**	**SKL U, H, and S**		**SKL Y**	
			Dword 1 [31:0]	**Dword 0 [31:0]**	**Dword 1 [31:0]**	**Dword 0 [31:0]**
400	400	0	0000 00ACh	0000 0018h	0000 00A1h	0000 0018h
400	600	3.5	0000 009Dh	0000 5012h	0000 00DFh	0000 5012h
400	800	6	0000 0088h	0000 7011h	0000 00CBh	8000 7011h[3]
450	450	0	0000 00A1h	0000 0018h	0000 00A4h	0000 0018h
600	600	0	0000 0098h	0000 0018h	0000 009Dh	0000 0018h
600	800	2.5	0000 0088h	0000 4013h	0000 0080h	0000 4013h
600	1000	4.5	0000 00CDh	8000 6012h[2]	0000 00C0h	8000 6013h[3]
800	800	0	0000 00DFh	0000 0018h	0000 008Ah	0000 0018h
[1]800	1000	2	0000 00CDh	8000 3015h[2]	0000 00C0h	8000 3015h[3]
1000	1250	2	0000 00C0h	8000 3015h[2]	0000 00C0h	**8**000 3015h[3]
1200	1200	0	0000 00C0h	8000 0018h[2]	0000 00C0h	8000 0018h[3]
Select one of the above values to program in DDI_BUF_TRANS entry 9.						

[1]This row is the recommended default if a specific value is not selected.

[2]I_boost needs to be configured to recommended level 0x1. DISPIO_CR_TX_BMU_CR0 tx_blnclegdisbl=0x00 and tx_blnclegsctl_<selected DDI>=0x1.

[3]I_boost needs to be configured to recommended level 0x3. DISPIO_CR_TX_BMU_CR0 tx_blnclegdisbl=0x00 and tx_blnclegsctl_<selected DDI>=0x3.

DDI AUX Channel

DDI_AUX_CTL

DDI_AUX_DATA

DDI_AUX_MUTEX

AUX programming sequence

A general purpose AUX functional programming sequence is provided below.

AUX Functional Sequence

Step	Description	Register	Notes
1	Display must already be initialized.		Power well1 enabled cdclk enabled AuxIO powered up in PWR_WELL_CTL.
2	Disable DC5 and DC6 before a DDI A AUX channel transaction is sent.	DC_STATE_EN	PG1 may disable automatically for DC5 or DC6.
Skip step 3, 4 if PSR1/SRD, PSR2 or GTC are NOT enabled. Do not skip step 3,4 if PSR1/SRD, PSR2 or GTC are enabled.			
3	Enable MUTEX without changing MUTEX status	DDI_AUX_MUTEX[31]='1'	
4	Read MUTEX status	DDI_AUX_MUTEX[30]	If MUTEX status is '1', wait for 500 us and poll for MUTEX status == '0'.
5	Program AUX data registers.	DDI_AUX_DATA_*_[0-4]	
6	Program control to configure AUX and START transaction.	DDI_AUX_CTL_*	Timeout timer value must be at least 600us. Timer values: 0: 400us 1: 600 us 2: 800 us 3: 1600 us START trigger: DDI_AUX_CTL_*[31]='1'
7	Wait for AUX transaction complete.		AUX Transaction complete interrupt if set OR when DDI_AUX_CTL_*[31:30] = '01'.
8	Check that receive data has no errors	DDI_AUX_CTL_*[25]	If set: write a '1' to clear this bit and skip reading AUX data registers.

Step	Description	Register	Notes
9	Read AUX data register	DDI_AUX_DATA_*_[0-4]	Condition: Aux Channel Control Register Send/Busy bit is NOT asserted
10	Clear status flags	DDI_AUX_CTL_*[30]	Transaction done status
Skip step 11 if PSR1/SRD, PSR2 or GTC are NOT enabled. Do not skip step 11 if PSR1/SRD, PSR2 or GTC are enabled.			
11	Release MUTEX	DDI_AUX_MUTEX[31:30]= "11" OR DDI_AUX_MUTEX[31:30]="01"	Release with (is optional) or without disabling MUTEX function.

DisplayPort Transport

There is one instance of these registers per each DDI.

DP_TP_CTL

DP_TP_STATUS

Global Time Code (GTC)

Global Time Code

Top Level GTC

GTC_CTL

GTC_DDA_M

GTC_DDA_N

GTC_LIVE

GTC Interrupt Bit Definition

GTC_IMR

GTC_IIR

DDI Level GTC

GTC_PORT_CTL
GTC_PORT_TX_CURR
GTC_PORT_TX_PREV
GTC_PORT_MISC

GTC Target Frequency Selection

For GTC top level logic, CDCLK is taken as an input and scaled to a "target frequency" which has a period which is an exact multiple of 0.5ns. This period is also known as the accumulator increment.

Once a target frequency + accumulator increment is selected, an M and N value can be picked and fine tuned to achieve the scaling from CDCLK to target frequency.

In order for the accumulated GTC Live value to match exactly with the real passage of time, the following must be true:

The target frequency selected must be the CLOSEST possible selection to CDCLK. This corresponds to rounding the accumulator increment to the NEAREST 0.5ns increment with respect to the period of CDCLK.

Example

	Frequency	Period
Given CDCLK	337.500 MHz	2.96 ns
1st closest target frequency/increment	333.333 MHz	3.00 ns
2nd closest target frequency/increment	400.000 MHz	2.50 ns
3rd closest target frequency/increment	285.714 MHz	3.50 ns
4th closest target frequency/increment	500.000 MHz	2.00 ns

Only using the 1st closest target frequency 333.333MHz / accumulator increment of 3.00 ns will result in the GTC Live Value correctly tracking the real passage of time.

Example Calculation Flow:

1. Find the period of CDCLK -- for 337.5 MHz = 2.96 ns.
2. Round to the nearest 0.5 ns -- 3.00 ns (this is your accumulator increment value).
3. Find the “target frequency” from the rounded period value of step 2 -- 333.333 MHz is the “target frequency”.
4. Find the ratio of target frequency / CDCLK = 0.987654321.
5. Choose M and N to satisfy M / N = same ratio as step 4.

South Display Engine Registers

The South Display Engine supports Hot Plug Detection, GPIO, GMBUS, Panel Power Sequencing, and Backlight Modulation.

Terminology

Access Field	Description	Should be implemented as
R/W (Read/Write)	The value written into this register will control hardware and is the same value that will be read.	Write data is stored. Read is from the stored data. Stored value is used to control hardware.
Reserved	Unused register bit. Don't assume a value for these bits. Writes have no effect.	Write data is ignored. Read is zero.
MBZ (Must Be Zero)	Always write a zero to this register.	May be implemented as Reserved or as R/W.
PBC (Preserve Bit Contents)	Software must write the original value back to this bit. This allows new features to be added using these bits.	May be implemented as Reserved or as R/W.
Read Only	The read value is determined by hardware. Writes to this bit have no effect.	Write data is ignored. Read is from a status signal or some other internal source.
Write Only	The value written into this register will control hardware. Reads return zero.	Write data is stored. Read is zero. Stored value is used to control hardware.
R/W Clear (Read/Write Clear)	Sticky status bit. Hardware will set the bit, software can clear it with a write of 1b.	Internal hardware events set a sticky bit. Read is from the sticky bit. A write of 1b clears the sticky bit.
Double Buffered	Write when desired and the written value will take effect at the time of the double buffer update point. Reads will return the written value, which is not necessarily the value being currently used to control hardware. Some have a specific arming sequence where a write to another register is required before the update can take place. This is used to ensure atomic updates of several registers.	Two stages of registers used. Write data is stored into first stage. Read is from the first stage stored data. First stage stored value is transferred to second stage storage at the double buffer update point. Second stage stored value is used to control hardware. Arm/disarm logic may be used for some registers to control the double

Access Field	Description	Should be implemented as
		buffer update point.
Write/Read Status	The value written into this register will control hardware. The read value is determined by hardware.	Write data is stored. Stored value is used to control hardware. Read is from a status signal or some other internal source.

Shared Functions

Fuses and Straps

SFUSE_STRAP

Raw Clock

RAWCLK_FREQ must be programmed to match the raw clock frequency.

Description
RAWCLK_FREQ

Description
Raw clock frequency = 24 MHz

Interrupts and Hot Plug

Description
SINTERRUPT **SHOTPLUG_CTL** **SHOTPLUG_CTL2** **SHPD_PULSE_CNT** **SHPD_FILTER_CNT** **South Display Engine Interrupt Bit Definition**

Panel Power and Backlight

Panel Power

Description
PP_DIVISOR **PP_STATUS** **PP_CONTROL**

Description
PP_ON_DELAYS
PP_OFF_DELAYS

Backlight

This section refers to the PCH display backlight control. For CPU display backlight control, see North Display Engine Registers.

The backlight PWM output frequency is determined by the PWM clock frequency, increment, and frequency divider.

PWM output frequency = PWM clock frequency / PWM increment / PWM frequency divider

The frequency divider minimum must be greater than or equal to the number of brightness levels required by software; typically 100 or 256.

Description
PWM clock frequency = 24 MHz
PWM increment = 128 or 16, selectable by software
PWM frequency divider maximum = 65,536
PWM output frequency range with 100 brightness levels and increment 128 = 3 to 1,875 Hz
PWM output frequency range with 100 brightness levels and increment 16 = 23 to 15,000 Hz
PWM output frequency range with 256 brightness levels and increment 128 = 3 to 732 Hz
PWM output frequency range with 256 brightness levels and increment 16 = 23 to 5,859 Hz

Backlight Enabling Sequence

Description
1. Set frequency and duty cycle in SBLC_PWM_CTL2 Backlight Modulation Frequency and Backlight Duty Cycle.
2. Set granularity in 0xC2000 bit 0 (0 = 16, 1 = 128).
3. Enable PWM output and set polarity in SBLC_PWM_CTL1 PWM PCH Enable and Backlight Polarity.
...
4. Change duty cycle as needed in SBLC_PWM_CTL2 Backlight Duty Cycle.

If needed, granularity, polarity, and override can be programmed earlier than shown.

Backlight Registers

Description
SBLC_PWM_CTL1
SBLC_PWM_CTL2

GMBUS and GPIO

Registers

Description
GPIO_CTL - GPIO Control
GMBUS0 - GMBUS Clock/Port Select
GMBUS1 - GMBUS Command/Status
GMBUS2 - GMBUS Status
GMBUS3 - GMBUS Data Buffer
GMBUS4 - GMBUS Interrupt Mask
GMBUS5 - GMBUS 2 Byte Index

Pin Usage

These GPIO pins allow the support of simple query and control functions such as DDC interface protocols. The GMBUS controller can be used to run the interface protocol, or the GPIO pins can be manually programmed for a "bit banging" interface.

The following tables describe the expected GPIO pin to register mapping. OEMs have the ability to remap these functions onto other pins as long as the hardware limitations are observed. The GPIO pins may also be muxed with other functions such that they are only available when the other function is not being used.

Port #	Name	Pin	Pull up/down	Description
5	DDID CTLDATA	DDPD_CTRLDATA	No (Weak pull down on reset)	DDC for port D. Digital port D present strap is set if pin is 1 at rising edge of PCH_PWROK.
	DDID CTLCLK	DDPD_CTRLCLK	No	
4	DDIB CTLDATA	DDPB_CTRLDATA	No (Weak pull down on	DDC for port B. Digital port B present strap is set if pin is 1 at rising

Port #	Name	Pin	Pull up/down	Description
			reset)	edge of PCH_PWROK.
	DDIB CTLCLK	DDPB_CTRLCLK	No	
3	DDIC CTLDATA	DDPC_CTRLDATA	No (Weak pull down on reset)	DDC for port C. Digital port C present strap is set if pin is 1 at rising edge of PCH_PWROK.
	DDIC CTLCLK	DDPC_CTRLCLK	No	

Program 0xC2020[31] = 0x1 for the entire time that GMBUS function is used. The value can be safely left at 0x1 when GMBUS is not being used.

This will disable GMBUS unit level clock gating and only rely on partition level clock gating.

GPIO Programming for I2C Bit Bashing

To drive GPIO pin low, program direction to "out" and data value to "0", along with mask bits.

To drive GPIO pin high (tristate to allow external pull up to activate), program direction to "in", along with mask bit. No need to set data value to "1".

GMBUS Controller Programming Interface

The GMBUS (Graphic Management Bus) is used to access/control devices connected to the GPIO pins.

Basic features:

1. I²C compatible.
2. Bus clock frequency of 50 KHz or 100 KHz.
3. Attaches to any of the GPIO pin pairs.
4. 7-bit or 10-bit Slave Address and 8-bit or 16-bit index.
5. Double buffered data register and a 9 bit counter support 0 byte to 256 byte transfers.
6. Supports stalls generated by the slave device pulling down the clock line (Slave Stall), or delaying the slave acknowledge response.
7. Status register indicates error conditions, data buffer busy, time out, and data complete acknowledgement.
8. Detects and reports time out conditions for a stall from a slave device, or a delayed or missing slave acknowledge.
9. Interrupts may optionally be generated.
10. Does not directly support segment pointer addressing as defined by the Enhanced Display Data Channel standard.

Segment pointer addressing as defined by the Enhanced Display Data Channel standard:

1. Use bit bashing (manual GPIO programming) to complete segment pointer write **without terminating in a stop or wait cycle**.
2. Terminate bit bashing phase with both I^2C lines pulled high by tri-stating the data line before the clock line. Follow EDDC requirement for response received from slave device.
3. Initiate GMBUS cycle as required to transfer EDID following normal procedure.

Display Watermark Programming

Watermark Overview

The display watermarks are used to control the display engine memory request behavior.

Description
The default settings of the watermark configuration registers will **not** allow the display engine to operate. The watermark values must be properly calculated and programmed in order to enable a display and achieve optimum power and performance. Incorrectly programmed watermark values can result in screen corruption.

The watermarks should be calculated and programmed when any of the watermark calculation inputs change. This includes planes enabling or disabling, plane source format or size changing, etc.

Besides programming the watermark registers, there are other display configuration requirements and registers that must be programmed in order for the display to operate in a low power mode, and there are memory controller configuration requirements which are not documented here.

Watermark Calculations

The display watermarks are calculated using information from the display configuration and memory latencies. The watermarks must be calculated and programmed before enabling a plane or changing a plane configuration.

For YUV 420 Planar formats, only the Y surface watermark value is calculated and programmed. Though the UV surface watermark value is not programmed separately, it must be calculated to make sure that the UV buffer allocation satisfies the latency requirements.

diThe ceiling function rounds any non-integer value up to the next greater integer. Example: ceiling[0.3]=1, ceiling[2.1]=3, ceiling[4.8]=5, ceiling[4]=4

Watermark Algorithm

1. Retrieve memory latency values
    - See the Memory Values section to find the memory latency values
    - The memory values do not change after boot, so software may cache them to avoid re-reading
2. For each enabled pipe (run each time pipe configuration changes)
    A. Calculate adjusted pipe pixel rate
        I. Adjusted pipe pixel rate = pixel rate for the screen resolution
            - If there will be dynamic switching between refresh rates, either use the fastest pixel rate, or re-calculate using the current pixel rate when the refresh rate is switched
            - If plane 90 or 270 rotation is enabled, use the rotated width and height in pixel rate calculations.
        II. If TRANS_CONF Interlaced Mode == PF-ID, adjusted pipe pixel rate = adjusted pipe pixel rate * 2
        III. If pipe scaling enabled, adjusted pipe pixel rate = adjusted pipe pixel rate * pipe down scale amount
            - See the Scaling section to find the down scale amount
    B. Program WM_LINETIME Line Time = roundup[8 * pipe horizontal total pixels / adjusted pipe pixel rate MHz]
3. For each enabled plane (run each time pipe or plane configuration changes)
    A. Calculate adjusted plane pixel rate
        I. Adjusted plane pixel rate = adjusted pipe pixel rate
        II. If plane scaling enabled, adjusted plane pixel rate = adjusted plane pixel rate * plane down scale amount
            - See the Scaling section to find the down scale amount
    B. For each valid memory latency level

Plane Bytes per pixel	Minimum Scanlines for Y Tile	
	0/180 Rotation	90/270 Rotation
2 or YUV422	4	8
4	4	4
8	4	N/A

        I. Calculate method 1
            - Method 1 = memory latency microseconds * adjusted plane pixel rate MHz * plane source bytes per pixel / 512
        II. Calculate method 2
            - plane bytes per line = plane source width pixels * plane source bytes per pixel

- plane blocks per line = Plane memory format is Y tile ? ceiling[Y tile minimum lines * plane bytes per line / 512]/Y tile minimum lines: ceiling[plane bytes per line / 512]
- Method 2 = ceiling[(memory latency microseconds * adjusted plane pixel rate MHz) / Pipe horizontal total number of pixels] * plane blocks per line

III. Calculate Y tile minimum
- Y tile minimum = Minimum Scanlines from table * plane blocks per line

IV. Select the watermark result
- If plane memory format is X tile or linear
  - If (plane buffer allocation / plane blocks per line) >=1
    - Selected Result Blocks = minimum[Method 1, Method 2]
  - Else
    - Selected Result Blocks = Method 1
- Else // Y tile
  - Selected Result Blocks = maximum[Method 2, Y tile minimum]

V. Convert result to blocks and lines
- Result Blocks = ceiling[Selected Result Blocks] + 1
- Result Lines = ceiling[Selected Result Blocks / plane blocks per line]
- If latency level 1 through 7 and Y tile:

  Result Lines = Result Lines + Y tile minimum lines

  Result Blocks = Result Lines * plane blocks per line

  If latency level 1 through 7 and not Y tile:

  Result Blocks = Result Blocks + 1
- If Render Decompression enabled and latency level 0:

  Result Lines = Result Lines + Y tile minimum lines

  Result Blocks = Result Blocks + (Y tile minimum lines * plane blocks per line)

  // For latency levels 1 through 7 ensure that the their result lines/blocks >= latency level 0 lines/blocks.

  For latency levels 1 through 7

  If latency level result blocks < latency level 0 blocks

  latency level result blocks = latency level 0 blocks

  If latency level result lines < latency level 0 lines

  latency level result lines = latency level 0 lines

VI. Compare against the maximum
- If (Result Blocks >= plane buffer allocation) or (Result Lines > 31), maximum exceeded for this latency level
- For YUV 420 Planar formats, perform the above check for both Y and UV planes.

4. For transition watermark
    A. Calculate transition offset
        - Transition Offset Blocks = Transition minimum + Transition amount
        - See Transition Watermark section for transition minimum and transition amount
    B. Calculate transition Y tile minimum
        - Transition Y tile minimum = 2 * memory latency level 0 Y tiled minimum
    C. Select the watermark result
        - If plane memory format is X tile or linear
            - Result Blocks = Memory latency level 0 Selected Result Blocks + Transition Offset Blocks
        - Else // Y tile
            - Result Blocks = maximum[Memory latency level 0 Selected Result Blocks, Transition Y tile minimum] + Transition Offset Blocks
    D. Convert result to blocks
        - Result Blocks = ceiling[Result Blocks] + 1
        - If not Y tile:

            Result Blocks = Result Blocks + 1
    E. Compare against the maximum
        - If (Result Blocks >= plane buffer allocation), maximum exceeded for transition watermark
        - For YUV 420 Planar formats, perform the above check for both Y and UV planes.
5. Program watermark registers
    A. For each latency level 0 to 7
        - If memory latency for this level is invalid, or the maximum was exceeded for this level or any previous level, program PLANE_WM_<latency level> Enable = 0
            - **If watermark latency level 0 exceeds the maximum, the plane must not be enabled.**
        - Else program PLANE_WM_<latency level> Enable = 1, Lines = Result Lines, Blocks = Result Blocks
            - The latency level 0 Lines value is ignored by hardware
            - The latency level 1-7 Lines values are used for Y tiling formats and the level 1-7 Block values are used for other formats.
    B. For transition watermark
        - If the maximum was exceeded for the transition watermark, program PLANE_WM_TRANS Enable = 0
        - Else program PLANE_WM_TRANS Enable = 1, Blocks = Transition Result Blocks
            - The transition watermark Lines value is ignored by hardware

C. Write the plane surface base address register to trigger update of the watermarks and other plane double buffered registers. This should be done only after all plane configuration is configured to match the new watermark values.

Transition Watermark

The transition watermark is used for Isochronous Priority Control (IPC). When IPC is enabled (ARB_CTL2 Enable IPC), plane read requests are sent at high priority until filling above the transition watermark, then the requests are sent at lower priority until dropping below the level 0 watermark. The lower priority requests allow other memory clients to have better memory access. If the transition watermark is not enabled, the plane behaves as if the transition watermark was programmed to the top of the plane buffer allocation. When IPC is disabled, all plane read requests are sent at high priority.

The transition watermark is programmed as a tunable amount above the level 0 watermark. Tuning to higher values will tend to cause longer periods of high priority reads followed by longer periods of lower priority reads. Tuning to lower values will tend to cause shorter periods of high and lower priority reads. The exact behavior depends on the memory bandwidth, display bandwidth, and other memory traffic in the system.

The transition watermark has a minimum value to ensure the demote does not happen before enough data has been read to meet the level 0 watermark requirements.

Transition Minimum
14 Blocks

Scaling

A scaler (pipe or plane scaler) is down scaling when it is enabled and the scaler input size is greater than the scaler output size.

Down scaling effectively increases the pixel rate. Up scaling does not reduce the pixel rate.

For plane scaling, the scaler input size is the plane size and the output size is the scaler window size.

For pipe scaling, the scaler input size is the pipe source size and the output size is the scaler window size.

Horizontal down scale amount = maximum[1, Horizontal source size / Horizontal destination size]

Vertical down scale amount = maximum[1, Vertical source size / Vertical destination size]

Total down scale amount = Horizontal down scale amount * Vertical down scale amount

Workaround	
Context:	Watermark Calculations

Workaround when calculating watermarks
Calculate arbitrated display bandwith:
For each pipe {
For each plane, except cursor, enabled on the pipe {
Plane bandwidth MB/s = pixel rate MHz * source pixel format in bytes * plane down scale	

Workaround	
Context:	Watermark Calculations

amount * pipe down scale amount

}

Pipe bandwidth MB/s = Number of planes, except cursor, enabled on the pipe * Max(bandwidth from each plane) // find plane with highest bandwidth and multiply with number of enabled planes

}

Arbitrated display bandwidth = Number of pipes enabled * Max(bandwidth from each pipe) // find pipe with highest bandwidth and multiply with number of enabled pipes

If there is any Ytile plane enabled and arbitrated display bandwidth > 20% of raw system memory bandwidth (# memory channels * memory frequency * 8 bytes) {

Double Ytile planes minimum lines and program all watermark levels accordingly. Do not change the latency values.

Increase Xtile planes watermark latency for all levels by 15us

}

If there is no Ytile plane enabled and arbitrated display bandwidth > 60% of raw system memory bandwidth (# memory channels * memory frequency * 8 bytes) {

Increase Xtile planes watermark latency for all levels by 15uS

}

System Agent Geyserville (SAGV)

SAGV dynamically adjusts the system agent voltage and clock frequencies depending on power and performance requirements. The display engine access to system memory is blocked during the adjustment time.

SAGV defaults to enabled. Software must use the GT-driver pcode mailbox to disable SAGV when the display engine is not able to tolerate the blocking time.

See the Memory Values section to find the SAGV block time.

Requirement before plane enabling or configuration change: Disable SAGV if any enabled plane will not be able to enable watermarks for memory latency >= SAGV block time, or any transcoder is interlaced. Else, enable SAGV.

If software ensures single pipe configurations always have enough data buffer allocation to tolerate SAGV, it can then simply disable SAGV anytime multiple display pipes are enabled or interlace is enabled, and re-enable SAGV when switching back to a single, non-interlaced pipe.

Sequence to Disable SAGV

1. Ensure any previous GT Driver Mailbox transaction is complete.
2. Write GT Driver Mailbox Data0 (GTTMMADDR offset 0x138128) = 0x00000000.
3. Write GT Driver Mailbox Interface (GTTMMADDR offset 0x13812C) = 0x80000021.
4. Poll for GT Driver Mailbox Interface Run/Busy == 0x0.
    - Timeout and fail after 150 us.
5. Read GT Driver Mailbox Data0, if bit 0 == 0x1, continue, else go to step 2.
    - If the condition in step 5 is not satisfied after cycling through steps 2-5 for 1 ms (typically <200 us), timeout and fail.
6. Continue with plane and pipe programming.

Sequence to Enable SAGV

1. Ensure any previous GT Driver Mailbox transaction is complete.
2. Write GT Driver Mailbox Data0 (GTTMMADDR offset 0x138128) = 0x00000003.
3. Write GT Driver Mailbox Interface (GTTMMADDR offset 0x13812C) = 0x80000021.

Continue with plane and pipe programming. There is no need to wait for SAGV enabling to complete.

Examples

Example pixel rate adjustments:

Pixel rate for screen resolution is 130 MHz. No interlacing. Pipe scale 1920x1080 pipe source size to 1714x1120 scaler window size. Plane scale 1920x1080 plane size to 800x600 scaler window size.

Pipe horizontal down scale amount = maximum[1, 1920 / 1714] = 1.12

Pipe vertical down scale amount = maximum[1, 1080 / 1120] = 1 **// Max condition was hit**

Pipe total down scale amount = 1.12 * 1 = 1.12

Adjusted pipe pixel rate = 130 MHz * 1.12 = 145.6 MHz

Plane horizontal down scale amount = maximum[1, 1920 / 800] = 2.4

Plane vertical down scale amount = maximum[1, 1080 / 600] = 1.8

Plane total down scale amount = 2.4 * 1.8 = 4.32

Adjusted plane pixel rate = 145.6 MHz * 4.32 = 628.99 MHz

Example method, block, and line calculations:

Plane source 4 Bpp, Plane X tile, Plane source width 1920 pixels, Horizontal total 2200 pixels, Adjusted plane pixel rate 148.5 MHz, memory latency 7.5 us

Method 1 = 148.5 MHz * 4 Bpp * 7.5 us / 512 = 8.7 blocks

Plane bytes per line = 1920 pixels * 4 Bpp = 7680 Bytes/line

Plane blocks per lines = ceiling[7680 / 512] = 15 blocks

Method 2 = ceiling[(7.5 us * 148.5 MHz) / 2200 pixels] * 15 blocks = 15 blocks

Y tile minimum = 4 * 15 blocks = 60 blocks

Result Blocks = minimum[8.7 blocks, 15 blocks] = 8.7 blocks // X tile so does not use Y tile minimum

Result Blocks = ceiling[8.7 blocks] + 1 block = 10 blocks

Result Lines = ceiling[8.7 blocks / 15] = 1 lines

Memory Values

Retrieve Memory Latency Data

1. Write GT Driver Mailbox Data0=0x0000_0000 (first set of latency values) and GT Driver Mailbox Data1=0x0000_0000
2. Write GT Driver Mailbox Interface Run/Busy=1, Address Control=All 0s, Command/Error Code=06h
3. Poll GT Driver Mailbox Interface for Run/Busy indication=0b and Command/Error Code=00h (success)
    - Timeout after 100 us and do not enable display planes.
4. Read GT Driver Mailbox Data0 for the first set of memory latency values
5. Write GT Driver Mailbox Data0=0x0000_0001 (second set of latency values) and GT Driver Mailbox Data1=0x0000_0000
6. Write GT Driver Mailbox Interface Run/Busy=1, Address Control=All 0s, Command/Error Code=06h
7. Poll GT Driver Mailbox Interface for Run/Busy indication=0b and Command/Error Code=00h (success)
    - Timeout after 100 us and do not enable display planes.
8. Read GT Driver Mailbox Data0 for the second set of memory latency values

Memory Latency Data Definition

First Set		
Data0 Bit	**Name**	**Description**
31:24	Level 3	Number of microseconds for level 3.
23:16	Level 2	Number of microseconds for level 2.
15:8	Level 1	Number of microseconds for level 1.
7:0	Level 0	Number of microseconds for level 0.

Second Set		
Data0 Bit	**Name**	**Description**
31:24	Level 7	Number of microseconds for level 7.
23:16	Level 6	Number of microseconds for level 6.
15:8	Level 5	Number of microseconds for level 5.
7:0	Level 4	Number of microseconds for level 4.

If level 1 or any higher level has a value of 0x00, that level and any higher levels are unused and invalid, so the associated watermark registers must not be enabled.

It is allowed to have the same value in adjacent levels.

Workaround	
Context:	Display Watermark Programming
The mailbox response data may not account for memory read latency. If the mailbox response data for level 0 is 0us, add 2 microseconds to the result for each valid level.	

SAGV Block Time

SAGV Block Time
30 us