Intel® Open Source HD Graphics Programmers' Reference Manual (PRM)

Volume 2, Part 4: Command Reference - Structures

For the 2014 Intel Atom™ Processors, Celeron™ Processors, and Pentium™ Processors based on the "BayTrail" Platform (ValleyView graphics)

Creative Commons License

You are free to Share — to copy, distribute, display, and perform the work

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

No Derivative Works. You may not alter, transform, or build upon this work

Notices and Disclaimers

Table of Contents

3DSTATE_CONSTANT(Body)

Source: RenderCS

Size (in bits): 192

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:16	**Constant Buffer 1 Read Length** Format: U16 read length This field specifies the length of the constant data to be loaded from memory in 256-bit units. **Programming Notes** • The sum of all four read length fields must be less than or equal to the size of 64 • Setting the value of the register to zero will disable buffer 1. • If disabled, the **Pointer to Constant Buffer 1** must be programmed to zero.
	15:0	**Constant Buffer 0 Read Length** Format: U16 read length This field specifies the length of the constant data to be loaded from memory in 256-bit units. **Programming Notes** • The sum of all four read length fields must be less than or equal to the size of 64 • Setting the value of the register to zero will disable buffer 0. • If disabled, the **Pointer to Constant Buffer 0** must be programmed to zero.
1	31:16	**Constant Buffer 3 Read Length** Format: U16 read length This field specifies the length of the constant data to be loaded from memory in 256-bit units. **Programming Notes** • The sum of all four read length fields must be less than or equal to the size of 64 • Setting the value of the register to zero will disable buffer 3. • If disabled, the **Pointer to Constant Buffer 3** must be programmed to zero.
	15:0	**Constant Buffer 2 Read Length** Format: U16 read length This field specifies the length of the constant data to be loaded from memory in 256-bit units. **Programming Notes** • The sum of all four read length fields must be less than or equal to the size of 64 • Setting the value of the register to zero will disable buffer 2.

3DSTATE_CONSTANT(Body)

		• If disabled, the **Pointer to Constant Buffer 2** must be programmed to zero.
2	31:5	**Pointer To Constant Buffer 0** Format: GraphicsAddress[31:5]ConstantBuffer This field points to the location of Constant Buffer 0. The state of **INSTPM<CONSTANT_BUFFER Address Offset Disable>** determines whether the Dynamic State Base Address is added to this pointer. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Constant Buffer Object Control State** Format: MEMORY_OBJECT_CONTROL_STATE Specifies the memory object control state for all constant buffers defined in this command.
3	31:5	**Pointer To Constant Buffer 1** Format: GraphicsAddress[31:5]ConstantBuffer This field points to the location of Constant Buffer 1. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ
4	31:5	**Pointer To Constant Buffer 2** Format: GraphicsAddress[31:5]ConstantBuffer This field points to the location of Constant Buffer 2. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ
5	31:5	**Pointer To Constant Buffer 3** Format: GraphicsAddress[31:5]ConstantBuffer This field points to the location of Constant Buffer 3. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ

AddrSubRegNum

Source:	EuIsa
Size (in bits):	3
Default Value:	0x00000000

Address Subregister Number
This field provides the subregister number for the address register. The address register contains 8 sub-registers. The size of each subregister is one word. The address register contains the register address of the operand, when the operand is in register-indirect addressing mode.
This field applies to the destination operand and the source operands. It is ignored (or not present in the instruction word) for an immediate source operand.
This field is present if the operand is in register-indirect addressing mode; it is not present if the operand is directly addressed.
An address subregister used for indirect addressing is often called an index register.

DWord	Bit	Description
0	2:0	**Address Subregister Number** Value: 0-7 — Name: Address Subregister Number

ARBITRATION_PRIORITY

Source: BSpec
Size (in bits): 2
Default Value: 0x00000000

This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.

DWord	Bit	Description
0	1:0	**Priority** Format: U2 Value / Name: 00b — Highest priority 01b — Second highest priority 10b — Third highest priority 11b — Lowest priority

AVC CABAC

Source: VideoCS
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15	**Reserved** Format: MBZ
	14	**Coefficient level out-of-bound Error** This flag indicates the coded coefficient level SEs in the bit-stream is out-of-bound.
	13	**Reserved** Format: MBZ
	12	**Reserved** Format: MBZ
	11	**Temporal Direction Motion Vector Out-of-Bound Error** This flag indicates motion vectors calculated from Temporal Direct Motion Vector is larger than the allowed range specified by the AVC spec.
	10	**Final Motion Vector Out-of-Bound Error** This flag indicates final reconstructed Motion Vector value is larger than the allowed range specified by the AVC spec.
	9	**Motion Vector Delta SE Out-of-Bound Error** This flag indicates inconsistent Motion Vector Delta SEs coded in the bit-stream.
	8	**Reference Index SE Out-of-Bound Error** This flag indicates inconsistent Reference Index SEs coded in the bit-stream.
	7	**MacroBlock QpDelta Error** This flag indicates out-of-bound MB QP delta SEs coded in the bit-stream.
	6	**Motion Vector Delta SE Error** This flag indicates out-of-bound motion vector delta SEs coded in the bit-stream.
	5	**Reference Index SE Error** This flag indicates out-of-bound Refidx SEs coded in the bit-stream.
	4	**Residual Error** This flag indicates out-of-bound absolute coefficient level SEs coded in the bit-stream.
	3	**Slice end Error** This flag indicates a pre-matured slice_end SE or inconsistent slice end on the last MB of a slice.
	2	**Chroma Intra prediction Mode Error** This flag indicates inconsistent Chroma Intra prediction mode SEs coded in the bit-stream.
	1	**Luma Intra prediction Mode Error** This flag indicates inconsistent luma Intra prediction mode SE coded in the bit-stream.
	0	**MB Concealment Flag** Each pulse from this flag indicates one MB is concealed by hardware.

AVC CAVLC

Source:	VideoCS
Size (in bits):	16
Default Value:	0x00000000

DWord	Bit	Description
0	15	**Total Zero out-of-bound Error** This flag indicates the Total zero SE count exceed the max number of coeffs allowed in an intra16x16 AC block.
	14	**Coefficient level out-of-bound Error** This flag indicates the coded coefficient level SEs in the bit-stream is out-of-bound.
	13	**RunBefore out-of-bound Error** This flag indicates the coded RunBefore SE value is larger than the remaining zero block count.
	12	**Total coefficient Out-of-bound Error** This flag indicates the coded total coeff SE count exceed the max number of coeffs allowed in an intra16x16 AC block.
	11	**Temporal Direction Motion Vector Out-of-Bound Error** This flag indicates motion vectors calculated from Temporal Direct Motion Vector is larger than the allowed range specified by the AVC spec.
	10	**Final Motion Vector Out-of-Bound Error** This flag indicates final reconstructed Motion Vector value is larger than the allowed range specified by the AVC spec.
	9	**Motion Vector Delta SE Out-of-Bound Error** This flag indicates inconsistent Motion Vector Delta SEs coded in the bit-stream.
	8	**Reference Index SE Out-of-Bound Error** This flag indicates inconsistent Reference Index SEs coded in the bit-stream.
	7	**RunBefore/TotalZero Error** This flag indicates one or more inconsistent RunBefore or TotalZero SEs coded in the bit-stream.
	6	**Exponential Golomb Error** This flag indicates hardware detects more than 18 leadzero for skip and more than 19 for other SEs from the Exponential Golomb Logic
	5	**Total Coeff SE Error** This flag indicates one or more inconsistent total coeff SEs coded in the bit-stream.
	4	**Macroblock Coded Block Pattern Error** This flag indicates inconsistent CBP SEs coded in the bit-stream.
	3	**Mbytpe/submbtype Error** This flag indicates inconsistent MBtype/SubMBtype SEs coded in the bit-stream.
	2	**Chroma Intra prediction Mode Error** This flag indicates inconsistent Chroma Intra prediction mode SEs coded in the bit-stream.
	1	**Luma Intra prediction Mode Error** This flag indicates inconsistent luma Intra prediction mode SE coded in the bit-stream.
	0	**MB Concealment Flag** Each pulse from this flag indicates one MB is concealed by hardware.

AVMUX_Packetization_Parameter

Source: BSpec

Size (in bits): 256

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:24	**GFX WNIC TTL offset in CL** Format: U8 Offset relatives to 2K/4K/8K TFD entry base address. This field is only valid when output is in TFD format - ignored when output is packed TS packed AV muxed data. **Programming Notes** HW ignores this field and TTL offset is written to 1984 (Last CL of TFD buffer).
	23:12	**Reserved** Format: MBZ
	11:0	**Reserved**
1	31:16	**GFX_WNIC_SHARED_DATABUFFER_STRIDE** Format: U16 This register contains the size (in bytes) of the stride - used to determine the start of the next TFD buffer. For e.g: if WNIC points each TFD descriptor to a separate 4KB page then stride would be 4KB, if WNIC maps adjacent TFD to top and middle of a page then stride would be 2 KB. This field is ignored by hardware if AV Muxing is disabled. This register is populated by the graphics driver in concert with the WNIC driver. The address must not be changed when a wireless session is active. If MMIO address needs to change due to PCI rebalancing, graphics driver must take steps to stop wireless session, program the register and then re-activate the wireless session. **Programming Notes** Hardware supports 2K stride size. HW ignores this field.
	15:0	**GFX_WNIC_SHARED_DATABUFFER_PACKSIZE** Exists If: //[AV Muxing] is Enabled Format: U16 This register contains the size (in bytes) of the data that must be output into a TFD. This is expected to be a multiple of 188 byte packets (MPEG transport packet size). This field is ignored by hardware if AV Muxing is disabled. **Programming Notes** Hardware only supports constant packet size of 1316 bytes. HW ignores this field.
2	31:17	**Reserved** Format: MBZ
	16	**PSI Stream Enable**

AVC CAVLC

		Format: U1
	15:0	**TFD message Timer Expiration Count** Format: U16 # of Clocks This field specified the time-out threshold value. If the idle timer is greater than this threshold, any completed TFD packets in the TFD output queue will be flushed to WNIC TFD Buffer. **Value / Name / Description** 0 / Timer Timeout is Disabled 1-FFFFh / Number of Clock to wait before flushing the output queue
3	31:28	**Reserved** Format: MBZ
	27:16	**Start 2k byte offset position for AV multiplexer** Format: U12 This field should be set to zero in functional mode (Reserved)Reserved.
	15:12	**Reserved** Exists If: //[AV Muxing] disabled Format: MBZ
	15:0	**GFX_WNIC_SHARED_DATABUFFER_ENTRY_SIZE** Exists If: //[AV Muxing] enabled Format: U16 This field specifies the number of the TFD entry contained in the shared databuffer. The size of each entry is specified in GFX_WNIC_SHARED_DATABUFFER_PACKSIZE bitfield (Refer to DW0, Bits[15:0] of this packet. Valid Sizes are in multiples of power of 2: 256, 512, 1K and 2K only. **Programming Notes** Please note that this field is ignored by hardware if AV Muxing is disabled.
	11:0	**Count 2K Buffer Minus 1** Exists If: //[AV Muxing] disabled Format: U12 This field specifies the number TFD entries / Size of the Circular AV mux buffer. For example a value of 63 indicates 64 TFD entries / 32 KB of circular buffer. A value of 4095 indicates 4096 TFD entries / 2 MB of circular buffer.
4	31:16	**Reserved**

AVC CAVLC

		Format: MBZ H/W ignores this fiels, place holder for future Interrupt Mask Register[15:0].
	15	**Reserved**
	14:2	**Reserved** Format: MBZ
	1	**Reserved**
	0	**Reserved**
5	31:13	**Reserved** Format: MBZ
	12:0	**Video Packet ID Header Parameter** Format: U13 This field specified the PID field of the MPEG header for each Video TS packets.
6	31:16	**PTS Delta Adjustment For Single Frame Transmission** Format: U16 This field specifies the PTS adjustment apply for the current large frame when the subsequent frame gets dropped due to large I-frame transmission. This value gets added to the original PTS value for the large frame.
	15:0	**TTL Delta Adjustment For Single Large Frame Transmission** Format: U16 This field specifies the TTL adjustment apply for the current large frame when the subsequent frame gets dropped due to large I-frame transmission. This value gets added to the original TTL value for the large frame.
7	31:16	**IntraMbConfSize - Max-bit size conformance Intra Flag** Format: U16 This field specifies the max number of bits allowed per MB to ensure spec conformance. The high order coefficients for the approximate blocks will be zero out when the current MB bit size exceed the programmed value. (panic mode - would give unpredictable quality result for the nonconformance MBs) N = Number of bits allowed per intra coded MB.

Value	Name	Description

AVC CAVLC

		0		Do not change intra macroblocks even if they are not 3200/6400 bit conformant for 4:2:0 and 4:4:4 chroma sub-sampling mode respectively.
	15:0	**InterMbConfSize - Max-bit size conformance Inter flag**		
		Format:		U16
		This field specifies the max number of bits allowed per MB to ensure spec conformance. The high order coefficients for the approximate blocks will be zero out when the current MB bit size exceed the programmed value. (panic mode - would give unpredictable quality result for the nonconformance MBs)		
		N = Number of bits allowed per inter coded MB.		
		Value	**Name**	**Description**
		0		Do not change inter macroblocks even if they are not 3200/6400 bit conformant for 4:2:0 and 4:4:4 chroma sub-sampling mode respectively.

BCS Hardware-Detected Error Bit Definitions

Source: BlitterCS
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15:3	**Reserved** Format: MBZ
	2	**Reserved** Format: MBZ
	1	**Reserved** Format: MBZ
	0	**Instruction Error** This bit is set when the Renderer Instruction Parser detects an error while parsing an instruction. Instruction errors include: • Client ID value (Bits 31:29 of the Header) is not supported (only MI, 2D and 3D are supported). • Defeatured MI Instruction Opcodes: **Value** | **Name** | **Description** 1 | | Instruction Error detected **Programming Notes** This error indications cannot be cleared except by reset (i.e., it is a fatal error).

BINDING_TABLE_STATE

Source:	BSpec
Size (in bits):	32
Default Value:	0x00000000

The binding table binds surfaces to logical resource indices used by shaders and other compute engine kernels. It is stored as an array of up to 256 elements, each of which contains one dword as defined here. The start of each element is spaced one dword apart. The first element of the binding table is aligned to a 32-byte boundary.

DWord	Bit	Description
0	31:5	**Surface State Pointer** Format: SurfaceStateOffset[31:5] This 32-byte aligned address points to a surface state block. This pointer is relative to the **Surface State Base Address**
	4:0	**Reserved** Format: MBZ

Bit Definition for Interrupt Control Registers - Render

Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:12	**Reserved** Format: MBZ Reserved for other command streamers - cannot be allocated by main command streamer.
	11:10	**Reserved** Format: MBZ These bits may be assigned to interrupts in future products or steppings.
	9	Reserved
	8	**Reserved**
	7	**Page Fault** Description: This bit is set whenever there is a pending GGTT/PPGTT (page or directory) fault in Render command streamer.
	6	**Timeout Counter Expired** Set when the render pipe timeout counter (0x02190) has reached the timeout threshold value (0x0217c).
	5	**Reserved** Format: MBZ
	4	**PIPE_CONTROL Notify Interrupt** The Pipe Control packet (Fences) specified in 3D pipeline document may optionally generate an Interrupt. The Store QW associated with a fence is completed ahead of the interrupt.
	3	**Render Command Parser Master Error** When this status bit is set, it indicates that the hardware has detected an error. It is set by the device upon an error condition and cleared by a CPU write of a one to the appropriate bit contained in the Error ID register followed by a write of a one to this bit in the IIR. Further information on the source of the error comes from the "Error Status Register" which along with the "Error Mask Register" determine which error conditions will cause the error status bit to be set and the interrupt to occur. **Page Table Error:** Indicates a page table error. **Instruction Parser Error:** The Render Instruction Parser encounters an error while parsing an instruction.
	2	**Sync Status**

Bit Definition for Interrupt Control Registers - Render		
		This bit is set in the Hardware Status Page DW offset 0 when the Instruction Parser completes a flush with the sync enable bit active in the INSTPM register. The toggle event will happen after the render engine is flushed. The HW Status DWord write resulting from this toggle will cause the CPU's view of graphics memory to be coherent as well (flush and invalidate the render cache). **It is the driver's responsibility to clear this bit before the next sync flush with HWSP write enabled.**
	1	**Reserved**
	0	**Render Command Parser User Interrupt** This status bit is set when an MI_USER_INTERRUPT instruction is executed on the Render Command Parser. Note that instruction execution is not halted and proceeds normally. A mechanism such as an MI_STORE_DATA instruction is required to associate a particular meaning to a user interrupt.

BLEND_STATE

Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000

The blend state is stored as an array of up to 8 elements, each of which contains the DWords described here. The start of each element is spaced 2 DWords apart. The first element of the blend state array is aligned to a 64-byte boundary, which is pointed to by a field in 3DSTATE_CC_STATE_POINTERS. The 3-bit Render Target Index field in the Render Target Write data port message header is used to select which of the 8 elements from BLEND_STATE that is used on the current message.

DWord	Bit	Description
0	31	**Color Buffer Blend Enable** Format: Enable Enables the ColorBufferBlending (nee "alpha blending") function of the Pixel Processing Pipeline for this render target. **Programming Notes** Enabling LogicOp and ColorBufferBlending at the same time is UNDEFINED
	30	**Independent Alpha Blend Enable** Format: Enable When enabled, the other fields in this instruction control the combination of the alpha components in the Color Buffer Blend stage. When disabled, the alpha components are combined in the same fashion as the color components.
	29	**Reserved** Format: MBZ
	28:26	**Alpha Blend Function** Format: 3D_ColorBufferBlendFunction This field specifies the function used to combine the alpha components in the Color Buffer blend stage of the Pixel Pipeline when the IndependentAlphaBlend state is enabled. Value / Name: 0 BLENDFUNCTION_ADD 1 BLENDFUNCTION_SUBTRACT 2 BLENDFUNCTION_REVERSE_SUBTRACT 3 BLENDFUNCTION_MIN 4 BLENDFUNCTION_MAX 5 - 7 Reserved
	25	**Reserved** Format: MBZ
	24:20	**Source Alpha Blend Factor** Format: 3D_ColorBufferBlendFactor Controls the "source factor" in alpha Color Buffer Blending stage. Note: For the

BLEND_STATE

source/destination alpha blend factors, the encodings indicating "COLOR" are the same as the encodings indicating "ALPHA", as the alpha component of the color is selected.

Value	Name
00h	Reserved
01h	BLENDFACTOR_ONE
02h	BLENDFACTOR_SRC_COLOR
03h	BLENDFACTOR_SRC_ALPHA
04h	BLENDFACTOR_DST_ALPHA
05h	BLENDFACTOR_DST_COLOR
06h	BLENDFACTOR_SRC_ALPHA_SATURATE
07h	BLENDFACTOR_CONST_COLOR
08h	BLENDFACTOR_CONST_ALPHA
09h	BLENDFACTOR_SRC1_COLOR
0Ah	BLENDFACTOR_SRC1_ALPHA
0Bh-10h	Reserved
11h	BLENDFACTOR_ZERO
12h	BLENDFACTOR_INV_SRC_COLOR
13h	BLENDFACTOR_INV_SRC_ALPHA
14h	BLENDFACTOR_INV_DST_ALPHA
15h	BLENDFACTOR_INV_DST_COLOR
16h	Reserved
17h	BLENDFACTOR_INV_CONST_COLOR
18h	BLENDFACTOR_INV_CONST_ALPHA
19h	BLENDFACTOR_INV_SRC1_COLOR
1Ah	BLENDFACTOR_INV_SRC1_ALPHA

19:15 **Destination Alpha Blend Factor**

Format:	3D_ColorBufferBlendFactor

Controls the "destination factor" in alpha Color Buffer Blending stage. Refer to Source Alpha Blend Factor for encodings.

14 **Reserved**

Format:	MBZ

13:11 **Color Blend Function**

Format:	3D_ColorBufferBlendFunction

This field specifies the function used to combine the color components in the ColorBufferBlending function of the Pixel Processing Pipeline. If Independent Alpha Blend Enable is disabled, this field will also control the blending of the alpha components in the ColorBufferBlending function.

BLEND_STATE

Value	Name
0	BLENDFUNCTION_ADD
1	BLENDFUNCTION_SUBTRACT
2	BLENDFUNCTION_REVERSE_SUBTRACT
3	BLENDFUNCTION_MIN
4	BLENDFUNCTION_MAX

	10	**Reserved** Format: MBZ
	9:5	**Source Blend Factor** Format: 3D_ColorBufferBlendFactor Controls the "source factor" in the ColorBufferBlending function. Refer to Source Alpha Blend Factor for encodings.
	4:0	**Destination Blend Factor** Format: 3D_ColorBufferBlendFactor Controls the "destination factor" in the ColorBufferBlending function. Refer to Source Alpha Blend Factor for encodings.
1	31	**AlphaToCoverage Enable** Format: Enable If set, Source0 Alpha is converted to a temporary 1/2/4-bit coverage mask and the mask bit corresponding to the sample# ANDed with the sample mask bit. If set, sample coverage is computed based on src0 alpha value. Value of 0 disables all samples and value of 1 enables all samples for that pixel. The same coverage needs to apply to all the RTs in MRT case. Further, any value of src0 alpha between 0 and 1 monotonically increases the number of enabled pixels. The same coverage needs to be applied to all the RTs in MRT case.
	30	**AlphaToOne Enable** Format: Enable If set, Source0 Alpha is set to 1.0f after (possibly) being used to generate the AlphaToCoverage coverage mask. The same coverage needs to be applied to all the RTs in MRT case. If Dual Source Blending is enabled, this bit must be disabled.
	29	**AlphaToCoverage Dither Enable** Format: Enable If set, sample coverage is computed based on src0 alpha value and it modulates the sample coverage based on screen coordinates. Value of 0 disables all samples and value of 1 enables all samples for that pixel. The same coverage needs to apply to all the RTs in MRT case. Further, any value of src0 alpha between 0 and 1 monotonically increases the number of enabled pixels. The same coverage needs to be applied to all the RTs in MRT case. If AlphaToCoverage is disabled, AlphaToCoverage Dither does not have any impact.

BLEND_STATE

28 **Reserved**

Format:	MBZ

27 **Write Disable Alpha**

Format:	Disable

This field controls the writing of the alpha component into the Render Target.

Value	Name	Description
0b	Enabled	Alpha component can be overwritten
1b	Disabled	Writes to the color buffer will not modify Alpha.

Programming Notes
For YUV surfaces, this field must be set to 0B (enabled).

26 **Write Disable Red**

Format:	Disable

This field controls the writing of the red component into the Render Target.

Value	Name	Description
0b	Enabled	Red component can be overwritten
1b	Disabled	Writes to the color buffer will not modify Red.

Programming Notes
For YUV surfaces, this field must be set to 0B (enabled).

25 **Write Disable Green**

Format:	Disable

This field controls the writing of the green component into the Render Target.

Value	Name	Description
0b	Enabled	Green component can be overwritten
1b	Disabled	Writes to the color buffer will not modify Green.

Programming Notes
For YUV surfaces, this field must be set to 0B (enabled).

24 **Write Disable Blue**

Format:	Disable

This field controls the writing of the Blue component into the Render Target.

Value	Name	Description
0b	Enabled	Blue component can be overwritten
1b	Disabled	Writes to the color buffer will not modify Blue.

Programming Notes

BLEND_STATE

		For YUV surfaces, this field must be set to 0B (enabled).
	23	**Reserved** Format: MBZ
	22	**Logic Op Enable** Format: Enable Enables the LogicOp function of the Pixel Processing pipeline. **Programming Notes** Enabling LogicOp and Color Buffer Blending at the same time is UNDEFINED
	21:18	**Logic Op Function** Format: 3D_LogicOpFunction This field specifies the function to be performed (when enabled) in the Logic Op stage of the Pixel Processing pipeline. Note that the encoding of this field is one less than the corresponding "R2_" ROP code defined in WINGDI. H, and is a rather contorted mapping of the OpenGL LogicOp encodings. However, this field was defined such that, when the 4 bits are replicated to 8 bits, they coincide with the ROP codes used in the Blter. Note: if the Logic Op Function does not depend on "D", the dest buffer is not read.
	17	**Reserved** Format: MBZ
	16	**Alpha Test Enable** Format: Enable

Value	Name	Description
0h	LOGICOP_CLEAR	BLACK; all 0's
1h	LOGICOP_NOR	NOTMERGEPEN; NOT (S OR D)
2h	LOGICOP_AND_INVERTED	MASKNOTPEN; (NOT S) AND D
3h	LOGICOP_COPY_INVERTED	NOTCOPYPEN; NOT S
4h	LOGICOP_AND_REVERSE	MASKPENNOT; S AND NOT D
5h	LOGICOP_INVERT	NOT; NOT D
6h	LOGICOP_XOR	XORPEN; S XOR D
7h	LOGICOP_NAND	NOTMASKPEN; NOT (S AND D)
8h	LOGICOP_AND	MASKPEN; S AND D
9h	LOGICOP_EQUIV	NOTXORPEN; NOT (S XOR D)
Ah	LOGICOP_NOOP	NOP; D
Bh	LOGICOP_OR_INVERTED	MERGENOTPEN; (NOT S) OR D
Ch	LOGICOP_COPY	COPYPEN; S
Dh	LOGICOP_OR_REVERSE	MERGEPENNOT; S OR NOT D
Eh	LOGICOP_OR	MERGEPEN; S OR D
Fh	LOGICOP_SET	WHITE; all 1's

BLEND_STATE

Enables the AlphaTest function of the Pixel Processing pipeline.

Programming Notes
Alpha Test can only be enabled if Pixel Shader outputs a float alpha value. Alpha Test is applied independently on each render target by comparing that render target's alpha value against the alpha reference value. If the alpha test fails, the corresponding pixel write will be supressed only for that render target. The depth/stencil update will occur if alpha test passes for any render target.
When Alpha Test is disabled, Alpha Test Function must be COMPAREFUNCTION_ALWAYS.

15:13 **Alpha Test Function**

Format:	3D_CompareFunction

This field specifies the comparison function used in the AlphaTest function

Value	Name	Description
0h	COMPAREFUNCTION_ALWAYS	Always pass
1h	COMPAREFUNCTION_NEVER	Never pass
2h	COMPAREFUNCTION_LESS	Pass if the value is less than the reference
3h	COMPAREFUNCTION_EQUAL	Pass if the value is equal to the reference
4h	COMPAREFUNCTION_LEQUAL	Pass if the value is less than or equal to the reference
5h	COMPAREFUNCTION_GREATER	Pass if the value is greater than the reference
6h	COMPAREFUNCTION_NOTEQUAL	Pass if the value is not equal to the reference
7h	COMPAREFUNCTION_GEQUAL	Pass if the value is greater than or equal to the reference

12 **Color Dither Enable**

Format:	Enable

Enables dithering of colors (including any alpha component) before they are written to the Color Buffer.

11:10 **X Dither Offset**

Format:	U2

Specifies offset to apply to pixel X coordinate LSBs when accessing dither table.

9:8 **Y Dither Offset**

Format:	U2

Specifies offset to apply to pixel Y coordinate LSBs when accessing dither table.

7:4 **Reserved**

Format:	MBZ

BLEND_STATE

3:2 **Color Clamp Range**

Specifies the clamped range used in Pre-Blend and Post-Blend Color Clamp functions if one or both of those functions are enabled. Note that this range selection is shared between those functions. This field is ignored if both of the Color Clamp Enables are disabled

Value	Name	Description
0	COLORCLAMP_UNORM	Clamp Range [0,1]
1	COLORCLAMP_SNORM	Clamp Range [-1,1]
2	COLORCLAMP_RTFORMAT	Clamp to the range of the RT surface format (Note: The Alpha component is clamped to FLOAT16 for R11G11B10_FLOAT format).
3	Reserved	Reserved

1 **Pre-Blend Color Clamp Enable**

Format:	Enable

This field specifies whether the source, destination and constant color channels are clamped prior to blending, regardless of whether blending is enabled. If DISABLED, no clamping is performed prior to blending. If ENABLED, all inputs to the blend function are clamped prior to the blend to the range specified by Color Clamp Range.

Value	Name	Description
0	Disabled	No clamping is performed prior to blending.
1	Enabled	All inputs to the blend function are clamped prior to the blend to the range specified by Color Clamp Range.

Programming Notes
See table in Pre-Blending Color Clamp subsection for programming restrictions as a function of RT format. This field is ignored (treated as DISABLED) for UINT and SINT RT surface formats. Blending is not supported for those RT surface formats. The device will automatically clamp source color channels to the respective RT surface range.

0 **Post-Blend Color Clamp Enable**

Format:	Enable

If blending is enabled, this field specifies whether the blending output channels are first clamped to the range specified by Color Clamp Range. Regardless of whether this clamping is enabled, the blending output channels will be clamped to the RT surface format just prior to being written.

Programming Notes
See table in Pre-Blending Color Clamp subsection for programming restrictions as a function of RT format. This field is ignored (treated as DISABLED) for UINT and SINT RT surface formats. Blending is not supported for those RT surface formats. The device will automatically clamp source color channels to the respective RT surface range.

BR00 - BLT Opcode and Control

Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31	**BLT Engine Busy** This bit indicates whether the BLT Engine is busy (1) or idle (0). This bit is replicated in the SETUP BLT Opcode and Control register. Value: 0 — Idle **[Default]**; 1 — Busy
	30	**Setup Instruction Instruction** Default Value: 0 The current instruction performs clipping (1).
	29	**Setup Monochrome Pattern** This bit is decoded from the Setup instruction opcode to identify whether a color (0) or monochrome (1) pattern is used with the SCANLINE_BLT instruction. Value: 0 — Color **[Default]**; 1 — Monochrome
	28:22	**Instruction Target (Opcode)** Default Value: 0000000b This is the contents of the Instruction Target field from the last BLT instruction. This field is used by the BLT Engine state machine to identify the BLT instruction it is to perform. The opcode specifies whether the source and pattern operands are color or monochrome.
	21:20	**32bpp Byte Mask** This field is only used for 32bpp. Value: 00b — **[Default]**; 1xb — Write Alpha Channel; x1b — Write RGB Channel
	19:17	**Monochrome Source Start** Default Value: 000b This field indicates the starting monochrome pixel bit position within a byte per scan line of the source operand. The monochrome source is word aligned which means that at the end of the scan line all bits should be discarded until the next word boundary.
	16	**Bit/Byte Packed** Byte packed is for the NT driver.

BR00 - BLT Opcode and Control

Value	Name
0b	Bit **[Default]**
1b	Byte

15 Src Tiling Enable

Value	Name
0b	Tiling Disabled (Linear) **[Default]**
1b	Tiling enabled: Tile-X or Tile-Y

14:12 Horizontal Pattern Seed

Default Value:	0b

This field indicates the pattern pixel position which corresponds to X = 0.

11 Dest Tiling Enable

When set to '1', this means that Blitter is executing in Tiled mode. If '0' it means that Blitter is in Linear mode. The Blitter supports both Tile-X and Tile-Y modes. On reset, this bit will be '0'. This definition applies to only X,Y Blits.

Value	Name
0b	Tiling Disabled (Linear blit) **[Default]**
1b	Tiling enabled: Tile-X or Tile-Y

10:8 Transparency Range Mode

These bits control whether or not the byte(s) at the destination corresponding to a given pixel will be conditionally written, and what those conditions are. This feature can make it possible to perform various masking functions in order to selectively write or preserve graphics data already at the destination.

Value	Name	Description
xx0b	**[Default]**	No color transparency mode enabled. This causes normal operation with regard to writing data to the destination.
001b		[Source color transparency] The Transparency Color Low: (Pixel Greater or Equal) (source background register) and the Transparency Color High: (Pixel Less or Equal) (source foreground register) are compared to the source pixels. The range comparisons are done on each component (R,G,B) and then logically ANDed. If the source pixel components are not within the range defined by the Transparency Color registers, then the byte(s) at the destination corresponding to the current pixel are written with the result of the bit-wise operation.
011b		[Source and Alpha color transparency] The Transparency Color Low: (Pixel Greater or Equal) (source background register) and the Transparency Color High: (Pixel Less or Equal) (source foreground register) are compared to the source pixels. The range comparisons are done on each component (A,R,G,B) and then logically ANDed. If the source pixel components are not within the range defined by the Transparency Color registers, then the byte(s) at the destination corresponding to the current pixel are written with the result of

BR00 - BLT Opcode and Control

				the bit-wise operation."
		101b		[Destination and Alpha color transparency] The Transparency Color Low: (Pixel Greater or Equal) (source background register) and the Transparency Color High: (Pixel Less or Equal) (source foreground register) are compared to the destination pixels. The range comparisons are done on each component (A,R,G,B) and then logically ANDed. If the destination pixels are within the range, then the byte(s) at the destination corresponding to the current pixel are written with the result of the bit-wise operation.
		111b		[Destination color transparency] The Transparency Color Low: (Pixel Greater or Equal) (source background register) and the Transparency Color High: (Pixel Less or Equal) (source foreground register) are compared to the destination pixels. The range comparisons are done on each component (R,G,B) and then logically ANDed. If the destination pixels are within the range, then the byte(s) at the destination corresponding to the current pixel are written with the result of the bit-wise operation.

7:5	**Pattern Vertical Seed** Default Value: 000b This field specifies the pattern scan line which corresponds to Y=0.
4	**Destination Read Modify Write** Default Value: 0b This bit is decoded from the last instruction's opcode field and Destination Transparency Mode to identify whether a Destination read is needed.
3	**Color Source** Default Value: 0b This bit is decoded from the last instructions opcode field to identify whether a color (1) source is used.
2	**Monochrome Source** Default Value: 0b This bit is decoded from the last instructions opcode field to identify whether a monochrome (1) source is used.
1	**Color Pattern** Default Value: 0b This bit is decoded from the last instructions opcode field to identify whether a color (1) pattern is used.

BR00 - BLT Opcode and Control

	0	**Monochrome Pattern** Default Value: 0b This bit is decoded from the last instructions opcode field to identify whether a monochrome (1) pattern is used.

BR01 - Setup BLT Raster OP, Control, and Destination Offset

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31	**Solid Pattern Select** This bit applies only when the pattern data is monochrome. This bit determines whether or not the BLT Engine actually performs read operations from the frame buffer in order to load the pattern data. Use of this feature to prevent these read operations can increase BLT Engine performance, if use of the pattern data is indeed not necessary. The BLT Engine is configured to accept either monochrome or color pattern data via the opcode field.

Value	Name	Description
0b	**[Default]**	This causes normal operation with regard to the use of the pattern data. The BLT Engine proceeds with the process of reading the pattern data, and the pattern data is used as the pattern operand for all bit-wise operations.
1b		The BLT Engine forgoes the process of reading the pattern data, the presumption is made that all of the bits of the pattern data are set to 0, and the pattern operand for all bit-wise operations is forced to the background color specified in the Color Expansion Background Color Register.

DWord	Bit	Description
	30	**Clipping Enabled**

Value	Name
0b	**[Default]**
1b	

DWord	Bit	Description
	29	**Monochrome Source Transparency Mode** This bit applies only when the source data is in monochrome. This bit determines whether or not the byte(s) at the destination corresponding to the pixel to which a given bit of the source data also corresponds will actually be written if that source data bit has the value of 0. This feature can make it possible to use the source as a transparency mask. The BLT Engine is configured to accepted either monochrome or color source data via the opcode field.

Value	Name	Description
0b	**[Default]**	This causes normal operation with regard to the use of the source data. Wherever a bit in the source data has the value of 0, the color specified in the background color register is used as the source operand in

BR00 - BLT Opcode and Control

		the bit-wise operation for the pixel corresponding to the source data bit, and the bytes at the destination corresponding to that pixel are written with the result.
1b		Wherever a bit in the source data has the value of 0, the byte(s) at the destination corresponding to the pixel to which the source data bit also corresponds are simply not written, and the data at those byte(s) at the destination are allowed to remain unchanged.

28 **Monochrome Pattern Transparency Mode**

This bit applies only when the pattern data is monochrome. This bit determines whether or not the byte(s) at the destination corresponding to the pixel to which a given bit of the pattern data also corresponds will actually be written if that pattern data bit has the value of 1. This feature can make it possible to use the pattern as a transparency mask. The BLT Engine is configured to accepted either monochrome or color pattern data via the opcode field.

Value	Name	Description
0b	**[Default]**	This causes normal operation with regard to the use of the pattern data. Wherever a bit in the pattern data has the value of 0, the color specified in the background color register is used as the pattern operand in the bit-wise operation for the pixel corresponding to the pattern data bit, and the bytes at the destination corresponding to that pixel are written with the result.
1b		Wherever a bit in the pattern data has the value of 0, the byte(s) at the destination corresponding to the pixel to which the pattern data bit also corresponds are simply not written, and the data at those byte(s) at the destination are allowed to remain unchanged.

27:26 **32bpp Byte Mask**

This bit applies only when the pattern data is monochrome. This bit determines whether or not the byte(s) at the destination corresponding to the pixel to which a given bit of the pattern data also corresponds will actually be written if that pattern data bit has the value of 1. This feature can make it possible to use the pattern as a transparency mask. The BLT Engine is configured to accepted either monochrome or color pattern data via the opcode field.

Value	Name
00b	**[Default]**
1xb	Write Alpha Channel

BR00 - BLT Opcode and Control

	x1b	Write RGB Channel

25:24 Color Depth

Value	Name
00b	8 Bit Color Depth **[Default]**
01b	16 Bit Color Depth
10b	16 Bit Color Depth
11b	32 Bit Color Depth

23:16 Raster Operation Select

These 8 bits are used to select which one of 256 possible raster operations is to be performed by the BLT Engine.

15:0 Destination Pitch (Offset)

For non-XY Blits, the signed 16bit field allows for specifying upto + 32Kbytes signed pitches in bytes (same as before). For X, Y Blits with tiled-X surfaces, the pitch for Destination will be 512Byte aligned and should be programmable upto + 128Kbytes. For X, Y Blits with tiled-Y surfaces, the pitch for Destination will be 128Byte aligned and should be programmable upto + 128Kbytes. In this case, this 16bit signed pitch field is used to specify upto + 32KDWords. For X, Y blits with nontiled surfaces (linear surfaces), this 16bit field can be programmed to byte specification of upto + 32Kbytes (same as before). These 16 bits store the signed memory address offset value by which the destination address originally specified in the Destination Address Register is incremented or decremented as each scan line's worth of destination data is written into the frame buffer by the BLT Engine, so that the destination address will point to the next memory address to
which the next scan line's worth of destination data is to be written. If the intended destination of a BLT operation is within on-screen frame buffer memory, this offset is normally set so that each subsequent scan line's worth of destination data lines up vertically with the destination data in the scan line, above. However, if the intended destination of a BLT operation is within off-screen memory, this offset can be set so that each subsequent scan line's worth of destination data is stored at a location immediately after the location where the destination data for the last scan line ended, in order to create a single contiguous block of bytes of destination data at the destination.

BR05 - Setup Expansion Background Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Setup Expansion Background Color Bits** These bits provide the one, two, or four bytes worth of color data that select the background color to be used in the color expansion of monochrome pattern or source data for either the SCANLINE_BLT or TEXT_BLT instructions. BR05 is also used as the solid pattern for the PIXEL_BLT instruction. Whether one, two, or three bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

BR06 - Setup Expansion Foreground Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Setup Expansion Foreground Color Bits** These bits provide the one, two, or four bytes worth of color data that select the foreground color to be used in the color expansion of monochrome pattern or source data for either the SCANLINE_BLT or TEXT_BLT instructions. Whether one, two, or three bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

BR07 - Setup Blit Color Pattern Address

Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:29	**Reserved** Format: MBZ
	28:6	**Setup Blit Color Pattern Address** Format: GraphicsAddress[28:6] These 26 bits specify the starting address of the (8X8) pixel color pattern from the SETUP_BLT instruction. This register works identically to the Pattern Address register (BR15), but this version is only used with the SCANLINE_BLT instruction execution (the actual programming for this, is done in XY_SETUP_BLT command). The pattern data must be located in linear memory. The pattern data must be located on a pattern-size boundary. The pattern is always of 8x8 pixels, and therefore, its size is dependent upon its pixel depth. The pixel depth may be 8, 16, or 32 bits per pixel if the pattern is in color (the pixel depth of a color pattern must match the pixel depth to which the graphics system has been set). Monochrome patterns require 8 bytes and is supplied through the instruction. Color patterns of 8, 16, and 32 bits per pixel color depth must start on 64-byte, 128-byte and 256-byte boundaries, respectively.
	5:0	**Reserved** Format: MBZ

BR09 - Destination Address

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Destination Address Bits** Format: GraphicsAddress[31:0] When tiling is enabled for XY-blits, this base address should be limited to 4KB. Otherwise for XY blits, there is no restriction and it is same as before. These specify the starting pixel address of the destination data. This register is also the working destination address register and changes as the BLT Engine performs the accesses. Used as the scan line address (Destination Y Address and Destination Y1 Address) for BLT instructions: PIXEL_BLT, SCANLINE_BLT, and TEXT_BLT. In this case the address points to the first pixel in a scan line and is compared with the ClipRect Y1 and Y2 address registers to determine whether the scan line should be written or not. The Destination Y1 address is the top scan line to be written for text. Note that for non-XY blits (COLOR_BLT, SRC_COPY_BLT), this address points to the first byte to be written. Note: Some instructions affect only one scan line (requiring only one coordinate); other instructions affect multiple scan lines and need both coordinates.

BR11 - BLT Source Pitch (Offset)

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:16	**Reserved**
	15:0	**Source Pitch (Offset)** For non-XY Blits with color source operand (SRC_COPY_BLT), the signed 16bit field allows for specifying upto + 32Kbytes signed pitch in bytes (same as before). For X, Y Blits with tiled-X surfaces, the pitch for Color Source will be 512Byte aligned and should be programmable upto + 128Kbytes. For X, Y Blits with tiled-Y surfaces, the pitch for Color Source will be 128Byte aligned and should be programmable upto + 128Kbytes. In this case, this 16bit signed pitch field is used to specify upto + 32KDWords. For X, Y blits with nontiled color source surfaces (linear surfaces), this 16bit field can be programmed to byte specification of upto + 32Kbytes (same as before). When the color source data is located within the frame buffer or AGP aperture, these signed 16 bits store the memory address offset (pitch) value by which the source address originally specified in the Source Address Register is incremented or decremented as each scan line's worth of source data is read from the frame buffer by the BLT Engine, so that the source address will point to the next memory address from which the next scan line's worth of source data is to be read. Note that if the intended source of a BLT operation is within on-screen frame buffer memory, this offset is normally set to accommodate the fact that each subsequent scan line's worth of source data lines up vertically with the source data in the scan line, above. However, if the intended source of a BLT operation is within off-screen memory, this offset can be set to accommodate a situation in which the source data exists as a single contiguous block of bytes where in each subsequent scan line's worth of source data is stored at a location immediately after the location where the source data for the last scan line ended.

BR12 - Source Address

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Source Address Bits** Format: GraphicsAddress[31:0] When tiling is enabled for XY-blits with Color source surfaces, this base address should be limited to 4KB. Otherwise for XY blits, there is no restriction and it is same as before, including for monosource and text blits. Note that for non-XY blit with Color Source (SRC_COPY_BLT), this address points to the first byte to be read. These specify the starting pixel address of the color source data. The lower 3 bits are used to indicate the position of the first valid byte within the first Quadword of the source data.

BR13 - BLT Raster OP, Control, and Destination Pitch

Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31	**Solid Pattern Select** This bit applies only when the pattern data is monochrome. This bit determines whether or not the BLT Engine actually performs read operations from the frame buffer in order to load the pattern data. Use of this feature to prevent these read operations can increase BLT Engine performance, if use of the pattern data is indeed not necessary. The BLT Engine is configured to accept either monochrome or color pattern data via the opcode field. **Value** / **Name** / **Description** 0 / **[Default]** / This causes normal operation with regard to the use of the pattern data. The BLT Engine proceeds with the process of reading the pattern data, and the pattern data is used as the pattern operand for all bit-wise operations. 1 / / The BLT Engine forgoes the process of reading the pattern data, the presumption is made that all of the bits of the pattern data are set to 0, and the pattern operand for all bit-wise operations is forced to the background color specified in the Color Expansion Background Color Register.
	30	**Clipping Enabled** Default Value: 0
	29	**Monochrome Source Transparency Mode** This bit applies only when the source data is in monochrome. This bit determines whether or not the byte(s) at the destination corresponding to the pixel to which a given bit of the source data also corresponds will actually be written if that source data bit has the value of 0. This feature can make it possible to use the source as a transparency mask. The BLT Engine is configured to accepted either monochrome or color source data via the opcode field. **Value** / **Name** / **Description** 0 / **[Default]** / This causes normal operation with regard to the use of the source data. Wherever a bit in the source data has the value of 0, the color specified in the background color register is used as the source operand in the bit-wise operation for the pixel corresponding to the source data bit, and the bytes at the destination corresponding to that pixel are written with the result. 1 / / Where a bit in the source data has the value of 0, the byte(s) at the destination corresponding to the pixel to which the source data bit also corresponds are simply not written, and the data at those byte(s) at the destination are allowed to remain unchanged.
	28	**Monochrome Pattern Transparency Mode** This bit applies only when the pattern data is monochrome. This bit determines whether or not the byte(s) at the destination corresponding to the pixel to which a given bit of the pattern data also corresponds will actually be written if that pattern data bit has the value of 1. This feature can make it possible to use the pattern as a transparency mask. The BLT Engine is configured to accepted either monochrome or color pattern data via the opcode in the Opcode and Control

BR13 - BLT Raster OP, Control, and Destination Pitch

register.

Value	Name	Description
0	**[Default]**	This causes normal operation with regard to the use of the pattern data. Where a bit in the pattern data has the value of 0, the color specified in the background color register is used as the pattern operand in the bit-wise operation for the pixel corresponding to the pattern data bit, and the bytes at the destination corresponding to that pixel are written with the result.
1		Wherever a bit in the pattern data has the value of 0, the byte(s) at the destination corresponding to the pixel to which the pattern data bit also corresponds are simply not written, and the data at those byte(s) at the destination are allowed to remain unchanged.

27:26 **32bpp Byte Mask**

This field is only used for 32bpp.

Value	Name
00b	**[Default]**
1xb	Write Alpha Channel
x1b	Write RGB Channel

25:24 **Color Depth**

Value	Name
00b	8 Bit Color Depth **[Default]**
01b	16 Bit Color Depth
10b	24 Bit Color Depth
11b	Reserved

23:16 **Raster Operation Select**

Default Value:	00000000b

These 8 bits are used to select which one of 256 possible raster operations is to be performed by the BLT Engine.

15:0 **Destination Pitch(Offset)**

These 16 bits store the signed memory address offset value by which the destination address originally specified in the Destination Address Register is incremented or decremented as each scan line's worth of destination data is written into the frame buffer by the BLT Engine, so that the destination address will point to the next memory address to which the next scan line's worth of destination data is to be written. If the intended destination of a BLT operation is within on-screen frame buffer memory, this offset is normally set so that each subsequent scan line's worth of destination data lines up vertically with the destination data in the scan line, above. However, if the intended destination of a BLT operation is within off-screen memory, this offset can be set so that each subsequent scan line's worth of destination data is stored at a location immediately after the location where the destination data for the last scan line ended, in order to create a single contiguous block of bytes of destination data at the destination.

BR14 - Destination Width and Height

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

BR14 contains the values for the height and width of the data to be BLT. If these values are not correct, such that the BLT Engine is either expecting data it does not receive or receives data it did not expect, the system can hang.

DWord	Bit	Description
0	31:29	**Reserved**
	28:16	**Destination Height** These 13 bits specify the height of the destination data in terms of the number of scan lines. This is a working register.
	15:13	**Reserved**
	12:0	**Destination Byte Width** These 13 bits specify the width of the destination data in terms of the number of bytes per scan line. The number of pixels per scan line into which this value translates depends upon the color depth to which the graphics system has been set.

BR15 - Color Pattern Address

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:29	**Reserved** Format: MBZ
	28:6	**Color Pattern Address** Format: GraphicsAddress[28:6] There is no change to the Color Pattern address specification due to Non-Power-of-2 change. It remains the same as before. The pattern data must be located in linear memory. These 26 bits specify the starting address of the (8X8) pixel color pattern. The pattern data must be located on a pattern-size boundary. The pattern is always of 8x8 pixels, and therefore, its size is dependent upon its pixel depth. The pixel depth may be 8, 16, or 32 bits per pixel if the pattern is in color (the pixel depth of a color pattern must match the pixel depth to which the graphics system has been set). Monochrome patterns require 8 bytes and are applied through the instruction. Color patterns of 8, 16, and 32 bits per pixel color depth must start on 64-byte, 128-byte and 256-byte boundaries, respectively.
	5:0	**Reserved** Format: MBZ

BR16 - Pattern Expansion Background and Solid Pattern Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Pattern Expansion Background Color Bits** These bits provide the one, two, or four bytes worth of color data that select the background color to be used in the color expansion of monochrome pattern data during BLT operations. Whether one, two, or four bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

BR17 - Pattern Expansion Foreground Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Pattern Expansion Background Color Bits** These bits provide the one, two, or four bytes worth of color data that select the foreground color to be used in the color expansion of monochrome pattern data during BLT operations. Whether one, two, or four bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

BR18 - Source Expansion Background and Destination Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Source Expansion Background Color Bits** These bits provide the one, two, or four bytes worth of color data that select the background color to be used in the color expansion of monochrome source data during BLT operations. This register is also used to support destination transparency mode and Solid color fill. Whether one, two, three, or four bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

BR19 - Source Expansion Foreground Color

Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:0	**Pattern/Source Expansion Foreground Color Bits** These bits provide the one, two, or four bytes worth of color data that select the foreground color to be used in the color expansion of monochrome source data during BLT operations. Whether one, two, or four bytes worth of color data is needed depends upon the color depth to which the BLT Engine has been set. For a color depth of 32bpp, 16bpp and 8bpp, bits [31:0], [15:0] and [7:0], respectively, are used.

CC_VIEWPORT

Source:	BSpec
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

The viewport state is stored as an array of up to 16 elements, each of which contains the DWords described here. The start of each element is spaced 2 DWords apart. The first element of the viewport state array is aligned to a 32-byte boundary. The Minimum Depth field in CC_Viewport state must be be greater than or equal to zero on D16_UNORM, D24_UNORM_X8_UINT, or D24_UNORM_S8_UINT depth formats

DWord	Bit	Description
0	31:0	**Minimum Depth** Format: IEEE_Float Indicates the minimum depth. The interpolated or computed depth is clamped to this value prior to the depth test.
1	31:0	**Maximum Depth** Format: IEEE_Float Indicates the maximum depth. The interpolated or computed depth is clamped to this value prior to the depth test.

COLOR_CALC_STATE

Source:	BSpec
Size (in bits):	192
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

It is pointed to by a field in 3DSTATE_CC_STATE_POINTERS, and stored at a 64-byte aligned boundary.

DWord	Bit	Description
0	31:24	**Stencil Reference Value** Format: U8.0 This field specifies the stencil reference value to compare against in the (front face) StencilTest function.
	23:16	**BackFace Stencil Reference Value** Format: U8.0 This field specifies the stencil reference value to compare against in the StencilTest function.
	15	**Round Disable Function Disable** Disables the round-disable function of the color calculator. Value / Name / Description: 0 / Cancelled / Dithering is cancelled based on the data used by blend to avoid drift. 1 / Not Cancelled / Dithering is NOT cancelled.
	14:1	**Reserved** Format: MBZ
	0	**Alpha Test Format** This field selects the format for Alpha Reference Value and the format in which Alpha Test is performed. Value / Name / Description: 0h / ALPHATEST_UNORM8 / UNorm8 1h / ALPHATEST_FLOAT32 / Float32 **Programming Notes** Alpha-test format is independent of RT format. When PS outputs UNIT/SINT alpha-value, it will be treated as IEEE 32bit float number for the purpose of alpha-test.
1	31:0	**Alpha Reference Value** Exists If: [Alpha Test Format] == 'ALPHATEST_FLOAT32' Format: IEEE_Float This field specifies the alpha reference value to compare against in the Alpha Test function.
	31:0	**Alpha Reference Value** Exists If: [Alpha Test Format] == 'ALPHATEST_UNORM8' Format: UNORM8 Upper 24 bits MBZ

COLOR_CALC_STATE

		This field specifies the alpha reference value to compare against in the Alpha Test function.
2	31:0	**Blend Constant Color Red** Format: IEEE_Float This field specifies the Red channel of the Constant Color used in Color Buffer Blending.
3	31:0	**Blend Constant Color Green** Format: IEEE_Float This field specifies the Green channel of the Constant Color used in Color Buffer Blending.
4	31:0	**Blend Constant Color Blue** Format: IEEE_Float This field specifies the Blue channel of the Constant Color used in Color Buffer Blending.
5	31:0	**Blend Constant Color Alpha** Format: IEEE_Float This field specifies the Alpha channel of the Constant Color used in Color Buffer Blending.

COLOR_PROCESSING_STATE - ACE State

Source: BSpec

Size (in bits): 416

Default Value: 0x00000068, 0x4C382410, 0x9C887460, 0xEBD8C4B0, 0x604C3824, 0xB09C8874, 0x0000D8C4, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

This state structure contains the ACE state used by the color processing function.
It corresponds to DW29..DW41 of the Color Processing State.

DWord	Bit	Description
0	31:7	**Reserved** Format: MBZ
	6:2	**Skin Threshold** Format: U5 Used for Y analysis (min/max) for pixels which are higher than skin threshold. Value / Name 1-31 26 **[Default]**
	1	**Full Image Histogram** Default Value: 0 Format: Enable Used to ignore the area of interest for full image histogram.
	0	**ACE Enable** Format: Enable
1	31:24	**Y3** Default Value: 76 Format: U8 The value of the y_pixel for point 3 in PWL.
	23:16	**Y2** Default Value: 56 Format: U8 The value of the y_pixel for point 2 in PWL.
	15:8	**Y1** Default Value: 36 Format: U8 The value of the y_pixel for point 1 in PWL.
	7:0	**Ymin**

COLOR_PROCESSING_STATE - ACE State

		Default Value: 16 Format: U8 The value of the y_pixel for point 0 in PWL.
2	31:24	**Y7** Default Value: 156 Format: U8 The value of the y_pixel for point 7 in PWL.
	23:16	**Y6** Default Value: 136 Format: U8 The value of the y_pixel for point 6 in PWL.
	15:8	**Y5** Default Value: 116 Format: U8 The value of the y_pixel for point 5 in PWL.
	7:0	**Y4** Default Value: 96 Format: U8 The value of the y_pixel for point 4 in PWL.
3	31:24	**Ymax** Default Value: 235 Format: U8 The value of the y_pixel for point 11 in PWL.
	23:16	**Y10** Default Value: 216 Format: U8 The value of the y_pixel for point 10 in PWL.
	15:8	**Y9** Default Value: 196 Format: U8 The value of the y_pixel for point 9 in PWL.

COLOR_PROCESSING_STATE - ACE State

	7:0	**Y8** Default Value: 176 Format: U8 The value of the y_pixel for point 8 in PWL.
4	31:24	**B4** Default Value: 96 Format: U8 The value of the bias for point 4 in PWL.
	23:16	**B3** Default Value: 76 Format: U8 The value of the bias for point 3 in PWL.
	15:8	**B2** Default Value: 56 Format: U8 The value of the bias for point 2 in PWL.
	7:0	**B1** Default Value: 36 Format: U8 The value of the bias for point 1 in PWL.
5	31:24	**B8** Default Value: 176 Format: U8 The value of the bias for point 8 in PWL.
	23:16	**B7** Default Value: 156 Format: U8 The value of the bias for point 7 in PWL.
	15:8	**B6** Default Value: 136 Format: U8 The value of the bias for point 6 in PWL.

COLOR_PROCESSING_STATE - ACE State

	7:0	**B5** Default Value: 116 Format: U8 The value of the bias for point 5 in PWL.
6	31:16	**Reserved** Format: MBZ
	15:8	**B10** Default Value: 216 Format: U8 The value of the bias for point 10 in PWL.
	7:0	**B9** Default Value: 196 Format: U8 The value of the bias for point 9 in PWL.
7	31:27	**Reserved** Format: MBZ
	26:16	**S1** Format: U1.10 The value of the slope for point 1 in PWL. The default is 1024/1024.
	15:11	**Reserved** Format: MBZ
	10:0	**S0** Format: U1.10 The value of the slope for point 0 in PWL. The default is 1024/1024.
8	31:27	**Reserved** Format: MBZ
	26:16	**S3** Format: U1.10 The value of the slope for point 3 in PWL. The default is 1024/1024.
	15:11	**Reserved**

COLOR_PROCESSING_STATE - ACE State

		Format: MBZ
	10:0	**S2** Format: U1.10 The value of the slope for point 2 in PWL. The default is 1024/1024.
9	31:27	**Reserved** Format: MBZ
	26:16	**S5** Format: U1.10 The value of the slope for point 5 in PWL. The default is 1024/1024.
	15:11	**Reserved** Format: MBZ
	10:0	**S4** Format: U1.10 The value of the slope for point 4 in PWL. The default is 1024/1024.
10	31:27	**Reserved** Format: MBZ
	26:16	**S7** Format: U1.10 The value of the slope for point 7 in PWL. The default is 1024/1024.
	15:11	**Reserved** Format: MBZ
	10:0	**S6** Format: U1.10 The value of the slope for point 6 in PWL. The default is 1024/1024.
11	31:27	**Reserved** Format: MBZ
	26:16	**S9** Format: U1.10 The value of the slope for point 9 in PWL.

COLOR_PROCESSING_STATE - ACE State		
		The default is 1024/1024.
	15:11	**Reserved** Format: MBZ
	10:0	**S8** Format: U1.10 The value of the slope for point 8 in PWL. The default is 1024/1024.
12	31:11	**Reserved** Format: MBZ
	10:0	**S10** Format: U1.10 The value of the slope for point 10 in PWL. The default is 1024/1024.

COLOR_PROCESSING_STATE - CGC State

Source: BSpec
Size (in bits): 96
Default Value: 0x0CD2911F, 0x30000334, 0x8A800000

This state structure contains the CGC state used by the color processing function.
It corresponds to DW64..DW66 of the Color Processing State.

DWord	Bit	Description
0	31	**Color Gamut Compression Enable**
	30	**Full Range Mapping Enable** Value: 0 — Basic Mode **[Default]** Value: 1 — Advanced Mode
	29:20	**d(in,default)** Default Value: 205 Format: U10 $d_{in.default}$**InnerTriangleMappingLength**
	19:10	**d(out,default)** Default Value: 164 Format: U10 $d_{out.default}$**OuterTriangleMappingLength**
	9:0	**d1(out)** Default Value: 287 Format: U10 d^{1}_{out}**OuterTriangleMappingLengthBelow**
1	31	**Reserved** Format: MBZ
	30:28	**Compression Line Shift** Value: 0-4 Value: 3 **[Default]**
	27:10	**Reserved** Format: MBZ
	9:0	**d1(in)** Default Value: 820 Format: U10 d^{1}_{in}**InnerTriangleMappingLengthBelow**

COLOR_PROCESSING_STATE - CGC State

2	31	**xvYcc Decode Encode Enable** **Value** / **Name** / **Description** 1 / **[Default]** / Both xvYcc decode and xvYcc encode are enabled 0 / / Disable both xvYcc decode and xvYcc encode **Programming Notes** This bit is valid only when **ColorGamutCompressionnEnable** is on.
	30	**Forced 444 for 444** Default Value: 0 Force the 4:4:4 operation when input video of 4:4:4 format
	29	**Forced 422 for 444** Default Value: 0 Force the 4:2:2 operation when input video of 4:4:4 format
	28	**Forced 444 for 422** Default Value: 0 Force the 4:4:4 operation when input video of 4:2:2 format
	27:26	**STD Factor Mode** **Value** / **Name** / **Description** 00b / STDMin / Select the minimum value of the STD factors 01b / STDMax / Select the maximum value of the STD factors 10b / STDAve **[Default]** / Select the average value of the STD factors 11b / Reserved / **Programming Notes** This field is only valid for input of 4:2:2 (**Forced444_for 422** is disabled), or when (**Forced422_for444** is enabled).
	25:24	**MV Dark Factor Mode** **Value** / **Name** / **Description** 00b / MVDarkMin / Select the minimum value of the MVDark factors 01b / MVDarkMax / Select the maximum value of the MVDark factors 10b / MVDarkAve **[Default]** / Select the average value of the MVDark factors 11b / Reserved / **Programming Notes**

COLOR_PROCESSING_STATE - CGC State

		This field is only valid for input of 4:2:2 (**Forced444_for 422** is disabled), or when (**Forced422_for444** is enabled).
	23:22	**Scaling Factor Mode** This mode is for color gamut compression module **Value** / **Name** / **Description** 00b / SFMin / Select the minimum value of the Scaling Factors 01b / SFMax / Select the maximum value of the Scaling Factors 10b / SFAve **[Default]** / Select the average value of the Scaling Factors 11b / Reserved / **Programming Notes** This field is only valid for input of 4:2:2 (**Forced444_for 422** is disabled), or when (**Forced422_for444** is enabled).
	21:5	**Reserved** Format: Reserved
	4	**Override Saturation Equal Zero** Format: MBZ **Programming Notes** This bit should always be 0.
	3:0	**Display Color Space Mode** **Value** / **Name** 0 / BT709 1 / BT601 2-15 / Reserved

COLOR_PROCESSING_STATE - CSC State

Source: BSpec

Size (in bits): 288

Default Value: 0x00002000, 0x00000000, 0x00000400, 0x00000000, 0x000004B4, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

This state structure contains the CSC state used by the color processing function.
It corresponds to DW55..DW63 of the Color Processing State.

DWord	Bit	Description
0	31:29	**Reserved** Format: MBZ
	28:16	**C1** Default Value: 0 Format: S2.10 2's complement Transform coefficient
	15:3	**C0** Default Value: 1024 Format: S2.10 2's complement Transform coefficient
	2	**YUV_IN** Default Value: 0 Format: YUV CSC input offset enable.
	1	**YUV_OUT** Default Value: 0 Format: RGB CSC output offset enable.
	0	**Transform Enable** Format: Enable
1	31:26	**Reserved** Format: MBZ
	25:13	**C3** Default Value: 0 Format: S2.10 2's complement Transform coefficient.

COLOR_PROCESSING_STATE - CSC State

	12:0	**C2** Default Value: 0 Format: S2.10 2's complement Transform coefficient.
2	31:26	**Reserved** Format: MBZ
	25:13	**C5** Default Value: 0 Format: S2.10 2's complement Transform coefficient.
	12:0	**C4** Default Value: 1024 Format: S2.10 2's complement Transform coefficient.
3	31:26	**Reserved** Format: MBZ
	25:13	**C7** Default Value: 0 Format: S2.10 2's complement Transform coefficient.
	12:0	**C6** Default Value: 0 Format: S2.10 2's complement Transform coefficient.
4	31:13	**Reserved** Format: MBZ
	12:0	**C8** Default Value: 1204 Format: S2.10 2's complement Transform coefficient.
5	31:20	**Reserved** Format: MBZ
	19:10	**Offset out 1**

COLOR_PROCESSING_STATE - CSC State

		Default Value: 0 Format: S9 2's complement Offset Out for Y/R.
	9:0	**Offset In 1** Default Value: 0 Format: S9 2's complement Offset in for Y/R.
6	31:20	**Reserved** Format: MBZ
	19:10	**Offset out 2** Default Value: 0 Format: S9 2's complement Offset out for U/G.
	9:0	**Offset in 2** Default Value: 0 Format: S9 2's complement Offset in for U/G.
7	31:20	**Reserved** Format: MBZ
	19:10	**Offset out 3** Default Value: 0 Format: S9 2's complement Offset out for V/B.
	9:0	**Offset in 3** Default Value: 0 Format: S9 2's complement Offset in for V/B.
8	31:17	**Reserved** Format: MBZ
	16	**Alpha from State Select** Format: U1 Enumerated Type **Value** / **Name** / **Description**

COLOR_PROCESSING_STATE - CSC State				
		0		Alpha is taken from message
		1		Alpha is taken from state
	15:0	**Color Pipe Alpha**		
		Format:		U16

COLOR_PROCESSING_STATE - PROCAMP State

Source: BSpec
Size (in bits): 64
Default Value: 0x00020001, 0x01000000

This state structure contains the PROCAMP state used by the color processing function.
It corresponds to DW53..DW54 of the Color Processing State.

DWord	Bit	Description
0	31:28	**Reserved** Format: MBZ
	27:17	**Contrast** Default Value: 1 Format: U4.7 Contrast magnitude.
	16:13	**Reserved** Format: MBZ
	12:1	**Brightness** Default Value: 0 Format: S7.4 2's complement Brightness magnitude.
	0	**PROCAMP Enable** Default Value: 1 Format: Enable
1	31:16	**Cos_c_s** Default Value: 256 Format: S7.8 2's complement UV multiplication cosine factor.
	15:0	**Sin_c_s** Default Value: 0 Format: S7.8 2's complement UV multiplication sine factor.

COLOR_PROCESSING_STATE - STD/STE State

Source:	BSpec
Size (in bits):	928
Default Value:	0x9A6E39F0, 0x400C0000, 0x00001180, 0xFE2F2E00, 0x000000FF, 0x00140000, 0xD82E0000, 0x8285ECEC, 0x00008282, 0x00000000, 0x02117000, 0xA38FEC96, 0x00008CC8, 0x00000000, 0x01478000, 0x0007C300, 0x00000000, 0x00000000, 0x1C180000, 0x00000000, 0x00000000, 0x00000000, 0x0007CF80, 0x00000000, 0x00000000, 0x1C080000, 0x00000000, 0x00000000, 0x00000000

This state structure contains the STD/STE state used by the color processing function.

DWord	Bit	Description
0	31:24	**V_Mid** Default Value: 154 Format: U8 Rectangle middle-point V coordinate
	23:16	**U_Mid** Default Value: 110 Format: U8 Rectangle middle-point U coordinate
	15:10	**Hue Max** Default Value: 14 Format: U6 Rectangle half width
	9:4	**Sat Max** Default Value: 31 Format: U6 Rectangle half length.
	3	**Reserved** Format: MBZ
	2	**Output Control** Value 0: Output Pixels **[Default]** Value 1: Output STD Decisions
	1	**STE Enable** Format: Enable
	0	**STD Enable** Format: Enable

COLOR_PROCESSING_STATE - STD/STE State

1	31	**Reserved** Format: MBZ
	30:28	**Diamond Margin** Default Value: 4 Format: U3
	27:21	**Diamond du** Default Value: 0 Format: S6 2's complement Rhombus center shift in the sat-direction, relative to the rectangle center.
	20:18	**HS Margin** Default Value: 3 Format: U3
	17:10	**Cos(α)** Format: S0.7 2's Compliment The default is 79/128
	9:8	**Reserved** Format: MBZ
	7:0	**Sin(α)** Format: S0.7 2's Compliment The default is 101/128
2	31:21	**Reserved** Format: MBZ
	20:13	**Diamond Alpha** Format: U2.6 1 / tan(β) The default is 100/64
	12:7	**Diamond Th** Default Value: 35 Format: U6 Half length of the rhombus axis in the sat-direction.
	6:0	**Diamond dv** Default Value: 0 Format: S6 2's complement

COLOR_PROCESSING_STATE - STD/STE State

3	31:24	**Y_point_3** Default Value: 254 Format: U8 Third point of the Y piecewise linear membership function.
	23:16	**Y_point_2** Default Value: 47 Format: U8 Second point of the Y piecewise linear membership function.
	15:8	**Y_point_1** Default Value: 46 Format: U8 First point of the Y piecewise linear membership function.
	7	**VY_STD_Enable** Format: Enable Enables STD in the VY subspace.
	6:0	**Reserved** Format: MBZ
4	31:18	**Reserved** Format: MBZ
	17:13	**Y_Slope_2** Format: U2.3 Slope between points Y3 and Y4. The default is 31/8.
	12:8	**Y_Slope_1** Format: U2.3 Slope between points Y1 and Y2. The default is 31/8.
	7:0	**Y_point_4** Default Value: 255 Format: U8 Fourth point of the Y piecewise linear membership function
5	31:16	**INV_skin_types_margin** Format: U0.16

COLOR_PROCESSING_STATE - STD/STE State

		1/(2* Skin_types_margin) Value / Name / Description: 20 / **[Default]** / Skin_Type_margin
	15:0	**Inverse Margin VYL** Format: U0.16 1 / Margin_VYL The default is 3300/65536
6	31:24	**P1L** Default Value: 216 Format: U8 Y Point 1 of the lower part of the detection PWLF.
	23:16	**P0L** Default Value: 46 Format: U8 Y Point 0 of the lower part of the detection PWLF.
	15:0	**Inverse Margin VYU** Format: U0.16 1 / Margin_VYU The default is 1600/65536.
7	31:24	**B1L** Default Value: 130 Format: U8 V Bias 1 of the lower part of the detection PWLF.
	23:16	**B0L** Default Value: 133 Format: U8 V Bias 0 of the lower part of the detection PWLF.
	15:8	**P3L** Default Value: 236 Format: U8 Y Point 3 of the lower part of the detection PWLF.
	7:0	**P2L**

COLOR_PROCESSING_STATE - STD/STE State

		Default Value: 236 Format: U8 Y point 2 of the lower part of the detection PWLF.
8	31:27	**Reserved** Format: MBZ
	26:16	**S0L** Format: S2.8 2's complement Slope 0 of the lower part of the detection PWLF. The default is -5/256.
	15:8	**B3L** Default Value: 130 Format: U8 V Bias 3 of the lower part of the detection PWLF.
	7:0	**B2L** Default Value: 130 Format: U8 V Bias 2 of the lower part of the detection PWLF.
9	31:22	**Reserved** Format: MBZ
	21:11	**S2L** Format: S2.8 2's complement Slope 2 of the lower part of the detection PWLF. The default is 0/256.
	10:0	**S1L** Format: S2.8 2's complement Slope 1 of the lower part of the detection PWLF. The default is 0/256.
10	31:27	**Reserved** Format: MBZ
	26:19	**P1U** Default Value: 66 Format: U8 Y Point 1 of the upper part of the detection PWLF.

COLOR_PROCESSING_STATE - STD/STE State

	18:11	**P0U** Default Value: 46 Format: U8 Y Point 0 of the upper part of the detection PWLF.
	10:0	**S3L** Format: S2.8 2's complement Slope 3 of the lower part of the detection PWLF. The default is 0/256.
11	31:24	**B1U** Default Value: 163 Format: U8 V Bias 1 of the upper part of the detection PWLF.
	23:16	**B0U** Default Value: 143 Format: U8 V Bias 0 of the upper part of the detection PWLF.
	15:8	**P3U** Default Value: 236 Format: U8 Y Point 3 of the upper part of the detection PWLF.
	7:0	**P2U** Default Value: 150 Format: U8 Y Point 2 of the upper part of the detection PWLF.
12	31:27	**Reserved** Format: MBZ
	26:16	**S0U** Format: S2.8 2's complement Slope 0 of the upper part of the detection PWLF. The default is 256/256.
	15:8	**B3U** Default Value: 140

COLOR_PROCESSING_STATE - STD/STE State

		Format: U8 V Bias 3 of the upper part of the detection PWLF.
	7:0	**B2U** Default Value: 200 Format: U8 V Bias 2 of the upper part of the detection PWLF.
13	31:22	**Reserved** Format: MBZ
	21:11	**S2U** Format: S2.8 2's complement Slope 2 of the upper part of the detection PWLF. The default is -179/256.
	10:0	**S1U** Format: S2.8 2's complement Slope 1 of the upper part of the detection PWLF. The default is -113/256.
14	31:28	**Reserved** Format: MBZ
	27:20	**Skin Types Margin** Default Value: 20 Format: U8 Skin types Y margin.
	19:12	**Skin Types Thresh** Default Value: 120 Format: U8 Skin types Y threshold.
	11	**Skin Type Enable** Format: Enable Treat differently bright and dark skin types. Value / Name / Description 0 / **[Default]** / Disable
	10:0	**S3U** Format: S2.8 2's complement

COLOR_PROCESSING_STATE - STD/STE State

		Slope 3 of the upper part of the detection PWLF. The default is 0/256.
15	31	**Reserved** Format: MBZ
	30:21	**SATB1** Format: S7.2 2's complement First bias for the saturation PWLF (bright skin). The default is -8/4.
	20:14	**SATP3** Default Value: 31 Format: S6 2's complement Third point for the saturation PWLF (bright skin).
	13:7	**SATP2** Default Value: 6 Format: S6 2's complement Second point for the saturation PWLF (bright skin).
	6:0	**SATP1** Format: S6 2's complement First point for the saturation PWLF (bright skin). The default is -6.
16	31	**Reserved** Format: MBZ
	30:20	**SATS0** Format: U3.8 Zeroth slope for the saturation PWLF (bright skin). The default is 297/256.
	19:10	**SATB3** Format: S7.2 2's complement Third bias for the saturation PWLF (bright skin). The default is 124/4.
	9:0	**SATB2** Format: S7.2 2's complement Second bias for the saturation PWLF (bright skin).

COLOR_PROCESSING_STATE - STD/STE State

		The default is 8/4.
17	31:22	**Reserved** Format: MBZ
	21:11	**SATS2** Format: U3.8 Second slope for the saturation PWLF (bright skin). The default is 297/256.
	10:0	**SATS1** Format: U3.8 First slope for the saturation PWLF (bright skin). The default is 85/256.
18	31:25	**HUEP3** Default Value: 14 Format: S6 2's complement Third point for the hue PWLF (bright skin)
	24:18	**HUEP2** Default Value: 6 Format: S6 2's complement Second point for the hue PWLF (bright skin)
	17:11	**HUEP1** Format: S6 2's complement First point for the hue PWLF (bright skin). The default is -6.
	10:0	**SATS3** Format: U3.8 Thrid slope for the saturation PWLF (bright skin). The default is 256/256.
19	31:30	**Reserved** Format: MBZ
	29:20	**HUEB3** Format: S7.2 2's complement Third bias for the hue PWLF (bright skin). The default is 56/4.

COLOR_PROCESSING_STATE - STD/STE State

	19:10	**HUEB2** Format: S7.2 2's complement Second bias for the hue PWLF (bright skin). The default is 8/4.
	9:0	**HUEB1** Format: S7.2 2's complement First bias for the hue PWLF (bright skin). The default is -8/4.
20	31:22	**Reserved** Format: MBZ
	21:11	**HUES1** Format: U3.8 First slope for the hue PWLF (bright skin) The default is 85/256.
	10:0	**HUES0** Format: U3.8 Zeroth slope for the hue PWLF (bright skin) The default is 384/256.
21	31:22	**Reserved** Format: MBZ
	21:11	**HUES3** Format: U3.8 Third slope for the hue PWLF (bright skin) The default is 256/256.
	10:0	**HUES2** Format: U3.8 Second slope for the hue PWLF (bright skin) The default is 384/256.
22	31	**Reserved** Format: MBZ
	30:21	**SATB1_DARK** Format: S7.2 2's complement First bias for the saturation PWLF (dark skin)

COLOR_PROCESSING_STATE - STD/STE State

		The default is 0/4.
	20:14	**SATP3_DARK** Default Value: 31 Format: S6 2's complement Third point for the saturation PWLF (dark skin)
	13:7	**SATP2_DARK** Default Value: 31 Format: S6 2's complement Second point for the saturation PWLF (dark skin)
	6:0	**SATP1_DARK** Format: S6 2's complement First point for the saturation PWLF (dark skin). The default is -11.
23	31	**Reserved** Format: MBZ
	30:20	**SATS0_DARK** Format: U3.8 Zeroth slope for the saturation PWLF (dark skin). The default is 397/256.
	19:10	**SATB3_DARK** Format: S7.2 2's complement Third bias for the saturation PWLF (dark skin). The default is 124/4.
	9:0	**SATB2_DARK** Format: S7.2 2's complement Second bias for the saturation PWLF (dark skin). The default is 124/4.
24	31:22	**Reserved** Format: MBZ
	21:11	**SATS2_DARK** Format: U3.8 Second slope for the saturation PWLF (dark skin). The default is 256/256.

COLOR_PROCESSING_STATE - STD/STE State

	10:0	**SATS1_DARK** Format: U3.8 First slope for the saturation PWLF (dark skin). The default is 189/256.
25	31:25	**HUEP3_DARK** Default Value: 14 Format: S6 2's complement Third point for the hue PWLF (dark skin).
	24:18	**HUEP2_DARK** Default Value: 2 Format: S6 2's complement Third point for the hue PWLF (dark skin).
	17:11	**HUEP1_DARK** Default Value: 0 Format: S6 2's complement Third point for the hue PWLF (dark skin).
	10:0	**SATS3_DARK** Format: U3.8 Third slope for the saturation PWLF (dark skin). The default is 256/256.
26	31:30	**Reserved** Format: MBZ
	29:20	**HUEB3_DARK** Format: S7.2 2's complement Third bias for the hue PWLF (dark skin). The default is 56/4.
	19:10	**HUEB2_DARK** Format: S7.2 2's complement Second bias for the hue PWLF (dark skin). The default is 0/4.
	9:0	**HUEB1_DARK** Format: S7.2 2's complement

COLOR_PROCESSING_STATE - STD/STE State		
		First bias for the hue PWLF (dark skin). The default is 0/4.
27	31:22	**Reserved** Format: MBZ
	21:11	**HUES1_DARK** Format: U3.8 First slope for the hue PWLF (dark skin). The default is 0/256.
	10:0	**HUES0_DARK** Format: U3.8 Zeroth slope for the hue PWLF (dark skin). The default is 256/256.
28	31:22	**Reserved** Format: MBZ
	21:11	**HUES3_DARK** Format: U3.8 Third slope for the hue PWLF (dark skin). The default is 256/256.
	10:0	**HUES2_DARK** Format: U3.8 Second slope for the hue PWLF (dark skin). The default is 299/256.

COLOR_PROCESSING_STATE - TCC State

Source: BSpec

Size (in bits): 352

Default Value: 0xDCDCDC00, 0xDCDCDC00, 0x1E34CC91, 0x3E3CCE91, 0x02E80195, 0x0197046B, 0x01790174, 0x00096000, 0x00000000, 0x03030000, 0x009201C0

This state structure contains the TCC state used by the color processing function.
It corresponds to DW42..DW52 of the Color Processing State.

DWord	Bit	Description
0	31:24	**SatFactor3** Default Value: 220 Format: U1.7 The saturation factor for yellow.
	23:16	**SatFactor2** Default Value: 220 Format: U1.7 The saturation factor for red.
	15:8	**SatFactor1** Default Value: 220 Format: U1.7 The saturation factor for magenta.
	7	**TCC Enable** Format: Enable
	6:0	**Reserved** Format: MBZ
1	31:24	**SatFactor6** Default Value: 220 Format: U1.7 The saturation factor for blue.
	23:16	**SatFactor5** Default Value: 220 Format: U1.7 The saturation factor for cyan.
	15:8	**SatFactor4** Default Value: 220

COLOR_PROCESSING_STATE - TCC State

		Format: U1.7 The saturation factor for green.
	7:0	**Reserved** Format: MBZ
2	31:30	**Reserved** Format: MBZ
	29:20	**Base Color 3** Default Value: 483 Format: U10
	19:10	**Base Color 2** Default Value: 307 Format: U10
	9:0	**Base Color 1** Default Value: 145 Format: U10
3	31:30	**Reserved** Format: MBZ
	29:20	**Base Color 6** Default Value: 995 Format: U10
	19:10	**Base Color 5** Default Value: 819 Format: U10
	9:0	**Base Color 4** Default Value: 657 Format: U10
4	31:16	**Color Transit Slope 23** Default Value: 744 Format: U0.16 The calculation result of 1 / (BC3 - BC2) [1/62]
	15:0	**Color Transit Slope 12** Default Value: 405 Format: U0.16 The calculation result of 1 / (BC2 - BC1) [1/57]

COLOR_PROCESSING_STATE - TCC State

5	31:16	**Color Transit Slope 45** Default Value: 407 Format: U0.16 The calculation result of 1 / (BC5 - BC4) [1/57]
	15:0	**Color Transit Slope 34** Default Value: 1131 Format: U0.16 The calculation result of 1 / (BC4 - BC3) [1/61]
6	31:16	**Color Transit Slope 61** Default Value: 377 Format: U0.16 The calculation result of 1 / (BC1 - BC6) [1/62]
	15:0	**Color Transit Slope 56** Default Value: 372 Format: U0.16 The calculation result of 1 / (BC6 - BC5) [1/62]
7	31:22	**Color Bias 3** Default Value: 0 Format: U2.8 Color bias for BaseColor3.
	21:12	**Color Bias 2** Default Value: 150 Format: U2.8 Color bias for BaseColor2.
	11:2	**Color Bias 1** Default Value: 0 Format: U2.8 Color bias for BaseColor1.
	1:0	**Reserved** Format: MBZ
8	31:22	**Color Bias 6** Default Value: 0

COLOR_PROCESSING_STATE - TCC State

		Format: U2.8 Color bias for BaseColor6.
	21:12	**Color Bias 5** Default Value: 0 Format: U2.8 Color bias for BaseColor5.
	11:2	**ColorBias4** Default Value: 0 Format: U2.8 Color bias for BaseColor4.
	1:0	**Reserved** Format: MBZ
9	31	**Reserved** Format: MBZ
	30:24	**UV Threshold** Default Value: 3 Format: U7 Low UV threshold.
	23:19	**Reserved** Format: MBZ
	18:16	**UV Threshold Bits** Default Value: 3 Format: U3 Low UV transition width bits.
	15:13	**Reserved** Format: MBZ
	12:8	**STE Threshold** Default Value: 0 Format: U5 Skin tone pixels enhancement threshold.
	7:3	**Reserved** Format: MBZ

COLOR_PROCESSING_STATE - TCC State

	2:0	**STE Slope Bits** Default Value: 0 Format: U3 Skin tone pixels enhancement slope bits.
10	31:16	**Inverse UVMax Color** Default Value: 146 Format: U0.16 1 / UVMaxColor. Used for the SFs2 calculation.
	15:9	**Reserved** Format: MBZ
	8:0	**UVMax Color** Default Value: 448 Format: U9 The maximum absolute value of the legal UV pixels. Used for the SFs2 calculation.

DEINTERLACE_SAMPLER_STATE

Source:	BSpec
Exists If:	//MessageType == 'Deinterlace'
Size (in bits):	256
Default Value:	0x00000800, 0x00000000, 0x04950100, 0x407D0000, 0x00000000, 0x00000000, 0x00000000, 0x005064A5

This state definition is used only by the *deinterlace* message. This state is stored as an array of up to 8 elements, each of which contains the dwords described here. The start of each element is spaced 8 dwords apart. The first element of the array is aligned to a 32-byte boundary. The index with range 0-7 that selects which element is being used is multiplied by 2 to determine the **Sampler Index** in the message descriptor.

DWord	Bit	Description
0	31:24	**Denoise STAD Threshold** Threshold for denoise sum of temporal absolute differences.
	23:16	**Denoise Maximum History** Maximum allowed value for denoise history. Value: 128-240 — Name: — Description: —
	15	**Reserved** Format: MBZ
	14	**VDI Walker Frame Sharing Enable** Format: U1 Enumerated Type For a GT2 system with 2 half-slices, this field controls how the frame is shared by the two deinterlacer walkers. 0 — There is only a single deinterlacer which must walk the entire frame. VDI Walker Y Stride is ignored. 1 — The screen is shared by the two deinterlacers as controlled by the VDI Walker Y Stride
	13:12	**VDI Walker Y Stride** Format: U2 Enumerated Type This field controls if the VDI walker skips pixels as it goes down the screen. This is used when a pair of VDI'S are splitting the frame between them. The stride also implies the offset used by the 2nd half-slice. 0 — Stride of 1 block (where a block is 4x4 when DI is enabled and 4x8 when DN only), offset for the 2nd half-slice is ½ the surface height. 1 — Stride of 2 blocks (every other row of blocks calculated by this VDI), offset for the 2nd half-slice is 1 block. 2 — Stride of 4 blocks (2 vertical blocks calclated by this VDI, then skip 2), offset for the 2nd half-slice is 2 blocks. 3 — Stride of 8 blocks (4 vertical blocks calculated by this VDI, then skip 4), offset for the 2nd half-slice is 4 blocks.

DEINTERLACE_SAMPLER_STATE

	11:8	**Denoise History Delta** Default Value: 8 Amount that denoise_history is increased.
	7:0	**Denoise ASD Threshold** Threshold for denoise absolute sum of differences. Value: 0-63
1	31:30	**Reserved** Format: MBZ
	29:24	**Temporal Difference Threshold** Programming Notes: The difference between **Temporal Difference Threshold** and **Low Temporal Difference Threshold** must be larger than 0 and less than or equal to 16, except when both thresholds are set to 0.
	23:22	**Reserved** Format: MBZ
	21:16	**Low Temporal Difference Threshold** Programming Notes: The difference between **Temporal Difference Threshold** and **Low Temporal Difference Threshold** must be larger than 0 and less than or equal to 16, except when both thresholds are set to 0.
	15:13	**STMM C2** Bias for divisor in STMM equation. The range represents values [1,8] Value: 0-7
	12:8	**Denoise Moving Pixel Threshold** Threshold for number of moving pixels to declare a block to be moving. Value: 0-16
	7:0	**Denoise Threshold for Sum of Complexity Measure**
2	31:30	**Reserved** Format: MBZ
	29:24	**Good Neighbor Threshold** Maximum difference from current pixel for neighboring pixels to be considered a good neighbor. Value: 4 **[Default]** depending on GNE of previous frame

DEINTERLACE_SAMPLER_STATE

	23:20	**CAT Slope** Format: U4-1 Determines the slope of the Content Adaptive Threshold. +1 added internally to get CAT_slope. Value: 9 — Name: **[Default]** — Description: CAT_slope value = 10
	19:16	**SAD Tight Threshold** Default Value: 5 Format: U4
	15:14	**Smooth MV Threshold** Format: U2
	13:12	**Reserved** Format: MBZ
	11:8	**BNE Edge Threshold** Default Value: 1 Format: U4 Threshold for detecting an edge in block noise estimate.
	7:0	**Block Noise Estimate Noise Threshold** Format: U8 Threshold for noise maximum/minimum. Value: 0-31 — Name: — Description:
3	31	**STMM Blending Constant Select** Format: U1 Value 0 — Use Minimum STMM for stmm_md_th Value 1 — Use Maximum STMM for stmm_md_th
	30:24	**Blending constant across time for large values of STMM** Default Value: 64 Format: U7
	23:16	**Blending constant across time for small values of STMM** Default Value: 125 Format: U8
	15:14	**Reserved** Format: MBZ
	13:8	**Multiplier for VECM**

DEINTERLACE_SAMPLER_STATE

		Format: U6 Determines the strength of the vertical edge complexity measure.
	7:0	**Maximum STMM** Format: U8 Largest allowed STMM in blending equations
4	31:24	**Minimum STMM** Format: U8 Smallest allowed STMM in blending equations
	23:22	**STMM Shift Down** Format: U2 Amount to shift STMM down (quantize to fewer bits)

Value	Name
0	Shift by 4
1	Shift by 5
2	Shift by 6
3	Reserved

	21:20	**STMM Shift Up** Format: U2 Amount to shift STMM up (set range).

Value	Name
0	Shift by 6
1	Shift by 7
2	Shift by 8
3	Reserved

	19:16	**STMM Output Shift** Format: U4 Amount to shift output of STMM blend equation

Value	Name	Description
0-16		

Programming Notes
The value of this field must satisfy the following equation: stmm_max - stmm_min = 2 ^ stmm_output_shift

	15:8	**SDI Threshold** Format: U8

DEINTERLACE_SAMPLER_STATE

		Threshold for angle detection in SDI algorithm.
	7:0	**SDI Delta** Format: U8 Delta value for angle detection in SDI algorithm.
5	31:24	**SDI Fallback Mode 1 T1 Constant** Format: U8
	23:16	**SDI Fallback Mode 1 T2 Constant** Format: U8
	15:8	**SDI Fallback Mode 2 Constant (Angle2x1)** Format: U8
	7:0	**FMD Temporal Difference Threshold** Format: U8
6	31:24	**FMD #1 Vertical Difference Threshold** Format: U8
	23:16	**FMD #2 Vertical Difference Threshold** Format: U8
	15:14	**CAT Threshold 1** Default Value: 0 Format: U2
	13:8	**FMD Tear Threshold** Format: U6
	7	**MCDI Enable** Use Motion Compensated Deinterlace algorithm. Ignored if DI Enable is off.
	6	**Progressive DN** Format: Enable Indicates that the denoise algorithm should assume progressive input when filtering neighboring pixels. **DI Enable** must be disabled when this field is enabled **Value** / **Name** 0 – DN assumes interlaced video and filters alternate lines together 1 – DN assumes progressive video and filters neighboring lines together
	5	**DN/DI First Frame** Format: Enable Indicates that this is the first frame of the stream, so previous clean is not available **Value** / **Name** 0 – Not first field; previous clean surface state is valid

DEINTERLACE_SAMPLER_STATE

	1	First field; previous clean surface state is invalid

4 DN/DI Stream ID

Format:	U1

Distinguishes between the two simultaneous streams that are supported. Used to update the GNE and FMD counters for that stream.

3 DN/DI Top First

Format:	Enable

Indicates the top field is first in sequence, otherwise bottom is first

Value	Name
0	Bottom field occurs first in sequence
1	Top field occurs first in sequence

2 DI Partial

Format:	Enable

If **DI Enable** and **DI Partial** are both enabled, the deinterlacer will output the partial VDI writeback message.

Value	Name
0	Output normal VDI writeback message (only if DI Enable is enabled also)
1	Output partial VDI writeback message (only if DI Enable is enabled also)

1 DI Enable

Format:	Enable

Deinterlacer is bypassed if this is disabled: the output is the same as the input (same as a 2:2 cadence). FMD and STMM are not calculated and the values in the response message are 0.

Value	Name
0	Do not calculate DI
1	Calculate DI

Programming Notes
DI Enable and **DN Enable** cannot both be disabled

0 DN Enable

Format:	Enable

Denoise is bypassed if this is low BNE is still calculated and output, but the denoised fields are not. VDI does not read in the denoised previous frame but uses the pointer for the original previous frame.

Value	Name
0	Do not denoise frame
1	Denoise frame

Programming Notes

DEINTERLACE_SAMPLER_STATE

		DI Enable and **DN Enable** cannot both be disabled
7	31:23	**Column Width Minus 1** Format: U9 This field specifies the (column width-1) / stride in units of blocks (Each blocks has width 16 pixels). A column width * 16 that equals the width of the frame means the walker will walk to the end of the frame. The value of this field is interpreted as binary value + 1, so the range represents column widths of [1,512]. **Value** / **Name** / **Description**: 0-511
	22:19	**Neighbor Pixel Threshold** Default Value: 10 Format: U4
	18	**VDI Walker Enable** Format: U1 **Value** 0: Walker Disabled. Use XY generated by Driver. **Value** 1: Walker Enabled. Use XY generated by VDIunit. **Programming Notes**: When enabled frame size should be aligned to 16x8 in DN only mode and 16x4 in DI enabled mode When walker is enabled in a GT2 system, the MEDIA_OBJECT commands dispatching work to the VDI must use the Half-Slice Destination Select field to split the work between the two half-slices; the Half-Slice Destination Select must never be set to 00 (either half-slice).
	17:16	**FMD for 2nd field of previous frame** Format: U2 **Value** 0: Deinterlace (not progressive output) **Value** 1: Put together with previous field in sequence (1st field of previous frame) **Value** 2: Put together with next field in sequence (1st field of current frame)
	15:10	**MC Pixel Consistency Threshold** Default Value: 25 Format: U6
	9:8	**FMD for 1st field of current frame** Format: U2

DEINTERLACE_SAMPLER_STATE

		Value / **Name** 0 — Deinterlace (not progressive output) 1 — Put together with previous field in sequence (2nd field of previous frame) 2 — Put together with next field in sequence (2nd field of current frame)
	7:4	**SAD Threshold B** Default Value: 10 Format: U4
	3:0	**SAD Threshold A** Default Value: 5 Format: U4

DEPTH_STENCIL_STATE

Source: BSpec
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000

The DEPTH_STENCIL_STATE is pointed to by a field in 3DSTATE_CC_STATE_POINTERS. It is stored at a 64-byte aligned boundary.

DWord	Bit	Description
0	31	**Stencil Test Enable** Format: Enable Enables StencilTest function of the Pixel Processing pipeline. **Programming Notes** If any of the render targets are YUV format, this field must be disabled.
	30:28	**Stencil Test Function** Format: 3D_CompareFunction This field specifies the comparison function used in the (front face) StencilTest function. Value / Name: 0h COMPAREFUNCTION_ALWAYS 1h COMPAREFUNCTION_NEVER 2h COMPAREFUNCTION_LESS 3h COMPAREFUNCTION_EQUAL 4h COMPAREFUNCTION_LEQUAL 5h COMPAREFUNCTION_GREATER 6h COMPAREFUNCTION_NOTEQUAL 7h COMPAREFUNCTION_GEQUAL
	27:25	**Stencil Fail Op** Format: 3D_StencilOperation This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test fails. Note: if all three stencil ops (Stencil Fail, Stencil Pass Depth Fail, and Stencil Pass Depth Pass) are KEEP, ZERO, or REPLACE, the stencil buffer is not read. Value / Name: 0 STENCILOP_KEEP 1 STENCILOP_ZERO 2 STENCILOP_REPLACE 3 STENCILOP_INCRSAT 4 STENCILOP_DECRSAT 5 STENCILOP_INCR 6 STENCILOP_DECR 7 STENCILOP_INVERT

DEPTH_STENCIL_STATE

24:22 Stencil Pass Depth Fail Op

Format:	3D_StencilOperation see Stencil Fail Op

This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes but the depth pass fails.

21:19 Stencil Pass Depth Pass Op

Format:	3D_StencilOperation see Stencil Fail Op

This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes and the depth pass passes (or is disabled).

18 Stencil Buffer Write Enable

Format:	Enable

Enables writes to the Stencil Buffer.

Programming Notes
If this field is enabled, Stencil Test Enable must also be enabled.

17:16 Reserved

Format:	MBZ

15 Double Sided Stencil Enable

Format:	Enable

Enable doubled sided stencil operations.

Value	Name	Description
1	Enable	Double Sided Stencil Enabled
0	Disable	Double Sided Stencil Disabled

Programming Notes
Back-facing primitives have a vertex winding order opposite to the currently selected Front Winding state. Culling of primitives is not affected by the double sided stencil stateBack-facing primitives will be rendered, honoring all current device state, as though it were a front-facing primitive with no implicitly overloaded state.

14:12 BackFace Stencil Test Function

Format:	3D_CompareFunction

This field specifies the comparison function used in the StencilTest function.

Value	Name
0h	COMPAREFUNCTION_ALWAYS
1h	COMPAREFUNCTION_NEVER
2h	COMPAREFUNCTION_LESS
3h	COMPAREFUNCTION_EQUAL
4h	COMPAREFUNCTION_LEQUAL
5h	COMPAREFUNCTION_GREATER

DEPTH_STENCIL_STATE

		6h COMPAREFUNCTION_NOTEQUAL 7h COMPAREFUNCTION_GEQUAL
	11:9	**Backface Stencil Fail Op** Format: 3D_StencilOperation This field specifies the operation to perform on the Stencil Buffer when the stencil test fails. 0 STENCILOP_KEEP STENCILOP_KEEP 1 STENCILOP_ZERO STENCILOP_ZERO 2 STENCILOP_REPLACE STENCILOP_REPLACE 3 STENCILOP_INCRSAT STENCILOP_INCRSAT 4 STENCILOP_DECRSAT STENCILOP_DECRSAT 5 STENCILOP_INCR STENCILOP_INCR 6 STENCILOP_DECR STENCILOP_DECR 7 STENCILOP_INVERT STENCILOP_INVERT
	8:6	**Backface Stencil Pass Depth Fail Op** Format: 3D_StencilOperation see Stencil Fail Op This field specifies the operation to perform on the Stencil Buffer when the stencil test passes but the depth pass fails.
	5:3	**Backface Stencil Pass Depth Pass Op** Format: 3D_StencilOperation see Stencil Fail Op This field specifies the operation to perform on the Stencil Buffer when the stencil test passes and the depth pass passes (or is disabled).
	2:0	**Reserved** Format: MBZ
1	31:24	**Stencil Test Mask** Format: U8 This field specifies a bit mask applied to stencil test values. Both the stencil reference value and value read from the stencil buffer will be logically ANDed with this mask before the stencil comparison test is performed.
	23:16	**Stencil Write Mask** Format: U8 This field specifies a bit mask applied to stencil buffer writes. Only those stencil buffer bits corresponding to bits set in this mask will be modified.
	15:8	**Backface Stencil Test Mask** Format: U8

DEPTH_STENCIL_STATE

		This field specifies a bit mask applied to backface stencil test values. Both the stencil reference value and value read from the stencil buffer will be logically ANDed with this mask before the stencil comparison test is performed.
	7:0	**Backface Stencil Write Mask** Format: U8 This field specifies a bit mask applied to backface stencil buffer writes. Only those stencil buffer bits corresponding to bits set in this mask will be modified.
2	31	**Depth Test Enable** Format: Enable Enables the DepthTest function of the Pixel Processing pipeline. **Programming Notes** If any of the render targets are YUV format, this field must be disabled.
	30	**Reserved** Format: MBZ
	29:27	**Depth Test Function** Format: 3D_DepthTestFunction Specifies the comparison function used in DepthTest function. **Value** / **Name** 0h COMPAREFUNCTION_ALWAYS 1h COMPAREFUNCTION_NEVER 2h COMPAREFUNCTION_LESS 3h COMPAREFUNCTION_EQUAL 4h COMPAREFUNCTION_LEQUAL 5h COMPAREFUNCTION_GREATER 6h COMPAREFUNCTION_NOTEQUAL 7h COMPAREFUNCTION_GEQUAL **Programming Notes** if the Depth Test Function is ALWAYS or NEVER, the depth buffer is not read.
	26	**Depth Buffer Write Enable** Format: Enable Enables writes to the Depth Buffer. **Programming Notes** A Depth Buffer must be defined before enabling writes to it, or operation is UNDEFINED.
	25:0	**Reserved** Format: MBZ

DstRegNum

Source:	EuIsa
Size (in bits):	8
Default Value:	0x00000000

Description

Register Number
This field provides the register number for the operand. For GRF register operand, it provides the portion of register address aligning to 256-bit. For an ARF register operand, this field is encoded such that MSBs identify the architecture register type and LSBs provide its register number.
This field together with the corresponding SubRegNum field provides the byte aligned address for the origin of the register region. Specifically, this field provides bits [12:5] of the byte address, while SubRegNum field provides bits [4:0].
This field applies to the destination operand and the source operands. It is ignored (or not present in the instruction word) for an immediate source operand.
This field is present if the operand is in direct addressing mode; it is not present if the operand is register-indirect addressed.

DWord	Bit	Description
0	7:0	**Destination Register Number**

Value	Name	Description
0-127	If {Dst/Src0/Src1/Src2}.RegFile==GRF	
0-0ffh	If {Dst/Src0/Src1/Src2}.RegFile==ARF	This field is used to encode the architecture register as well as providing the register number. See GEN Execution Environment chapter for details.

DstSubRegNum

Source:	EuIsa
Size (in bits):	5
Default Value:	0x00000000

Description

Subregister Number

This field provides the sub-register number for the operand. For a GRF register operand, it provides the byte address within a 256-bit register. For an ARF register operand, this field also provides the sub-register number according to the encoding defined for the given architecture register.

This field together with the corresponding RegNum field provides the byte aligned address for the origin of the register region. Specifically, this field provides bits [4:0] of the byte address, while the RegNum field provides bits [12:5].

This field applies to the destination operand and the source operands. It is ignored (or not present in the instruction word) for an immediate source operand.

This field is present if the operand is in direct addressing mode; it is not present if the operand is register-indirect addressed.

Programming Notes

Note: The recommended instruction syntax uses subregister numbers within the GRF in units of actual data element size, corresponding to the data type used. For example for the F (Float) type, the assembler syntax uses subregister numbers 0 to 7, corresponding to subregister byte addresses of 0 to 28 in steps of 4, the element size.

DWord	Bit	Description
0	4:0	**Destination Sub Register Number**

Value	Name	Description
0-31	If {Dst/Src0/Src1/Src2}.RegFile==GRF	
0-0ffh	If {Dst/Src0/Src1/Src2}.RegFile==ARF	This field is used to encode the architecture register as well as providing the register number. See GEN Execution Environment chapter for details.

EU_INSTRUCTION_BASIC_ONE_SRC

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:64	**ImmSource** Exists If: ([Operand Controls][Src0.RegFile]=='IMM') Format: EU_INSTRUCTION_SOURCES_IMM32
	127:64	**RegSource** Exists If: ([Operand Controls][Src0.RegFile]!='IMM') Format: EU_INSTRUCTION_SOURCES_REG
	63:32	**Operand Controls** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BASIC_THREE_SRC

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:126	**Reserved** Format: MBZ
	125:106	**Source 2** Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC
	105	**Reserved** Format: MBZ
	104:85	**Source 1** Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC
	84	**Reserved** Format: MBZ
	83:64	**Source 0** Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC
	63:56	**Destination Register Number** Format: DstRegNum
	55:53	**Destination Subregister Number** Format: DstSubRegNum[2:0]
	52:49	**Destination Channel Enable** Format: ChanEn[4] Four channel enables are defined for controlling which channels are written into the destination region. These channel mask bits are applied in a modulo-four manner to all ExecSize channels. There is 1-bit Channel Enable for each channel within the group of 4. If the bit is cleared, the write for the corresponding channel is disabled. If the bit is set, the write is enabled. Mnemonics for the bit being set for the group of 4 are *x*, *y*, *z*, and *w*, respectively, where *x* corresponds to Channel 0 in the group and *w* corresponds to channel 3 in the group
	48	**Reserved** Format: MBZ
	47	**NibCtrl** Format: NibCtrl
	46	**Reserved** Format: MBZ
	45:44	**Destination Data Type** This field contains the data type for the destination

EU_INSTRUCTION_BASIC_THREE_SRC

		Value: 00b — Name: Single Precision Float Value: 01b — Name: DWord Value: 10b — Name: Unsigned DWord Value: 11b — Name: Double Precision Float
	43:42	**Source Data Type** This field contains the data type for all three sources Value: 00b — Name: Single Precision Float Value: 01b — Name: DWord Value: 10b — Name: Unsigned DWord Value: 11b — Name: Double Precision Float
	41:40	**Source 2 Modifier** Exists If: ([Property[Source Modification]=='true') Format: SrcMod
	39:38	**Source 1 Modifier** Exists If: ([Property[Source Modification]=='true') Format: SrcMod
	41:36	**Reserved** Exists If: ([Property[Source Modification]=='false') Format: MBZ
	37:36	**Source 0 Modifier** Exists If: ([Property[Source Modification]=='true') Format: SrcMod
	35	**Reserved** Format: MBZ
	34	**Flag Register Number** This field contains the flag register number for instructions with a non-zero Conditional Modifier.
	33	**Flag Subregister Number** This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.
	32	**Reserved** Format: MBZ
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BASIC_TWO_SRC

Source: EuIsa

Size (in bits): 128

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:64	**ImmSource** Exists If: ([ImmSource][Src1.RegFile]=='IMM') Format: EU_INSTRUCTION_SOURCES_REG_IMM
	127:64	**RegSource** Exists If: ([RegSource][Src1.RegFile]!='IMM') Format: EU_INSTRUCTION_SOURCES_REG_REG
	63:32	**Operand Controls** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BRANCH_CONDITIONAL

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:64	**Sources** Exists If: ([Src1.RegFile]=='IMM') Format: EU_INSTRUCTION_SOURCES_REG_IMM
	127:64	**Sources** Exists If: ([Src1.RegFile]!='IMM') Format: EU_INSTRUCTION_SOURCES_REG_REG
	63:48	**JIP** Format: S15 Jump Target Offset. The jump distance in number of eight-byte units if a jump is taken for the instruction.
	47	**Reserved** Format: MBZ
	46:44	**Src1.SrcType** Format: DataType This field specifies the numeric data type of the source operand src1. The bits of a source operand are interpreted as the identified numeric data type, rather than coerced into a type implied by the operator. Depending on RegFile field of the source operand, there are two different encoding for this field. If a source is a register operand, this field follows the Source Register Type Encoding. If a source is an immediate operand, this field follows the Source Immediate Type Encoding. **Programming Notes** Both source operands, src0 and src1, support immediate types, but only one immediate is allowed for a given instruction and it must be the last operand. Halfbyte integer vector (v) type can only be used in instructions in packed-word execution mode. Therefore, in a two-source instruction where src1 is of type :v, src0 must be of type :b, :ub, :w, or :uw.
	43:42	**Src1.RegFile** Format: RegFile
	41:39	**Src0.SrcType** Format: DataType
	38:37	**Src0.RegFile** Format: RegFile
	36:34	**Destination Data Type** Format: DataType

EU_INSTRUCTION_BRANCH_CONDITIONAL

This field specifies the numeric data type of the destination operand dst. The bits of the destination operand are interpreted as the identified numeric data type, rather than coerced into a type implied by the operator. For a send instruction, this field applies to the CurrDst ? the current destination operand.

33:32 Destination Register File

Format:	RegFile

Value	Name	Description
11b	Reserved	Note that it is obvious that immediate cannot be a destination operand.

31:0 Header

Format:	EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BRANCH_ONE_SRC

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:112	**Reserved** Format: MBZ
	111:96	**JIP** Format: S15 Jump Target Offset. The relative offset in 64-bit units if a jump is taken for the instruction.
	95:91	**Reserved** Format: MBZ
	90	**Flag Register Number** Added a second flag register
	89	**Flag Subregister Number** This field specifies the sub-register number for a flag register operand. There are two sub-registers in the flag register. Each sub-register contains 16 flag bits. The selected flag sub-register is the source for predication if predication is enabled for the instruction. It is the destination to store conditional flag bits if conditional modifier is enabled for the instruction. The same flag sub-register can be both the predication source and conditional destination, if both predication and conditional modifier are enabled.
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16
	63:32	**Operand Control** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BRANCH_TWO_SRC

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:112	**UIP** Format: S15 The jump distance in number of eight-byte units if a jump is taken for the channel.
	111:96	**JIP** Format: S15 The jump distance in number of eight-byte units if a jump is taken for the instruction.
	95:64	**Reserved** Format: MBZ
	63:32	**Operand Control** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_COMPACT_TWO_SRC

Source:	EuIsa
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

The following table describes the EU compact instruction format. Instructions with three source operands cannot be compacted.

DWord	Bit	Description
0..1	63:56	**Src1.RegNum** Exists If: ([DataTypeIndex][Src1.RegFile]=='IMM') Maps to 103:96 (Imm32[7:0])
	63:56	**Src1.RegNum** Exists If: ([DataTypeIndex][Src1.RegFile]!='IMM') Format: SrcRegNum Maps to 108:101 (Src1.RegNum)
	55:48	**Src0.RegNum** Format: SrcRegNum Maps to 76:69 (Src0.RegNum)
	47:40	**Dst.RegNum** Format: DstRegNum Maps to 60:53 (Dst.RegNum)
	39:35	**Src1Index** Exists If: ([DataTypeIndex][Src1.RegFile]=='IMM') If an immediate operand, does not do any lookup. The 5-bit value directly maps to bits 108:104 (Imm32[12:8]) and the upper bit (bit 39 in the compact format, bit 108 in the native format) is replicated to provide bits 127:109 (Imm32[31:13]) in the native format. Maps to 108:104
	39:35	**Src1Index** Exists If: ([DataTypeIndex][Src1.RegFile]!='IMM') Format: SrcIndex Lookup one of 32 12-bit values. If not an immediate operand, maps to bits 120:109, covering the Src1.AddrMode, Src1.ChanSel[7:4], Src1.HorzStride, Src1.SrcMod, Src1.VertStride, and Src1.Width bit fields. Maps to 120:109
	34:30	**Src0Index** Format: SrcIndex

EU_INSTRUCTION_COMPACT_TWO_SRC

		Maps to 88:77
	29	**Compaction Control** Format: CmptCtrl
	28	**Reserved** Format: MBZ
	27:24	**Conditional Modifier** Exists If: (Property[Conditional Modifier]=='true') Format: CondModifier
	27:24	**Reserved** Exists If: (Property[Conditional Modifier]=='false') Format: MBZ
	23	**Accumulator Write Control** Format: AccWrCtrl
	22:18	**SubRegIndex** Lookup one of 32 15-bit values. That value is used (from MSB to LSB) for various fields for Src1, Src0, and Dst, including ChanEn/ChanSel, SubRegNum, and AddrImm[4] or AddrImm[4:0], depending on AddrMode and AccessMode. Maps to 100:96, 68:64, 52:48

Value	Name	Description
0	000000000000000	0 | 0 | 0 |
1	000000000000001	0.x | 0.xx | 0.xx
2	000000000001000	8 | 0 | 0 |
3	000000000001111	0.xyzw | 0.xx | 0.xx
4	000000000010000	16 | 0 | 0 |
5	000000010000000	0 | 4 | 0 |
6	000000100000000	0 | 8 | 0 |
7	000000110000000	0 | 12 | 0 |
8	000001000000000	0 | 16 | 0 |
9	000001000010000	16 | 16 | 0 |
10	000001010000000	0 | 20 | 0 |
11	001000000000000	0 | 0 | 4 |
12	001000000000001	0.x | 0.xx | 0.xy
13	001000010000001	0.x | 0.xy | 0.xy
14	001000010000010	0.y | 0.xy | 0.xy
15	001000010000011	0.xy | 0.xy | 0.xy

EU_INSTRUCTION_COMPACT_TWO_SRC

16	001000010000100	0.z | 0.xy | 0.xy
17	001000010000111	0.xyz | 0.xy | 0.xy
18	001000010001000	0.w | 0.xy | 0.xy
19	001000010001110	0.yzw | 0.xy | 0.xy
20	001000010001111	0.xyzw | 0.xy | 0.xy
21	001000110000000	0 | 12 | 4 |
22	001000111101000	0.w | 0.ww | 0.xy
23	010000000000000	0 | 0 | 8 |
24	010000110000000	0 | 12 | 8 |
25	011000000000000	0 | 0 | 12 |
26	011110010000111	0.xyz | 0.xy | 0.ww
27	100000000000000	0 | 0 | 16 |
28	101000000000000	0 | 0 | 20 |
29	110000000000000	0 | 0 | 24 |
30	111000000000000	0 | 0 | 28 |
31	111000000011100	28 | 0 | 28 |

17:13 **DataTypeIndex**

Lookup one of 32 18-bit values. That value is used (from MSB to LSB) for the Dst.AddrMode, Dst.HorzStride, Dst.DstType, Dst.RegFile, Src0.SrcType, Src0.RegFile, Src1.SrcType, and Src1.RegType bit fields.

Maps to 63:61, 46:32

Value	Name	Description
0	001000000000000001	r:ud | a:ud | a:ud | <1> | dir |
1	001000000000100000	a:ud | r:ud | a:ud | <1> | dir |
2	001000000000100001	r:ud | r:ud | a:ud | <1> | dir |
3	001000000001100001	r:ud | i:ud | a:ud | <1> | dir |
4	001000000010111101	r:f | r:d | a:ud | <1> | dir |
5	001000001011111101	r:f | i:vf | a:ud | <1> | dir |
6	001000001110100001	r:ud | r:f | a:ud | <1> | dir |
7	001000001110100101	r:d | r:f | a:ud | <1> | dir |
8	001000001110111101	r:f | r:f | a:ud | <1> | dir |
9	001000010000100001	r:ud | r:ud | r:ud | <1> | dir |
10	001000110000100000	a:ud | r:ud | i:ud | <1> | dir |
11	001000110000100001	r:ud | r:ud | i:ud | <1> | dir |
12	001001010010100101	r:d | r:d | r:d | <1> | dir |
13	001001110010100100	a:d | r:d | i:d | <1> | dir |

EU_INSTRUCTION_COMPACT_TWO_SRC

14	0010011100101001 01	r:d | r:d | i:d | <1> | dir |
15	0011110011101111 01	r:f | r:f | a:f | <1> | dir |
16	0011110111100111 01	r:f | a:f | r:f | <1> | dir |
17	0011110111101111 00	a:f | r:f | r:f | <1> | dir |
18	0011110111101111 01	r:f | r:f | r:f | <1> | dir |
19	0011111111101111 00	a:f | r:f | i:f | <1> | dir |
20	0000000010000011 00	a:w | a:ub | a:ud | <0> | dir |
21	0010000000001111 01	r:f | r:ud | a:ud | <1> | dir |
22	0010000000101001 01	r:d | r:d | a:ud | <1> | dir |
23	0010000100001000 00	a:ud | r:ud | r:ud | <1> | dir |
24	0010010010101001 00	a:d | r:d | r:d | <1> | dir |
25	0010011100100001 00	a:d | a:d | i:d | <1> | dir |
26	0010100101000010 01	r:uw | a:uw | r:uw | <1> | dir |
27	0011011111101111 01	r:f | r:f | i:vf | <1> | dir |
28	0011111111101111 01	r:f | r:f | i:f | <1> | dir |
29	0010111101101011 00	a:w | r:w | i:w | <1> | dir |
30	0010100101001010 00	a:uw | r:uw | r:uw | <1> | dir |
31	0010100101001010 00	a:uw | r:uw | i:uw | <1> | dir |

12:8 **ControlIndex**

Lookup one of 32 19-bit values. That value is used (from MSB to LSB) for the FlagRegNum, FlagSubRegNum, Saturate, ExecSize, PredInv, PredCtrl, ThreadCtrl, QtrCtrl, DepCtrl, MaskCtrl, and AccessMode bit fields.

Maps to 90:89, 31, 23:8

Value	Name	Description
0	0000000000000000 10	Align1 | We | (1) | f0.0
1	0000100000000000 00	Align1 | (4) | f0.0
2	0000100000000000 01	Align16 | (4) | f0.0
3	0000100000000000 10	Align1 | We | (4) | f0.0
4	0000100000000000 11	Align16 | We | (4) | f0.0
5	0000100000000001 00	Align1 | NoDDClr | (4) | f0.0
6	0000100000000001 01	Align16 | NoDDClr | (4) | f0.0
7	0000100000000001 11	Align16 | We | NoDDClr | (4) | f0.0
8	0000100000000010 00	Align1 | NoDDChk | (4) | f0.0
9	0000100000000010 01	Align16 | NoDDChk | (4) | f0.0
10	0000100000000011 01	Align16 | NoDDClr, NoDDChk | (4) | f0.0
11	0000110000000000 00	Align1 | Q1 | (8) | f0.0

EU_INSTRUCTION_COMPACT_TWO_SRC

12	0000110000000000001	Align16 | Q1 | (8) | f0.0
13	0000110000000000010	Align1 | We | Q1 | (8) | f0.0
14	0000110000000000011	Align16 | We | Q1 | (8) | f0.0
15	0000110000000000100	Align1 | NoDDClr | Q1 | (8) | f0.0
16	0000110000000000101	Align16 | NoDDClr | Q1 | (8) | f0.0
17	0000110000000000111	Align16 | We | NoDDClr | Q1 | (8) | f0.0
18	0000110000000001001	Align16 | NoDDChk | Q1 | (8) | f0.0
19	0000110000000001101	Align16 | NoDDClr, NoDDChk | Q1 | (8) | f0.0
20	0000110000000010000	Align1 | Q2 | (8) | f0.0
21	0000110000100000000	Align1 | Q1 | +f.xyzw | (8) | f0.0
22	0001000000000000000	Align1 | H1 | (16) | f0.0
23	0001000000000000010	Align1 | We | H1 | (16) | f0.0
24	0001000000000000100	Align1 | NoDDClr | H1 | (16) | f0.0
25	0001000000100000000	Align1 | H1 | +f.xyzw | (16) | f0.0
26	0010110000000000000	Align1 | Q1 | (8) | .sat | f0.0
27	0010110000000010000	Align1 | Q2 | (8) | .sat | f0.0
28	0011000000000000000	Align1 | H1 | (16) | .sat | f0.0
29	0011000000100000000	Align1 | H1 | +f.xyzw | (16) | .sat | f0.0
30	0101000000000000000	Align1 | H1 | (16) | f0.1
31	0101000000100000000	Align1 | H1 | +f.xyzw | (16) | f0.1

7	**DebugCtrl**
	Format: DebugCtrl
6:0	**Opcode**

EU_INSTRUCTION_CONTROLS_A

Source: EuIsa
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15:13	**ExecSize** Format: ExecSize This field determines the number of channels operating in parallel for this instruction. The size cannot exceed the maximum number of channels allowed for the given data type.
	12	**Reserved** Exists If: (Property[Predication]=='false')
	12	**PredInv** Exists If: (Property[Predication]=='true') This field, together with PredCtrl, enables and controls the generation of the predication mask for the instruction. When it is set, the predication uses the inverse of the predication bits generated according to setting of Predicate Control. In other words, effect of PredInv happens after PredCtrl. This field is ignored by hardware if Predicate Control is set to 0000 - there is no predication. Value / Name: 0 — Positive polarity of predication **[Default]** 1 — Negative polarity of predication
	11:8	**Reserved** Exists If: (Property[Predication]=='false') Format: PredCtrl
	11:8	**PredCtrl** Exists If: (Property[Predication]=='true') Format: PredCtrl This field, together with PredInv, enables and controls the generation of the predication mask for the instruction. It allows per-channel conditional execution of the instruction based on the content of the selected flag register. Encoding depends on the access mode. In Align16 access mode, there are eight encodings (including no predication). All encodings are based on group-of-4 predicate bits, including channel sequential, replication swizzles and horizontal any|all operations. The same configuration is repeated for each group-of-4 execution channels.
	7:6	**Thread Control** Format: ThreadCtrl
	5:4	**QtrCtrl** Format: QtrCtrl Quarter Control

EU_INSTRUCTION_CONTROLS_A

This field provides explicit control for ARF selection.
This field combined with NibCtrl and ExecSize determines which channels are used for the ARF registers.

3:2 **DepCtrl**

Format:	DepCtrl

Destination Dependency Control. This field selectively disables destination dependency check and clear for this instruction.

1 **MaskCtrl**

Mask Control (formerly Write Enable Control). This field determines if the the per channel write enables are used to generate the final write enable. This field should be normally "0".

Value	Name	Description
0	Normal **[Default]**	
1	Write all channels	Except channels killed with predication control

Programming Notes
MaskCtrl = NoMask skips the check for PcIP[n] == ExIP before enabling a channel, as described in the Evaluate Write Enable section.

0 **AccessMode**

Access Mode
This field determines the operand access for the instruction. It applies to all source and destination operands.
When it is cleared (Align1), the instruction uses byte-aligned addressing for source and destination operands. Source swizzle control and destination mask control are not supported.
When it is set (Align16), the instruction uses 16-byte-aligned addressing for all source and destination operands. Source swizzle control and destination mask control are supported in this mode.

Value	Name
0	Align1 **[Default]**
1	Align16

EU_INSTRUCTION_CONTROLS_B

Source: EuIsa
Size (in bits): 4
Default Value: 0x00000000

DWord	Bit	Description
0	3	**Reserved** Exists If: (Property[Saturation]=='false') Format: MBZ
	3	**Saturate** Exists If: (Property[Saturation]=='true') This field controls the destination saturation. When it is set, output data to the destination register are saturated. The saturation operation depends on the destination data type. Saturation is the operation that converts any data that is outside the saturation target range for the data type to the closest representable value with the target range. If destination type is float, saturation target range is [0, 1]. For example, any positive number greater than 1 (including +INF) is saturated to 1 and any negative number (including -INF) is saturated to 0. A NaN is saturated to 0, For integer data types, the maximum range for the given numerical data type is the saturation target range. When it is not set, output data to the destination register are not saturated. For example, a wrapped result (modular) is output to the destination for an overflowed integer data. More details can be found in the Data Types chapter. Value / Name / Description: 0 — No destination modification **[Default]** 1 — sat — Saturate the output
	2	**DebugCtrl** This field allows the insertion of a breakpoint at the current instruction. When the bit is set, hardware automatically stores the current IP in CR register and jumps to the System IP (SIP) BEFORE executing the current instruction. Value / Name: 0 — No Breakpoint **[Default]** 1 — Breakpoint
	1	**CmptCtrl** Compaction Control Indicates whether the instruction is compacted to the 64-bit compact instruction format. When this bit is set, the 64-bit compact instruction format is used. The EU decodes the compact format using lookup tables internal to the hardware, but documented for use by software tools. Only some instruction variations can be compacted, the variations supported by those lookup tables and the compact format. Value / Name / Description: 0 — NoCompaction — No compaction. 128-bit native instruction supporting all instruction options. 1 — Compacted — Compaction is enabled. 64-bit compact instruction supporting only some

EU_INSTRUCTION_CONTROLS_B

				instruction variations.
	0	**AccWrCtrl** AccWrCtrl. This field allows per instruction accumulator write control.		
		Value	**Name**	**Description**
		0	Don't write to ACC **[Default]**	
		1	Update ACC	Write result to the ACC, and destination

EU_INSTRUCTION_CONTROLS

Source: EuIsa

Size (in bits): 24

Default Value: 0x00000000

Most fields in Instruction Operation Doubleword (DW0) apply to all instructions. Bit field [27:24] is one exception. It is CondModifier for most instructions but is SFID[3:0] field for the send instruction.
The descriptions in the table below are shared between the 1-src/2-src instructions and 3-src instructions.

DWord	Bit	Description
0	23:20	**Controls B** Format: EU_INSTRUCTION_CONTROLS_B
	19:16	**CondModifier** Exists If: (Property[Conditional Modifier]=='true') Format: CondModifier This field sets the flag register based on the internal conditional signals output from the execution pipe such as sign, zero, overflow and NaNs, etc. If this field is set to 0000, no flag registers are updated. Flag registers are not updated for instructions with embedded compares. This field may also be referred to as the flag destination control field. Does not exist for send/sendc/math/branch/break-continue opcodes
	19:16	**Reserved** Exists If: (Property[Conditional Modifier]=='false') Format: MBZ
	15:0	**Controls A** Format: EU_INSTRUCTION_CONTROLS_A

EU_INSTRUCTION_FLAGS

Source: EuIsa
Size (in bits): 7
Default Value: 0x00000000

DWord	Bit	Description
0	6:2	**Reserved** Format: MBZ
	1	**Flag Register Number** Added a second flag register
	0	**Flag Subregister Number** This field specifies the sub-register number for a flag register operand. There are two sub-registers in the flag register. Each sub-register contains 16 flag bits. The selected flag sub-register is the source for predication if predication is enabled for the instruction. It is the destination to store conditional flag bits if conditional modifier is enabled for the instruction. The same flag sub-register can be both the predication source and conditional destination, if both predication and conditional modifier are enabled.

EU_INSTRUCTION_HEADER

Source: EuIsa
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:8	**Control** Format: EU_INSTRUCTION_CONTROLS
	7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_ILLEGAL

Source:	EuIsa
Size (in bits):	128
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_MATH

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:64	**RegSource** Format: EU_INSTRUCTION_SOURCES_REG_REG
	63:32	**Operand Control** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:28	**Controls B** Format: EU_INSTRUCTION_CONTROLS_B
	27:24	**Function Control (FC)** Format: FC
	23:8	**Controls A** Format: EU_INSTRUCTION_CONTROLS_A
	7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_NOP

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:31	**Reserved** Format: MBZ
	30	**DebugCtrl** This field allows the insertion of a breakpoint at the current instruction. When the bit is set, hardware automatically stores the current IP in CR register and jumps to the System IP (SIP) BEFORE executing the current instruction. Value: 0 — Name: No Breakpoint **[Default]** Value: 1 — Name: Breakpoint
	29:7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_OPERAND_CONTROLS

Source: EuIsa
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:16	**Destination Register Region** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_DST_ALIGN16
	31:16	**Destination Register Region** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_DST_ALIGN1
	15	**NibCtrl**
	14:12	**Src1.SrcType** Format: DataType This field specifies the numeric data type of the source operand src1. The bits of a source operand are interpreted as the identified numeric data type, rather than coerced into a type implied by the operator. Depending on RegFile field of the source operand, there are two different encoding for this field. If a source is a register operand, this field follows the Source Register Type Encoding. If a source is an immediate operand, this field follows the Source Immediate Type Encoding. **Programming Notes** Both source operands, src0 and src1, support immediate types, but only one immediate is allowed for a given instruction and it must be the last operand. Halfbyte integer vector (v) type can only be used in instructions in packed-word execution mode. Therefore, in a two-source instruction where src1 is of type :v, src0 must be of type :b, :ub, :w, or :uw.
	11:10	**Src1.RegFile** Format: RegFile
	9:7	**Src0.SrcType** Format: DataType
	6:5	**Src0.RegFile** Format: RegFile
	4:2	**Destination Data Type** Format: DataType This field specifies the numeric data type of the destination operand dst. The bits of the destination operand are interpreted as the identified numeric data type, rather than coerced into a type implied by the operator. For a send instruction, this field applies to the CurrDst ? the current destination operand.
	1:0	**Destination Register File**

EU_INSTRUCTION_OPERAND_CONTROLS

Format:	RegFile

Value	Name	Description
11b	Reserved	Note that it is obvious that immediate cannot be a destination operand.

EU_INSTRUCTION_OPERAND_DST_ALIGN1

Source: EuIsa
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15	**Destination Addressing Mode** Format: AddrMode For a send instruction, this field applies to PostDst - the post destination operand. Addressing mode for CurrDst (current destination operand) is fixed as Direct. (See Instruction Reference chapter for CurrDst and PostDst.)
	14:13	**Destination Horizontal Stride** Format: HorzStride For a send instruction, this field applies to CurrDst. PostDst only uses the register number.
	12:10	**Destination Address Subregister Number** Exists If: ([Destination Addressing Mode]=='Indirect') Format: AddrSubRegNum For a send instruction, this field applies to PostDst
	12:5	**Destination Register Number** Exists If: ([Destination Addressing Mode]=='Direct') Format: DstRegNum For a send instruction, this field applies to PostDst.
	9:0	**Destination Address Immediate** Exists If: ([Destination Addressing Mode]=='Indirect') Format: S9 For a send instruction, this field applies to PostDst.
	4:0	**Destination Subregister Number** Exists If: ([Destination Addressing Mode]=='Direct') Format: DstSubRegNum For a send instruction, this field applies to CurrDst.

EU_INSTRUCTION_OPERAND_DST_ALIGN16

Source: EuIsa
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15	**Destination Addressing Mode** Format: AddrMode For a send instruction, this field applies to PostDst - the post destination operand. Addressing mode for CurrDst (current destination operand) is fixed as Direct. (See Instruction Reference chapter for CurrDst and PostDst.)
	14:13	**Reserved** Value: 01b — Name: See Programming Note **Programming Notes** Although Dst.HorzStride is a don?t care for Align16, HW needs this to be programmed as ?01?.
	12:10	**Destination Address Subregister Number** Exists If: ([Destination Addressing Mode]=='Indirect') Format: AddrSubRegNum For a send instruction, this field applies to PostDst
	12:5	**Destination Register Number** Exists If: ([Destination Addressing Mode]=='Direct') Format: DstRegNum For a send instruction, this field applies to PostDst.
	9:4	**Destination Address Immediate[9:4]** Exists If: ([Destination Addressing Mode]=='Indirect') Format: S9[9:4] For a send instruction, this field applies to PostDst
	4	**Destination Subregister Number** Exists If: ([Destination Addressing Mode]=='Direct') Format: DstSubRegNum[4:4] For a send instruction, this field applies to CurrDst.

EU_INSTRUCTION_OPERAND_DST_ALIGN16

	3:0	**Destination Channel Enable** Format: ChanEn[4] For a send instruction, this field applies to the CurrDst

EU_INSTRUCTION_OPERAND_SEND_MSG

Source: EuIsa
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31	**EOT** **Description** For a send or sendc instruction, this bit controls thread termination. It is not used for other instructions. For a send or sendcinstruction, if this field is set, the EU terminates the thread and also sets the EOT bit in the message sideband. **Value** / **Name** 0 — Thread is not terminated 1 — EOT
	30:29	**Reserved** Format: MBZ
	28:0	**Message Descriptor** Exists If: (Structure[EU_INSTRUCTION_SEND][Src1.RegFile]=='IMM') Format: MsgDescpt31
	28:0	**Reg32** Exists If: (Structure[EU_INSTRUCTION_SEND][Src1.RegFile]!='IMM') In a send or sendc instruction refers to the option of providing the message descriptor field DWord, of which bits 28:0 are used, in the first two words of the Address Register rather than as an immediate operand.

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1

Source: EuIsa
Size (in bits): 25
Default Value: 0x00000000

DWord	Bit	Description
0	24:21	**Source Vertical Stride** Format: VertStride
	20:18	**Source Width** Format: Width
	17:16	**Source Horizontal Stride** Format: HorzStride
	15	**Source Addressing Mode** Format: AddrMode
	14:13	**Reserved** Exists If: (Property[Source Modifier]=='false') Format: MBZ
	14:13	**Source Modifier** Exists If: (Property[Source Modifier]=='true') Format: SrcMod
	12:10	**Source Address Subregister Number** Exists If: ([Source Addressing Mode]=='Indirect') Format: AddrSubRegNum
	12:5	**Source Register Number** Exists If: ([Source Addressing Mode]=='Direct') Format: SrcRegNum
	9:0	**Source Address Immediate** Exists If: ([Source Addressing Mode]=='Indirect') Format: S9
	4:0	**Source Subregister Number** Exists If: ([Source Addressing Mode]=='Direct') Format: SrcSubRegNum

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16

Source: EuIsa
Size (in bits): 25
Default Value: 0x00000000

DWord	Bit	Description
0	24:21	**Source Vertical Stride** Format: VertStride
	20	**Reserved** Format: MBZ
	19:16	**Source Channel Select[7:4]** Format: ChanSel[4][7:4]
	15	**Source Addressing Mode** Format: AddrMode
	14:13	**Reserved** Exists If: (Property[Source Modifier]=='false') Format: MBZ
	14:13	**Source Modifier** Exists If: (Property[Source Modifier]=='true') Format: SrcMod
	12:10	**Source Address Subregister Number** Exists If: ([Source Addressing Mode]=='Indirect') Format: AddrSubRegNum
	12:5	**Source Register Number** Exists If: ([Source Addressing Mode]=='Direct') Format: SrcRegNum
	9:4	**Source Address Immediate[9:4]** Exists If: ([Source Addressing Mode]=='Indirect') Format: S9[9:4]
	4	**Source Subregister Number[4:4]** Exists If: ([Source Addressing Mode]=='Direct') Format: SrcSubRegNum[4:4]
	3:0	**Source Channel Select[3:0]** Format: ChanSel[4][3:0]

EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC

Source: EuIsa
Size (in bits): 20
Default Value: 0x00000000

DWord	Bit	Description
0	19:12	**Source Register Number** Format: SrcRegNum
	11:9	**Source Subregister Number [4:2]** Format: SrcSubRegNum[4:2]
	8:1	**Source Swizzle** Format: ChanSel[4]
	0	**Source Replicate Control** Format: RepCtrl

EU_INSTRUCTION_SEND

Source: EuIsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:96	**Message** Format: EU_INSTRUCTION_OPERAND_SEND_MSG
	95:89	**Flags** Format: EU_INSTRUCTION_FLAGS
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16
	63:32	**Operand Control** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:28	**Controls B** Format: EU_INSTRUCTION_CONTROLS_B
	27:24	**Shared Function ID (SFID)** Format: SFID
	23:8	**Controls A** Format: EU_INSTRUCTION_CONTROLS_A
	7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_SOURCES_IMM32

Source:	EuIsa
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Single source, immediate (32-bit)

DWord	Bit	Description
0..1	63:32	**Source 0 Immediate**
	31:25	**Flags** Format: EU_INSTRUCTION_FLAGS
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A]AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16

EU_INSTRUCTION_SOURCES_REG_IMM

Source:	EuIsa
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Dual sources, one register, one immediate

DWord	Bit	Description
0..1	63:32	**Source 1 Immediate**
	31:25	**Flags** Format: EU_INSTRUCTION_FLAGS
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16

EU_INSTRUCTION_SOURCES_REG

Source:	EuIsa
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Single source, register

DWord	Bit	Description
0..1	63:32	**Reserved**
	31:25	**Flags** Format: EU_INSTRUCTION_FLAGS
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16

EU_INSTRUCTION_SOURCES_REG_REG

Source:	EuIsa
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Dual sources, both are registers

DWord	Bit	Description
0..1	63:57	**Reserved** Format: MBZ
	56:32	**Source 1** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16
	56:32	**Source 1** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	31:25	**Flags** Format: EU_INSTRUCTION_FLAGS
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1

ExtMsgDescpt

Source: EuIsa
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:16	**Extended Function Control** Format: U16 This field is intended to control the target function unit. Refer to the section on the specific target function unit for details on the contents of this field.
	15:6	**Reserved** Format: MBZ
	5	**EOT** Format: U1 This field, if set, indicates that this is the final message of the thread and the thread's resources can be reclaimed. Value: 0 — Name: No Termination Value: 1 — Name: EOT
	4	**Reserved** Format: MBZ
	3:0	**Target Function ID** Format: U4 If set, indicates that the message includes a header. Depending on the target shared function, this field may be restricted to either enabled or disabled. Refer to the specific shared function section for details. Value: 0000b — Name: Null Value: 0001b — Name: Reserved Value: 0010b — Name: SamplingEngine Value: 0011b — Name: MessageGateway Value: 0100b — Name: DataPortSamplerCache Value: 0101b — Name: DataPortRenderCache Value: 0110b — Name: URB Value: 0111b — Name: ThreadSpawner Value: 1000b — Name: VideoMotionEstimation Value: 1001b — Name: ConstantCache Value: 1010b-1111b — Name: Reserved

FrameDeltaQp

Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:56	**FrameDeltaQp[7]** Format: S7
	55:48	**FrameDeltaQp[6]** Format: S7
	47:40	**FrameDeltaQp[5]** Format: S7
	39:32	**FrameDeltaQp[4]** Format: S7
	31:24	**FrameDeltaQp[3]** Format: S7
	23:16	**FrameDeltaQp[2]** Format: S7
	15:8	**FrameDeltaQp[1]** Format: S7
	7:0	**FrameDeltaQp[0]** Format: S7

FrameDeltaQpRange

Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:56	**FrameDeltaQpRange[7]** Format: U8
	55:48	**FrameDeltaQpRange[6]** Format: U8
	47:40	**FrameDeltaQpRange[5]** Format: U8
	39:32	**FrameDeltaQpRange[4]** Format: U8
	31:24	**FrameDeltaQpRange[3]** Format: U8
	23:16	**FrameDeltaQpRange[2]** Format: U8
	15:8	**FrameDeltaQpRange[1]** Format: U8
	7:0	**FrameDeltaQpRange[0]** Format: U8

FunctionControl

Source: EuIsa
Size (in bits): 6
Default Value: 0x00000000

DWord	Bit	Description
0	5:4	**Reserved**
	3:0	**Target Function ID**

Value	Name
0000b	Reserved
0001b	INV (Reciprocal)
0010b	LOG
0011b	EXP
0100b	SQRT
0101b	RSQ
0110b	SIN
0111b	COS
1000b	Reserved
1001b	FDIV
1010b	POW
1011b	INT DIV Quotient and remainder
1100b	INT DIV Quotient only
1101b	INT DIV Remainder only
1110b-1111b	Reserved

Hardware-Detected Error Bit Definitions

Source: RenderCS

Size (in bits): 32

Default Value: 0x00000000

DWord	Bit	Description
0	31:3	**Reserved** Format: MBZ
	2	**Reserved** Format: MBZ
	1	**Reserved** Format: MBZ
	0	**Instruction Error** This bit is set when the Renderer Instruction Parser detects an error while parsing an instruction. Instruction errors include: • Client ID value (Bits 31:29 of the Header) is not supported (only MI, 2D and 3D are supported). • Defeatured MI Instruction Opcodes: **Value** | **Name** | **Description** 1 | | Instruction Error detected **Programming Notes** This error indications cannot be cleared except by reset (i.e., it is a fatal error).

Hardware Status Page Layout

Source:	RenderCS, VideoCS, BlitterCS
Size (in bits):	32768
Default Value:	All bits zero

DWord	Bit	Description
0	31:0	**Interrupt Status Register Storage** The content of the ISR register is written to this location whenever an "unmasked" bit of the ISR (as determined by the HWSTAM register) changes state.
1..3	31:0	**Reserved** Must not be used.
4	31:0	**Ring Head Pointer Storage** The contents of the Ring Buffer Head Pointer register (register DWord 1) are written to this location either as result of an MI_REPORT_HEAD instruction or as the result of an "automatic report" (see RINGBUF registers).
5..15	31:0	**Reserved** Must not be used.
16..27	31:0	**Reserved**
28..30	31:0	**Reserved** Must not be used.
31	31:0	**Reserved**
32..39	31:0	**Reserved**
40..46	31:0	**Reserved**
47	31:0	**Reserved**
48..1023	31:0	**General Purpose** These locations can be used for general purpose via the MI_STORE_DATA_INDEX or MI_STORE_DATA_IMM instructions.

Inline Data Description for MFD_AVC_BSD_Object

Source: VideoCS

Size (in bits): 64

Default Value: 0x00000000, 0x00000000

This structure includes all the required Slice Header parameters and error handling settings for AVC_BSD_OBJECT Command (DW3..DW5).

DWord	Bit	Description
0	31	**Concealment Method** This field specifies the method used for concealment when error is detected. If set, a copy from collocated macroblock location is performed from the concealment reference indicated by the ConCeal_Pic_Id field. If it is not set, a copy from the current picture is performed using Intra 16x16 Prediction method. Value / Name / Description: 0 — Intra 16x16 Prediction 1 — Inter P Copy
	30	**Init Current MB Number** When set, the current Slice_Start_MB_Num, Slice_MB_Start_Hor_Pos and Slice_MB_Start_Vert_Pos fields will be used to initialize the Current_MB_Number register. This effectively disables the concealment capability.
	29	**Reserved** Format: MBZ
	28:27	**MB Error Concealment B Temporal Prediction mode** These two bits control how the reference L0/L1 are overridden in B temporal slice. Value / Name / Description: 00b — **[Default]** — Both Reference Indexes L0/L1 are forced to 0 during Concealment 01b — Only Reference Index L1 is forced to 0; Reference Index L0 is forced to -1 10b — Only Reference Index L0 is forced to 0; Reference Index L1 is forced to -1 11b — Reserved — Invalid
	26	**MB Error Concealment B Temporal Reference Index Override Enable Flag** During MB Error Concealment on B slice with Temporal Direct Prediction, either L0 or L1 or both can be forced to 0 (MB Error Concealment B Temporal Reference Index Override Mode from above will control which one) This bit can be set to use the predicted reference indexes instead. Value / Name / Description: 0 — **[Default]** — Predicted Reference Indexes L0/L1 are used during MB Concealment. 1 — Reference Indexes L0/L1 are overridden to 0 during MB Concealment.
	25	**MB Error Concealment B Temporal Motion Vectors Override Enable Flag** During MB Error Concealment on B slice with Temporal Direct Prediction, motion vectors are forced to 0 to improve image quality. This bit can be set to preserve the original weight prediction. Value / Name / Description:

Inline Data Description for MFD_AVC_BSD_Object

0	**[Default]**	Predicted Motion Vectors are used during MB Concealment
1		Motion Vectors are Overridden to 0 during MB Concealment

24 — MB Error Concealment B Temporal Weight Prediction Disable Flag

During MB Error Concealment on B slice with Temporal Direct Prediction, weight prediction is disabled to improve image quality.

This bit can be set to preserve the original weight prediction.

Value	Name	Description
0	**[Default]**	Weight Prediction is Disabled during MB Concealment
1		Weight Prediction will not be overridden during MB Concealment

23:22 — Reserved

Format:	MBZ

21:16 — Concealment Picture ID

This field identifies the picture in the reference list to be used for concealment. This field is only valid if **Concealment Method** is Inter P Copy.

Bit Filed	Value	Defenition
21	0	Frame Picture
21	1	Field picture
20:16		Frame Store Index[4:0]

15 — Reserved

Format:	MBZ

14 — BSD Premature Complete Error Handling

BSD Premature Complete Error occurs in situation where the Slice decode is completed but there are still data in the bitstream.

Value	Name	Description
1		Set the interrupt to the driver (provide MMIO registers for MB address R/W)
0		Ignore the error and continue (masked the interrupt), assume the hardware automatically performs the error handling

13 — Reserved

Format:	MBZ

12 — MPR Error (MV out of range) Handling

Software must follow the action for each Value as follow:

Value	Name	Description
1		Set the interrupt to the driver (provide MMIO registers for MB address R/W)
0		Ignore the error and continue (masked the interrupt), assume the hardware automatically performs the error handling

11 — Reserved

Format:	MBZ

10 — Entropy Error Handling

Inline Data Description for MFD_AVC_BSD_Object

Software must follow the action for each Value as follow:

Value	Name	Description
1		Set the interrupt to the driver (provide MMIO registers for MB address R/W).
0		Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling.

9 **Reserved**

Format:	MBZ

8 **MB Header Error Handling**

Software must follow the action for each Value as follow:

Value	Name	Description
1		Set the interrupt to the driver (provide MMIO registers for MB address R/W).
0		Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error concealment.

7:6 **MB Error Concealment B Spatial Prediction mode**

These two bits control how the reference L0/L1 are overridden in B spatial slice.

Value	Name	Description
00b	**[Default]**	Both Reference Indexes L0/L1 are forced to 0 during Concealment
01b		Only Reference Index L1 is forced to 0; Reference Index L0 is forced to -1
10b		Only Reference Index L0 is forced to 0; Reference Index L1 is forced to -1
11b	Reserved	Invalid

5 **MB Error Concealment B Spatial Reference Index Override Disable Flag**

During MB Error Concealment on B slice with Spatial Direct Prediction, either L0 or L1 or both can be forced to 0 (MB Error Concealment B Spatial Reference Index Override Mode from above will control which one)

This bit can be set to use the predicted reference indexes instead.

Value	Name	Description
0	**[Default]**	Reference Indexes L0/L1 are overridden during MB Concealment
1		Predicted Reference Indexes L0/L1 are used during MB Concealment

4 **MB Error Concealment B Spatial Motion Vectors Override Disable Flag**

During MB Error Concealment on B slice with Spatial Direct Prediction, motion vectors are forced to 0 to improve image quality.

This bit can be set to use the predicted motion vectors instead.

This bit does not affect normal decoded MB.

Value	Name	Description
0	**[Default]**	Motion Vectors are Overridden to 0 during MB Concealment
1		Predicted Motion Vectors are used during MB Concealment

3 **MB Error Concealment B Spatial Weight Prediction Disable Flag**

During MB Error Concealment on B slice with Spatial Direct Prediction, weight prediction is disabled to improve image quality.

Inline Data Description for MFD_AVC_BSD_Object

This bit can be set to preserve the original weight prediction.
This bit does not affect normal decoded MB.

Value	Name	Description
0	**[Default]**	Weight Prediction is Disabled during MB Concealment.
1		Weight Prediction will not be overridden during MB Concealment.

2 — MB Error Concealment P Slice Reference Index Override Disable Flag

During MB Error Concealment on P slice reference index L0 is forced to 0.
This bit can be set to use the predicted reference indexes instead.
This bit does not affect normal decoded MB.

Value	Name	Description
0	**[Default]**	Reference Indexes L0 are force to 0
1		Predicted Reference Indexes L0 are used during MB Concealment.

1 — MB Error Concealment P Slice Motion Vectors Override Disable Flag

During MB Error Concealment on P slice, motion vectors are forced to 0 to improve image quality.
This bit can be set to use the predicted motion vectors instead.
This bit does not affect normal decoded MB.

Value	Name	Description
0	**[Default]**	Motion Vectors are Overridden to 0 during MB Concealment
1		Predicted Motion Vectors are used during MB Concealment

0 — MB Error Concealment P Slice Weight Prediction Disable Flag

During MB Error Concealment on P slice, weight prediction is disabled to improve image quality.
This bit can be set to preserve the original weight prediction.
This bit does not affect normal decoded MB.

Value	Name	Description
0	**[Default]**	Weight Prediction is Disabled during MB Concealment.
1		Weight Prediction will not be overridden during MB Concealment.

1 — 31:16 First MB Byte Offset of Slice Data or Slice Header

Description
Long Format: It gives the byte offset to locate the Slice Header in the bitstream for a slice, provided by the Indirect BSD Data Start Address. It does not include any Emulation Byte count present in the Slice Header. HW will take care of the Emulation Byte adjustment to this offset.
Short Format:it should be programmed to be 0. HW will parse the Slice Header.

Programming Notes
MFX supports only DXVA2 Long and Short Format.

15:8 Reserved

Format:	MBZ

7 Fix Prev Mb Skipped

Inline Data Description for MFD_AVC_BSD_Object

		Enables an alternative method for decoding mb_skipped, to cope with an encoder that codes a skipped MB as a direct MB with no coefficient.
	6:5	**Reserved** Format: MBZ **Programming Notes** Please note that the field MUST be set to '0' at this time.
	4	**Emulation Prevention Byte Present** Value / Name / Description 0 / / H/W needs to perform Emulation Byte Removal 1 / / H/W does not need to perform Emulation Byte Removal
	3	**LastSlice Flag** It is needed for both error concealment at the end of a picture. It is also needed to know to set the last MB in a picture correctly. Value / Name / Description 1 / / If the current Slice to be decoded is the very last slice of the current picture. 0 / / If the current Slice to be decoded is any slice other than the very last slice of the current picture
	2:0	**First Macroblock (MB)Bit Offset** Exists If: //AVC Long Format Only Format: U3 This field provides the bit offset of the first macroblock of the Slice in the first byte of the input compressed bitstream.

INTERFACE_DESCRIPTOR_DATA

Source: RenderCS

Size (in bits): 256

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:6	**Kernel Start Pointer** Format: InstructionBaseOffset[31:6]Kernel Specifies the 64-byte aligned address offset of the first instruction in the kernel. This pointer is relative to the **Instruction Base Address**.
	5:0	**Reserved** Format: MBZ
1	31:26	**Reserved** Format: MBZ
	25:20	**Reserved** Format: MBZ
	19	**Reserved** Format: MBZ
	18	**Single Program Flow (SPF)** Specifies whether the kernel program has a single program flow (SIMDnxm with m = 1) or multiple program flows (SIMDnxm with m > 1). Value: 0h — Multiple Program Flow Value: 1h — Single Program Flow
	17	**Thread Priority** Specifies the priority of the thread for dispatch Value: 0h — Normal Priority Value: 1h — High Priority
	16	**Floating Point Mode** Specifies the floating point mode used by the dispatched thread. Value: 0h — Use IEEE-754 Rules Value: 1h — Use alternate rules
	15:14	**Reserved** Format: MBZ
	13	**Illegal Opcode Exception Enable** Format: Enable

INTERFACE_DESCRIPTOR_DATA

		This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.
	12	**Reserved** Format: MBZ
	11	**MaskStack Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[11]. See Exceptions and ISA Execution Environment.
	10:8	**Reserved** Format: MBZ
	7	**Software Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[13] (note the bit # difference). See Exceptions and ISA Execution Environment.
	6:0	**Reserved** Format: MBZ
2	31:5	**Sampler State Pointer** Format: DynamicStateOffset[31:5]SAMPLER_STATE Specifies the 32-byte aligned address offset of the sampler state table. This pointer is relative to the Dynamic State Base Address. This field is ignored for child threads.
	4:2	**Sampler Count** Specifies how many samplers (in multiples of 4) the kernel uses. Used only for prefetching the associated sampler state entries. This field is ignored for child threads. If this field is not zero, sampler state is prefetched for the first instance of a root thread upon the startup of the media pipeline.
	1:0	**Reserved** Format: MBZ
3	31:5	**Binding Table Pointer** Format: SurfaceStateOffset[31:5]BINDING_TABLE_STATE*256

Value	Name
[0,4]	
0h	No samplers used
1h	Between 1 and 4 samplers used
2h	Between 5 and 8 samplers used
3h	Between 9 and 12 samplers used
4h	Between 13 and 16 samplers used

INTERFACE_DESCRIPTOR_DATA

Specifies the 32-byte aligned address of the binding table. This pointer is relative to the **Surface State Base Address**.
This field is ignored for child threads.

4:0 Binding Table Entry Count

Format:	U5

Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state.
This field is ignored for child threads.
If this field is not zero, binding table and surface state are prefetched for the first instance of a root thread upon the startup of the media pipeline.

Programming Notes
The maximum number of prefetched binding table entries is limited to 31. For kernels using a large number of binding table entries, it may be wise to set this field to zero to avoid prefetching too many entries and thrashing the state cache.

4 31:16 Constant URB Entry Read Length

Format:	U16

Specifies the amount of URB data read and passed in the thread payload for the Constant URB entry, in 8-DW register increments. A value 0 means that no Constant URB Entry will be loaded. The Constant URB Entry Read Offset field will then be ignored. In GPGPU mode this describes how much data is delivered in a single dispatch. Multiple dispatches in a thread group will deliver constant data offset by this value. The total amount of constant data is (Constant URB Read Length * Number of Threads in GPGPU Thread Group).

Value	Name
[0,63]	

15:0 Constant URB Entry Read Offset

Format:	U16

Specifies the offset (in 8-DW units) at which Constant URB data is to be read from the URB before being included in the thread payload.

Value	Name	Description
[0,2015]		Indicating [0,2015] 256-bit register increments. ROB has 64KB of storage; 2048 entries. However, lowest 32 entries are reserved for VFE/TS to store interface descriptor data. Hence, URB Entry Read Offset plus Read Length shall not exceed 2016.

5 31:24 Reserved

Format:	MBZ

23:22 Rounding Mode

Value	Name	Description
00b	RTNE **[Default]**	Round to Nearest Even

INTERFACE_DESCRIPTOR_DATA

		01b: RU — Round toward +Infinity 10b: RD — Round toward -Infinity 11b: RTZ — Round to toward Zero
	21	**Barrier Enable** Format: Enable This field specifies whether the thread group requires a barrier. If not, it can be dispatched without allocating one.
	20:16	**Shared Local Memory Size** Format: U5 This field indicates how much shared local memory the thread group requires. The amount is specified in 4k blocks from 0k up to the maximum of 64k per half-slice. Value: [0,16] — Name: — Description: encodes 0k to 64k
	15:8	**Reserved** Format: MBZ
	7:0	**Number of Threads in GPGPU Thread Group** Format: U8 Specifies the number of threads that are in this thread group. Used to program the barrier for the number of messages to expect. MBZ for threads that do not use a barrier. Value: [0,16] — Name: Value: [0,16] — Name: Value: [0,31] — Name:
6	31:0	**Reserved** Format: MBZ
7	31:0	**Reserved** Format: MBZ

INTERFACE_DESCRIPTOR_DATA

Source: RenderCS

Size (in bits): 256

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:6	**Kernel Start Pointer** Format: InstructionBaseOffset[31:6]Kernel Specifies the 64-byte aligned address offset of the first instruction in the kernel. This pointer is relative to the **Instruction Base Address.**
	5:0	**Reserved** Format: MBZ
1	31:19	**Reserved** Format: MBZ
	18	**Single Program Flow (SPF)** Specifies whether the kernel program has a single program flow (SIMDnxm with m = 1) or multiple program flows (SIMDnxm with m > 1). Value: 0h — Name: Multiple Program Flow Value: 1h — Name: Single Program Flow
	17	**Thread Priority** Specifies the priority of the thread for dispatch. Value: 0h — Name: Normal Priority Value: 1h — Name: High Priority
	16	**Floating Point Mode** Specifies the floating point mode used by the dispatched thread. Value: 0h — Name: Use IEEE-754 Rules Value: 1h — Name: Use alternate rules
	15:14	**Reserved** Format: MBZ
	13	**Illegal Opcode Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[12] (note the bit # difference). See *Exceptions and ISA Execution Environment*.
	12	**Reserved**

INTERFACE_DESCRIPTOR_DATA

		Format: MBZ
	11	**MaskStack Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[11]. See *Exceptions and ISA Execution Environment*.
	10:8	**Reserved** Format: MBZ
	7	**Software Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[13] (note the bit # difference). See *Exceptions and ISA Execution Environment*.
	6:0	**Reserved** Format: MBZ
2	31:5	**Sampler State Pointer** Format: DynamicStateOffset[31:5]SAMPLER_STATE Specifies the 32-byte aligned address offset of the sampler state table. This pointer is relative to the **Dynamic State Base Address.** *This field is ignored for child threads.*
	4:2	**Sampler Count** Format: U3 Specifies how many samplers (in multiples of 4) the kernel uses. Used only for prefetching the associated sampler state entries. *This field is ignored for child threads.* *If this field is not zero, sampler state is prefetched for the first instance of a root thread upon the startup of the media pipeline.*
	1:0	**Reserved** Format: MBZ
3	31:16	**Reserved** Format: MBZ
	15:5	**Binding Table Pointer**

Value	Name
[0,4]	
0h	No samplers used
1h	Between 1 and 4 samplers used
2h	Between 5 and 8 samplers used
3h	Between 9 and 12 samplers used
4h	Between 13 and 16 samplers used

INTERFACE_DESCRIPTOR_DATA

		Format: SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 Specifies the 32-byte aligned address of the binding table. This pointer is relative to the **Surface State Base Address.** *This field is ignored for child threads.*
	4:0	**Binding Table Entry Count** Format: U5 Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state. **Note:** The maximum number of prefetched binding table entries is limited to 31. For kernels using a large number of binding table entries, it may be wise to set this field to zero to avoid prefetching too many entries and thrashing the state cache. *This field is ignored for child threads.* *If this field is not zero, binding table and surface state are prefetched for the first instance of a root thread upon the startup of the media pipeline.* Value: [0,31] — Name:
4	31:16	**Constant URB Entry Read Length** Format: U16 Description: Specifies the amount of URB data read and passed in the thread payload for the Constant or Indirect URB entry, in 8-DW register increments. A value 0 means that no Constant or Indirect URB Entry will be loaded. The Constant URB Entry Read Offset field will then be ignored. In GPGPU mode this describes how much data is delivered in a single dispatch. Multiple dispatches in a thread group will deliver constant data offset by this value. The total amount of constant data is (Constant URB Read Length * Number of Threads in GPGPU Thread Group). Value: [0,63] — Name:
	15:0	**Constant URB Entry Read Offset** Format: U16 Specifies the offset (in 8-DW units) at which Constant URB data is to be read from the URB before being included in the thread payload. Value: [0,2015] — Name: — Description: indicating [0,2015] 256-bit register increments. ROB has 64KB of storage; 2048 entries. However, lowest 32 entries are reserved for VFE/TS to store interface descriptor data. Hence, URB Entry Read Offset plus Read Length shall not exceed 2016.
5	31:24	**Reserved**

INTERFACE_DESCRIPTOR_DATA

	Format:	MBZ

31:24 Barrier Return GRF Offset

Format:	U8

This field specifies the offset into the GRF that the barrier return byte will be written to.

Value	Name
0,127	

23:22 Rounding Mode

Format:	U2

Value	Name	Description
00b	RTNE **[Default]**	Round to Nearest Even
01b	RU	Round toward +Infinity
10b	RD	Round toward -Infinity
11b	RTZ	Round toward Zero

21 Barrier Enable

Format:	Enable

This field specifies whether the thread group requires a barrier. If not, it can be dispatched without allocating one.

20:16 Shared Local Memory Size

Format:	U5

This field indicates how much shared local memory the thread group requires. The amount is specified in 4k blocks, but only powers of 2 are allowed: 0, 4k, 8k, 16k, 32k and 64k per half-slice.

Value	Name	Description
[0,16]		Encodes 0k to 64k in powers of 2

15:8 Reserved

Format:	MBZ

15:8 Barrier Return Byte

Format:	U8

This field specifies the byte that will be returned by the gateway when the barrier is reached.

7:0 Number of Threads in GPGPU Thread Group

Format:	U8

Specifies the number of threads that are in this thread group. Used to program the barrier for the number of messages to expect. The minimum value is 0 (which will disable the barrier), while the maximum value is the number of threads in a subslice for local barriers. See Configurations chapter for the number of threads per subslice for different products.

Value	Name
[0,64]	

INTERFACE_DESCRIPTOR_DATA			
		[0,16]	
		[0,31]	
6	31:0	**Reserved**	
		Format:	MBZ
7	31:0	**Reserved**	
		Format:	MBZ

JPEG

Source: VideoCS

Size (in bits): 16

Default Value: 0x00000000

DWord	Bit	Description
0	15:5	**Reserved** Format: MBZ
	4	**Inconsistent VLD SE Error** This flag indicates an inconsistent SE coded in the bit-stream. Bit-stream does not match any entries in the hauffman table.
	3	**Extra Block Error** This flag indicates extra block coded within an ECS data boundary.
	2	**Missing block Error** This flag indicates one or more blocks are missing within an ECS data boundary.
	1	**Extra ECS Error** This flag indicates extra ECS' coded in the bit-stream SCAN payload data.
	0	**Missing ECS Error** This flag indicates one or more ECS' are missing from the bit-stream SCAN payload data.

MEDIA_SURFACE_STATE

Source:	BSpec
Exists If:	//([MessageType] == 'Deinterlace') OR ([MessageType] == 'Sample_8x8')
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

This is the SURFACE_STATE used by only deinterlace, sample_8x8, and VME messages.

DWord	Bit	Description
0	31:0	**Surface Base Address** Format: GraphicsAddress[31:0] Specifies the byte-aligned base address of the surface **Programming Notes** For SURFTYPE_BUFFER render targets, this field specifies the base address of first element of the surface. The surface is interpreted as a simple array of that single element type. The address must be naturally-aligned to the element size (e.g., a buffer containing R32G32B32A32_FLOAT elements must be 16-byte aligned). For SURFTYPE_BUFFER non-rendertarget surfaces, this field specifies the base address of the first element of the surface, computed in software by adding the surface base address to the byte offset of the element in the buffer. Mipmapped, cube and 3D sampling engine surfaces are stored in a "monolithic" (fixed) format, and only require a single address for the base texture. Linear render target surface base addresses must be element-size aligned, for non-YUV surface formats, or a multiple of 2 element-sizes for YUV surface formats. Other linear surfaces have no alignment requirements (byte alignment is sufficient.)Linear depth buffer surface base addresses must be 64-byte aligned. Note that while render targets (color) can be SURFTYPE_BUFFER, depth buffers cannot.Tiled surface base addresses must be 4KB-aligned. Note that only the offsets from Surface Base Address are tiled, Surface Base Address itself is not transformed using the tiling algorithm. Tiled surface base addresses must be 4KB-aligned. Note that only the offsets from Surface Base Address are tiled, Surface Base Address itself is not transformed using the tiling algorithm. For tiled surfaces, the actual start of the surface can be offset from the Surface Base Address by the X Offset and Y Offset fields. Certain message types used to access surfaces have more stringent alignment requirements. Please refer to the specific message documentation for additional restrictions.
1	31:18	**Height** Format: U14 This field specifies the height of the surface in units of pixels. For PLANAR surface formats, this field indicates the height of the Y (luma) plane. Value / Name / Description: [0,16383] / / representing heights [1,16384] **Programming Notes** Height (field value + 1) must be a multiple of 2 for PLANAR_420 surfaces. If Vertical Line Stride is 1, this field indicates the height of the field, not the height of the frame.
	17:4	**Width** Format: U14

MEDIA_SURFACE_STATE

This field specifies the width of the surface in units of pixels. For PLANAR surface formats, this field indicates the width of the Y (luma) plane.

Value	Name	Description
[0,16383]		representing widths [1,16384]

Programming Notes

- The Width specified by this field multiplied by the pixel size in bytes must be less than or equal to the surface pitch (specified in bytes via the Surface Pitch field).
- Width (field value + 1) must be a multiple of 2 for PLANAR_420, PLANAR_422, and all YCRCB_* and Y16_UNORM surfaces, and must be a multiple of 4 for PLANAR_411 and Y8_UNORM_VA surfaces.
- For deinterlace messages, the Width (field value + 1) must be a multiple of 8.

3:2 **Picture Structure**

Specifies the encoding of the current picture.

Value	Name
00b	Frame Picture
01b	Top Field Picture
10b	Bottom Field Picture
11b	Invalid, not allowed

1:0 **Cr(V)/Cb(U) Pixel Offset V Direction**

Format:	U0.2

Specifies the distance to the U/V values with respect to the even numbered Y channels in the V direction

Programming Notes

This field is ignored for all formats except PLANAR_420_8

2 31:28 **Surface Format**

Specifies the format of the surface. All of the Y and G channels will use table 0 and all of the Cr/Cb/R/B channels will use table 1

Value	Name	Description
0	YCRCB_NORMAL	
1	YCRCB_SWAPUVY	
2	YCRCB_SWAPUV	
3	YCRCB_SWAPY	
4	PLANAR_420_8	
5	PLANAR_411_8	Deinterlace only
6	PLANAR_422_8	Deinterlace only
7	STMM_DN_STATISTICS	Deinterlace only
8	R10G10B10A2_UNORM	Sample_8x8 only

MEDIA_SURFACE_STATE

9	R8G8B8A8_UNORM	Sample_8x8 only
10	R8B8_UNORM (CrCb)	Sample_8x8 only
11	R8_UNORM (Cr/Cb)	Sample_8x8 only
12	Y8_UNORM	
15	Reserved	

27 Interleave Chroma

Format:	Enable

This field indicates that the chroma fields are interleaved in a single plane rather than stored as two separate planes. This field is only used for PLANAR surface formats.

26 Reserved

Format:	MBZ

25:22 Surface Object Control State (MEMORY_OBJECT_CONTROL_STATE)

This 4-bit field is used in various state commands and indirect state objects to define cacheability including graphics data type for memory objects.

21 Reserved

Format:	MBZ

20:3 Surface Pitch

Format:	U18-1 pitch in Bytes

This field specifies the surface pitch in (#Bytes - 1).

Value	Name	Description
[0,262143]		For other linear surfaces: representing [1B, 256KB]
[511, 262143]		For X-tiled surface: representing [512B, 256KB] = [1 tile, 512 tiles]
[127, 262143]		For Y-tiled surfaces: representing [128B, 256KB] = [1 tile, 2048 tiles]

Programming Notes

For tiled surfaces, the pitch must be a multiple of the tile widthIf Half Pitch for Chroma is set, this field must be a multiple of two tile widths for tiled surfaces, or a multiple of 2 bytes for linear surfaces. The Surface Pitches of current picture and reference picture should be declared as the identical type in VDI mode with identical Height, Width and Format.

2 Half Pitch for Chroma

Format:	Enable

This field indicates that the chroma plane(s) will use a pitch equal to half the value specified in the Surface Pitch field. This field is only used for PLANAR surface formats.

1:0 Tile Mode

Format:	U2 Enumerated Type

This field specifies the type of memory tiling (Linear, WMajor, XMajor, or YMajor) employed to tile this surface. See Memory Interface Functions for details on memory tiling and restrictions.

MEDIA_SURFACE_STATE

Value	Name	Description
0h	TILEMODE_LINEAR	Linear mode (no tiling)
1h	Reserved	Reserved
2h	TILEMODE_XMAJOR	X major tiling
3h	TILEMODE_YMAJOR	Y major tiling

Programming Notes

- Refer to *Memory Data Formats* for restrictions on TileMode direction for the various buffer types. (Of particular interest is the fact that YMAJOR tiling is not supported for display/overlay buffers).
- The corresponding cache(s) must be invalidated before a previously accessed surface is accessed again with an altered state of this field.
- Linear surfaces can be mapped to Main Memory (uncached) or System Memory (cacheable, snooped). Tiled (X/Y/W) surfaces can only be mapped to Main Memory.

3 | 31:30 | **Reserved**

Format:	MBZ

29:16 | **X Offset for U(Cb)**

Format:	U14 Pixel Offset

Description

For non planar surfaces this field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the surface.

For Planar surfaces this field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave Chroma is enabled.

Programming Notes

For PLANAR_420 and PLANAR_422 surface formats, this field must indicate an even number of pixels.

15:14 | **Reserved**

Format:	MBZ

13:0 | **Y Offset for U(Cb)**

Format:	U14 Row Offset

Description

For non planar surfaces this field specifies the vertical offset in pixels from the Surface Base Address to the start (origin) of the surface.

For Planar surfaces this field specifies the vertical offset in rows from the Surface Base Address to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave

MEDIA_SURFACE_STATE

		Chroma is enabled.
		Programming Notes This field must indicate an even number (bit [0] = 0)
4	31:30	**Reserved** Format: MBZ
	29:16	**X Offset for V(Cr)** Exists If: //([Surface Format] is one of planar) AND ([Interleave Chroma] == '0') Format: U14 Pixel Offset **Description** This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the V(Cr) plane. **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must indicate an even number of pixels.
	15	**Reserved** Format: MBZ
	14:0	**Y Offset for V(Cr)** Exists If: //([Surface Format] is one of planar) AND ([Interleave Chroma] == '0') Format: U15 Row Offset **Description** This field specifies the veritical offset in rows from the Surface Base Address to the start (origin) of the V(Cr) plane. **Value** 0,16380 — **Name** **Programming Notes** This field must indicate an even number (bit 0 = 0).
5	31:30	**Reserved** Format: MBZ
	29:20	**Reserved** Format: MBZ
	19:18	**Reserved** Format: MBZ

MEDIA_SURFACE_STATE		
	17:7	**Reserved** Format: MBZ
	6:0	**Reserved** Format: MBZ
6	31:0	**Reserved** Format: MBZ
7	31:16	**Reserved** Format: MBZ
	15:0	**Reserved** Format: MBZ

MEMORY_OBJECT_CONTROL_STATE

Source: BSpec
Size (in bits): 4
Default Value: 0x00000000

DWord	Bit	Description
0	3	**Reserved**
	2:1	**Coherency Control (when GT is a non-caching agent)** This field controls coherency with the CPU core caches. Bit[2] is used for snooping the processor L2 vs not snooping. The snoops should only be enabled for the surfaces that are prepared by driver (IA core) and defined as Write-back (WB) for the memory type in processor's memory map. Snooping accesses will have reduced performance due to longer latency to get the content from the CPU caches rather than main memory. **Value** / **Name** 0xb: Data in this page may not be coherent with CPU caches, h/w uses GTT entry to decide 1xb: Data accesses in this page must be snooped in the CPU caches
	0	**L3 Cacheability Control (L3CC)** This field is used to control the L3 cacheability (allocation) of the stream. Note: even if the surface is not cacheable in L3, it is still kept coherent with L3 content. **Value** / **Name** 0h: Not cacheable within L3 1h: Cacheable in L3

Message Descriptor - Render Target Write

Source: BSpec
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31	**Reserved** Format: MBZ
	30	**Reserved** Format: MBZ
	29:14	**Reserved** Format: MBZ
	13	**Reserved** Format: MBZ
	12	**Last Render Target Select** This bit must be set on the last render target write message sent for each group of pixels. For single render target pixel shaders, this bit is set on all render target write messages. For multiple render target pixel shaders, this bit is set only on messages sent to the last render target. This bit must be zero for SIMD8 Image Write message. **Programming Notes** In general, when threads are not launched by 3D FF, this bit must be zero.
	11	**Slot Group Select** This field selects whether slots 15:0 or slots 31:16 are used for bypassed data. Bypassed data includes the antialias alpha, multisample coverage mask, and if the header is not present also includes the X/Y addresses and pixel enables. For 8- and 16-pixel dispatches, SLOTGRP_LO must be selected on every message. For 32-pixel dispatches, this field must be set correctly for each message based on which slots are currently being processed. **Value / Name / Description** 0 / SLOTGRP_LO / choose bypassed data for slots 15:0 1 / SLOTGRP_HI / choose bypassed data for slots 31:16 **Programming Notes** For SIMD8 Image Write message thsi field MBZ.
	10:8	**Message Type** This field specifies the type of render target message. For the SIMD8_DUALSRC_xx messages, the low bit indicates which slots to use for the pixel enables, X/Y addresses, and oMask. **Value / Name / Description** 000b / SIMD16 / SIMD16 single source message 001b / SIMD16_REPDATA / SIMD16 single source message with replicated data 010b / SIMD8_DUALSRC_LO / SIMD8 dual source message, use slots 7:0

Message Descriptor - Render Target Write

011b	SIMD8_DUALSRC_HI	SIMD8 dual source message, use slots 15:8
100b	SIMD8_LO	SIMD8 single source message, use slots 7:0
111b	SIMD16_REPDATA	It's only supported when accessing *Tiled Memory*. Using this Message Type to access linear *(Untiled)* memory is UNDEFINED.

Programming Notes
the above slots indicated are within the 16 slots selected by **Slot Group Select**. If SLOTGRP_HI is selected, the SIMD8 message types above reference slots 23:16 or 31:24 instead of 7:0 or 15:8, respectively.
SIMD16 messages are not supported for 8X MSAA when PS outputs depth.
SIMD16_REPDATA message must not be used in SIMD8 pixel-shaders.

7:0 **Reserved**

Format:	MBZ

MFD_MPEG2_BSD_OBJECT Inline Data Description

Source: VideoCS

Size (in bits): 64

Default Value: 0x00000000, 0x00000000

DW0..1 corresponds to DW3..4 of the MFD_MPEG2_BSD_OBJECT.

DWord	Bit	Description
0	31	**Reserved** Format: MBZ
	30:24	**Slice Horizontal Position** Format: U7 in Macroblocks This field indicates the horizontal position (in macroblock units) of the first macroblock in the slice.
	23	**Reserved** Format: MBZ
	22:16	**Slice Vertical Position** Format: U7 in Macroblocks This field indicates the vertical position (in macroblock units) of the first macroblock in the slice.
	15	**Reserved** Format: MBZ
	14:8	**Macroblock Count** Format: U7 in Macroblocks This field indicates the number of macroblocks in the slice, including skipped macroblocks.
	7:6	**Reserved** Format: MBZ
	5	**Last Pic Slice** This bit is added to support error concealment at the end of a picture. Value / Name / Description: 1h — The current Slice is the last Slice of the entire picture 0h — The current Slice is not the last Slice of current picture
	4	**Reserved**
	3	**Is Last MB** Value / Name / Description: 1h — The current MB is the last MB in the current Slice 0h — The current MB is not the last MB in the current Slice
	2:0	**First Macroblock Bit Offset** Format: U3

MFD_MPEG2_BSD_OBJECT Inline Data Description

		This field provides the bit offset of the first macroblock in the first byte of the input bitstream.
1	31:29	**Reserved** Format: MBZ
	28:24	**Quantizer Scale Code** Format: U5 This field sets the quantizer scale code of the inverse quantizer. It remains in effect until changed by a decoded quantizer scale code in a macroblock. This field is decoded from the slice header by host software.
	23:0	**Reserved** Format: MBZ

MPEG2

Source: VideoCS
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15:6	**Reserved** Format: MBZ
	5	**Missing EOB Error** This flag indicates missing EOB SEs coded in the bit-stream. Missing EOBs are concealed to match CBP of the error MB.
	4	**Inconsistent starting position Error - overlapping MBs** This flag indicates two slices overlapping one another by one or more MBs. Duplicate MBs decoded off the second slice shall be discarded.
	3	**Slice out-of-bound Error** This flag indicates a slice is running beyond the width of the picture. Out-of-bound MBs shall be discarded.
	2	**Premature frame end Error** This flag indicates missing slices/MBs coded in the bit-stream of a frame. One or more MBs are concealed to reach end of picture.
	1	**Inconsistent starting position Error - Missing MBs** This flag indicates one or more MBs are being concealed due to inconsistent MB starting and ending positions between slices.
	0	**MB Concealment Flag** . Each pulse from this flag indicates one MB is concealed by hardware.

MsgDescpt31

Source: EuIsa
Size (in bits): 29
Default Value: 0x00000000

DWord	Bit	Description
0	28:25	**Message Length** This field specifies the number of 256-bit MRF registers starting from <curr_dest> to be sent out on the request message payload. Valid value ranges from 1 to 15. A value of 0 is considered erroneous. Value: 1-15 — Name: Number of MRF Registers
	24:20	**Response Length** This field indicates the number of 256-bit registers expected in the message response. The valid value ranges from 0 to 16. A value 0 indicates that the request message does not expect any response. The largest response supported is 16 GRF registers. Value: 0-16 — Name: Number of Registers
	19	**Header Present** Format: Enable If set, indicates that the message includes a header. Depending on the target shared function, this field may be restricted to either enabled or disabled. Refer to the specific shared function section for details.
	18:0	**Function Control** This field is intended to control the target function unit. Refer to the section on the specific target function unit for details on the contents of this field.

MDP_RTW_M16REP - OM Replicated SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	512
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
1.0-1.7	255:0	**RGBA** Format: MDPR_RGBA RGBA for all slots [15:0]

MDP_RTW_MA8 - OM S0A SIMD8 Render Target Data Payload

Source:	BSpec
Size (in bits):	1536
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**oMask** Format: MDPR_OMASK Slots [7:0] oMask. Upper half ignored.
2.0-2.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha

MDP_RTW_MA16 - OM S0A SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	2816
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Source 0 Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source 0 Alpha
2.0-2.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
3.0-3.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
4.0-4.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
5.0-5.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8

MDP_RTW_MA16 - OM S0A SIMD16 Render Target Data Payload

		Slots [7:0] Green
6.0-6.7	255:0	**Green[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Green
7.0-7.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Blue
8.0-8.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Blue
9.0-9.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
10.0-10.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha

MDP_RTW_M8DS - OM SIMD8 Dual Source Render Target Data Payload

Source: BSpec

Size (in bits): 2304

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK oMask for slots [7:0] and [15:8]. Operation selects upper or lower half.
1.0-1.7	255:0	**Src0 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Red
2.0-2.7	255:0	**Src0 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Green
3.0-3.7	255:0	**Src0 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Blue
4.0-4.7	255:0	**Src0 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Alpha
5.0-5.7	255:0	**Src1 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Red
6.0-6.7	255:0	**Src1 Green**

MDP_RTW_M8DS - OM SIMD8 Dual Source Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Green
7.0-7.7	255:0	**Src1 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Blue
8.0-8.7	255:0	**Src1 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Alpha

MDP_RTW_M8 - OM SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1280

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK Slots [7:0] oMask. Upper half ignored.
1.0-1.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha

MDP_RTW_M16 - OM SIMD16 Render Target Data Payload

Source: BSpec

Size (in bits): 2304

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
1.0-1.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
3.0-3.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Green[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Green
5.0-5.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Blue
6.0-6.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8

MDP_RTW_M16 - OM SIMD16 Render Target Data Payload

		Slots [15:8] Blue
7.0-7.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
8.0-8.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha

Reference_List_Entry

Source: BSpec
Size (in bits): 8
Default Value: 0x00000000

The byte definition for a reference picture:

DWord	Bit	Description
0	7	**Non-Existing** Format: U1 Indicates that frame store index that should have been at this entry did not exist and was replaced by an index 0 (a valid entry) for error concealment. Value: 0 = Existing; 1 = NonExisting
	6	**Long term bit** Format: U1 Set this reference picture to be used as long term reference. Value: 0 = ShortTerm; 1 = LongTerm
	5	**Field picture flag** Format: U1 Indicates frame/field. Value: 0,1 = FrameOrField
	4:0	**Frame store index or Frame Store ID** Format: U5 Bit 4:1 is used to form the binding table index.

RENDER_SURFACE_STATE

Source:	BSpec
Exists If:	//([MessageType] != 'Deinterlace') && ([MessageType] != 'Sample_8x8')
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

This is the normal surface state used by all messages that use SURFACE_STATE except those that use MEDIA_SURFACE_STATE.

DWord	Bit	Description
0	31:29	**Surface Type** Format: U3 Enumerated Type This field defines the type of the surface.

Value	Name	Description
0h	SURFTYPE_1D	Defines a 1-dimensional map or array of maps
1h	SURFTYPE_2D	Defines a 2-dimensional map or array of maps.
2h	SURFTYPE_3D	Defines a 3-dimensional (volumetric) map.
3h	SURFTYPE_CUBE	Defines a cube map or array of cube maps.
4h	SURFTYPE_BUFFER	Defines an element in a buffer.
5h	SURFTYPE_STRBUF	Defines a structured buffer surface.
6h	Reserved	
7h	SURFTYPE_NULL	Defines a null surface.

Programming Notes

A null surface is used in instances where an actual surface is not bound. When a write message is generated to a null surface, no actual surface is written to. When a read message (including any sampling engine message) is generated to a null surface, the result is all zeros. Note that a null surface type is allowed to be used with all messages, even if it is not specificially indicated as supported. All of the remaining fields in surface state are ignored for null surfaces, with the following exceptions: Width, Height, Depth, LOD, and Render Target View Extent fields must match the depth buffer's corresponding state for all render target surfaces, including null. All sampling engine and data port messages support null surfaces with the above behavior, even if not mentioned as specifically supported, except for the following: Data Port Media Block Read/Write messages. The Surface Type of a surface used as a render target (accessed via the Data Port's Render Target Write message) must be the same as the Surface Type of all other render targets and of the depth buffer (defined in 3DSTATE_DEPTH_BUFFER), unless either the depth buffer or render targets are SURFTYPE_NULL.

	28	**Surface Array** Format: Enable This field, if enabled, indicates that the surface is an array. If this field is enabled, the Surface Type must be SURFTYPE_1D, SURFTYPE_2D, or

RENDER_SURFACE_STATE

SURFTYPE_CUBE. If this field is disabled and Surface Type is SURFTYPE_1D, SURFTYPE_2D, or SURFTYPE_CUBE, the Depth field must be set to zero.

27 **Reserved**

Format:	MBZ

26:18 **Surface Format**

Format:	SURFACE_FORMAT

Specifies the format of the surface or element within this surface. Refer to the table in section 1.12.4.1.2 for the formats supported and their encodings.

Programming Notes
YUV (YCRCB) surfaces used as render targets can only be rendered to using 3DPRIM_RECTLIST with even X coordinates on all of its vertices, and the pixel shader cannot kill pixels. If Number of Multisamples is set to a value other than MULTISAMPLECOUNT_1, this field cannot be set to the following formats: any format with greater than 64 bits per element, if Number of Multisamples is MULTISAMPLECOUNT_8, any compressed texture format (BC*), and any YCRCB* format.
This field cannot be a YUV (YCRCB*) format if the **Surface Type** is SURFTYPE_BUFFER or SURFTYPE_STRBUF.

17:16 **Surface Vertical Alignment**

Format:	U2 Enumerated Type

Description
For Sampling Engine Uncompressed and Render Target Surfaces: This field specifies the vertical alignment requirement for the surface. Refer to the "Memory Data Formats" chapter for details on how this field changes the layout of the surface in memory. This field applies to surface formats other than compressed formats. For other surfaces this field is ignored.
A value of 1 is not supported for formats YCRCB_NORMAL (0x182), YCRCB_SWAPUVY (0x183), YCRCB_SWAPUV (0x18f), or YCRCB_SWAPY (0x190).

Value	Name	Description
2h-3h	Reserved	Reserved

Programming Notes
This field is intended to be set to VALIGN_4 if the surface was rendered as a depth buffer, for a multisampled (4x) render target, or for a multisampled (8x) render target, since these surfaces support only alignment of 4. Use of VALIGN_4 for other surfaces is supported, but uses more memory. This field must be set to VALIGN_4 for all tiled Y Render Target surfaces.
Value of 1 is not supported for format YCRCB_NORMAL (0x182), YCRCB_SWAPUVY (0x183), YCRCB_SWAPUV (0x18f), YCRCB_SWAPY (0x190)

RENDER_SURFACE_STATE

If Number of Multisamples is not MULTISAMPLECOUNT_1, this field must be set to VALIGN_4.

Programming Notes
Restriction: VALIGN_4 is not supported for surface format R32G32B32_FLOAT.

15 **Surface Horizontal Alignment**

Format:	U1 Enumerated Type

For Sampling Engine Uncompressed and Render Target Surfaces: This field specifies the horizontal alignment requirement for the surface. Refer to the "Memory Data Formats" chapter for details on how this field changes the layout of the surface in memory. This field applies to surface formats other than compressed formats. For other surfaces, this field is ignored.

Value	Name	Description
0h	HALIGN_4	Horizontal alignment factor j = 4
1h	HALIGN_8	Horizontal alignment factor j = 8

Programming Notes
This field is intended to be set to HALIGN_8 only if the surface was rendered as a depth buffer with Z16 format or a stencil buffer, since these surfaces support only alignment of 8. Use of HALIGN_8 for other surfaces is supported, but uses more memory.
This field must be set to HALIGN_4 if the Surface Format is BC*.
This field must be set to HALIGN_8 if the Surface Format is FXT1.

14 **Tiled Surface**

Format:	U1 Enumerated Type

This field specifies whether the surface is tiled.

Value	Name	Description
0h	FALSE	Linear surface
1h	TRUE	Tiled surface

Programming Notes
Linear surfaces can be mapped to Main Memory (uncached) or System Memory (cacheable, snooped). Tiled surfaces can only be mapped to Main Memory. The corresponding cache(s) must be invalidated before a previously accessed surface is accessed again with an altered state of this bit. If Surface Type is SURFTYPE_BUFFER, this field must be FALSE (because buffers are supported only in linear memory). If Surface Type is SURFTYPE_NULL, this field must be TRUE.
If **Number of Multisamples** is not MULTISAMPLECOUNT_1, this field must be TRUE.

13 **Tile Walk**

Format:	U1 Enumerated Type

This field specifies the type of memory tiling (XMajor or YMajor) used to tile this surface. See Memory Interface Functions for details on memory tiling and restrictions.

Value	Name	Description

RENDER_SURFACE_STATE

0b	TILEWALK_XMAJOR	X major tiling.
1b	TILEWALK_YMAJOR	Y major tiling.

Programming Notes

Refer to Memory Data Formats for restrictions on TileWalk direction for the various buffer types. (Of particular interest is the fact that YMAJOR tiling is not supported for display/overlay buffers). The corresponding caches must be invalidated before a previously accessed surface is accessed again with an altered state of this bit.This field is ignored when the surface is linear.

Programming Notes

Set Tile Walk to TILEWALK_XMAJOR if Tiled Surface is False.

12 **Vertical Line Stride**

Format:	U1 in lines to skip between logically adjacent lines

For 2D non-array surfaces accessed via the Sampling Engine or Data Port: Specifies the number of lines (0 or 1) to skip between logically adjacent lines and supports interleaved (field) surfaces as textures.
For other surfaces, Vertical Line Stride must be zero.

Programming Notes

This bit must not be set if the surface format is a compressed type (BCn*).

If this bit is set on a sampling engine surface, the mip mode filter must be set to MIPFILTER_NONE.

11 **Vertical Line Stride Offset**

Format:	U1 in lines of initial offset (when Vertical Line Stride == 1)

For 2D non-array Surfaces accessed via the Sampling Engine or Data Port: Specifies the offset of the initial line from the beginning of the buffer. Ignored when Vertical Line Stride is 0.
For other surfaces, Vertical Line Stride Offset must be zero.

10 **Surface Array Spacing**

Format:	U1 Enumerated Type

For 1D Array, 2D Array, Cube, and 2D Multisampled Surfaces: This field specifies whether space is reserved between array slices for additional LODs beyond LOD 0. Refer to the "Memory Data Formats" chapter for details on how this field changes the QPitch equation used to determine spacing between array slices in memory. For other surfaces, this field is ignored.

Value	Name	Description
0h	ARYSPC_FULL	Memory space between array slices is reserved for all possible LOD's.
1h	ARYSPC_LOD0	Memory space is optimized for surfaces which contain only LOD 0.

Programming Notes

RENDER_SURFACE_STATE

If Multisampled Surface Storage Format is MSFMT_MSS and Number of Multisamples is *not* MULTISAMPLECOUNT_1, this field must be set to ARYSPC_LOD0.

9 **Reserved**

Format:	MBZ

8 **Render Cache Read Write Mode**

Format:	U1 Enumerated Type

For Surfaces accessed via the Data Port to Render Cache: This field specifies the way Render Cache treats a write request. If clear, Render Cache allocates a write-only cache line for a write miss. If set, Render Cache allocates a read-write cache line for a write miss. For Surfaces accessed via the Sampling Engine or Data Port to Texture Cache or Data Cache: This field is reserved and MBZ.

Value	Name	Description
0h		Allocating write-only cache for a write miss
1h		Allocating read-write cache for a write miss

Programming Notes
This field is provided for performance optimization for Render Cache read/write accesses (from Gen4 EU's point of view).

7:6 **Media Boundary Pixel Mode**

Format:	U2 Enumerated Type

For 2D Non-Array Surfaces accessed via the Data Port Media Block Read Message: This field enables control of which rows are returned on vertical out-of-bounds reads using the Data Port Media Block Read Message. In the description below, frame mode refers to Vertical Line Stride = 0, field mode is Vertical Line Stride = 1 in which only the even or odd rows are addressable. The frame refers to the entire surface, while the field refers only to the even or odd rows within the surface. For other surfaces this field is reserved and MBZ.

Value	Name	Description
0h	NORMAL_MODE	the row returned on an out-of-bound access is the closest row in the frame or field. Rows from the opposite field are never returned.
1h	Reserved	
2h	PROGRESSIVE_FRAME	the row returned on an out-of-bound access is the closest row in the frame, even if in field mode.
3h	INTERLACED_FRAME	In field mode, the row returned on an out-of-bound access is the closest row in the field. In frame mode, even out-of-bound rows return the nearest even row while odd out-of-bound rows return the nearest odd row.

5:0 **Cube Face Enables**

Format:	U6 bit mask of enables

For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine: Bits 5:0 of this field enable the

RENDER_SURFACE_STATE

individual faces of a cube map. Enabling a face indicates that the face is present in the cube map, while disabling it indicates that that face is represented by the texture map's border color. Refer to Memory Data Formats for the correlation between faces and the cube map memory layout. Note that storage for disabled faces must be provided. For other surfaces this field is reserved and MBZ.

Value	Name
1xxxxxb	-X face
x1xxxxb	+X face
xx1xxxb	-Y face
xxx1xxb	+Y face
xxxx1xb	-Z face
xxxxx1b	+Z face

Programming Notes

When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 111111b (all faces enabled). This field is ignored unless the Surface Type is SURFTYPE_CUBE.

1 | 31:0 | **Surface Base Address**

Format:	GraphicsAddress[31:0]

Specifies the byte-aligned base address of the surface.

Programming Notes

- For SURFTYPE_BUFFER render targets, this field specifies the base address of first element of the surface. The surface is interpreted as a simple array of that single element type. The address must be naturally-aligned to the element size (e.g., a buffer containing R32G32B32A32_FLOAT elements must be 16-byte aligned)
- For SURFTYPE_BUFFER non-rendertarget surfaces, this field specifies the base address of the first element of the surface, computed in software by adding the surface base address to the byte offset of the element in the buffer.
- Mipmapped, cube and 3D sampling engine surfaces are stored in a "monolithic" (fixed) format, and only require a single address for the base texture.
- The Base Address for linear render target surfaces and surfaces accessed with the typed surface read/write data port messages must be element-size aligned, for non-YUV surface formats, or a multiple of 2 element-sizes for YUV surface formats. Other linear surfaces have no alignment requirements (byte alignment is sufficient).
- Linear depth buffer surface base addresses must be 64-byte aligned. Note that while render targets (color) can be SURFTYPE_BUFFER, depth buffers cannot.
- Tiled surface base addresses must be 4KB-aligned. Note that only the offsets from Surface Base Address are tiled, Surface Base Address itself is not transformed using the tiling algorithm.
- For tiled surfaces, the actual start of the surface can be offset from the Surface Base Address by the X Offset and Y Offset fields.

RENDER_SURFACE_STATE

- Certain message types used to access surfaces have more stringent alignment requirements. Please refer to the specific message documentation for additional restrictions.

2 — 31:30 Reserved

Format:	MBZ

29:16 Height

Format:	U14

This field specifies the height of the surface. If the surface is MIP-mapped, this field contains the height of the base MIP level. For buffers, this field specifies a portion of the buffer size.

Value	Name	Description
0		SURFTYPE_1D: must be zero
[0,16383]		SURFTYPE_2D: height of surface - 1 (y/v dimension)
[0,2047]		SURFTYPE_3D: height of surface - 1 (y/v dimension)
[0,16383]		SURFTYPE_CUBE: height of surface - 1 (y/v dimension)
[0,16383]		SURFTYPE_BUFFER/STRBUF: contains bits [20:7] of the number of entries in the buffer - 1

Programming Notes

For typed buffer and structured buffer surfaces, the number of entries in the buffer ranges from 1 to 2^{27}. For raw buffer surfaces, the number of entries in the buffer is the number of bytes which can range from 1 to 2^{30}. After subtracting one from the number of entries, software must place the fields of the resulting 27-bit value into the Height, Width, and Depth fields as indicated, right-justified in each field. Unused upper bits must be set to zero. If Vertical Line Stride is 1, this field indicates the height of the field, not the height of the frameThe Height of a render target must be the same as the Height of the other render targets and the depth buffer (defined in 3DSTATE_DEPTH_BUFFER), unless Surface Type is SURFTYPE_1D or SURFTYPE_2D with Depth = 0 (non-array) and LOD = 0 (non-mip mapped).

If this surface in memory is accessed with Vertical Line Stride set to both 0 and 1, this field must be an even value when Vertical Line Stride is 0.

If Media Pixel Boundary Mode is not set to NORMAL_MODE, this field must be an even value.

If the surface is a stencil buffer, the height must be set to 1/2x the value true surface height, as the stencil buffer is stored with two rows interleaved. For details on the separate stencil buffer storage format in memory, see GPU Overview (vol1a), Memory Data Formats, Surface Layout, 2D Surfaces, Stencil Buffer Layout (section 8.20.4.8).

If **Surface Format** is PLANAR*, this field must be a multiple of 4

15:14 Reserved

Format:	MBZ

13:0 Width

Format:	U14-1

RENDER_SURFACE_STATE

This field specifies the width of the surface. If the surface is MIP-mapped, this field specifies the width of the base MIP level. The width is specified in units of pixels or texels. For buffers, this field specifies a portion of the buffer size.

For surfaces accessed with the Media Block Read/Write message, this field is in units of DWords except when used for IECP and the output surface format is NV12 (R16_UNORM), this field is in units of Words.

Value	Name	Description
[0, 16383]		SURFTYPE_1D: width of surface - 1 (x/u dimension)
[0, 16383]		SURFTYPE_2D: width of surface - 1 (x/u dimension)
[0, 2047]		SURFTYPE_3D: width of surface - 1 (x/u dimension)
[0, 16383]		SURFTYPE_CUBE: width of surface - 1 (x/u dimension)
[0, 127]		SURFTYPE_BUFFER/STRBUF: contains bits [6:0] of the number of entries in the buffer - 1

Programming Notes

For surface types other than SURFTYPE_BUFFER or STRBUF The Width specified by this field must be less than or equal to the surface pitch (specified in bytes viathe Surface Pitch field). For cube maps, Width must be set equal to the Height.For MONO8 textures, Width must be a multiple of 32 texels. The Width of a render target must be the same as the Width of the other render target(s) and the depth buffer (defined in 3DSTATE_DEPTH_BUFFER), unless Surface Type is SURFTYPE_1D or SURFTYPE_2D with Depth = 0 (non-array) and LOD = 0 (non-mip mapped). The Width of a render target with YUV surface format must be a multiple of 2.For SURFTYPE_BUFFER: The low two bits of this field must be 11 if the Surface Format is RAW (the size of the buffer must be a multiple of 4 bytes).

If the surface is a stencil buffer, the width must be set to 2x the value true surface width, as the stencil buffer is stored with two rows interleaved. For details on the separate stencil buffer storage format in memory, see GPU Overview (vol1a), Memory Data Formats, Surface Layout, 2D Surfaces, Stencil Buffer Layout (section 8.20.4.8).

If **Surface Format** is PLANAR*, this field must be a multiple of 4

3 | 31:21 **Depth**

Format:	U11

This field specifies the total number of levels for a volume texture or the number of array elements allowed to be accessed starting at the Minimum Array Element for arrayed surfaces. If the volume texture is MIP-mapped, this field specifies the depth of the base MIP level. For buffers, this field specifies a portion of the buffer size.

Value	Name	Description
[0,2047]		SURFTYPE_1D: number of array elements - 1
[0,2047]		SURFTYPE_2D: number of array elements - 1

RENDER_SURFACE_STATE

[0,2047]		SURFTYPE_3D: depth of surface - 1 (z/r dimension)
[0,2047]		SURFTYPE_CUBE: number of array elements - 1 [see programming notes for range]
[0,1023]		SURFTYPE_BUFFER: contains bits [30:21] of the number of entries in the buffer - 1 for Surface Format RAW.
[0,63]		SURFTYPE_BUFFER: Contains bits [26:21] of the number of entries in the buffer - 1 for other surface formats.
[0,63]		SURFTYPE_STRBUF: contains bits [26:21] of the number of entries in the buffer - 1

Programming Notes
The Depth of a render target must be the same as the Depth of the other render target(s) and of the depth buffer (defined in 3DSTATE_DEPTH_BUFFER). For SURFTYPE_CUBE:For Sampling Engine Surfaces, the range of this field is [0,340], indicating the number of cube array elements (equal to the number of underlying 2D array elements divided by 6). For other surfaces, this field must be zero. For SURFTYPE_BUFFER: The range of this field is [0,63] unless the Surface Format is RAW and Surface Ptich is 1 byte.
For SURFTYPE_1D, 2D, and CUBE: The range of this field is reduced by one for each increase from zero of **Minimum Array Element**. For example, if **Minimum Array Element** is set to 1024 on a 2D surface, the range of this field is reduced to [0,1023].

Programming Notes
Restriction: For SURFTYPE_CUBE sampling engine surfaces, the range of this field is limited to [0,85].
Restriction: If Surface Array is enabled, and Depth is between 1024 and 2047, an incorrect array slice may be accessed if the requested array index in the message is greater than or equal to 4096.

20:18 **Reserved**

Format:	MBZ

17:0 **Surface Pitch**

Format:	U18 pitch in (#Bytes - 1)

This field specifies the surface pitch in (#Bytes - 1). For surfaces of type SURFTYPE_BUFFER and SURFTYPE_STRBUF, this field indicates the size of the structure.

Value	Name	Description
[0,2047]		For surfaces of type SURFTYPE_BUFFER: representing [1B, 2048B]
[0,2047]		For surfaces of type SURFTYPE_STRBUF: representing [1B, 2048B]
[0,262143]		For other linear surfaces: representing [1B, 256KB]
[511,262143]		For X-tiled surface: representing [512B, 256KB] = [1 tile, 512 tiles]
[127,262143]		For Y-tiled surfaces: representing [128B, 256KB] = [1 tile, 2048 tiles]

RENDER_SURFACE_STATE

		Programming Notes For linear render target surfaces and surfaces accessed with the typed data port messages, the pitch must be a multiple of the element size for non-YUV surface formats. Pitch must be a multiple of 2 * element size for YUV surface formats. For linear surfaces with Surface Type of SURFTYPE_STRBUF, the pitch must be a multiple of 4 bytes. For other linear surfaces, the pitch can be any multiple of bytes. For tiled surfaces, the pitch must be a multiple of the tile width. If the surface is a stencil buffer, the pitch must be set to 2x the value computed based on width, as the stencil buffer is stored with two rows interleaved. For details on the separate stencil buffer storage format in memory, see GPU Overview (vol1a), Memory Data Formats, Surface Layout, 2D Surfaces, Stencil Buffer Layout (section 8.20.4.8).
4	31	**Reserved** Exists If: [Surface Type] != 'SURFTYPE_STRBUF' Format: MBZ
	30:29	**Render Target Rotation** Exists If: [Surface Type] != 'SURFTYPE_STRBUF' Format: U2 Enumerated Type For Render Target Surfaces: This field specifies the rotation of this render target surface when being written to memory. For Other Surfaces:This field is ignored. Value / Name / Description: 0h / RTROTATE_0DEG / No rotation (0 degrees) 1h / RTROTATE_90DEG / Rotate by 90 degrees 2h / Reserved / 3h / RTROTATE_270DEG / Rotate by 270 degrees **Programming Notes** Rotation is not supported for render targets of any type other than simple, non-mip-mapped, non-array 2D surfaces. The surface must be using tiled with X major. Width and Height fields apply to the dimensions of the surface before rotation. For 90 and 270 degree rotated surfaces, the Height (rather than the Width) must be less than or equal to the Surface Pitch (specified in bytes). For 90 and 270 degree rotated surfaces, the actual Height and Width of the surface in pixels (not the field value which is decremented) must both be even. Rotation is supported only for surfaces with the following surface formats: B5G6R5_UNORM, B5G6R5_UNORM_SRGB, R8G8B8[A|X]8_UNORM, R8G8B8[A|X]8_UNORM_SRGB, B8G8R8[A|X]8_UNORM, B8G8R8[A|X]8_UNORM_SRGB, B10G10R10[A|X]2_UNORM, B10G10R10A2_UNORM_SRGB, R10G10B10A2_UNORM, R10G10B10A2_UNORM_SRGB, R16G16B16A16_FLOAT, R16G16B16X16_FLOAT. Rotation is not supported for typed UAV messages
	31:27	**Reserved** Exists If: [Surface Type] == 'SURFTYPE_STRBUF' Format: MBZ
	28:18	**Minimum Array Element**

RENDER_SURFACE_STATE

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'
Format:	U11

For Sampling Engine, Render Target, and Typed 1D and 2D Surfaces:This field indicates the minimum array element that can be accessed as part of this surface. This field is added to the delivered array index before it is used to address the surface. For Render Target 3D Surfaces:This field indicates the minimum 'R' coordinate on the LOD currently being rendered to. This field is added to the delivered array index before it is used to address the surface. For Sampling Engine Cube Surfaces:This field indicates the minimum array element in the underlying 2D surface array that can be accessed as part of this surface (the cube array index is multipled by 6 to compute this value, although this field is not restricted to only multiples of 6). This field is added to the delivered array index before it is used to address the surface.

For Other Surfaces:This field must be set to zero.

Value	Name	Description
[0,2047]		1D/2D/cube surfaces
[0,2047]		3D surfaces

Programming Notes
If Number of Multisamples is not MULTISAMPLECOUNT_1, this field must be set to zero if this surface is used with sampling engine messages.

17:7 **Render Target View Extent**

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'
Format:	U11

For Render Target 3D Surfaces:This field indicates the extent of the accessible 'R' coordinates minus 1 on the LOD currently being rendered to. For Render Target 1D and 2D Surfaces:This field must be set to the same value as the Depth field. For Other Surfaces:This field is ignored.

Value	Name	Description
[0,2047]		to indicate extent of [1,2048]

6 **Multisampled Surface Storage Format**

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'
Format:	U1 Enumerated Type

This field indicates the storage format of the multisampled surface.

Value	Name	Description
0h	MSFMT_MSS	Multsampled surface was/is rendered as a render target
1h	MSFMT_DEPTH_STENCIL	Multisampled surface was rendered as a depth or stencil buffer

Programming Notes
All multisampled render target surfaces must have this field set to MSFMT_MSSIF this field is MSFMT_DEPTH_STENCIL, the only sampling engine messages allowed are "ld2dms", "resinfo", and "sampleinfo".

RENDER_SURFACE_STATE

	This field is ignored if Number of Multisamples is MULTISAMPLECOUNT_1
	If the surface's Number of Multisamples is MULTISAMPLECOUNT_8, Width is >= 8192 (meaning the actual surface width is >= 8193 pixels), this field must be set to MSFMT_MSS.
	If the surface's Number of Multisamples is MULTISAMPLECOUNT_8, ((Depth+1) * (Height+1)) is > 4,194,304, OR if the surface's Number of Multisamples is MULTISAMPLECOUNT_4, ((Depth+1) * (Height+1)) is > 8,388,608, this field must be set to MSFMT_DEPTH_STENCIL. This field must be set to MSFMT_DEPTH_STENCIL if Surface Format is one of the following: I24X8_UNORM, L24X8_UNORM, A24X8_UNORM, or R24_UNORM_X8_TYPELESS.

5:3 Number of Multisamples

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'
Format:	U3 Enumerated Type

This field indicates the number of multisamples on the surface.

Value	Name
0h	MULTISAMPLECOUNT_1
1h	Reserved
2h	MULTISAMPLECOUNT_4
3h	MULTISAMPLECOUNT_8
4h-7h	Reserved

Programming Notes
If this field is any value other than MULTISAMPLECOUNT_1, the Surface Type must be SURFTYPE_2D This field must be set to MULTISAMPLECOUNT_1 unless the surface is a Sampling Engine surface or Render Target surface. This field must be set to MULTISAMPLECOUNT_1 for SINT MSRTs when all RT channels are not written If this field is any value other than MULTISAMPLECOUNT_1, Surface Min LOD, Mip Count / LOD, and Resource Min LOD must be set to zero

26:0 Minimum Array Element

Exists If:	[Surface Type] == 'SURFTYPE_STRBUF'

This field indicates the minimum array element that can be accessed as part of this surface. This field is added to the delivered array index before it is used to address the surface.

Value	Name
[0,226]	

2:0 Multisample Position Palette Index

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'

This field indicates the index into the sample position palette that the multisampled surface is using. This field is only used as a return value for the sampleinfo message, and is otherwise not used by hardware.

Value	Name

RENDER_SURFACE_STATE

		[0,7]

5 — 31:25 X Offset

Format:	PixelOffset[8:2]

This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the surface. This field effectively loosens the alignment restrictions on the origin of tiled surfaces. Previously, tiled surface origin was (by definition) located at the base address, and thus needed to satisfy the 4KB base address alignment restriction. Now the origin can be specified at a finer (4-wide x 2-high pixel) resolution.

Value	Name	Description
[0,508]		in multiples of 4 (low 2 bits missing)

Programming Notes

For linear surfaces, this field must be zero.
For surfaces accessed with the Data Port Media Block Read/Write message, the pixel size is assumed to be 32 bits in width.
For Surface Format with other than 8, 16, 32, 64, or 128 bits per pixel, this field must be zero.
If Render Target Rotation is set to other than RTROTATE_0DEG, this field must be zero.
If Surface Type is SURFTYPE_STRBUF, this field must be zero.
This field must be zero if Surface Format is PLANAR*. For all other surfaces, Xoffset must be programmed such that (max X of the draw rectangle)+Xoffset < 16K (max surface width)
For YUV422 surfaces, the pixel offset is in multiples of 2. Pixel offset specified in this case is PixelOffset[7:1]

24 — Reserved

Format:	MBZ

23:20 Y Offset

Format:	RowOffset[4:1]

This field specifies the vertical offset in rows from the Surface Base Address to the start of the surface. (See additional description in the X Offset field)

Value	Name	Description
[0,30]		in multiples of 2 (low bit missing)

Programming Notes

For linear surfaces, this field must be zero. For render targets in which the Render Target Array Index is not zero, this field must be zero. For Surface Format with other than 8, 16, 32, 64, or 128 bits per pixel, this field must be zero. If Render Target Rotation is set to other than RTROTATE_0DEG, this field must be zero. For surfaces accessed in field mode (Vertical Line Stride = 1 or equivalent Media Block Read/Write message override), this field must be set to a multiple of 4. If Surface Type is SURFTYPE_STRBUF, this field must be zero. This field must be zero if Surface Format is PLANAR*. For all other surfaces, Yoffset must be programmed such that (Maximum Yof draw rectangle) + Yoffset < 16K (max surface height)

19:16 Surface Object Control State

RENDER_SURFACE_STATE

	Format:	MEMORY_OBJECT_CONTROL_STATE

Specifies the memory object control state for this surface.

15:8 Reserved

Format:	MBZ

7:4 Surface Min LOD

Format:	U4 in LOD units

For Sampling Engine and Typed Surfaces:This field indicates the most detailed LOD that can be accessed as part of this surface. This field is added to the delivered LOD (sample_l, ld, or resinfo message types) before it is used to address the surface. For Other Surfaces:This field is ignored.

Value	Name
[0,14]	

Programming Notes
This field must be zero if the Surface Format is MONO8

3:0 MIP Count / LOD

Format:	Sampling Engine and Typed Surfaces: U4 in (LOD units - 1)Render Target Surfaces: U4 in LOD units

For Sampling Engine Surfaces:

This field indicates the number of MIP levels allowed to be accessed starting at **Surface Min LOD**, which must be less than or equal to the number of MIP levels actually stored in memory for this surface. For sample* messages, the mip map access is clamped to be between the mipmap specified by the integer bits of the Min LOD and the ceiling of the value specified here. For ld* messages, out-of-bounds behavior results for LODs outside of the range specified in this field.

For Render Target Surfaces:

This field defines the MIP level that is currently being rendered into. This is the absolute MIP level on the surface and is not relative to the Surface Min LOD field, which is ignored for render target surfaces.

For Other Surfaces:

This field is reserved: MBZ

Value	Name	Description
[0,14]		Sampling Engine and Typed Surfaces: representing [1,15] MIP levels
[0,14]		Render Target Surfaces: representing LOD
0		Other Surfaces
0h	Disable	
1h	Enable	

RENDER_SURFACE_STATE

		Programming Notes The LOD of a render target must be the same as the LOD of the other render target(s) and of the depth buffer (defined in 3DSTATE_DEPTH_BUFFER). For render targets with YUV surface formats, the LOD must be zero. It is not legal to have more than one 1x1 mipmap. Software must ensure that MIP Count is set to end on the first 1x1 mipmap (or before).
6	31:30	**Reserved: MBZ** Exists If: [Surface Format] == 'PLANAR' Format: MBZ
	29:16	**X Offset for UV Plane** Exists If: [Surface Format] == 'PLANAR' Format: U14 Row Offset This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the interleaved UV plane. This field is only used for PLANAR surface formats. **Programming Notes** This field must indicate an even number of pixels.
	15:14	**Reserved** Exists If: [Surface Format] == 'PLANAR' Format: MBZ
	31:12	**MCS Base Address** Exists If: ([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Enabled') Format: GraphicsAddress[31:12] Specifies the 4kbyte-aligned base address of the MCS surface associated with the MSS surface specified in other 32 fields. **Programming Notes** The MCS surface must be stored as Tile Y. The MCS surface shares Height, Width, Depth, Surface Min LOD, MIP Count / LOD, Surface Object Control State, Surface Array Spacing, and Minimum Array Element with the primary surface.
	31:6	**Append Counter Address** Exists If: ([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Disabled') Format: GraphicsAddress[31:6] Specifies the 64byte-aligned base address of the Append counter associated with this surface specified in other SURFACE_STATE fields.
	11:3	**MCS Surface Pitch** Exists If: ([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Enabled') Format: U9-1 pitch in #Tiles

RENDER_SURFACE_STATE

This field specifies the MCS surface pitch in (#Tiles - 1).

Value	Name	Description
[0,511]		representing [1 tile, 512 tiles]

5:2 Reserved

Exists If:	([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Disabled')
Format:	MBZ

2:1 Reserved

Exists If:	([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Enabled')
Format:	MBZ

1 Append Counter Enable

Exists If:	([Surface Format] != 'PLANAR') AND ([MCS Enable] == 'Disabled')
Format:	Enable

Enables the use of the Append Counter with this surface. If disabled, all other Append counter fields are ignored.

13:0 Y Offset for UV Plane

Exists If:	[Surface Format] == 'PLANAR'
Format:	14 Row Offset

This field specifies the vertical offset in rows from the Surface Base Address to the start (origin) of the interleaved UV plane. This field is only used for PLANAR surface formats.

Programming Notes
This field must indicate an even number (bit 0 = 0).

0 MCS Enable

Exists If:	[Surface Format] != 'PLANAR'
Format:	Enable

Enables the use of the MCS with this surface. If disabled, all other MCS fields are ignored.
For Render Target and Sampling Engine Surfaces:If the surface is multisampled (Number of Multisamples any value other than MULTISAMPLECOUNT_1), this field must be enabled.
For Other Surfaces:This field and the other MCS fields are ignored.

Programming Notes
When accessing a multisampled surface using the sampling engine, the MCS surface is read in a separate pass and is considered by hardware to be an independent surface. This same bitfield is used when MCS is enabled; also when disabled.

7 31 Red Clear Color

Format:	U1 Enumerated Type

For Sampling Engine Multisampled Surfaces and Render Targets:Specifies the clear value for the red channel. For Other Surfaces:This field is ignored.

Value	Name	Description
0	CC_ZERO	Clear color value is 0.0, correctly interpreted based on surface

RENDER_SURFACE_STATE

		format.
1	CC_ONE	Clear color value is 1.0, correctly interpreted based on surface format.

30 **Green Clear Color**

Format:	U1 Enumerated Type

For Sampling Engine Multisampled Surfaces and Render Targets:Specifies the clear value for the green channel. For Other Surfaces:This field is ignored.

Value	Name	Description
0	CC_ZERO	Clear color value is 0.0, correctly interpreted based on surface format.
1	CC_ONE	Clear color value is 1.0, correctly interpreted based on surface format.

29 **Blue Clear Color**

Format:	U1 Enumerated Type

For Sampling Engine Multisampled Surfaces and Render Targets:Specifies the clear value for the blue channel. For Other Surfaces:This field is ignored.

Value	Name	Description
0	CC_ZERO	Clear color value is 0.0, correctly interpreted based on surface format.
1	CC_ONE	Clear color value is 1.0, correctly interpreted based on surface format.

28 **Alpha Clear Color**

Format:	U1 Enumerated Type

For Sampling Engine Multisampled Surfaces and Render Targets:Specifies the clear value for the alpha channel. For Other Surfaces:This field is ignored.

Value	Name	Description
0	CC_ZERO	Clear color value is 0.0, correctly interpreted based on surface format.
1	CC_ONE	Clear color value is 1.0, correctly interpreted based on surface format.

27:16 **Reserved**

Format:	MBZ

15:12 **Reserved**

Format:	MBZ

11:0 **Resource Min LOD**

Format:	U4.8 in LOD units

For Sampling Engine Surfaces:This field indicates the most detailed LOD that is present in the resource underlying the surface. Refer to the "LOD Computation Pseudocode" section for the use of this field. For Other Surfaces:This field is ignored.

RENDER_SURFACE_STATE

Value	Name
[0,14]	

Programming Notes
This field must be zero if the Surface Format is MONO8
This field must be zero if the **ChromaKey Enable** is enabled in the associated sampler.

MDP_RTW_16REP - Replicated SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**RGBA** Format: MDPR_RGBA RGBA for all slots [15:0]

RoundingPrecisionTable_3_Bits

Source: BSpec
Size (in bits): 3
Default Value: 0x00000000

DWord	Bit	Description
0	2:0	**Rounding Precision** Format: U3 Value / Name: 000b +1/16 001b +2/16 010b +3/16 011b +4/16 100b +5/16 101b +6/16 110b +7/16 111b +8/16

MDP_RTW_A8 - S0A SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1280

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha

MDP_RTW_A16 - S0A SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	2560
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Source 0 Alpha[15:7]** Format: MDP_DW_SIMD8 Slots [15:8] Source 0 Alpha
2.0-2.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
4.0-4.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
5.0-5.7	255:0	**Green[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Green
6.0-6.7	255:0	**Blue[7:0]**

MDP_RTW_A16 - S0A SIMD16 Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots [7:0] Blue
7.0-7.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Blue
8.0-8.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
9.0-9.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha

SAMPLER_8x8_STATE

Source:	BSpec
Exists If:	//MessageType == 'Sample_8x8'
Size (in bits):	4416
Default Value:	All bits are zeros.

The 8x8 coefficients and other state used by the sample_8x8 message are stored as indirect state, pointed to by a field in SAMPLER_STATE. There are four different tables loaded using this structure (0X, 0Y, 1X, and 1Y). Each table is stored as an array of 17 elements, each with either 4 or 8 coefficients.

DWord	Bit	Description
0	31:24	**Table 0X Filter Coefficient[0,3]** Format: S1.6 In 2's complement format **Description** Range: [-2.0, +2.0)
	23:16	**Table 0X Filter Coefficient[0,2]** Format: S1.6 In 2's complement format Range: [-1, +1)
	15:8	**Table 0X Filter Coefficient[0,1]** Format: S1.6 In 2's complement format Range = $[-2^{-1}, +2^{-1})$ **Programming Notes** Must be zero if the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM.
	7:0	**Table 0X Filter Coefficient[0,0]** Format: S1.6 In 2's complement format Range = $[-2^{-2}, +2^{-2})$ **Programming Notes** Must be zero if the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM
1	31:24	**Table 0X Filter Coefficient[0,7]** Format: S1.6 In 2's complement format Range = $[-2^{-2}, +2^{-2})$
	23:16	**Table 0X Filter Coefficient[0,6]** Format: S1.6 In 2's complement format Range = $[-2^{-1}, +2^{-1})$
	15:8	**Table 0X Filter Coefficient[0,5]** Format: S1.6 In 2's complement format Range: [-1, +1)

SAMPLER_8x8_STATE

	7:0	**Table 0X Filter Coefficient[0,4]** Format: S1.6 In 2's complement format Description: Range: [-2.0, +2.0)
2..3	31:24	**Table 0Y Filter Coefficient[0,7]** Format: S1.6 In 2's complement format Range = $[-2^{-2}, +2^{-2})$
	23:16	**Table 0Y Filter Coefficient[0,6]** Format: S1.6 In 2's complement format Range = $[-2^{-1}, +2^{-1})$
	15:8	**Table 0Y Filter Coefficient[0,5]** Format: S1.6 In 2's complement format Range: [-1, +1)
	7:0	**Table 0Y Filter Coefficient[0,4]** Format: S1.6 In 2's complement format Description: Range: [-2.0, +2.0)
4	31:24	**Table 1X Filter Coefficient[0,3]** Format: S1.6 In 2's complement format Range: [0.0, +2.0)
	23:16	**Table 1X Filter Coefficient[0,2]** Format: S1.6 In 2's complement format Range: [-1, +1)
	15	**Adaptive Filter for all channels** Only to be enabled if 8-tap Adaptive filter mode is on. Else it should be disabled. Value 1: Enable adaptive filter on UV/RB channels Value 0: Disable adaptive filter on UV/RB channels
	14	**Enable RGB Adaptive for RGB input only :** This should be always set to 0 for YUV input and can be enabled/disabled for RGB input. This should be enabled only if we enable 8-tap adaptive filter for RGB input Value | Name

SAMPLER_8x8_STATE

		1 Enable the RGB Adaptive filter using the equation (Y=(R+2G+B)>>2) 0 Disable the RGB Adaptive equation and use G-Ch directly for adaptive filter
	13:0	**Reserved** Format: MBZ
5	31:16	**Reserved** Format: MBZ
	15:8	**Table 1X Filter Coefficient[0,5]** Format: S1.6 In 2's complement format Range: [-1, +1)
	7:0	**Table 1X Filter Coefficient[0,4]** Format: S1.6 In 2's complement format Range: [0.0, +2.0)
6..7	31:16	**Reserved** Format: MBZ
	15:8	**Table 1Y Filter Coefficient[0,5]** Format: S1.6 In 2's complement format Range: [-1, +1)
	7:0	**Table 1Y Filter Coefficient[0,4]** Format: S1.6 In 2's complement format Range: [0.0, +2.0)
8..15	31:0	**Filter Coefficient[1,7:0]**
16..23	31:0	**Filter Coefficient[2,7:0]**
24..31	31:0	**Filter Coefficient[3,7:0]**
32..39	31:0	**Filter Coefficient[4,7:0]**
40..47	31:0	**Filter Coefficient[5,7:0]**
48..55	31:0	**Filter Coefficient[6,7:0]**
56..63	31:0	**Filter Coefficient[7,7:0]**
64..71	31:0	**Filter Coefficient[8,7:0]**
72..79	31:0	**Filter Coefficient[9,7:0]**
80..87	31:0	**Filter Coefficient[10,7:0]**
88..95	31:0	**Filter Coefficient[11,7:0]**
96..103	31:0	**Filter Coefficient[12,7:0]**
104..111	31:0	**Filter Coefficient[13,7:0]**
112..119	31:0	**Filter Coefficient[14,7:0]**

SAMPLER_8x8_STATE

120..127	31:0	**Filter Coefficient[15,7:0]**
128..135	31:0	**Filter Coefficient[16,7:0]**
136	31:24	**Default Sharpness Level** Format: U8 When adaptive scaling is off, determines the balance between sharp and smooth scalers. Value: 0 — Name: Contribute 1 from the smooth scalar Value: 255 — Name: Contribute 1 from the sharp scalar
	23:16	**Max Derivative 4 Pixels** Format: U8 Used in adaptive filtering to specify the lower boundary of the smooth 8 pixel area.
	15:8	**Max Derivative 8 Pixels** Format: U8 Used in adaptive filtering to specify the lower boundary of the smooth 8 pixel area.
	7	**Reserved** Format: MBZ
	6:4	**Transition Area with 4 Pixels** Format: U3 Used in adaptive filtering to specify the width of the transition area for the 4 pixel calculation.
	3	**Reserved** Format: MBZ
	2:0	**Transition Area with 8 Pixels** Format: U3 Used in adaptive filtering to specify the width of the transition area for the 8 pixel calculation.
137	31:23	**Reserved** Format: MBZ
	22	**Bypass X Adaptive Filtering** Format: Disable When disabled, the X direction will use **Default Sharpness Level** to blend between the smooth and sharp filters rather than the calculated value. Value: 1 — Name: Disable X adaptive filtering Value: 0 — Name: Enable X adaptive filtering
	21	**Bypass Y Adaptive Filtering**

SAMPLER_8x8_STATE

		Format: Disable When disabled, the Y direction will use **Default Sharpness Level** to blend between the smooth and sharp filters rather than the calculated value. **Value** / **Name** 1 – Disable X adaptive filtering 0 – Enable X adaptive filtering
	20:2	**Reserved** Format: MBZ
	1	**Reserved** Format: MBZ
	0	**Reserved** Format: MBZ

SAMPLER_BORDER_COLOR_STATE

Source:	BSpec
Size (in bits):	128
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000

This structure is pointed to by a field in SAMPLER_STATE. The interpretation of the border color depends on the Texture Border Color Mode field in SAMPLER_STATE as follows:In DX9 mode, the border color is 8-bit UNORM format, regardless of the surface format chosen. For surface formats with one or more channels missing (i.e. R5G6R5_UNORM is missing the alpha channel), the value from the border color, if selected, will be used even for the missing channels. In DX10/OGL mode, the format of the border color is R32G32B32A32_FLOAT, regardless of the surface format chosen. For surface formats with one or more channels missing, the value from the border color is not used for the missing channels, resulting in these channels resulting in the overall default value (0 for colors and 1 for alpha) regardless of whether border color is chosen. The surface formats with "L" and "I" have special behavior with respect to the border color. The border color value used for the replicated channels (RGB for "L" formats and RGBA for "I" formats) comes from the red channel of border color. In these cases, the green and blue channels, and also alpha for "I", of the border color are ignored. The format of this state depends on the Texture Border Color Mode field.

Programming Notes

- DX9 mode is not supported for surfaces with more than 16 bits in any channel, other than 32-bit float formats which are supported.
- The conditions under which this color is used depend on the **Surface Type** - 1D/2D/3D surfaces use the border color when the coordinates extend beyond the surface extent; cube surfaces use the border color for "empty" (disabled) faces.
- The border color itself is accessed through the texture cache hierarchy rather than the state cache hierarchy. Thus, if the border color is changed in memory, the texture cache must be invalidated and the state cache does not need to be invalidated.
- MAPFILTER_MONO: The border color is ignored. Border color is fixed at a value of 0 by hardware.

DWord	Bit	Description
0	31:24	**Border Color Alpha** Exists If: Structure[SAMPLER_STATE][Texture Border Color Mode] == 'DX9' Format: UNORM8 Texture Border Color Mode = DX9
	23:16	**Border Color Blue** Exists If: Structure[SAMPLER_STATE][Texture Border Color Mode] == 'DX9' Format: UNORM8 Texture Border Color Mode = DX9
	15:8	**Border Color Green** Exists If: Structure[SAMPLER_STATE][Texture Border Color Mode] == 'DX9' Format: UNORM8

SAMPLER_BORDER_COLOR_STATE

		Texture Border Color Mode = DX9
	31:0	**Border Color Red - (DX10/0GL)** Exists If: Structure[SAMPLER_STATE][Texture Border Color Mode] == 'DX10/0GL' Format: IEEE_FP Texture Border Color Mode = DX10/OGL
	7:0	**Border Color Red - (DX9)** Exists If: Structure[SAMPLER_STATE][Texture Border Color Mode] == 'DX9' Format: UNORM8 Texture Border Color Mode = DX9
1	31:0	**Border Color Green** Format: IEEE_FP Texture Border Color Mode = DX10/OGL
2	31:0	**Border Color Blue** Format: IEEE_FP Texture Border Color Mode = DX10/OGL
3	31:0	**Border Color Alpha** Format: IEEE_FP Texture Border Color Mode = DX10/OGL

SAMPLER_STATE

Source:	BSpec
Exists If:	//(MessageType != 'Deinterlace') && (MessageType != 'Sample_8x8')
Size (in bits):	128
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000

This is the normal sampler state used by all messages that use SAMPLER_STATE except sample_8x8 and deinterlace. The sampler state is stored as an array of up to 16 elements, each of which contains the dwords described here. The start of each element is spaced 4 dwords apart. The first element of the sampler state array is aligned to a 32-byte boundary.

DWord	Bit	Description
0	31	**Sampler Disable** Format: Disable This field allows the sampler to be disabled. If disabled, all output channels will return 0.
	30	**Reserved** Format: MBZ
	29	**Texture Border Color Mode** For some surface formats, the 32 bit border color is decoded differently based on the border color mode. In addition, the default value of channels not included in the surface may be affected by this field. Refer to the "Sampler Output Channel Mapping" table for the values of these channels, and for surface formats that may only support one of these modes. Also refer to the definition of SAMPLER_BORDER_COLOR_STATE for more details on the behavior of the two modes defined by this field.

Value	Name	Description
0h	DX10/OGL	DX10/OGL mode for interpreting the border color
1h	DX9	DX9 and earlier mode for interpreting the border color

Programming Notes
This field is required to be the same for every message over a period of time. A flush of the sampler cache must occur before a message with the opposite state of this field is delivered.
This field must be set to DX9 mode when used with surfaces that have Surface Format P4A4_UNORM or A4P4_UNORM.
This field must be set to DX10/OGL mode when used with surfaces that have Surface Format YCRCB_SWAPUV or YCRCB_SWAPY.
This field must be set to DX10/OGL mode if **Surface Format** for the associated surface is UINT OR SINT.
This field must be set to DX10/OGL mode if REDUCTION_MINIMUM or REDUCTION_MAXIMUM or message type is sample_min or sample_max.
This field must be set to DX10/OGL mode if either **Min** or **Mag Mode Filter** is set to MAPFILTER_FLEXIBLE.

	28	**LOD PreClamp Enable**

SAMPLER_STATE

Format:	U1 Enumerated Type

When enabled, the computed LOD is clamped to [max,min] mip level before the mag-vs-min determination is performed. This is how the OpenGL API currently performs min/mag determination, and therefore it is expected that an OpenGL driver would need to set this bit.

Value	Name	Description
0h	Reserved	
1h	OGL	OGL Mode (LOD PreClamp enabled)

27 **Reserved**

Format:	MBZ

26:22 **Base Mip Level**

Format:	U4.1

Range: [0.0, 14.0]

Specifies which mip level is considered the "base" level when determining mag-vs-min filter and selecting the "base" mip level.

21:20 **Mip Mode Filter**

Format:	U2 Enumerated Type

This field determines if and how mip map levels are chosen and/or combined when texture filtering.

Value	Name	Description
0h	NONE	Disable mip mapping - force use of the mipmap level corresponding to Min LOD.
1h	NEAREST	Nearest, Select the nearest mip map
2h	Reserved	
3h	LINEAR	Linearly interpolate between nearest mip maps (combined with linear min/mag filters this is analogous to "Trilinear" filtering).

Programming Notes
MIPFILTER_LINEAR is not supported for surface formats that do not support "Sampling Engine Filtering" as indicated in the Surface Formats table unless using the sample_c message type or minimum/maximum operation.

19:17 **Mag Mode Filter**

Format:	U3 Enumerated Type

This field determines how texels are sampled/filtered when a texture is being "magnified" (enlarged). For volume maps, this filter mode selection also applies to the 3rd (inter-layer) dimension.

Value	Name	Description

SAMPLER_STATE

0h	NEAREST	Sample the nearest texel
1h	LINEAR	Bilinearly filter the 4 nearest texels
2h	ANISOTROPIC	Perform an "anisotropic" filter on the chosen mip level
4h-5h	Reserved	
6h	MONO	Perform a monochrome convolution filter
7h	Reserved	

Programming Notes
Only MAPFILTER_NEAREST and MAPFILTER_LINEAR are supported for surfaces of type SURFTYPE_3D.
Only MAPFILTER_NEAREST is supported for surface formats that do not support "Sampling Engine Filtering" as indicated in the Surface Formats table unless using the sample_c message type or minimum/maximum operation.
MAPFILTER_MONO: Only CLAMP_BORDER texture addressing mode is supported. . Both Mag Mode Filter and Min Mode Filter must be programmed to MAPFILTER_MONO. Mip Mode Filter must be MIPFILTER_NONE. Only valid on surfaces with Surface Format MONO8 and with Surface Type SURFTYPE_2D.
MAPFILTER_FLEXIBLE: The Surface Type of the surface being sampled must be SURFTYPE_2D.
MAPFILTER_ANISOTROPIC may cause artifacts at cube edges if enabled for cube maps with the TEXCOORDMODE_CUBE addressing mode.
MAPFILTER_ANISOTROPIC will be overridden to MAPFILTER_LINEAR when using a sample_l or sample_l_c message type or when Force LOD to Zero is set in the message header.

16:14 **Min Mode Filter**

Format:	U3 Enumerated Type

This field determines how texels are sampled/filtered when a texture is being "minified" (shrunk). For volume maps, this filter mode selection also applies to the 3rd (inter-layer) dimension. See Mag Mode Filter

Value	Name	Description
0h	NEAREST	Sample the nearest texel
1h	LINEAR	Bilinearly filter the 4 nearest texels
2h	ANISOTROPIC	Perform an "anisotropic" filter on the chosen mip level
4h-5h	Reserved	
6h	MONO	Perform a monochrome convolution filter
7h	Reserved	

13:1 **Texture LOD Bias**

Format:	S4.8 2's complement

Range: [-16.0, 16.0)
This field specifies the signed bias value added to the calculated texture map LOD prior to min-

SAMPLER_STATE

vs-mag determination and mip-level clamping. Assuming mipmapping is enabled, a positive LOD bias will result in a somewhat blurrier image (using less-detailed mip levels) and possibly higher performance, while a negative bias will result in a somewhat crisper image (using more-detailed mip levels) and may lower performance.

Programming Notes
There is no requirement or need to offset the LOD Bias in order to produce a correct LOD for texture filtering (as was required for correct bilinear and anisotropic filtering in some legacy devices).

0 — Anisotropic Algorithm

Format:	U1 Enumerated Type

Controls which algorithm is used for anisotropic filtering. Generally, the EWA approximation algorithm results in higher image quality than the legacy algorithm.

Value	Name	Description
0h	LEGACY	Use the legacy algorithm for anisotropic filtering
1h	EWA Approximation	Use the new EWA approximation algorithm for anisotropic filtering

1, 31:20 — Min LOD

Format:	U4.8 in LOD units

Range: [0.0, 14.0], where the upper limit is also bounded by the Max LOD.

This field specifies the minimum value used to clamp the computed LOD after LOD bias is applied. Note that the minification-vs.-magnification status is determined after LOD bias and before this maximum (resolution) mip clamping is applied. The integer bits of this field are used to control the "maximum" (highest resolution) mipmap level that may be accessed (where LOD 0 is the highest resolution map). The fractional bits of this value effectively clamp the inter-level trilinear blend factor when trilinear filtering is in use.

Programming Notes
If Min LOD is greater than Max LOD, Min LOD takes precedence, i.e. the resulting LOD will always be Min LOD.
This field must be zero if the Min or Mag Mode Filter is set to MAPFILTER_MONO

19:8 — Max LOD

Format:	U4.8 in LOD units

Range: [0.0, 14.0]

This field specifies the maximum value used to clamp the computed LOD after LOD bias is applied. Note that the minification-vs.-magnification status is determined after LOD bias and before this minimum (resolution) mip clamping is applied. The integer bits of this field are used to control the "minimum" (lowest resolution) mipmap level that may be accessed. The fractional bits of this value effectively clamp the inter-level trilinear blend factor when trilinear filtering is

SAMPLER_STATE

		in use. Force the mip map access to be between the mipmap specified by the integer bits of the Min LOD and the ceiling of the value specified here.
	7:4	**Reserved** Format: MBZ
	3:1	**Shadow Function** Format: U3 Enumerated Type This field is used for shadow mapping support via the sample_c message type, and specifies the specific comparison operation to be used. The comparison is between the texture sample red channel (except for alpha-only formats which use the alpha channel), and the "ref" value provided in the input message. **Value** / **Name** 0h PREFILTEROP ALWAYS 1h PREFILTEROP NEVER 2h PREFILTEROP LESS 3h PREFILTEROP EQUAL 4h PREFILTEROP LEQUAL 5h PREFILTEROP GREATER 6h PREFILTEROP NOTEQUAL 7h PREFILTEROP GEQUAL
	0	**Cube Surface Control Mode** Format: U1 Enumerated Type When sampling from a SURFTYPE_CUBE surface, this field controls whether the TC* Address Control Mode fields are interpreted as programmed or overridden to TEXCOORDMODE_CUBE. **Value** / **Name** 0h PROGRAMMED 1h OVERRIDE **Programming Notes** This field must be set to CUBECTRLMODE_PROGRAMMED
2	31:5	**Border Color Pointer** Format: DynamicStateOffset[31:5]SAMPLER_BORDER_COLOR_STATE **Description** This field specifies the pointer to SAMPLER_BORDER_COLOR_STATE, which contains the "border" color to be used when accessing texels not contained within the texture map. This pointer is relative to the Dynamic State Base Address. Field definition if Flexible Filter Mode = FLEX_NONSEP:

SAMPLER_STATE

	4:0	**Reserved** Format: MBZ
3	31:26	**Reserved** Format: MBZ
	25	**ChromaKey Enable** Format: Enable This field enables the chroma key function. **Programming Notes** Supported only on a specific subset of surface formats. See section "Surface Formats" for supported formats. This field must be disabled if min or mag filter is MAPFILTER_MONO or MAPFILTER_ANISOTROPIC. This field must be disabled if used with a surface of type SURFTYPE_3D.
	24:23	**ChromaKey Index** This field specifies the index of the ChromaKey Table entry associated with this Sampler. This field is a "don't care" unless ChromaKey Enable is ENABLED.
	22	**ChromaKey Mode** Format: U1 Enumerated Type This field specifies the behavior of the device in the event of a ChromaKey match. This field is ignored if ChromaKey is disabled. KEYFILTER_KILL_ON_ANY_MATCH:In this mode, if any contributing texel matches the chroma key, the corresponding pixel mask bit for that pixel is cleared. The result of this operation is observable only if the Killed Pixel Mask Return flag is set on the input message. KEYFILTER_REPLACE_BLACK:In this mode, each texel that matches the chroma key is replaced with (0,0,0,0) (black with alpha=0) prior to filtering. For YCrCb surface formats, the black value is A=0, R(Cr)=0x80, G(Y)=0x10, B(Cb)=0x80. This will tend to darken/fade edges of keyed regions. Note that the pixel pipeline must be programmed to use the resulting filtered texel value to gain the intended effect, e.g., handle the case of a totally keyed-out region (filtered texel alpha==0) through use of alpha test, etc.

Value	Name
0h	KEYFILTER_KILL_ON_ANY_MATCH
1h	KEYFILTER_REPLACE_BLACK

	21:19	**Maximum Anisotropy** Format: U3 Enumerated Type This field clamps the maximum value of the anisotropy ratio used by the MAPFILTER_ANISOTROPIC filter (Min or Mag Mode Filter).

Value	Name	Description
0h	RATIO 2:1	At most a 2:1 aspect ratio filter is used
1h	RATIO 4:1	At most a 4:1 aspect ratio filter is used
2h	RATIO 6:1	At most a 6:1 aspect ratio filter is used
3h	RATIO 8:1	At most a 8:1 aspect ratio filter is used
4h	RATIO 10:1	At most a 10:1 aspect ratio filter is used

SAMPLER_STATE

5h	RATIO 12:1	At most a 12:1 aspect ratio filter is used
6h	RATIO 14:1	At most a 14:1 aspect ratio filter is used
7h	RATIO 16:1	At most a 16:1 aspect ratio filter is used

18 **U Address Mag Filter Rounding Enable**

Format:	Enable

Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode.

Programming Notes

Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.

17 **U Address Min Filter Rounding Enable**

Format:	Enable

Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode.

Programming Notes

Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.

16 **V Address Mag Filter Rounding Enable**

Format:	Enable

Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode.

Programming Notes

Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.

15 **V Address Min Filter Rounding Enable**

Format:	Enable

Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode.

Programming Notes

Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.

14 **R Address Mag Filter Rounding Enable**

Format:	Enable

Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode.

SAMPLER_STATE

		Programming Notes Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.
	13	**R Address Min Filter Rounding Enable** Format: Enable Controls whether the texture address is rounded or truncated before being used to select texels to sample. Provides independent control of rounding on one texture address dimension (U/V/R) in either mag or min filter mode. **Programming Notes** Hardware will force rounding enable to 0 when message is **gather4**, **gather4_po**, **gather4_c**, or **gather4_po_c**.
	12:11	**Trilinear Filter Quality** Format: U2 Enumerated Type Selects the quality level for the trilinear filter.

Value	Name	Description
0	FULL	Full Quality. Both mip maps are sampled under all circumstances.
2	MED	Medium Quality. If the contribution of one mip map is less than 25%, only the other mip map contributes.
3	LOW	Low Quality. If the contribution of one mip map is less than 37.5%, only the other mip map contributes.

	10	**Non-normalized Coordinate Enable** Format: Enable This field, if enabled, specifies that the input coordinates (U/V/R) are in non-normalized space, where each integer increment is one texel on LOD 0. If disabled, coordinates are normalized, where the range 0 to 1 spans the entire surface.

Programming Notes

The following state must be set as indicated if this field is *enabled*:

- TCX/Y/Z Address Control Mode must be TEXCOORDMODE_CLAMP, TEXCOORDMODE_HALF_BORDER, or TEXCOORDMODE_CLAMP_BORDER.
- Surface Type must be SURFTYPE_2D or SURFTYPE_3D.
- Mag Mode Filter must be MAPFILTER_NEAREST or MAPFILTER_LINEAR.
- Min Mode Filter must be MAPFILTER_NEAREST or MAPFILTER_LINEAR.
- Mip Mode Filter must be MIPFILTER_NONE.
- Min LOD must be 0.
- Max LOD must be 0.
- MIP Count must be 0.
- Surface Min LOD must be 0.
- Texture LOD Bias must be 0.

SAMPLER_STATE

	9	**Reserved** Format: MBZ
	8:6	**TCX Address Control Mode** Format: Texture Coordinate Mode Enumerated Type Controls how the 1st (TCX, aka U) component of input texture coordinates are mapped to texture map addresses - specifically, how coordinates "outside" the texture are handled (wrap/clamp/mirror). The setting of this field is subject to being overridden by the Cube Surface Control Mode field when sampling from a SURFTYPE_CUBE surface. **Programming Notes** When using cube map texture coordinates, only TEXCOORDMODE_CLAMP and TEXCOORDMODE_CUBE settings are valid, and each TC component must have the same Address Control mode. When TEXCOORDMODE_CUBE is not used accessing a cube map, the map's Cube Face Enable field must be programmed to 111111b (all faces enabled). MAPFILTER_MONO: Texture addressing modes must all be set to TEXCOORDMODE_CLAMP_BORDER. The **Border Color** is ignored in this mode, a constant value of 0 is used for border color. Software must pad the border texels within the map itself with 0. If **Surface Format** is PLANAR*, this field must be set to TEXCOORDMODE_CLAMP.
	5:3	**TCY Address Control Mode** Format: Texture Coordinate Mode Enumerated Type Controls how the 2nd (TCY, aka V) component of input texture coordinates are mapped to texture map addresses - specifically, how coordinates "outside" the texture are handled (wrap/clamp/mirror). See Address TCX Control Mode above for details **Programming Notes** If this field is set to TEXCOORDMODE_CLAMP_BORDER or TEXCOORDMODE_HALF_BORDER and a 1D surface is sampled, incorrect blending with the border color in the vertical direction may occur.
	2:0	**TCZ Address Control Mode** Format: Texture Coordinate Mode Enumerated Type **Description** Controls how the 3rd (TCZ) component of input texture coordinates are mapped to texture map addresses - specifically, how coordinates "outside" the texture are handled (wrap/clamp/mirror). See Address TCX Control Mode above for details If this field is set to TEXCOORDMODE_CLAMP_BORDER for 3D maps on formats without an alpha channel, samples straddling the map in the Z direction may have their alpha channels off by 1.

SAMPLER_STATE for Sample_8x8 Message

Source: BSpec
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x0D090801, 0x721A03C6

. This state definition is used only by the sample_8x8 message. This state is stored as an array of up to 16 elements, each of which contains the dwords described here. The start of each element is spaced 4 dwords apart. The first element of the array is aligned to a 32-byte boundary.
The index with range 0-15 that selects which element is being used to determine the **Sampler Index** in the message descriptor.

DWord	Bit	Description
0	31:30	**Reserved** Format: MBZ
	29	**Reserved**
	28:19	**Reserved** Format: MBZ
	18	**ChromaKey Enable** Format: Enable This field enables chroma keying when accessing this particular texture map. **Programming Notes** • For sample_8x8 instructions KEYFILTER_REPLACE_BLACK is assumed if chromakey is enabled. • For 10 bit formats only the 8 MSBs will be compared.
	17:16	**ChromaKey Index** Format: U2 This field specifies the index of the ChromaKey Table entry associated with this Sampler. This field is a "don't care" unless ChromaKey Enable is ENABLED. Value: [0,3] — Name:
	15:8	**Reserved** Format: MBZ
	7:0	**Global Noise Estimation** Format: U8 Global noise estimation of previous frame.
1	31:5	**Sampler 8x8 State Pointer** Format: DynamicStateOffset[31:5] This field specifies the pointer to the SAMPLER_8x8_STATE structure. This pointer is relative to the Dynamic State Base Address. **Programming Notes**

SAMPLER_STATE for Sample_8x8 Message

		• This field must be set to the same value in all sample_8x8 type SAMPLER_STATE instances applied to a given primitive. • PIPE_CONTROL with State/Instruction Cache Invalidate set and the CS Stall field set is required between primitives that use different values of this field.
	4:0	**Reserved** Format: MBZ
2	31	**Reserved** Format: MBZ
	30:26	**Reserved**
	25:21	**Reserved**
	20:16	**Reserved**
	15:14	**Reserved** Format: MBZ
	13:8	**Strong Edge Threshold** Default Value: 8 Format: U6 If EM > Strong Edge Threshold, the basic VSA detects a strong edge.
	7:6	**Reserved** Format: MBZ
	5:0	**Weak Edge Threshold** Default Value: 1 Format: U6 If Strong Edge Threshold > EM > Weak Edge Threshold, the basic VSA detects a weak edge.
3	31	**Reserved**
	30:28	**Strong Edge Weight** Default Value: 7 Format: U3 Sharpening strength when a strong edge is found in basic VSA..
	27	**Reserved** Format: MBZ
	26:24	**Regular Weight** Default Value: 2 Format: U3 Sharpening strength when a weak edge is found in basic VSA.

SAMPLER_STATE for Sample_8x8 Message

	23	**Reserved** Format: MBZ
	22:20	**Non Edge Weight** Default Value: 1 Format: U3 Sharpening strength when no edge is found in basic VSA.
	19:14	**Gain Factor** Default Value: 40 Format: U6 User control sharpening strength.
	13:11	**Reserved** Format: MBZ
	10:6	**Reserved**
	5	**Reserved** Format: MBZ
	4:0	**Reserved**

SCISSOR_RECT

Source: RenderCS

Size (in bits): 64

Default Value: 0x00000000, 0x00000000

The viewport-specific state used by the SF unit (SCISSOR_RECT) is stored as an array of up to 16 elements, each of which contains the DWords described below. The start of each element is spaced 2 DWords apart. The location of first element of the array, as specified by Pointer to SCISSOR_RECT, is aligned to a 32-byte boundary.

DWord	Bit	Description
0	31:16	**Scissor Rectangle Y Min** Format: U16 Pixels from Drawing Rectangle origin (upper left corner) Specifies Y Min coordinate of (inclusive) Scissor Rectangle used for scissor test. Pixels with (Draw Rectangle-relative) Y coordinates less than Y Min will be clipped out if Scissor Rectangle is enabled. NOTE: If Y Min is set to a value greater than Y Max, all primitives will be discarded for this viewport. Value: [0,16383] — Name:
	15:0	**Scissor Rectangle X Min** Format: U16 Pixels from Drawing Rectangle origin (upper left corner) Specifies X Min coordinate of (inclusive) Scissor Rectangle used for scissor test. Pixels with (Draw Rectangle-relative) X coordinates less than X Min will be clipped out if Scissor Rectangle is enabled. NOTE: If X Min is set to a value greater than X Max, all primitives will be discarded for this viewport. Value: [0,16383] — Name:
1	31:16	**Scissor Rectangle Y Max** Format: U16 Pixels from Drawing Rectangle origin (upper left corner) Specifies Y Max coordinate of (inclusive) Scissor Rectangle used for scissor test. Pixels with (Draw Rectangle-relative) Y coordinates greater than Y Max will be clipped out if Scissor Rectangle is enabled. Value: [0,16383] — Name:
	15:0	**Scissor Rectangle X Max** Format: U16 Pixels from Drawing Rectangle origin (upper left corner) Specifies X Max coordinate of (inclusive) Scissor Rectangle used for scissor test. Pixels with (Draw Rectangle-relative) Y coordinates greater than X Max will be clipped out if Scissor Rectangle is enabled. Value: 0-16383 — Name:

SF_CLIP_VIEWPORT

Source: RenderCS

Size (in bits): 512

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:0	**Viewport Matrix Element m00** Format: IEEE_Float
1	31:0	**Viewport Matrix Element m11** Format: IEEE_Float
2	31:0	**Viewport Matrix Element m22** Format: IEEE_Float Total Length - 2
3	31:0	**Viewport Matrix Element m30** Format: IEEE_Float Total Length - 2
4	31:0	**Viewport Matrix Element m31** Format: IEEE_Float
5	31:0	**Viewport Matrix Element m32** Format: IEEE_Float
6	31:0	**Reserved**
7	31:0	**Reserved** Format: MBZ
8	31:0	**X Min Clip Guardband** Default Value: 0h Excludes DWord (0,1) Format: FLOAT32 . This 32-bit float represents the XMin guardband boundary (normalized to Viewport.XMin == -1.0f). This corresponds to the left boundary of the NDC guardband.
9	31:0	**X Max Clip Guardband** Default Value: 0h Excludes DWord (0,1) Format: FLOAT32 This 32-bit float represents the XMax guardband boundary (normalized to Viewport..XMax == 1.0f). This corresponds to the right boundary of the NDC guardband.
10	31:0	**Y Min Clip Guardband** Format: FLOAT32 This 32-bit float represents the YMin guardband boundary (normalized to Viewport.YMin == -1.0f). This corresponds to the bottom boundary of the NDC guardband.

SF_CLIP_VIEWPORT		
11	31:0	**Y Max Clip Guardband:** Format: FLOAT32 This 32-bit float represents the YMax guardband boundary (normalized to Viewport.YMax == 1.0f). This corresponds to the top boundary of the NDC guardband.
12..15	31:0	**Reserved** Format: MBZ

MDP_RTW_8DS - SIMD8 Dual Source Render Target Data Payload

Source: BSpec

Size (in bits): 2048

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Src0 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Red
1.0-1.7	255:0	**Src0 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Green
2.0-2.7	255:0	**Src0 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Blue
3.0-3.7	255:0	**Src0 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Alpha
4.0-4.7	255:0	**Src1 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Red
5.0-5.7	255:0	**Src1 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Green
6.0-6.7	255:0	**Src1 Blue** Format: MDP_DW_SIMD8

MDP_RTW_8DS - SIMD8 Dual Source Render Target Data Payload

		Slots[7:0] or [15:8] of Src1 Blue
7.0-7.7	255:0	**Src1 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Alpha

MDP_RTW_8 - SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1024

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
2.0-2.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
3.0-3.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha

MAP32B_USU_SIMD8 - SIMD8 Untyped BUFFER Surface 32-Bit Address Payload

Source:	BSpec
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U** Format: MACR_32b Specifies the U channel for slots [7:0]

MAP64B_USU_SIMD8 - SIMD8 Untyped BUFFER Surface 64-Bit Address Payload

Source:	BSpec
Size (in bits):	512
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U3_U0** Format: MACR_64b Specifies the U channel for slots [3:0]
1.0-1.7	255:0	**U7_U4** Format: MACR_64b Specifies the U channel for slots [7:4]

MAP32B_USUV_SIMD8 - SIMD8 Untyped STRBUF Surface 32-Bit Address Payload

Source:	BSpec
Size (in bits):	512
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U** Format: MACR_32b Specifies the U channel for slots [7:0]
1.0-1.7	255:0	**V** Format: MACR_32b Specifies the V channel for slots [7:0]

MDP_RTW_16 - SIMD16 Render Target Data Payload

Source: BSpec

Size (in bits): 2048

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD16 Slots [15:8] Red
2.0-2.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Green[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Green
4.0-4.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Blue
6.0-6.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha

MDP_RTW_16 - SIMD16 Render Target Data Payload

7.0-7.7	255:0	**Alpha[15:7]** Format: MDP_DW_SIMD8 Slots [15:7] Alpha

MAP32B_USU_SIMD16 - SIMD16 Untyped BUFFER Surface 32-Bit Address Payload

Source:	BSpec
Size (in bits):	512
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U[7:0]** Format: MACR_32b Specifies the U channel for slots [7:0]
1.0-1.7	255:0	**U[15:8]** Format: MACR_32b Specifies the U channel for slots [15:8]

MAP64B_USU_SIMD16 - SIMD16 Untyped BUFFER Surface 64-Bit Address Payload

Source:	BSpec
Size (in bits):	1024
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U3_U0** Format: MACR_64b Specifies the U channel for slots [3:0]
1.0-1.7	255:0	**U7_U4** Format: MACR_64b Specifies the U channel for slots [7:4]
2.0-2.7	255:0	**U11_U8** Format: MACR_64b Specifies the U channel for slots [11:8]
3.0-3.7	255:0	**U15_U12** Format: MACR_64b Specifies the U channel for slots [15:12]

MAP32B_USUV_SIMD16 - SIMD16 Untyped STRBUF Surface 32-Bit Address Payload

Source:	BSpec
Size (in bits):	1024
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**U7_U0** Format: MACR_32b Specifies the U channel for slots [7:0]
1.0-1.7	255:0	**U15_U8** Format: MACR_32b Specifies the U channel for slots [15:8]
2.0-2.7	255:0	**V7_V0** Format: MACR_32b Specifies the V channel for slots [7:0]
3.0-3.7	255:0	**V15_V8** Format: MACR_32b Specifies the V channel for slots [15:8]

SO_DECL

Source:	RenderCS
Size (in bits):	16
Default Value:	0x00000000

A list of SO_DECL structures are passed in the 3DSTATE_SO_DECL_LIST command. Each structure specifies either (a) the source and destination of an up-to-4-DWord appending write into an SO buffer, or (b) how many DWords to skip over in the destination SO buffer (i.e., a "hole" where the previous buffer contents are maintained).

DWord	Bit	Description
0	15:14	**Reserved** Format: MBZ
	13:12	**Output Buffer Slot** Format: U2 Buffer Index This field selects the destination output buffer slot.
	11	**Hole Flag** Format: Flag If set, the Component Mask field indirectly specifies a number of 32-bit locations to skip over (leave unmodified in memory) in the selected output buffer. The Register Index field is ignored. The only permitted Component Mask values are as follows: 0x0 No Dwords are skipped over (SO_DECL performs no operation) 0x1 (X) Skip 1 DWord 0x3 (XY) Skip 2 DWords 0x7 (XYZ) Skip 3 DWords 0xF (XYZW) Skip 4 DWords
	10	**Reserved** Format: MBZ
	9:4	**Register Index** Format: U6 128-bit granular offset into the source vertex read data If Hole Flag is clear, this field specifies the 128-bit offset into the source vertex data which supplies the source data to be written to the destination buffer, where the individual 32-component destination locations are selected by Component Mask. e.g., Register Index 0 corresponds with the first 128 bits of the data read from the vertex URB entry (as per corresponding Vertex Read Offset state) There is only enough internal storage for the 128-bit vertex header and 32 128-bit vertex attributes. Value: [0,32] — Name:

SO_DECL

		0h **[Default]**

Programming Notes

It is the responsibility of software to map any API-visible source data specifications (e.g., vertex register number) into 128-bit granular URB read offsets.

3:0 **Component Mask**

Format:	MASK 4-bit Mask

This field is a 4-bit bitmask that selects which contiguous 32-bit component(s) are either written or skipped-over in the destination buffer.

If this field is zero the SO_DECL operation is effectively a no-op. No data will be appended to the destination and the destination buffer's write pointer will not be advanced.

If the **Hole Flag** is set, this field (if non-zero) indirectly specifies how much the destination buffer's write pointer should be advanced. See **Hole Flag** description above for restrictions on this field.

If the **Hole Flag** is clear, this field (if non-zero) selects which source components are to be written to the destination buffer. The components must be contiguous, e.g. YZW is legal, but XZW is not. The selected source components are written to the destination buffer starting at the current write pointer, and then the write pointer is advanced past the written data. E.g., if YZW is specified, the three (YZW) components of the source register will be written to the destination buffer at the current write pointer, and the write pointer will be advanced by 3 DWords.

Value	Name
0h	**[Default]**
xxx1b	SO_DECL_COMPMASK_X
xx1xb	SO_DECL_COMPMASK_Y
x1xxb	SO_DECL_COMPMASK_Z
1xxxb	SO_DECL_COMPMASK_W

SplitBaseAddress4KByteAligned

Source:	BSpec
Size (in bits):	32
Default Value:	0x00000000

Specifies a 64-bit, 4K-byte aligned memory base address.

DWord	Bit	Description
0	31:12	**Base Address Low** Format: GraphicsAddress[31:12]
	11:0	**Reserved** Format: MBZ

SplitBaseAddress64ByteAligned

Source:	BSpec
Size (in bits):	32
Default Value:	0x00000000

Specifies a 64-bit, 64-byte aligned memory base address.

DWord	Bit	Description
0	31:6	**Base Address Low** Format: GraphicsAddress[31:6]
	5:0	**Reserved** Format: MBZ

SrcRegNum

Source:	EuIsa
Size (in bits):	8
Default Value:	0x00000000

Description
Register Number This field provides the register number for the operand. For GRF register operand, it provides the portion of register address aligning to 256-bit. For an ARF register operand, this field is encoded such that MSBs identify the architecture register type and LSBs provide its register number. This field together with the corresponding SubRegNum field provides the byte aligned address for the origin of the register region. Specifically, this field provides bits [12:5] of the byte address, while SubRegNum field provides bits [4:0]. This field applies to the destination operand and the source operands. It is ignored (or not present in the instruction word) for an immediate source operand. This field is present if the operand is in direct addressing mode; it is not present if the operand is register-indirect addressed.

DWord	Bit	Description
0	7:0	**Source Register Number**

Value	Name	Description
0-127	If {Dst/Src0/Src1/Src2}.RegFile==GRF	
0-0ffh	If {Dst/Src0/Src1/Src2}.RegFile==ARF	This field is used to encode the architecture register as well as providing the register number. See GEN Execution Environment chapter for details.

SrcSubRegNum

Source:	EuIsa
Size (in bits):	5
Default Value:	0x00000000

Description

Subregister Number
This field provides the sub-register number for the operand. For a GRF register operand, it provides the byte address within a 256-bit register. For an ARF register operand, this field also provides the sub-register number according to the encoding defined for the given architecture register.
This field together with the corresponding RegNum field provides the byte aligned address for the origin of the register region. Specifically, this field provides bits [4:0] of the byte address, while the RegNum field provides bits [12:5].
This field applies to the destination operand and the source operands. It is ignored (or not present in the instruction word) for an immediate source operand.
This field is present if the operand is in direct addressing mode; it is not present if the operand is register-indirect addressed.

Programming Notes

Note: The recommended instruction syntax uses subregister numbers within the GRF in units of actual data element size, corresponding to the data type used. For example for the F (Float) type, the assembler syntax uses subregister numbers 0 to 7, corresponding to subregister byte addresses of 0 to 28 in steps of 4, the element size.

DWord	Bit	Description
0	4:0	**Source Sub Register Number**

Value	Name	Description
0-31	If {Dst/Src0/Src1/Src2}.RegFile==GRF	
0-0ffh	If {Dst/Src0/Src1/Src2}.RegFile==ARF	This field is used to encode the architecture register as well as providing the register number. See GEN Execution Environment chapter for details.

SURFACE_STATE

Source: BSpec
Size (in bits): 160
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000003, 0x00000000

DWord	Bit	Description
0	31:2	**Reserved** Format: MBZ
	1:0	**Surface mode** Format: U2 Value / Name / Description 0 / Display Surface / 1 / Reconstructed Surfaces / 2 / Scaled surfaces / 3 / Reserved /
1	31:4	**Reserved** Format: MBZ Surface Base Address is NOT used for codec H/W. This field is reserved for 3D surface state compatibility. VDEnc pipeline gets this address from VDENC_PIPE_BUF_ADDR_STATE for different buffers.
	3:0	**Reserved** Format: MBZ
2	31:18	**Height** Format: U14-1 This field specifies the height of the Picture in units of pixels. For PLANAR surface formats, this field indicates the height of the Y (luma) plane. Note: Video Codecs must program less than and equal to 4K. Value / Name / Description [0,16383] / / representing heights [1,16384] **Programming Notes** • For AVC: For frame picture is a multiple of 16.
	17:4	**Width** Format: U14-1 This field specifies the width of the Picture in units of pixels/residuals. For PLANAR surface formats, this field indicates the width of the Y (luma) plane. Value / Name / Description [0,16383] / / representing widths [1,16384]

SURFACE_STATE

		Programming Notes • The Width specified by this field multiplied by the pixel size in bytes must be less than or equal to the surface pitch (specified in bytes via the Surface Pitch field). • Width (field value + 1) must be a multiple of 2 for PLANAR_420, • VDEnc HW does not use this field, the picture width is read from IMG State instead, because this field may not equal to the actual picture width. This field is used by the KMD to allocate surface in GTT.
	3:2	**Reserved** Format: MBZ
	1:0	**Cr(V)/Cb(U) Pixel Offset V Direction** Format: U0.2 Exactly as shown in the original spec. Specifies the distance to the U/V values with respect to the even numbered Y channels in the V direction. **Programming Notes** This field is ignored for all formats except PLANAR_420_8.
3	31:28	**Surface Format** Format: U4 Specifies the format of the surface. This field must be set to 4 - PLANAR_420_8 for VDEnc usage.

Value	Name	Description
0	YUV 4:2:2	Input format from DE: YUV 16-bit 4:2:2 packed (8:8:8:8 MSB- Y1: U1: Y2: V1; Drop U2: and V2)
1	RGBA 4:4:4:4	RGBA 32-bit 4:4:4:4 packed (8:8:8:8 MSB-X:B:G:R)
2	YUV 4:4:4	YUV 32-bit 4:4:4:4 packed (8:8:8:8 MSB-Y:U:X:V)
3	Y8_UNORM	
4	PLANAR_420_8	(NV12, IMC1,2,3,4, YV12)
5,15	Reserved	

	27	**Interleave Chroma** Format: Enable This field indicates that the chroma fields are interleaved in a single plane rather than stored as two separate planes. This field is only used for PLANAR surface formats.

SURFACE_STATE

Value	Name
0	Disable
1	Enable

Programming Notes

26 **Reserved**

Format:	MBZ

25:22 **Reserved**

Format:	MBZ

21:20 **Reserved**

Format:	MBZ

19:3 **Surface Pitch**

Format:	U17-1

This field specifies the surface pitch in (#Bytes).

Value	Name	Description
[0,2047]		to [1B, 2048B]

Programming Notes
For tiled surfaces, the pitch must be a multiple of the tile width (i.e.128 bytes aligned). If Half Pitch for Chroma is set, this field must be a multiple of two tile widths for tiled surfaces, or a multiple of 2 bytes for linear surfaces. For Y-tiled surfaces: Range = [127, 524287] to [128B,256KB] = [1 tile, 2048 tiles].

2 **Half Pitch for Chroma**

Format:	Enable

(This field must be set to Disable.) This field indicates that the chroma plane(s) will use a pitch equal to half the value specified in the Surface Pitch field. This field is only used for PLANAR surface formats.

Value	Name	Description
0	Disable **[Default]**	
1	Enable	

1 **Tiled Surface**

Format:	Boolean

(This field must be set to TRUE: Tiled.) This field specifies whether the surface is tiled. This field is ignored by VDEnc usage.

Value	Name	Description
0	False	Linear
1	True **[Default]**	Tiled

SURFACE_STATE

		Programming Notes Linear surfaces can be mapped to Main Memory (uncached) or System Memory (cacheable, snooped). Tiled surfaces can only be mapped to Main Memory. The corresponding cache(s) must be invalidated before a previously accessed surface is accessed again with an altered state of this bit.
	0	**Tile Walk** Format: 3D_Tilewalk (This field must be set to 1: TILEWALK_YMAJOR.) This field specifies the type of memory tiling (XMajor or YMajor) employed to tile this surface. See Memory Interface Functions for details on memory tiling and restrictions. This field is ignored when the surface is linear. Internally H/W is always treated this set to 1 for all VDEnc usage. **Value** / **Name** / **Description** 0h / XMAJOR / TILEWALK_XMAJOR 1h / YMAJOR **[Default]** / TILEWALK_YMAJOR **Programming Notes** The corresponding cache(s) must be invalidated before a previously accessed surface is accessed again with an altered state of this bit.
4	31	**Reserved** Format: MBZ
	30:16	**X Offset for U(Cb)** Format: U15 Pixel Offset This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave Chroma is enabled. This field is only used for PLANAR surface formats. This field must be set to zero. X Offset for U(Cb) in pixel (This field must be zero for NV12 and IMC 1 and 3). **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must be zero.
	15	**Reserved** Format: MBZ
	14:0	**Y Offset for U(Cb)** Format: U15 Pixel Row Offset This field specifies the vertical offset in rows from the Surface Base Address to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave Chroma is enabled. This field is only used for PLANAR surface formats.

SURFACE_STATE

		Programming Notes For PLANAR_420 and PLANAR_422 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs.

SurfaceAddress

Source: BSpec
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Base Address 4KByte Aligned** Format: BaseAddress4KByteAligned
2	31:0	**Surface Address Attributes** Format: SurfaceAddressAttributes

SurfaceAddress_64ByteAligned_CM

Source: BSpec
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Base Address 64Byte Aligned** Format: BaseAddress64ByteAligned
2	31:0	**Surface Address Attributes With CM** Format: SurfaceAddressAttributesWithCompressionMode

SurfaceAddress_64ByteAligned_CM_ScratchBufferCacheSelect

Source:	BSpec
Size (in bits):	96
Default Value:	0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Base Address 64Byte Aligned** Format: BaseAddress64ByteAligned
2	31:0	**Surface Address Attributes With CM AND SBCS** Format: SurfaceAddressAttributes_CM_ScratchBufferCacheSelect

SurfaceAddress_CM_ScratchBufferCacheSelect

Source: BSpec
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Base Address 4KByte Aligned** Format: BaseAddress4KByteAligned
2	31:0	**Surface Address Attributes With CM AND SBCS** Format: SurfaceAddressAttributes_CM_ScratchBufferCacheSelect

SurfaceAddressAttributes

Source: BSpec
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:15	**Reserved** Format: MBZ
	14:13	**Reserved**
	12:11	**Reserved** Format: MBZ
	10:9	**Reserved** Format: MBZ
	8:7	**Arbitration Priority Control** Format: ARBITRATION_PRIORITY
	6:1	**Index to Memory Object Control State (MOCS) Tables** Format: U6 The index to define the L3 and system cache memory properties. The details of the controls are further defined in L3 and Page walker (memory interface) control registers. The field is defined to populate 64 different surface controls to be used concurrently. Related control registers can be updated during runtime.
	0	**Reserved**

SurfaceAddressAttributes_CM_ScratchBufferCacheSelect

Source: BSpec
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:15	**Reserved** Format: MBZ
	14:13	**Reserved**
	12	**Scratch Buffer Cache Select** Format: U1 This field must be 1, indicating Internal Media Storage.
	11	**Reserved** Format: MBZ
	10	**Memory Compression Mode** Format: U1 Distinguishes vertical from horizontal compression. Please refer to vol1a **Memory Data Formats chapter - section** media Memory Compression for more details. Value: 0b — Horizontal Compression Mode Value: 1b — Vertical Compression Mode
	9	**Memory Compression Enable** Format: Enable Memory compression will be attempted for this surface.
	8:7	**Arbitration Priority Control** Format: ARBITRATION_PRIORITY
	6:1	**Index to Memory Object Control State (MOCS) Tables** Format: U6 The index to define the L3 and system cache memory properties. The details of the controls are further defined in L3 and Page walker (memory interface) control registers. The field is defined to populate 64 different surface controls to be used concurrently. Related control registers can be updated during runtime.
	0	**Reserved**

SurfaceAddressAttributesWithCompressionMode

Source: BSpec
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:15	**Reserved** Format: MBZ
	14:13	**Reserved**
	12:11	**Reserved** Format: MBZ
	10	**Memory Compression Mode** Format: U1 Distinguishes vertical from horizontal compression. Please refer to vol1a **Memory Data Formats chapter - section** media Memory Compression for more details. Value: 0b — Horizontal Compression Mode Value: 1b — Vertical Compression Mode
	9	**Memory Compression Enable** Format: Enable Memory compression will be attempted for this surface.
	8:7	**Arbitration Priority Control** Format: ARBITRATION_PRIORITY
	6:1	**Index to Memory Object Control State (MOCS) Tables** Format: U6 The index to define the L3 and system cache memory properties. The details of the controls are further defined in L3 and Page walker (memory interface) control registers. The field is defined to populate 64 different surface controls to be used concurrently. Related control registers can be updated during runtime.
	0	**Reserved**

SurfaceAddressWithCompression

Source: BSpec
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Base Address 4KByte Aligned** Format: BaseAddress4KByteAligned
2	31:0	**Surface Address Attributes With CM** Format: SurfaceAddressAttributesWithCompressionMode

MDP_RTW_ZMA8 - SZ OM S0A SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1792

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**oMask** Format: MDPR_OMASK Slots [7:0] oMask. Upper half ignored.
2.0-2.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
6.0-6.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

MDP_RTW_ZMA8 - SZ OM S0A SIMD8 Render Target Data Payload

MDP_RTW_ZMA16 - SZ OM S0A SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	3328
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-1.7	511:0	**Source 0 Alpha** Format: MDP_DW_SIMD16 Slots [15:0] Source 0 Alpha
2.0-2.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
3.0-4.7	511:0	**Red** Format: MDP_DW_SIMD16 Slots [15:0] Red
5.0-6.7	511:0	**Green** Format: MDP_DW_SIMD16 Slots [15:0] Green
7.0-8.7	511:0	**Blue** Format: MDP_DW_SIMD16 Slots [15:0] Blue

MDP_RTW_ZMA16 - SZ OM S0A SIMD16 Render Target Data Payload		
9.0-10.7	511:0	**Alpha** Format: MDP_DW_SIMD16 Slots [15:0] Alpha
11.0-12.7	511:0	**Source Depth** Format: MDP_DW_SIMD16 Slots [15:0] Source Depth

MDP_RTW_ZM8DS - SZ OM SIMD8 Dual Source Render Target Data Payload

Source:	BSpec
Size (in bits):	2560
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK oMask for slots [7:0] and [15:8]. Operation selects upper or lower half.
1.0-1.7	255:0	**Src0 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Red
2.0-2.7	255:0	**Src0 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Green
3.0-3.7	255:0	**Src0 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Blue
4.0-4.7	255:0	**Src0 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Alpha
5.0-5.7	255:0	**Src1 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Red

MDP_RTW_ZM8DS - SZ OM SIMD8 Dual Source Render Target Data Payload

6.0-6.7	255:0	**Src1 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Green
7.0-7.7	255:0	**Src1 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Blue
8.0-8.7	255:0	**Src1 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Alpha
9.0-9.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] or [15:8] of Source Depth

MDP_RTW_ZM8 - SZ OM SIMD8 Render Target Data Payload

Source:	BSpec
Size (in bits):	1536
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK Slots [7:0] oMask. Upper half ignored.
1.0-1.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

MDP_RTW_ZM16 - SZ OM SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	2816
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
1.0-1.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
3.0-3.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Green[15:7]** Format: MDP_DW_SIMD8 Slots [15:8] Green
5.0-5.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8

MDP_RTW_ZM16 - SZ OM SIMD16 Render Target Data Payload

		Slots [7:0] Blue
6.0-6.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Blue
7.0-7.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
8.0-8.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha
9.0-9.7	255:0	**Source Depth[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
10.0-10.7	255:0	**Source Depth[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

MDP_RTW_ZA8 - SZ S0A SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1536

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

MDP_RTW_ZA16 - SZ S0A SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	3072
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Source 0 Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source 0 Alpha
2.0-2.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
4.0-4.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
5.0-5.7	255:0	**Green[15:8]**

MDP_RTW_ZA16 - SZ S0A SIMD16 Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots [15:8] Green
6.0-6.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Blue
7.0-7.7	255:0	**Blue[15:7]** Format: MDP_DW_SIMD8 Slots [15:8] Blue
8.0-8.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
9.0-9.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha
10.0-10.7	255:0	**Source Depth[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
11.0-11.7	255:0	**Source Depth[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

MDP_RTW_Z8DS - SZ SIMD8 Dual Source Render Target Data Payload

Source:	BSpec
Size (in bits):	2304
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Src0 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Red
1.0-1.7	255:0	**Src0 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Green
2.0-2.7	255:0	**Src0 Blue** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Blue
3.0-3.7	255:0	**Src0 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src0 Alpha
4.0-4.7	255:0	**Src1 Red** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Red
5.0-5.7	255:0	**Src1 Green** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Green
6.0-6.7	255:0	**Src1 Blue**

MDP_RTW_Z8DS - SZ SIMD8 Dual Source Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Blue
7.0-7.7	255:0	**Src1 Alpha** Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Alpha
8.0-8.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] or [15:8] of Source Depth

MDP_RTW_Z8 - SZ SIMD8 Render Target Data Payload

Source: BSpec

Size (in bits): 1280

Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Red** Format: MDP_DW_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Green** Format: MDP_DW_SIMD8 Slots [7:0] Green
2.0-2.7	255:0	**Blue** Format: MDP_DW_SIMD8 Slots [7:0] Blue
3.0-3.7	255:0	**Alpha** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
4.0-4.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

MDP_RTW_Z16 - SZ SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	2560
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Red[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Red[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Red
2.0-2.7	255:0	**Green[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Green[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Green
4.0-4.7	255:0	**Blue[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Blue[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Blue

MDP_RTW_Z16 - SZ SIMD16 Render Target Data Payload

6.0-6.7	255:0	**Alpha[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Alpha
7.0-7.7	255:0	**Alpha[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Alpha
8.0-8.7	255:0	**Source Depth[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
9.0-9.7	255:0	**Source Depth[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

Thread Spawn Message Descriptor

Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000

DWord	Bit	Description
0	31:20	**Reserved** Format: MBZ
	19	**Header Present** Format: MBZ **Programming Notes** This bit MBZ for all Thread Spawner messages.
	18:5	**Reserved** Format: MBZ
	4	**Resource Select** This field specifies the resource associated with the action taken by the Opcode. Value / Name / Description / Exists If: 0 / / Spawn a Child Thread / [Opcode] == 'Spawn Thread' 1 / / Spawn a Root Thread / [Opcode] == 'Spawn Thread' 0 / / The URB Handle is Dereferenced / [Opcode] == 'Dereference Resource' 1 / / The URBHhandle is NOT Dereferenced / [Opcode] == 'Dereference Resource'
	3:2	**Reserved** Format: MBZ
	1	**Requester Type** This field indicates whether the requesting thread is a root thread or a child thread. If it is a root thread, when Opcode is 0, FF managed resources are dereferenced. If it is a child thread and Opcode is 0, no resource is dereferenced; no action is required by the TS. Value / Name: 0 / Root Thread 1 / Child Thread
	0	**Opcode** Indicates the operation performed by the message. A root thread must terminate with a message to TS (Opcode == 0 and EOT == 1). A child thread should also terminate with such a message. A thread cannot terminate with an Opcode of "spawn thread". Value / Name / Description: 0 / Dereference Resource / also used for end of thread 1 / Spawn Thread /

VC1

Source: VideoCS
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15:8	**Reserved** Format: MBZ
	7	**Syncmarker Error** This flag indicates missing sync marker SEs coded in the bit-stream.
	6	**Mbmode SE Error** This flag indicates inconsistent Macroblock SEs coded in the bit-stream.
	5	**Transformtype SE Error** This flag indicates inconsistent transform type SEs coded in the bit-stream.
	4	**Coefficient Error** This flag indicates inconsistent Coefficient SEs coded in the bit-stream.
	3	**Motion Vector SE Error** This flag indicates inconsistent Motion Vector SEs coded in the bit-stream.
	2	**Coded Block Pattern CY SE Error** This flag indicates inconsistent CBPCY SEs coded in the bit-stream.
	1	**Mquant Error** This flag indicates inconsistent MQUANT SEs coded in the bit-stream.
	0	**MB Concealment Flag** . Each pulse from this flag indicates one MB is concealed by hardware.

VCS Hardware-Detected Error Bit Definitions

Source: VideoCS
Size (in bits): 16
Default Value: 0x00000000

DWord	Bit	Description
0	15:3	**Reserved** Format: MBZ
	2	**Reserved** Format: MBZ
	1	**Reserved** Format: MBZ
	0	**Instruction Error** This bit is set when the Renderer Instruction Parser detects an error while parsing an instruction. Instruction errors include: • Client ID value (Bits 31:29 of the Header) is not supported (only MI, 2D and 3D are supported). • Defeatured MI Instruction Opcodes: **Value** / **Name** / **Description** 1 / / Instruction Error detected **Programming Notes** This error indications cannot be cleared except by reset (i.e., it is a fatal error).

VEBOX_Ch_Dir_Filter_Coefficient

Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:56	**Filter Coefficient[7]** Format: S1.6 2's Complement **Range:** [-2, +2)
	55:48	**Filter Coefficient[6]** Format: S1.6 2's Complement **Range:** [-2, +2)
	47:40	**Filter Coefficient[5]** Format: S1.6 2's Complement **Range:** [-2, +2)
	39:32	**Filter Coefficient[4]** Format: S1.6 2's Complement **Range:** [-2, +2)
	31:24	**Filter Coefficient[3]** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**Filter Coefficient[2]** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**Filter Coefficient[1]** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**Filter Coefficient[0]** Format: S1.6 2's Complement **Range:** [-2, +2)

VEBOX_Filter_Coefficient

Source:	BSpec
Size (in bits):	8
Default Value:	0x00000000

DWord	Bit	Description
0	7:0	**2's Complement Filter Coefficient** Format: S1.6 2's Complement **Range:** [-2, +2)

VERTEX_BUFFER_STATE

Source: RenderCS
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000

This structure is used in 3DSTATE_VERTEX_BUFFERS to set the state associated with a VB. The VF function will use this state to determine how/where to extract vertex element data for all vertex elements associated with the VB.
The VERTEX_BUFFER_STATE structure is 4 DWords for both INSTANCEDATA and VERTEXDATA buffers. A VB is defined as a 1D array of vertex data structures, accessed via a computed index value. The VF function therefore needs to know the starting address of the first structure (index 0) and size of the vertex data structure.

Programming Notes

Vertex element accesses which straddle or go past the VB's End Address will return 0's for all elements.

DWord	Bit	Description
0	31:26	**Vertex Buffer Index** Format: U6 Index This field contains an index value which selects the VB state being defined. Value: [0,32] — Name:
	25:21	**Reserved** Format: MBZ
	20	**Buffer Access Type** This field determines how vertex element data is extracted from this VB. This control applies to all vertex elements associated with this VB. 00b — VERTEXDATA — For SEQUENTIAL vertex access, each vertex of an instance is sourced from sequential structures within the VB. For RANDOM vertex access, each vertex of an instance is looked up (separately) via a computed index value 01b — INSTANCEDATA — Each vertex of an instance is sourced with the same (instance) data. Subsequent instances may be sourced with the same or different data, depending on Instance Data Step Rate.
	19:16	**Vertex Buffer Memory Object Control State** Format: MEMORY_OBJECT_CONTROL_STATE Specifies the memory object control state for this vertex buffer.
	15	**Reserved** Format: MBZ
	14	**Address Modify Enable**

VERTEX_BUFFER_STATE

If set, the Buffer Starting Address and End Address fields are used to update the state of this buffer. If clear, those fields are ignored and the previously-programmed values are maintained.

13 Null Vertex Buffer

Format:	Enable

This field enabled causes any fetch for vertex data to return 0.

12 Vertex Fetch Invalidate

Default Value:	0h

Invalidate the Vertex overfetch cache when this bit is set. For multiple vertex buffer state structures in one packet, this bit may be set only once in the entire packet.

11:0 Buffer Pitch

Format:	U12 Count of bytes

This field specifies the pitch in bytes of the structures accessed within the VB. This information is required in order to access elements in the VB via a structure index.

Value	Name	Description
[0,2048]		Bytes

Programming Notes

- Different VERTEX_BUFFER_STATE structures can refer to the same memory region using different Buffer Pitch values.
- See note on 64-bit float alignment in Buffer Starting Address.

1 31:0 Buffer Starting Address

Format:	GraphicsAddress[31:0]

Description

This field contains the byte-aligned Graphics Address of the first element of interest within the VB. Software must program this value with the combination (sum) of the base address of the memory resource and the byte offset from the base address to the starting structure within the buffer.

If the Address ModifyEnable bit is clear, this field is ignored and the previous value of Buffer Starting Address for this buffer is maintained.

Programming Notes

64-bit floating point values must be 64-bit aligned in memory, or UNPREDICTABLE data will be fetched. When accessing an element containing 64-bit floating point values, the Buffer Starting Address and Source Element Offset values must add to a 64-bit aligned address, and BufferPitch must be a multiple of 64-bits.

VBs can only be allocated in linear (not tiled) graphics memory.

As computed index values are, by definition, interpreted as unsigned values, there is no issue with accesses to locations before (lower address value) the start of the buffer. However, these

VERTEX_BUFFER_STATE

		wrapped indices are subject to Max Index checking (see below).
2	31:0	**End Address** Format: GraphicsAddress[31:0]U32 **Description** This field defines the address of the last valid byte in this particular VB. Access of a vertex element which either straddles or is beyond this address will return 0's for any data read. If the Address ModifyEnable bit is clear, this field is ignored and the previous value of End Address for this buffer is maintained. **Value** / **Name** [0,FFFFFFFFh] 0h **[Default]**
3	31:0	**Instance Data Step Rate** Format: U32 This field only applies to INSTANCEDATA buffers - it is ignored (but still present) for VERTEXDATA buffers). Only after the number of instances specified by this field is generated is new (sequential) instance data provided. This process continues for each group of instances defined in the draw command. For example, a value of 1 in this field causes new instance data to be supplied with each sequential (instance) group of vertices. A value of 2 causes every other instance group of vertices to be provided with new instance data. The special value of 0 causes all vertices of all instances generated by the draw command to be provided with the same instance data. (The same effect can be achieved by setting this field to its maximum value.)

VERTEX_ELEMENT_STATE

Source:	RenderCS
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Description
This structure is used in 3DSTATE_VERTEX_ELEMENTS to set the state associated with a vertex element. A vertex element is defined as an entity supplying from 1 to 4 DWord vertex components to be stored in the vertex URB entry. The number of supported vertex elements is:
34
The VF function will use this state, and possibly the state of the associated vertex buffer, to fetch/generate the source vertex element data, perform any required format conversions, padding with zeros, and store the resulting destination vertex element data into the vertex URB entry.

Programming Notes
The SourceElementFormat needs to be a single-component format with an element which has edge flag enabled.

DWord	Bit	Description
0	31:26	**Vertex Buffer Index** Format: U6 This field specifies which vertex buffer the element is sourced from. Value: [0,32] — Name: Up to 33 VBs are supported **Programming Notes**: It is possible for a vertex element to include only internally-generated data (VertexID, etc.), in which case the associated vertex buffer state is ignored.
	25	**Valid** Format: Boolean Value: 1h — Name: TRUE — Description: this vertex element is used in vertex assembly Value: 0h — Name: FALSE — Description: this vertex element is not used.
	24:16	**Source Element Format** Format: SURFACE_FORMAT Range: Valid formats are found in the 3D Primitive Processing FormatConversion portion of the vertex fetch chapter. Format: The encoding of this field is identical the Surface Format field of the SURFACE_STATE

VERTEX_ELEMENT_STATE

structure, as described in the Sampler chapter.

This field specifies the format in which the memory-resident source data for this particular vertex element is stored in the memory buffer. This only applies to elements stored with VFCOMP_STORE_SRC component control. (All other component types have an explicit format).

15 **Edge Flag Enable**

Format:	Enable

Description

When ENABLED, the source element is interpreted as an EdgeFlag for the vertex. If the source element is zero, the EdgeFlag will be set to FALSE. If the source element is non-zero, the EdgeFlag will be set to TRUE. The EdgeFlag bit will travel down the fixed function pipeline along with the vertex handle, etc. and not be stored in the vertex data like the other vertex elements. Refer to the fixed function descriptions for how this EdgeFlag affects rendering.
Edge flags are supported for the following primitive topology types only, otherwise EdgeFlagEnable must not be ENABLED.

- 3DPRIM_TRILIST*
- 3DPRIM_TRISTRIP*
- 3DPRIM_TRIFAN*
- 3DPRIM_POLYGON

If this bit is DISABLED for all valid VERTEX_ELEMENTs, the vertex will be assigned a default EdgeFlag of TRUE.

Edge flags are supported for all primitive topology types.

Programming Notes

- This bit must only be ENABLED on the last valid VERTEX_ELEMENT structure.
- When set, Component 0 Control must be set to VFCOMP_STORE_SRC, and Component 1-3 Control must be set to VFCOMP_NOSTORE.

14:12 **Reserved**

Format:	MBZ

11:0 **Source Element Offset**

Format:	U12 byte offset

Byte offset of the source vertex element data in the structures comprising the vertex buffer.

Value	Name
[0,4095]	

Programming Notes

See note on 64-bit float alignment in Buffer Starting Address.

VERTEX_ELEMENT_STATE

1	31	**Reserved** Format: MBZ
	30:28	**Component 0 Control** Format: 3D_VertexComponentControl Refer to the 3D_VertexComponentControl table below
	27	**Reserved** Format: MBZ
	26:24	**Component 1 Control** Format: 3D_VertexComponentControl Refer to the 3D_VertexComponentControl table below
	23	**Reserved** Format: MBZ
	22:20	**Component 2 Control** Format: 3D_VertexComponentControl Refer to the 3D_VertexComponentControl table below
	19	**Reserved** Format: MBZ
	18:16	**Component 3 Control** Format: 3D_VertexComponentControl Refer to the 3D_VertexComponentControl table below
	15:8	**Reserved** Format: MBZ
	7:0	**Reserved** Format: MBZ

VFE_STATE_EX

Source: RenderCS

Size (in bits): 256

Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:8	**Reserved**
	7:0	**Reserved** Format: MBZ
1	31:0	**VFE Control** This field is used by VFE depending on the mode of operation. See the following tables for details. If VFE Mode = AVC-IT or AVC-MC, this field is valid as defined in Table 1 13. If VFE Mode = VC1-IT, this field is valid as defined in Table 1 14. Otherwise, this field is reserved.
2	31:0	**Interface Descriptor Remap Table** This field contains the interface descriptor remap table entries for the first 8 kernel indices. Each table entry has 4 bits, providing a remapping range of [0, 15]. The input of this table is the Interface Descriptor Offset within the MEDIA_OBJECT or MEDIA_OBJECT_EX command. As the table is limited to map the first 16 values, any Interface Descriptor Offset greater than 15 is not remapped. Bits 31:28: Remap for index = 7 Bits 27:24: Remap for index = 6 Bits 23:20: Remap for index = 5 Bits 19:16: Remap for index = 4 Bits 15:12: Remap for index = 3 Bits 11:8: Remap for index = 2 Bits 7:4: Remap for index = 1 Bits 3:0: Remap for index = 0
3	31:0	**Interface Descriptor Remap Table (cont)** This field contains the interface descriptor remap table entries for the next 8 kernel indices (index = 8...15). Each table entry has 4 bits, providing a remapping range of [0, 15]. Bits 31:28: Remap for index = 15 Bits 27:24: Remap for index = 14 Bits 23:20: Remap for index = 13 Bits 19:16: Remap for index = 12 Bits 15:12: Remap for index = 11

VFE_STATE_EX

		Bits 11:8: Remap for index = 10 Bits 7:4: Remap for index = 9 Bits 3:0: Remap for index = 8
4	31	**Scoreboard Enable** This field enables and disables the hardware scoreboard in the Media Pipeline. If this field is cleared, hardware ignores the following scoreboard state fields. **Value** / **Name** 0 – Scoreboard disabled 1 – Scoreboard enabled
	30	**Scoreboard Type** This field selects the type of scoreboard in use. **Value** / **Name** 0 – Stalling Scoreboard 1 – Reserved (for Non-stalling scoreboard)
	29:8	**Reserved** Format: MBZ
	7:0	**Scoreboard Mask** Format: Boolean Each bit indicates the corresponding dependency scoreboard is enabled. The scoreboard is based on the relative (X, Y) distance from the current threads' (X, Y) position. **Value** / **Name** / **Description** [0,7] / Bit n / Score n is enabled
5	31:28	**Scoreboard 3 Delta Y** Format: S3 Relative vertical distance of the dependent instance assigned to scoreboard 3, in the form of 2's compliment.
	27:24	**Scoreboard 3 Delta X** Format: S3 Relative horizontal distance of the dependent instance assigned to scoreboard 3, in the form of 2's compliment.
	23:16	**Scoreboard 2 Delta (X, Y)**
	15:8	**Scoreboard 1 Delta (X, Y)**
	7:0	**Scoreboard 0 Delta (X, Y)**
6	31:24	**Scoreboard 7 Delta (X, Y)**
	23:16	**Scoreboard 6 Delta (X, Y)**

VFE_STATE_EX		
	15:8	**Scoreboard 5 Delta (X, Y)**
	7:0	**Scoreboard 4 Delta (X, Y)**
7	31:0	**Reserved** Format: MBZ