Intel® UHD Graphics Open Source

Programmer's Reference Manual

For the 2020 Intel Core™ Processors with Intel Hybrid Technology based on the "Lakefield" Platform

Volume 2d: Command Reference: Structures

May 2021, Revision 1.0

Notices and Disclaimers

Intel technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

Customer is responsible for safety of the overall system, including compliance with applicable safety-related requirements or standards.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

Table of Contents

3DSTATE_BINDING_TABLE_POINTERS_BODY

3DSTATE_BINDING_TABLE_POINTERS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:21	**Reserved** Format: MBZ
	20:16	**Reserved** Format: MBZ
	15:5	**Pointer to Binding Table** Format: SurfaceStateOffset[15:5] When Binding Table Pool is disabled and HW Binding Table Alignment is not set to 256B alignment. Format: SurfaceStateOffset[16:6] When Binding Table Pool is enabled and HW Binding Table Alignment is not set to 256B alignment. Format: SurfaceStateOffset[18:8] When HW Binding Table Alignment is set to 256B alignment. Specifies an aligned address offset of the function's BINDING_TABLE_STATE. The offset's base and alignment differ depending on whether HW Binding Table is enabled and the setting of HW Binding Table Alignment field: If HW Binding Table Pool is **disabled** and the HW Binding Table Alignment is not set to 256B, the offset is relative to **Surface State Base Address** and the alignment is **32B**. If HW Binding Table Pool is **enabled** and the HW Binding Table Alignment is not set to 256B, the offset is relative to the **Binding Table Pool Base Address** and the alignment is **64B**. If HW Binding Table Pool is **disabled** and the HW Binding Table Alignment is set to 256B, the offset is relative to the **Surface State Base Address** and the alignment is **256B**. If HW Binding Table Pool is **enabled** and the HW Binding Table Alignment is set to 256B, the offset is relative to the **Binding Table Pool Base Address** and the alignment is **256B**.
	4:0	**Reserved** Format: MBZ

3DSTATE_BLEND_STATE_POINTERS_BODY

3DSTATE_BLEND_STATE_POINTERS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:6	**Blend State Pointer** Format: DynamicStateOffset[31:6]BLEND_STATE*8 Specifies the 64-byte aligned offset of the BLEND_STATE. This offset is relative to the **Dynamic State Base Address**.
	5:1	**Reserved** Format: MBZ
	0	**Blend State Pointer Valid** Format: Enable This bit, if set, indicates that the BLEND_STATE pointer has changed and new state needs to be fetched.

3DSTATE_CC_STATE_POINTERS_BODY

3DSTATE_CC_STATE_POINTERS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:6	**Color Calc State Pointer** Format: DynamicStateOffset[31:6]COLOR_CALC_STATE Specifies the 64-byte aligned offset of the COLOR_CALC_STATE. This offset is relative to the **Dynamic State Base Address**.
	5:1	**Reserved** Format: MBZ
	0	**Color Calc State Pointer Valid** Format: Enable If set, the hardware will fetch the CC state. This bit is context saved and restored so the CC state is considered undefined once this bit is cleared due to the possiblity of the CC state changing between context switches.

3DSTATE_CLEAR_PARAMS_BODY

RenderCS - 3DSTATE_CLEAR_PARAMS_BODY

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

DWord	Bit	Description
0	31:0	**Depth Clear Value** Format: IEEE_FLOAT32 This field defines the clear value that will be applied to the depth buffer if the Depth Buffer Clear field is enabled. It is valid only if Depth Buffer Clear Value Valid is set. **Programming Notes** The clear value must be between the min and max depth values (inclusive) defined in the CC_VIEWPORT. If the depth buffer format is D32_FLOAT, then values must be limited to the range of +0.0f and 1.0f inclusive; values outside this range are reserved.
1	31:1	**Reserved** Format: MBZ
	0	**Depth Clear Value Valid** Format: Boolean This field enables the **Depth Clear Value**. If clear, the depth clear value is obtained from interpolated depth of an arbitrary pixel of the primitive rendered with **Depth Buffer Clear** set in WM_STATE or 3DSTATE_WM. If set, the depth clear value is obtained from the **Depth Clear Value** field of this command.

3DSTATE_CLIP_BODY

3DSTATE_CLIP_BODY	
Source:	RenderCS
Size (in bits):	96
Default Value:	0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:21	**Reserved** Format: MBZ
	20	**Force User Clip Distance Cull Test Enable Bitmask** Format: Enable This field provides a work around override for the computation of SOL_INT::Render_Enable Value: 0h — Name: Normal — Description: Clip_INT::User Clip Distance Cull Test Enable Bitmask normally Value: 1h — Name: Force — Description: Forces Clip_INT::User Clip Distance Cull Test Enable Bitmask to use the value in 3DSTATE_CLIP:: User Clip Distance Cull Test Enable Bitmask
	19	**Vertex Sub Pixel Precision Select** Format: U1 Selects the number of fractional bits maintained in the vertex data Value: 0h — Name: 8 Bit — Description: 8 sub pixel precision bits maintained Value: 1h — Name: 4 Bit — Description: 4 sub pixel precision bits maintained
	18	**Early Cull Enable** Format: Enable This field is used to enable/disable the EarlyCull function. When this bit is set triangles are checked if they are backface culled before proceeding through must clip function. **Programming Notes** Setting this bit must not impact functionality, this state only controls the performance of the must clip function. Vertex Sub Pixel Precision Select precision must be set to "8 bit" in order avoid precision issues. In POSH pipe, this state will be used to control the forcing of Z-Only clipped triangles to Trivial accept. This is similar functionality as early cull in replay pipe. This bit must be set for better performance.
	17	**Force User Clip Distance Clip Test Enable Bitmask** Format: Enable This field provides a work around override for the computation of SOL_INT::Render_Enable. Value: 0b — Name: Normal — Description: Clip_INT:: User Clip Distance Clip Test Enable Bitmask normally Value: 1b — Name: Force — Description: Forces Clip_INT:: User Clip Distance Clip Test Enable Bitmask to use the value in 3DSTATE_CLIP::User Clip Distance Clip Test Enable Bitmask

3DSTATE_CLIP_BODY

DWord	Bit	Description
	16	**Force Clip Mode** Format: Enable This field provides a work around override for the computation of SOL_INT::Render_Enable. **Value** / **Name** / **Description**: 0b / Normal / Clip_INT::Clip Mode is computed normally. 1b / Force / Forces Clip_INT::Clip Mode to use the value in 3DSTATE_CLIP::User Clip Mode.
	15:12	**Reserved** Format: MBZ
	11:10	**Clipper Statistics Enable** This bit controls whether Clip-unit-specific statistics register(s) can be incremented. **Value** / **Name** / **Description**: 00h / Disable / CL_INVOCATIONS_COUNT cannot increment 01h / Increment by one / CL_INVOCATIONS_COUNT can increment 03h / Reserved /
	9:8	**Reserved** Format: MBZ
	7:0	**User Clip Distance Cull Test Enable Bitmask** Format: Enable[8] This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept determination needs to be made (does not cause a must clip).DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.
1	31	**Clip Enable** Format: Enable Specifies whether the Clip function is enabled or disabled (pass-through).
	30	**API Mode** Controls the definition of the NEAR clipping plane **Value** / **Name** / **Description**: 0h / OGL / NEAR VP boundary == 0.0 (NDC)
	29	**Reserved** Format: MBZ
	28	**Viewport XY Clip Test Enable** Format: Enable This field is used to control whether the Viewport X, Y extents [-1,1] are considered in VertexClipTest. If both the Guardband and Viewport XY ClipTest are DISABLED, all vertices are considered "visible" with respect to the XY directions.
	27	**Reserved** Format: MBZ

<table>
<tr><th colspan="3">3DSTATE_CLIP_BODY</th></tr>
<tr><td></td><td>26</td><td>Guardband Clip Test Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>This field is used to control whether the Guardband X, Y extents are considered in VertexClipTest for non-point objects. If the Guardband ClipTest is DISABLED but the Viewport XY ClipTest is ENABLED, ClipDetermination operates as if the Guardband were coincident with the Viewport. If both the Guardband and Viewport XY ClipTest are DISABLED, all vertices are considered "visible" with respect to the XY directions.</td></tr>
<tr><td></td><td>25:24</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>23:16</td><td>User Clip Distance Clip Test Enable Bitmask
<table><tr><td>Format:</td><td>Enable[8]</td></tr></table>This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept / must clip determination needs to be made.DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.</td></tr>
<tr><td></td><td>15:13</td><td>Clip Mode
This field specifies a general mode of the CLIP unit, when the CLIP unit is ENABLED.<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>NORMAL</td><td>TrivialAccept objects are passed down the pipeline, MustClip objects Clipped in the Fixed Function Clipper HW, TrivialReject and BAD objects are discarded</td></tr><tr><td>1h</td><td>Reserved</td><td></td></tr><tr><td>2h</td><td>Reserved</td><td></td></tr><tr><td>3h</td><td>REJECT_ALL</td><td>All objects are discarded</td></tr><tr><td>4h</td><td>ACCEPT_ALL</td><td>All objects (except BAD objects) are trivially accepted. This effectively disables the clip-test/clip-determination function. Note that the CLIP unit will still filter out adjacency information, which may be required since the SF unit does not accept primitives with adjacency.</td></tr><tr><td>5h-7h</td><td>Reserved</td><td></td></tr></table></td></tr>
<tr><td></td><td>12:10</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>9</td><td>Perspective Divide Disable
<table><tr><td>Format:</td><td>Disable</td></tr></table>This field disables the Perspective Divide function performed on homogeneous position read from the URB. This feature can be used by software to submit pre-transformed "screen-space" geometry for rasterization. This likely requires the W component of positions to contain "rhw" (aka 1/w) in order to support perspective-correct interpolation of vertex attributes. Likewise, the X, Y, Z components will likely be required to be X/W, Y/W, Z/W. Note that the device does not support clipping when perspective divide is disabled. Software must specify CLIPMODE_ACCEPT_ALL whenever it disables perspective divide. This implies that software must ensure that object positions are completely contained within the "guardband" screen-space limits imposed by the SF unit (e.g., by clipping in CPU SW before submitting the objects).</td></tr>
</table>

3DSTATE_CLIP_BODY

	8	**Non-Perspective Barycentric Enable**
		Format: Enable
		This field enables computation of non-perspective barycentric parameters in the clipper, which are sent to SF unit in the must clip case. This field must be enabled if any non-perspective interpolation modes are used in pixel shader.
		Programming Notes
		This field must be set whenever Enable bits 3 or 4 or 5 of 3DSTATE_WM:Barycentric Interpolation Mode is set. This indicates that one of the Non-perspective barycentric interpolation modes are used.
		This field must be set if the 3DSTATE_PS_EXTRA:Pixel Shader Requires Non-Perspective Bary Plane Coefficients is set.
	7:6	**Reserved**
		Format: MBZ
	5:4	**Triangle Strip/List Provoking Vertex Select**
		Format: U2
		enumerated type
		This field selects which vertex of a triangle (in a triangle strip or list primitive) is considered the "provoking vertex".

Value	Name
0h	0
1h	1
2h	2
3h	Reserved

	3:2	**Line Strip/List Provoking Vertex Select**
		Format: U2
		enumerated type
		This field selects which vertex of a line (in a line strip or list primitive) is considered the "provoking vertex".

Value	Name
0h	0
1h	1
2h	Reserved
3h	Reserved

	1:0	**Triangle Fan Provoking Vertex Select**

3DSTATE_CLIP_BODY

		Format: U2 enumerated type This field selects which vertex of a triangle (in a triangle fan primitive) is considered the "provoking vertex". **Value** / **Name** 0h / 0 1h / 1 2h / 2 3h / Reserved
2	31:28	**Reserved** Format: MBZ
	27:17	**Minimum Point Width** Format: U8.3 pixels This value is used to clamp read-back PointWidth values.
	16:6	**Maximum Point Width** Format: U8.3 pixels This value is used to clamp read-back PointWidth values.
	5	**Force Zero RTA Index Enable** Format: Enable If set, the Clip unit will ignore the read-back RTAIndex and operate as if the value 0 was read-back. If clear, the read-back value is used.
	4	**Reserved** Format: MBZ
	3:0	**Maximum VP Index** Format: U4-1 index value (# of viewports) This field specifies the maximum valid VPIndex value, corresponding to the number of active viewports. If the source of the VPIndex exceeds this maximum value, a VPIndex value of 0 is passed down the pipeline. Note that this clamping does not affect a VPIndex value stored in the URB.

3DSTATE_CONSTANT_ALL_DATA

3DSTATE_CONSTANT_ALL_DATA		
Source:	RenderCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0..1	63:5	**Pointer To Constant Buffer** Format: GraphicsAddress63-5 The value of this field is the virtual address of the location of the push constant buffer. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Constant Buffer Read Length** Format: U5 This field specifies the length of the constant data to be loaded from memory in 256-bit units. **Programming Notes** • The sum of the read length fields for all pointers must be less than or equal to the size of 64 • Zero means there no data to fetch for this buffer pointer.

3DSTATE_CONSTANT(Body)

3DSTATE_CONSTANT(Body)		
Source:	RenderCS	
Size (in bits):	320	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 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..3	63:5	**Pointer To Constant Buffer 0** Format: GraphicsAddress63-5 **Description** The value of this field is the virtual address of the location of the push constant buffer 0. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47]. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ
4..5	63:5	**Pointer To Constant Buffer 1** Format: GraphicsAddress63-5 This field points to the location of Constant Buffer 1. If gather constants are enabled This field is an offset of constant Buffer1 from the Gather Pool BASE ADDRESS. If gather constants is disabled, the value of this field is the virtual address of the location of the push constant buffer. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47]. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ
6..7	63:5	**Pointer To Constant Buffer 2** Format: GraphicsAddress63-5 The value of this field is the virtual address of the location of the push constant buffer 2. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47]. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ
8..9	63:5	**Pointer To Constant Buffer 3** Format: GraphicsAddress63-5 The value of this field is the virtual address of the location of the push constant buffer 3.

3DSTATE_CONSTANT(Body)		
		GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47]. **Programming Notes** Constant buffers must be allocated in linear (not tiled) graphics memory.
	4:0	**Reserved** Format: MBZ

3DSTATE_CPS_BODY

<table>
<tr><th colspan="3">3DSTATE_CPS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="5">0</td><td>31:27</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>26:16</td><td>MinCPSizeY
Format: S3.7
This bit-field defines the minimum shading ratio in Y dimension in screen space. This value is used only when Coarse Pixel Shading is enabled. It also defines the floor of the non-quantized CPSizeY for Mode 1. HW quantizes this value to determine Decoupled Rate. This value is used to clamp the CPSizeY for the lowest bound.</td></tr>
<tr><td>15</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>14</td><td>ScaleAxis
Format: U1
This bit defines which dimension (along X- or Y- axis) should be scaled when computing Coarse Pixel Size values along ellipse in Mode1.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>X axis</td><td>Use aspect to scale X-dimension</td></tr>
<tr><td>1h</td><td>Y axis</td><td>Use aspect to scale Y-dimension</td></tr>
</table></td></tr>
<tr><td>13:12</td><td>Coarse Pixel Shading Mode
Format: U2
This bit-field defines Coarse Pixel Shading Mode.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>CPS_MODE_NONE</td><td>Coarse Pixel Shading is disabled. HW may be required to drive default values to shader inputs e.g. ScaleX = ScaleY = 1 and LODCompX = LODCompY = 1.</td></tr>
<tr><td>1h</td><td>CPS_MODE_CONSTANT</td><td>Coarse Pixel Shading Ratios are defined per DRAW based on MinCPSizeX and MinCPSizeY fields in this state (constant across render target).</td></tr>
<tr><td>2h</td><td>CPS_MODE_RADIAL</td><td>Coarse Pixel Shading Ratio varies radially from a focal point defined by (X_Focal, Y_Focal). This mode is typically used when there is Depth of Field or Ring of Confusion camera effects are desired.</td></tr>
</table></td></tr>
</table>

3DSTATE_CPS_BODY

		3h | Reserved
		Programming Notes It is a valid configuration to set the CPS mode other than CPS_MODE_NONE and request per-pixel dispatch in 3DSTATE_PS_EXTRA. In such case, 3DSTATE_PS_EXTRA configuration overrides 3DSTATE_CPS configuration, and effective CPS mode is set to CPS_MODE_NONE for this draw primitive. It is an INVALID configuration to set the CPS mode other than CPS_MODE_NONE and request per-sample dispatch in 3DSTATE_PS_EXTRA. Such configuration should be disallowed at the API level, and rendering results are undefined. It is a valid configuration to set the CPS mode to CPS_MODE_NONE and at the same time set Pixel Shader Is Per Coarse Pixel in 3DSTATE_PS_EXTRA. In such case, 3DSTATE_PS_EXTRA bit is ignored and shader is dispatched at pixel-rate; shader inputs specific to coarse-rate have undefined value (ActualCoarsePixelSize for example).
	11	**Statistics Enable** Format: | Enable This bit-field defines statistics gathering. When enabled, CPS Invocation Counter is enabled.
	10:0	**MinCPSizeX** Format: | S3.7 This bit-field defines the minimum shading ratio in X dimension in screen space. This value is used only when Coarse Pixel Shading is enabled. It also defines the floor of the non-quantized ScaleX for Mode 1. HW quantizes this value to determine Decoupled Rate.
1	31:27	**Reserved** Format: | MBZ
	26:16	**MaxCPSizeY** Format: | S3.7 This bit-field defines the maximum shading ratio in Y dimension in screen space. This value is used only when Coarse Pixel Shading is enabled and Coarse Pixel Shading Mode is set to CPS_MODE_RADIAL. This value is used to clamp the CPSizeY for the highest bound. MaxCPSizeY must be greater than or equal to MinCPSizeY when this value is used.
	15:11	**Reserved** Format: | MBZ
	10:0	**MaxCPSizeX** Format: | S3.7 This bit-field defines the maximum shading ratio in X dimension in screen space. This value is used only when Coarse Pixel Shading is enabled and Coarse Pixel Shading Mode is set to CPS_MODE_RADIAL. This value is used to clamp the CPSizeX for the highest bound. MaxCPSizeX must be greater than or equal to MinCPSizeX when this value is used.

3DSTATE_CPS_BODY

2	31:16	**Reserved** Format: MBZ
	15:0	**Y_Focal** Format: S15 The valid data range is (-2^14 to 2^14-1) This field defines the Y-coordinate for a focal point with respect to which shading ratio is computed in Mode1.
3	31:16	**Reserved** Format: MBZ
	15:0	**X_Focal** Format: S15 The valid data range is (-2^14 to 2^14-1) This field defines the X-coordinate for a focal point with respect to which shading ratio is computed in Mode1.
4	31:0	**My** Format: IEEE_FLOAT32 This field defines the slope of the Transfer function for computing CPSizeY for Mode1. **Programming Notes** SW needs to compute this from API supplied parameters: $(M_x, M_y) = \left(\frac{S_x^{max}-S_x^{min}}{R_{max}-R_{min}}, \frac{S_y^{max}-S_y^{min}}{R_{max}-R_{min}}\right)$ My must be greater than or equal to zero.
5	31:0	**Mx** Format: IEEE_FLOAT32 This field defines the slope of the Transfer function for computing CPSizeX for Mode1. **Programming Notes** SW needs to compute this from API supplied parameters: $(M_x, M_y) = \left(\frac{S_x^{max}-S_x^{min}}{R_{max}-R_{min}}, \frac{S_y^{max}-S_y^{min}}{R_{max}-R_{min}}\right)$ Mx must be greater than or equal to zero
6	31:0	**Rmin** Format: IEEE_FLOAT32 This field defines (smaller) radius of the inner ellipse for Mode1. All points on inner ellipse have coarse point size = (MinCPSizeX, MinCPSizeY).
7	31:0	**Aspect**

3DSTATE_CPS_BODY		
		Format: IEEE_FLOAT32 This field defines aspect for both inner and outer ellipses in Mode1. The aspect parameter must be within <0,1> range and Driver must program it as ratio of smallest ellipse radius to larger ellipse radius: Aspect = min(radiusX, radiusY) / max(radiusX, radiusY) where radiusX and radiusY define ellipse radius along x- and y- axes respectively. Note: Aspect must be same for both inner and outer ellipses.

3DSTATE_DS_BODY

3DSTATE_DS_BODY

Source: RenderCS

Size (in bits): 320

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

DWord	Bit	Description
0..1	63:6	**Kernel Start Pointer** Format: InstructionBaseOffset[63:6] This field specifies the starting location of the kernel program run by threads spawned by this FF unit. It is specified as a 64-byte-granular offset from the Instruction Base Address. This field is ignored if DS Function Enable is DISABLED.
	5:0	**Reserved** Format: MBZ
2	31	**Reserved** Format: MBZ
	30	**Vector Mask Enable** Format: Enable Enumerated Type Upon subsequent DS thread dispatches, this bit is loaded into the EU's Vector Mask Enable (VME, cr0.0[3]) thread state. Refer to EU documentation for the definition and use of VME state. Value / Name / Description: 0h / Dmask / The EU will use the Dispatch Mask (supplied by the DS stage) for instruction execution. 1h / Vmask / The EU will use the Vector Mask (derived from the Dispatch Mask) for instruction execution. **Programming Notes** Under normal conditions SW shall specify DMask, as the DS stage will provide a Dispatch Mask appropriate to SIMD4x2 or SIMD8 thread execution (as a function of dispatch mode). E.g., for SIMD4x2 thread execution, the DS stage will generate a Dispatch Mask that is equal to what the EU would use as the Vector Mask. For SIMD8 execution there is no known usage model for use of Vector Mask (as there is for PS shaders).
	29:27	**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 if DS Function Enable is DISABLED. Value / Name / Description

3DSTATE_DS_BODY

		0h	No Samplers	No samplers used
		1h	1-4 Samplers	between 1 and 4 samplers used
		2h	5-8 Samplers	between 5 and 8 samplers used
		3h	9-12 Samplers	between 9 and 12 samplers used
		4h	13-16 Samplers	between 13 and 16 samplers used

26 **Reserved**

Format:	MBZ

25:18 **Binding Table Entry Count**

Format:	U8

When HW Generated Binding Table is disabled: Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state. **Note:** 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 if DS Function Enable is DISABLED.
When HW Generated Binding Table bit is enabled: This field indicates which cache lines (512bit units - 32 Binding Table Entry section) should be fetched. Each bit in this field corresponds to a cache line. Only the 1st 4 non-zero Binding Table entries of each 32 Binding Table entry section prefetched will have its surface state prefetched.

Value	Name
[0,255]	

Programming Notes
When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.

17 **Thread Dispatch Priority**

Format:	U1 Enumerated Type

Specifies the priority of the thread for dispatch: This field is ignored if DS Function Enable is DISABLED.

Value	Name	Description
0h	Normal	Normal Priority
1h	High	High Priority

16 **Floating Point Mode**

Format:	U1 Enumerated Type

Specifies the initial floating point mode used by the dispatched thread. This field is ignored if DS Function Enable is DISABLED.

Value	Name	Description
0h	IEEE-754	Use IEEE-754 Rules

3DSTATE_DS_BODY		
		1h | Alternate | Use alternate rules
	15	**Reserved** Format: | MBZ
	14	**Accesses UAV** Format: | Enable This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment. **Programming Notes** This field must not be set when DS Function Enable is disabled.
	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. This field is ignored if DS Function Enable is DISABLED.
	12:8	**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. This field is ignored if DS Function Enable is DISABLED.
	6:0	**Reserved** Format: | MBZ
3..4	63:32	**Reserved** Format: | MBZ
	31:10	**Scratch Space Base Pointer** Format: | GeneralStateOffset[31:10] Specifies the starting location of the scratch space area allocated to this FF unit as a 1K-byte aligned offset from the General State Base Address. If required, each thread spawned by this FF unit will be allocated some portion of this space, as specified by Per-Thread Scratch Space. The computed offset of the thread-specific portion will be passed in the thread payload as Scratch Space Offset. The thread is expected to utilize "stateless" DataPort read/write requests to access scratch space, where the DataPort will cause the General State Base Address to be added to the offset passed in the request header. This field is ignored if DS Function Enable is DISABLED.
	9:4	**Reserved** Format: | MBZ
	3:0	**Per-Thread Scratch Space**

3DSTATE_DS_BODY

		Format: U4 power of 2 Bytes over 1K Bytes Specifies the amount of scratch space to be allocated to each thread spawned by this FF unit. The driver must allocate enough contiguous scratch space, starting at the Scratch Space Base Pointer, to ensure that the Maximum Number of Threads can each get Per-Thread Scratch Space size without exceeding the driver-allocated scratch space. This field is ignored if DS Function Enable is DISABLED. **Value** | **Name** [0,11] | indicating [1K Bytes, 2M Bytes] **Programming Notes** This amount is available to the kernel for information only. It will be passed verbatim (if not altered by the kernel) to the Data Port in any scratch space access messages, but the Data Port will ignore it.
5	31:25	**Reserved** Format: MBZ
	24:20	**Dispatch GRF Start Register For URB Data** Format: U5 **Description** Specifies the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload. This field is ignored if DS Function Enable is DISABLED. When SIMD8_SINGLE_OR_DUAL_PATCH dispatch mode is selected, HW shall increment the GRF start register by 1 when a dual patch simd8 thread is dispatched. **Value** | **Name** | **Description** [0,31] | | indicating GRF [R0, R31]
	19:18	**Reserved** Format: MBZ
	17:11	**Patch URB Entry Read Length** Format: U7 Specifies how much data (in 256-bit units) is to be read from the Patch URB entry and passed in the DS thread payload. This field is ignored if DS Function Enable is DISABLED. **Value** | **Name** [0,64] |
	10	**Reserved** Format: MBZ

<table>
<tr><th colspan="3">3DSTATE_DS_BODY</th></tr>
<tr><td></td><td>9:4</td><td>Patch URB Entry Read Offset
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U6</td></tr></table>
Specifies the offset (in 256-bit units) at which Patch URB data is to be read from the URB before being included in the thread payload. This field is ignored if DS Function Enable is DISABLED.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,63]</td><td></td></tr></table></td></tr>
<tr><td></td><td>3:0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>6</td><td>31</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>30:21</td><td>Maximum Number of Threads
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U10-1 Thread Count</td></tr></table>
Specifies the maximum number of simultaneous DS threads allowed to be active. Used to avoid using up the scratch space. Programming the value of the max threads over the number of threads based off number of threads supported in the execution units may improve performance since the architecture allows threads to be buffered between the check for max threads and the actual dispatch into the EU. Programming the max values to a number less than the number of threads supported in the execution units may reduce performance. This field is ignored if DS Function Enable is DISABLED.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>[0,363]</td><td></td><td>indicating thread count of [1,364]</td></tr></table></td></tr>
<tr><td></td><td>20:11</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>10</td><td>Statistics Enable
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table>
If ENABLED, this FF unit will engage in statistics gathering. Refer to the Statistics Gathering section.
If DISABLED, statistics information associated with this FF stage will be left unchanged.
This field is ignored if DS Function Enable is DISABLED.</td></tr>
<tr><td></td><td>9</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

3DSTATE_DS_BODY

	8:5	**Reserved** Format: MBZ
	4:3	**Dispatch Mode** Format: U2 This field specifies how the DS stage generates DS thread requests, and correspondingly impacts the DS thread payload. The setting of this field must agree with how the DS kernel was compiled. This field is ignored if DS Function Enable is DISABLED.

Value	Name	Description	Programming Notes
1h	SIMD8_SINGLE_PATCH	DS threads are passed one patch, up to 8 domain point inputs, and up to 8 output vertex handles. All the domain points are associated with the single input patch. The DS kernel (at KSP) is expected to run in SIMD8 execution mode. The DUAL_PATCH KSP is ignored.	
2h	SIMD8_SINGLE_OR_DUAL_PATCH	SIMD8_SINGLE_OR_DUAL_PATCH This mode enables use of both the KSP and the DUAL_PATCH KSP. The KSP kernel operates just like in SIMD8_SINGLE_PATCH mode. See DUAL_PATCH Thread Execution for a discussion of how the DUAL_PATCH KSP is used.	At least 2 HS URB handles must be allocated in order to enable this mode.
3h	Reserved		

Programming Notes
SIMD4X2 mode is no longer allowed.

	2	**Compute W Coordinate Enable** Format: Enable If ENABLED, the DS unit will (for each domain point) compute W = 1 - (U + V) and pass the result as a floating point value in the DS thread payload. If DISABLED, 0.0 will be passed. This field must only be ENABLED for the tessellation of TRI domains, where UVW coordinates are required. This field must be DISABLED for other domains (as they only require UV coordinates) otherwise the computed W coordinate is UNDEFINED. This field is ignored if DS Function Enable is DISABLED.
	1	**Cache Disable**

3DSTATE_DS_BODY

		Format: Disable This bit controls the operation of the DS Cache. This field is ignored if DS Function Enable is DISABLED. If the DS Cache is DISABLED and the DS Function is ENABLED, the DS Cache is not used and all incoming domain points will be passed to DS threads. If the DS Cache is ENABLED and the DS Function is ENABLED, incoming domain points that do not hit in the DS Cache will be passed to DS threads. The DS Cache is invalidated whenever the DS Cache becomes DISABLED , whenever the DS Function Enable toggles, and between patches.
	0	**Function Enable** Format: Enable If ENABLED, DS threads will be spawned to process incoming domain points which miss in the DS cache. If DISABLED, the DS stage goes into pass-through mode and performs no specific processing. This field is always used. **Programming Notes** The tessellation stages (HS, TE and DS) must be enabled/disabled as a group. I.e., draw commands can only be issued if all three stages are enabled or all three stages are disabled, otherwise the behavior is UNDEFINED.
7	31:27	**Reserved** Format: MBZ
	26:21	**Vertex URB Entry Output Read Offset** Format: U6 Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB by SBE. Value: [0,63] — Name:
	20:16	**Vertex URB Entry Output Length** Format: U5 Specifies the amount of URB data written for each Vertex URB entry, in 256-bit register increments. Value: [1,16] — Name: **Programming Notes** This length does not include the vertex header.
	15:8	**User Clip Distance Clip Test Enable Bitmask**

		Format: U8 This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept / must clip determination needs to be made. DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.
	7:0	**User Clip Distance Cull Test Enable Bitmask** Format: U8 This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept determination needs to be made (does not cause a must clip). DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.
8..9	63:6	**DUAL_PATCH Kernel Start Pointer** Format: InstructionBaseOffset[63:6] This field specifies the starting location of the DUAL_PATCH kernel program run by threads spawned by this FF unit. It is specified as a 64-byte-granular offset from the Instruction Base Address. This field is ignored if DS Function Enable is DISABLED. See DUAL_PATCH Thread Execution for a discussion of how the DUAL_PATCH KSP is used.
	5:0	**Reserved** Format: MBZ

intel

3DSTATE_GS_BODY

<table>
<tr><th colspan="3">3DSTATE_GS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 288
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0..1</td><td>63:6</td><td>Kernel Start Pointer
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>InstructionBaseOffset[63:6]</td></tr></table>
This field specifies the starting location (1st EU core instruction) of the kernel program run by threads spawned by this FF unit. It is specified as a 64-byte-granular offset from the Instruction Base Address.</td></tr>
<tr><td>5:0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td rowspan="2">2</td><td>31</td><td>Single Program Flow
<table><tr><td>Format:</td><td>Enable</td></tr></table>
Specifies the initial condition of the kernel program as either a single program flow (SIMDnxm with m = 1) or as multiple program flows (SIMDnxm with m > 1). See CR0 description in ISA Execution Environment.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Disable</td><td>Single Program Flow disabled</td></tr>
<tr><td>1h</td><td>Enable</td><td>Single Program Flow enabled</td></tr></table></td></tr>
<tr><td>30</td><td>Vector Mask Enable
<table><tr><td>Format:</td><td>Enable Enumerated Type</td></tr></table>
Upon subsequent GS thread dispatches, this bit is loaded into the EU's Vector Mask Enable (VME, cr0.0[3]) thread state. Refer to EU documentation for the definition and use of VME state.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Dmask</td><td>The EU will use the Dispatch Mask (supplied by the GS stage) for instruction execution.</td></tr>
<tr><td>1h</td><td>Vmask</td><td>The EU will use the Vector Mask (derived from Dispatch Mask) for instruction execution.</td></tr></table>
<table><tr><th>Programming Notes</th></tr>
<tr><td>Under normal conditions SW shall specify DMask, as the GS stage will provide a Dispatch Mask appropriate to SIMD4x2 or SIMD8 thread execution (as a function of dispatch mode). E.g., for SIMD4x2 execution, the GS stage will generate a Dispatch Mask that is equal to what the EU</td></tr></table></td></tr>
</table>

3DSTATE_GS_BODY

	would use as the Vector Mask. For SIMD8 execution there is no known usage model for use of Vector Mask (as there is for PS shaders).
29:27	**Sampler Count**

Format:	U3

Specifies how many samplers (in multiples of 4) the geometry shader kernel uses. Used only for prefetching the associated sampler state entries.

Value	Name	Description
0h	No Samplers	No Samplers used
1h	1-4 Samplers	Between 1 and 4 samplers used
2h	5-8 Samplers	Between 5 and 8 samplers used
3h	9-12 Samplers	Between 9 and 12 samplers used
4h	13-16 Samplers	Between 13 and 16 samplers used
5h-7h	Reserved	

26	**Reserved**
25:18	**Binding Table Entry Count**

Format:	U8

When **HW Generated Binding Table** is disabled: Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state. Note: 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. When **HW Generated Binding Table** bit is enabled: This field indicates which cache lines (512bit units - 32 Binding Table Entry section) should be fetched. Each bit in this field corresponds to a cache line. Only the 1st 4 non-zero Binding Table entries of each 32 Binding Table entry section prefetched will have its surface state prefetched.

Programming Notes
When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.

17	**Thread Dispatch Priority**

Specifies the priority of the thread for dispatch.

Value	Name	Description
0h	Normal	Normal thread dispatch priority
1h	High	High thread dispatch priority

16	**Floating Point Mode**

Specifies the initial floating point mode used by the dispatched thread.

Value	Name	Description
0h	IEEE-754	Use IEEE-754 Rules
1h	Alternate	Use alternate rules

15:14	**Reserved**

3DSTATE_GS_BODY

		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	**Accesses UAV** Format: Enable This field must be set when GS has a UAV access. **Programming Notes** This field must not be set when GS Function Enable is disabled.
	11	**Mask Stack 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	**Reserved** Format: MBZ
	5:0	**Expected Vertex Count** Format: U6 Specifies the number of vertices per input object expected by the GS thread. Input topologies not matching this expect value are discarded. Note that **DiscardAdjacency** is also considered (e.g., if the value programmed is 3 and DiscardAdjacency is set, TRILIST_ADJ and TRISTRIP_ADJ topologies are <u>not</u> discarded as they will pass 3 vertices/object to the GS threads). **Value** / **Name** [1,32] /
3..4	63:32	**Reserved** Format: MBZ
	31:10	**Scratch Space Base Pointer** Format: GeneralStateOffset[31:10] Specifies the starting location of the scratch space area allocated to this FF unit as a 1K-byte aligned offset from the General State Base Address. If required, each thread spawned by this FF unit will be allocated some portion of this space, as specified by Per-Thread Scratch Space. The

3DSTATE_GS_BODY		
		computed offset of the thread-specific portion will be passed in the thread payload as Scratch Space Offset. The thread is expected to utilize "stateless" DataPort read/write requests to access scratch space, where the DataPort will cause the General State Base Address to be added to the offset passed in the request header. This field is ignored if VS Function Enable is DISABLED.
	9:4	**Reserved** Format: MBZ
	3:0	**Per-Thread Scratch Space** Format: U4 power of 2 Bytes over 1K Bytes Specifies the amount of scratch space to be allocated to each thread spawned by this FF unit. The driver must allocate enough contiguous scratch space, starting at the Scratch Space Base Pointer, to ensure that the Maximum Number of Threads can each get Per-Thread Scratch Space size without exceeding the driver-allocated scratch space. Value: [0,11] — Name: (none) — Description: indicating [1K Bytes, 2M Bytes]
5	31	**Reserved** Format: MBZ
	30:29	**Dispatch GRF Start Register For URB Data [5:4]** Format: U2 Specifies bit [5:4] of the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload. The **Dispatch GRF Start Register For URB Data [3:0]** field is used to specify bits [3:0] of the starting GRF register number.
	28:23	**Output Vertex Size** Format: U6 [0,63] indicating [1,64] 16B units Specifies the size of each vertex stored in the GS output entry (following any Control Header data) as a number of 128-bit units (minus one). **Programming Notes** Programming Restrictions: The vertex size must be programmed as a multiple of 32B units with the following exception: Rendering is disabled (as per SOL stage state) and the vertex size output by the GS thread is 16B. If rendering is enabled (as per SOL state) the vertex size must be programmed as a multiple of 32B units. In other words, the only time software can program a vertex size with an odd number of 16B units is when rendering is disabled.

3DSTATE_GS_BODY

	22:17	**Output Topology** Format: **3D_Prim_Topo_Type** This field specifies the topology type (3DPrimType) to be associated with GS-thread output vertices (if any).
	16:11	**Vertex URB Entry Read Length** Specifies the amount of URB data read and passed in the thread payload for each Vertex URB entry, in 256-bit register increments. **Programming Notes** Programming Restriction:This field must be a non-zero value if Include Vertex Handles is cleared to zero.
	10	**Include Vertex Handles** Format: Boolean If set, all the input Vertex URB handles are included in the payload. These are referred to as "pull model" URB handles, as the thread will use them to read from the URB. **Programming Notes** Programming Restriction: This field must be set if Vertex URB Entry Read Length is cleared to zero.
	9:4	**Vertex URB Entry Read Offset** Format: U6 Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB before being included in the thread payload. This offset applies to all Vertex URB entries passed to the thread.
	3:0	**Dispatch GRF Start Register For URB Data** Format: U4 **Description** Specifies the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload. The Dispatch GRF Start Register for URB Data [5:4] field is used to extend the range of the starting GRF register number to [0,63]. **Value** | **Name** | **Description** [0,15] | | indicating bits [3:0] of the GRF number **Programming Notes** If Include Vertex Handles is enabled (pull or hybrid handles case), then For simd4x2: For DUAL_OBJECT dispatch mode this field should be: (((2*numVerticesPerObject) + 8 - 1)/8) + 1 For SINGLE and DUAL_INSTANCE dispatch modes this field should be:

3DSTATE_GS_BODY

DWord	Bit	Description
		((numVerticesPerObject +8 - 1)/8) + 1 If Include Primitive ID is set, then add 1 to the value obtained by using the above
		If Include Vertex Handles is enabled (pull or hybrid handles case), then For simd8: For InstanceCount == 1: numVerticesPerObject 2 For InstanceCount > 1: ((numVerticesPerObject 8 - 1)/8) 2 If Include Primitive ID is set, then add 1 to the value obtained by using the above
6	31:26	**Reserved** Format: MBZ
	25:24	**Reserved** Format: MBZ
	23:20	**Control Data Header Size** Format: U4 Specifies the number of 32B units of control data header located at the start of the GS URB entry. The value 0 indicates there is no control data header, and Control Data Format is ignored and neither Cut nor StreamID bits are defined. Software must ensure that the Control Data Header Size is sufficient to accommodate the maximum number of vertices possibly output by the GS thread. It is UNDEFINED for a GS thread to report more output vertices than can be accomodated in a non-zero-sized header. Value: [0,8] — Name: 32B Units
	19:15	**Instance Control** Format: U5-1 #Instances Specifies the number of instances (minus one) for each input object. To avoid confusion, this document uses the term **"InstanceCount"** to refer to InstanceControl+1, with a range of [1,32] If **InstanceCount>1**, DUAL_OBJECT mode is invalid. Software will likely want to use DUAL_INSTANCE mode for higher performance, but SINGLE mode is also supported. When **InstanceCount=1** (one instance per object), software can decide which dispatch mode to use. DUAL_OBJECT mode would likely be the best choice for performance, followed by SINGLE mode. DUAL_INSTANCE mode is not recommended but is supported. Value: [0,31] — Name: — Description: Indicating [1,31] instances
	14:13	**Default Stream Id** Format: U2 When the GS is enabled, unless the GS output entry contains StreamID bits in the control header, this field specifies the default StreamID associated with any GS-thread output vertices. When the GS is disabled, StreamID will be output as 0.
	12:11	**Dispatch Mode** Format: U2 This field specifies how the GS unit dispatches multiple instances and/or multiple objects. Value | Name | Description | Programming Notes

3DSTATE_GS_BODY

3h	SIMD8	Each thread shades up to 8 different objects or (if InstanceCount >1) 8 instances of a single object.	[] The driver must send pipe control with a cs stall after a 3dstate_gs state change and the Dispatch Mode is simd8 and the number of handles allocated to gs is less than 16.

Programming Notes
The GS must be allocated at least two URB handles or behavior is UNDEFINED for Dual Instance or Dual Object mode.
The only valid Dispatch Mode is SIMD8.

10 **Statistics Enable**

Format:	Enable

This bit controls whether GS-unit-specific statistics register(s) can be incremented.

Value	Name	Description
0h	Disable	GS_INVOCATIONS_COUNT and GS_PRIMITIVES_COUNT cannot increment
1h	Enable	GS_INVOCATIONS_COUNT and GS_PRIMITIVES_COUNT can increment

9:5 **Invocations Increment Value**

Format:	U5

Specifies how much to increment the GS_INVOCATIONS_COUNT for each instance of each object. This control is provided to allow software to process multiple instances (from an API POV) in a single kernel invocation. In SINGLE dispatch mode, the counter will increment by this value for each dispatch (as it's only one instance of one object). In DUAL_INSTANCE mode, the counter will be incremented by the value if only one instance is included in the dispatch (i.e., the last odd instance), otherwise the counter will be incremented by twice this value. In DUAL_OBJECT dispatch mode, the counter will be incremented by the value if only one object is included in the dispatch (i.e., a forced dispatch of one object), otherwise the counter will be incremented by twice this value.

Value	Name	Description
[0,31]		indicating an increment of [1,32]

4 **Include Primitive ID**

Format:	Boolean

If set, R1 of the payload is written with Primitive ID value(s). If clear, these Primitive ID values are not included in the payload R1.

3 **Hint**

Format:	U1

This state bit is simply passed in GS thread payloads for use by the GS kernel - it has no other impact on hardware operation.

2 **Reorder Mode**

This bit controls how vertices of triangle objects resulting from TRISTRIP[_ADJ][_REV] topologies are [re]ordered when passed in the GS thread payload See Object Vertex Ordering table (below).

3DSTATE_GS_BODY

		<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>LEADING</td><td>Reorder the vertices of alternating triangles of a TRISTRIP[_ADJ] such that the leading (first) vertices are in consecutive order starting at v0. A similar reordering is performed on alternating triangles in a TRISTRIP_REV.</td></tr><tr><td>1h</td><td>TRAILING</td><td>Reorder the vertices of alternating triangles of a TRISTRIP[_ADJ] such that the trailing (last) vertices are in consecutive order starting at v2. A similar reordering is performed on alternating triangles in a TRISTRIP_REV.</td></tr></table>
	1	**Discard Adjacency** <table><tr><td>Format:</td><td>Enable</td></tr></table> <table><tr><td>When set, adjacent vertices <u>will not be passed</u> in the GS payload when objects with adjacency are processed. Instead, only the non-adjacent vertices will be passed in the same fashion as the without-adjacency form of the primitive. Software should set this bit whenever a GS kernel is used that <u>does not expect</u> adjacent vertices. This allows both with-adjacency/without-adjacency variants of the primitive to be submitted to the pipeline (via 3DPRIMITIVE) - the GS unit will silently discard any adjacent vertices and present the GS thread with only the internal object. When clear, adjacent vertices <u>will be passed</u> to the GS thread, as dictated by the incoming primitive type. Software should only clear this bit when a GS kernel is used that does expect adjacent vertices. E.g., if the GS kernel is compiled to expect a TRIANGLE_ADJ object, software must clear this bit. Software should also clear this bit if the GS kernel expects a POINT or PATCHLIST_n object (which don't have with-adjacency variants).</td></tr><tr><td>The only hardware assistance is to allow the submission of a with-adjacency variant of a primitive when operating with a GS kernel that expects the without-adjacency variant of the object. (E.g., when the GS kernel is compiled to expect a TRIANGLE object, software should set this bit just in case a TRILIST_ADJ is submitted to the pipeline.) Note that the GS unit is otherwise not aware of the object type that is expected by the GS kernel. It is up to software to ensure that the submitted primitive type (in 3DPRIMITIVE) is otherwise compatible with the object type expected by the GS kernel. (E.g., if the GS kernel expects a LINE_ADJ object, only LINELIST_ADJ or LINESTRIP_ADJ should be submitted, otherwise the GS kernel will produce unpredictable results.) Also note that it is possible to craft a GS kernel which can accept any object type that's thrown at it by first examining the PrimType passed in the payload and then using this info to correctly interpret the number of vertices passed in the payload.</td></tr></table>
	0	**Enable** <table><tr><td>Format:</td><td>Enable</td></tr></table> Specifies whether the GS stage is enabled or disabled (pass-through).
7	31	**Control Data Format** <table><tr><td>Format:</td><td>U1</td></tr></table> This field specifies the format of the control data header (if any). <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>CUT</td><td>The control data header contains Cut bits.</td></tr><tr><td>1h</td><td>SID</td><td>The control data header contains StreamID bits. Output Topology must be set to</td></tr></table>

3DSTATE_GS_BODY

DWord	Bit	Description
		POINTLIST, or behavior is UNDEFINED.
	30	**Static Output** Format: Enable Specifies whether the GS shader outputs a static number of vertices per invocation. If this bit is clear, the number of vertices output by each GS shader invocation is stored by the GS thread in Vertex Count at the very beginning of the output URB entry (see GS URB Entry description).
	29:27	**Reserved** Format: MBZ
	26:16	**Static Output Vertex Count** Format: U11 Count of object vertices If **GSEnable** is ENABLED and **StaticOutput** is ENABLED, this field specifies the total number of vertices output each GS shader invocation. If **GSEnable** is ENABLED and **StaticOutput** is DISABLED (i.e., variable GS output), the total number of vertices output by a GS shader invocation is stored by the thread at the very beginning of the output URB entry, and this field is ignored. (See GS URB Entry description). Value: [0,1024] — Name:
	15:9	**Reserved** Format: MBZ
	8:0	**Maximum Number of Threads** Format: U9-1 Thread count Specifies the maximum number of simultaneous threads allowed to be active. Used to avoid using up the scratch space. Programming the value of the max threads over the number of threads based off number of threads supported in the execution units may improve performance since the architecture allows threads to be buffered between the check for max threads and the actual dispatch into the EU. Programming the max values to a number less than the number of threads supported in the execution units may reduce performance. Value: [0,223] — Name: — Description: indicating thread count of [1,224]
8	31:27	**Reserved** Format: MBZ
	26:21	**Vertex URB Entry Output Read Offset** Format: U6 Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB by SBE. Value: [0,63] — Name:

3DSTATE_GS_BODY

	20:16	**Vertex URB Entry Output Length** Format: U5 Specifies the amount of URB data written for each Vertex URB entry, in 256-bit register increments. **Value** / **Name**: [1,16] **Programming Notes** This length does not include the vertex header.
	15:8	**User Clip Distance Clip Test Enable Bitmask** Format: Enable[8] This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept / must clip determination needs to be made. DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.
	7:0	**User Clip Distance Cull Test Enable Bitmask** Format: Enable[8] This 8 bit mask field selects which of the 8 user clip distances against which trivial reject / trivial accept determination needs to be made (does not cause a must clip). DX10 allows simultaneous use of ClipDistance and Cull Distance test of up to 8 distances.

3DSTATE_HS_BODY

<table>
<tr><th colspan="3">3DSTATE_HS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="5">0</td><td>31:30</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>29:27</td><td>Sampler Count
Format: U3
Specifies how many samplers (in multiples of 4) the HS kernels use. Used only for prefetching the associated sampler state entries.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>No Samplers</td><td>no samplers used</td></tr>
<tr><td>1h</td><td>1-4 Samplers</td><td>between 1 and 4 samplers used</td></tr>
<tr><td>2h</td><td>5-8 Samplers</td><td>between 5 and 8 samplers used</td></tr>
<tr><td>3h</td><td>9-12 Samplers</td><td>between 9 and 12 samplers used</td></tr>
<tr><td>4h</td><td>13-16 Samplers</td><td>between 13 and 16 samplers used</td></tr>
<tr><td>5h-7h</td><td>Reserved</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>26</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>25:18</td><td>Binding Table Entry Count
Format: U8
When HW Generated Binding Table is disabled:
Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state.
Note: 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.
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.</td></tr>
</table></td></tr>
<tr><td>17</td><td>Thread Dispatch Priority</td></tr>
</table>

<table>
<tr><th colspan="3">3DSTATE_HS_BODY</th></tr>
<tr><td></td><td></td><td>Specifies the priority of the thread for dispatch
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Normal</td><td>Normal Priority</td></tr>
<tr><td>1h</td><td>High</td><td>High Priority</td></tr>
</table></td></tr>
<tr><td></td><td>16</td><td>Floating Point Mode
<table><tr><td></td><td></td></tr></table>
Specifies the initial floating point mode used by the dispatched thread.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>IEEE-754</td><td>Use IEEE-754 Rules</td></tr>
<tr><td>1h</td><td>alternate</td><td>Use alternate rules</td></tr>
</table></td></tr>
<tr><td></td><td>15:14</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>13</td><td>Illegal Opcode Exception Enable
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table>
This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.</td></tr>
<tr><td></td><td>12</td><td>Software Exception Enable
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table>
This bit gets loaded into EU CRO1[13] (note the bit # difference). See Exceptions and ISA Execution Environment.</td></tr>
<tr><td></td><td>11:8</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>7:0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>1</td><td>31</td><td>Enable
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table>
Specifies whether the HS function is enabled or disabled (pass-through). If ENABLED MI_TOPOLOGY_FILTER must be used to silently discard any topologies that the HS kernel is not expecting. E.g., if the HS kernel is expecting PATCHLIST_32 topologies, MI_TOPOLOGY_FILTER must be set to PATCHLIST_32 so only those topologies can reach the enabled HS.</td></tr>
</table>

3DSTATE_HS_BODY

		Programming Notes The tessellation stages (HS, TE and DS) must be enabled/disabled as a group. I.e., draw commands can only be issued if all three stages are enabled or all three stages are disabled, otherwise the behavior is UNDEFINED.
	30	**Reserved** Format: MBZ
	29	**Statistics Enable** Format: Enable This bit controls whether HS-unit-specific statistics register(s) will increment (for each patch).
	28:27	**Reserved** Format: MBZ
	26:18	**Reserved** Format: MBZ
	17	**Reserved** Format: MBZ
	16:8	**Maximum Number of Threads** Format: U9-1 Specifies the maximum number of simultaneous threads allowed to be active. Used to avoid using up the scratch space. Programming the value of the max threads over the number of threads based off number of threads supported in the execution units may improve performance since the architecture allows threads to be buffered between the check for max threads and the actual dispatch into the EU. Programming the max values to a number less than the number of threads supported in the execution units may reduce performance. Value: [0,223] — Name: — Description: indicating thread count of [1,224]
	7:5	**Reserved** Format: MBZ
	4	**Reserved** Format: MBZ
	3:0	**Instance Count**

3DSTATE_HS_BODY

DWord	Bit	Description
		Format: U4-1 This field determines the number of threads (minus one) spawned per input patch. If the HS kernel uses a barrier function, software must restrict the **Instance Count** to the number of threads that can be simultaneously active within a subslice. Factors which must be considered includes scratch memory availability. **Value** | **Name** | **Description** [0,15] | | representing [1,16] instances **Programming Notes** Instance Count must be programmed to 0 (1 instance) whenever DispatchMode is programmed to DUAL_PATCH.
2..3	63:6	**Kernel Start Pointer** Format: InstructionBaseOffset[63:6] This field specifies the starting location (1st EU core instruction) of the kernel program run by threads spawned by this FF unit. It is specified as a 64-byte-granular offset from the Instruction Base Address.
	5:0	**Reserved** Format: MBZ
4..5	63:32	**Reserved** Format: MBZ
	31:10	**Scratch Space Base Pointer** Format: GeneralStateOffset[31:10] **Value** | **Name** | **Description** [0,31] | | Specifies the location of the scratch space area allocated to this FF unit, specified as a 1KB-granular offset from the General State Base Address. If required, each thread spawned by this FF unit will be allocated some portion of this space, as specified by Per-Thread Scratch Space.
	9:4	**Reserved** Format: MBZ
	3:0	**Per-Thread Scratch Space**

3DSTATE_HS_BODY

		Format: U4 power of 2 Bytes over 1K Bytes Specifies the amount of scratch space to be allocated to each thread spawned by this FF unit. The driver must allocate enough contiguous scratch space, starting at the Scratch Space Base Pointer, to ensure that the Maximum Number of Threads can each get Per-Thread Scratch Space size without exceeding the driver-allocated scratch space. **Value** | **Name** | **Description** [0,11] | | Indicating[1K Bytes, 2M Bytes
6	31:29	**Reserved** Format: MBZ
	28	**Dispatch GRF Start Register For URB Data [5]** Format: U1 Specifies bit [5] of the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload. The **Dispatch GRF Start Register For URB Data [4:0]** field is used to specify bits [4:0] of the starting GRF register number.
	27	**Single Program Flow** Format: Enable Specifies the initial condition of the kernel program as either a single program flow (SIMDnxm with m = 1) or as multiple program flows (SIMDnxm with m > 1). See CR0 description in *ISA Execution Environment*. **Value** | **Name** | **Description** 0h | Reserved | 1h | Enable | Single Program Flow Enabled
	26	**Vector Mask Enable** Format: Enable Enumerated Type Upon subsequent HS thread dispatches, this bit is loaded into the EU's Vector Mask Enable (VME, cr0.0[3]) thread state. Refer to the EU documentation for the definition and use of VME state. **Value** | **Name** | **Description** 0h | Dmask | The EU will use the Dispatch Mask (supplied by the HS stage) for instruction execution. 1h | Vmask | The EU will use the Vector Mask (derived from the Dispatch Mask) for instruction execution. **Programming Notes** Under normal conditions SW shall specify DMask, as the HS stage will provide a Dispatch Mask

3DSTATE_HS_BODY

		appropriate to SIMD4x2 or SIMD8 thread execution (as a function of dispatch mode). E.g., for SIMD4x2 thread execution, the HS state will generate a Dispatch Mask that is equal to what the EU would use as a Vector Mask. For SIMD8 execution there is no known usage model for use of Vector Mask (as there is for PS shaders).
	25	**Accesses UAV** Format: Enable This field must be set when HS has a UAV access **Programming Notes** This field must not be set when HS Function Enable is disabled.
	24	**Include Vertex Handles** Format: Boolean If set, all the input Vertex URB handles are included in payloads. This field is ignored if **HS Function Enable** is DISABLED. **Programming Notes** **Programming Restriction:** This field must be set if value if **Vertex URB Entry Read Length** is cleared to zero.
	23:19	**Dispatch GRF Start Register For URB Data** Format: U5 **Description** Specifies the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload. This field is ignored if **HS Function Enable** is DISABLED. The Dispatch GRF Start Register for URB Data [5] field is used to extend the range of the starting GRF register number to [0,63]. **Value** | **Name** | **Description** [0,31] | | indicating bits [4:0] of the GRF number **Programming Notes** When Include Vertex Handles is set for non-instanced 8_PATCH dispatch of PATCHLIST_30..32 objects, pushed vertex data and/or pushed constants cannot be used as they would need to start in the payload beyond the range of this field (i.e., beyond R31). When Include PrimitiveID is also set, this issue extends to non-instanced 8_PATCH dispatch of PATCHLIST_29..32 objects.
	18:17	**Dispatch Mode** Format: U2 This field is unused to set the current thread dispatch mode for the HS stage.

3DSTATE_HS_BODY

Value	Name	Description
0h	SINGLE_PATCH	HS threads are passed inputs and an output handle associated with a single input patch.
2h	8_PATCH	HS threads are passed inputs and an output handle associated with (up to) 8 patches in SIMD8 fashion. See 8_PATCH Thread Payload.
3h	Reserved	

Programming Notes
DUAL_PATCH is not supported.

16:11 Vertex URB Entry Read Length

Format:	U6

Specifies the amount of URB data read and passed in the thread payload for each Vertex URB entry, in 256-bit register increments. This field is ignored if HS Function Enable is DISABLED.

Value	Name
[0,63]	

Programming Notes
Programming Restriction: This field must be a non-zero value if **Include Vertex Handles** is cleared to zero.

10 Reserved

Format:	MBZ

9:4 Vertex URB Entry Read Offset

Format:	U6

Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB before being included in the thread payload. This offset applies to all Vertex URB entries passed to the thread. This field is ignored if HS Function Enable is DISABLED.

Value	Name
[0,63]	

3:1 Reserved

Format:	MBZ

0 Include Primitive ID

Format:	Enable

If set, R1 of the payload is written with Primitive ID value(s). If clear, these Primitive IDs are not included in the payload R1.

<table>
<tr><th colspan="3">3DSTATE_HS_BODY</th></tr>
<tr><td></td><td></td><td><table><tr><th>Programming Notes</th></tr><tr><td>This field is only used when DUAL_PATCH DispatchMode is specified. In SINGLE_PATCH, the single Primitive ID is always passed in R0.</td></tr></table></td></tr>
<tr><td>7</td><td>31:0</td><td>Reserved<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

3DSTATE_INDEX_BUFFER_BODY

3DSTATE_INDEX_BUFFER_BODY

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

DWord	Bit	Description
0	31:12	**Reserved** Format: MBZ
	11	**Reserved** Format: MBZ
	10	**Reserved** Format: MBZ
	9:8	**Index Format** Format: U2 Enumerated type This field specifies the data format of the index buffer. All index values are UNSIGNED. Value / Name: 0h: BYTE 1h: WORD 2h: DWORD
	7	**Reserved** Format: MBZ
	6:0	**Memory Object Control State** Format: MEMORY_OBJECT_CONTROL_STATE Specifies the memory object control state for this index buffer.
1..2	63:0	**Buffer Starting Address** Format: GraphicsAddress[63:0] This field contains the size-aligned (as specified by Index Format) Graphics Address LSBs of the first element of interest within the index buffer. Software must program this value with the combination (sum) of the base address of the memory resource and the byte offset from the

3DSTATE_INDEX_BUFFER_BODY		
		base address to the starting structure within the buffer. **Programming Notes** Index Buffers can only be allocated in linear (not tiled) graphics memory.
3	31:0	**Buffer Size** Format: U32 Count of bytes This field specifies the size of the buffer in bytes. Index accesses which straddle or go past the end of the buffer will return 0..Note that BufferSize=0 indicates that there is no valid data in the buffer. **Value** / **Name** [0, FFFFFFFFh]

3DSTATE_MULTISAMPLE_BODY

<table>
<tr><th colspan="3">3DSTATE_MULTISAMPLE_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:6</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>5</td><td>Pixel Position Offset Enable
Format: Enable
Enables the device to offset pixel positions by 0.5 both in horizontal and vertical directions.
Programming Notes
Setting this field along with setting the Pixel Location to upper left and number of multisamples to greater than one will cause the device to offset pixel positions by 0.5 both in horizontal and vertical directions.
It is to be noted this is done to adjust the pixel co-ordinate system to DX9 like, so any WM_HZ_OP screen space rectangles (eg: legacy HiZ Clear, Resolve etc) generated internally by driver in this mode needs to be aware of this offset adjustment and send the rectangles according to alignment restriction taking this offset adjustment into consideration.
SW can choose to set this bit only for DX9 API. DX10/OGL API's should not have any effect by setting or not setting this bit.</td></tr>
<tr><td></td><td>4</td><td>Pixel Location
Format: U1
This field specifies where the device evaluates "pixel" (vs. centroid or sample) values/attributes.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>CENTER</td><td>Use the pixel center (0.5, 0.5 offset)</td></tr><tr><td>1h</td><td>UL_CORNER</td><td>Use the pixel upper-left corner</td></tr></table>Programming Notes
The programming of this field is assumed to be a function of the API being supported. Specifically, it is expected that OpenGL and DX10+ APIs require CENTER selection, while DX9- APIs require UL_CORNER selection.
When 3DSTATE_RASTER::ForcedSampleCount is other than NUMRASTSAMPLES_0, this field must be 0h.</td></tr>
<tr><td></td><td>3:1</td><td>Number of Multisamples</td></tr>
</table>

3DSTATE_MULTISAMPLE_BODY

Format:	U3

This field specifies how many samples/pixel exist in all RTs and the Depth Buffer, as log2(#samples). This field is valid regardless of the setting of **Multisample Rasterization Mode**.

Value	Name	Description
0h	1	1 sample/pixel
1h	2	2 samples/pixel
2h	4	4 samples/pixel
3h	8	8 samples/pixel
4h	16	16 samples/pixel
5h-7h	Reserved	

Programming Notes
The setting of this field must match the **Number of Multisamples** field in SURFACE_STATE of all bound render targets.

0 **Reserved**

Format:	MBZ

3DSTATE_PS_BLEND_BODY

3DSTATE_PS_BLEND_BODY		
Source: RenderCS Size (in bits): 32 Default Value: 0x00000000		
DWord	**Bit**	**Description**
0	31	**Alpha To Coverage Enable** Format: Enable If set, indicates that AlphaToCoverage is on RT[0], since this bit must be set the same for all RTs in the MRT case.
	30	**Has Writeable RT** Format: Enable When set indicates the there is at least one non-null RT w/ at least one channel write enabled
	29	**Color Buffer Blend Enable** Format: Enable When set indicates that RT[0] has color buffer blend enabled.
	28:24	**Source Alpha Blend Factor** Format: 3D_Color_Buffer_Blend_Factor Indicates the "source factor" in alpha Color Buffer Blending stage for RT[0]
	23:19	**Destination Alpha Blend Factor** Format: 3D_Color_Buffer_Blend_Factor Indicates the "destination factor" in alpha Color Buffer Blending stage for RT[0]
	18:14	**Source Blend Factor** Format: 3D_Color_Buffer_Blend_Factor Indicates the "source factor" in Color Buffer Blending stage for RT[0]
	13:9	**Destination Blend Factor**

![intel]

3DSTATE_PS_BLEND_BODY

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**3D_Color_Buffer_Blend_Factor**</td></tr></table> Indicates the "destination factor" in Color Buffer Blending stage for RT[0]
	8	**Alpha Test Enable** <table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table> Indicates the AlphaTestEnable for RT[0]
	7	**Independent Alpha Blend Enable** <table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table> Indicates the Independent Alpha Blend Enable for RT[0] 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.
	6:0	**Reserved** <table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>

3DSTATE_PS_BODY

<table>
<tr><th colspan="3">3DSTATE_PS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 352
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0..1</td><td>63:6</td><td>Kernel Start Pointer 0
Format: InstructionBaseOffset
Specifies the 64-byte aligned address offset of the first instruction in the kernel[0]. This pointer is relative to the Instruction Base Address.</td></tr>
<tr><td>5:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="3">2</td><td>31</td><td>Single Program Flow
Single Program Flow (SPF) specifies the initial condition of the kernel program as either a single program flow (SIMDnxm with m = 1) or as multiple program flows (SIMDnxm with m > 1). See CR0 description in ISA Execution Environment.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Multiple</td><td>Multiple Program Flows</td></tr>
<tr><td>1h</td><td>Single</td><td>Single Program Flows</td></tr>
</table></td></tr>
<tr><td>30</td><td>Vector Mask Enable
Format: Enable
When SPF=0, Vector Mask Enable (VME) specifies which mask to use to initialize the initial channel enables. When SPF=1, VME specifies which mask to use to generate execution channel enables.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Dmask</td><td>Channels are enabled based on the dispatch mask</td></tr>
<tr><td>1h</td><td>Vmask</td><td>Channels are enabled based on the vector mask</td></tr>
</table></td></tr>
<tr><td>29:27</td><td>Sampler Count
Specifies how many samplers (in multiples of 4) the vertex shader 0 kernel uses. Used only for prefetching the associated sampler state entries.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>[0,4]</td><td></td><td></td></tr>
<tr><td>0h</td><td>No Samplers</td><td>no samplers used</td></tr>
<tr><td>1h</td><td>1-4 Samplers</td><td>between 1 and 4 samplers used</td></tr>
<tr><td>2h</td><td>5-8 Samplers</td><td>between 5 and 8 samplers used</td></tr>
<tr><td>3h</td><td>9-12 Samplers</td><td>between 9 and 12 samplers used</td></tr>
<tr><td>4h</td><td>13-16 Samplers</td><td>between 13 and 16 samplers used</td></tr>
<tr><td>5h-7h</td><td></td><td>Reserved</td></tr>
</table></td></tr>
</table>

3DSTATE_PS_BODY

	26	**Single Precision Denormal Mode** Specifies the single precision denormal mode used by the dispatched thread.

Value	Name	Description
0h	Flushed to Zero	Single Precision denormals are flushed to zero
1h	Retained	Single Precision denormals are retained

25:18 Binding Table Entry Count

Description
Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state. **Note:** For kernels using a large number of binding table entries, it may be advantageous to set this field to zero to avoid prefetching too many entries and thrashing the state cache. This field is ignored if [PS Function Enable] is DISABLED.
When [HW Generated Binding Table] bit is enabled: This field indicates which cache lines (512bit units - 32 Binding Table Entry section) should be fetched. Each bit in this field corresponds to a cache line. Only the 1st 4 non-zero Binding Table entries of each 32 Binding Table entry section prefetched will have its surface state prefetched. See 3D Pipeline for more information.

Programming Notes
When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.

17 Thread Dispatch Priority

Specifies the priority of the thread for dispatch.

Value	Name	Description
0h	Normal	Normal Priority
1h	High	High Priority

16 Floating Point Mode

Specifies the floating point mode used by the dispatched thread.

Value	Name	Description
0h	IEEE-754	Use IEEE-754 rules
1h	Alternate	Use alternate rules

15:14 Rounding Mode

Specifies the rounding mode used by the dispatched thread.

Value	Name	Description
0h	RTNE	Round to Nearest Even
1h	RU	Round toward +infinity
2h	RD	Round toward -infinity
3h	RTZ	Round toward zero

13 Illegal Opcode Exception Enable

3DSTATE_PS_BODY		
		Format: Enable This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.
	12	**Reserved** Format: MBZ
	11	**Mask Stack Exception Enable** Format: Enable This bit gets loaded into EU CR0.1[12] (note the bit # difference). 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
3..4	63:32	**Reserved** Format: MBZ
	31:10	**Scratch Space Base Pointer** Format: GeneralStateOffset[31:10]ScratchSpace Specifies the 1k-byte aligned address offset to scratch space for use by the kernel. This pointer is relative to the General State Base Address. **Programming Notes** Scratch Space per slice is computed based on 4 sub-slices. SW must allocate scratch space enough so that each slice has 4 slices allowed.
	9:4	**Reserved** Format: MBZ
	3:0	**Per Thread Scratch Space** Format: U4 Specifies the amount of scratch space allowed to be used by each thread. The driver must allocate enough contiguous scratch space, pointed to by the Scratch Space Pointer, to ensure that the Maximum Number of Threads each get Per Thread Scratch Space size without exceeding the driver-allocated scratch space. **Value** — **Name** [0,11] — indicating [1k bytes, 2M bytes] in powers of two
5	31:23	**Maximum Number of Threads Per PSD** Specifies the maximum number of simultaneous virtual threads allowed to be active per Pixel Shader Dispatch(PSD). PSD serves a pair of subslices. This bit-field can be programmed in the

3DSTATE_PS_BODY		
		range: [0,63] each integer in the range linearly maps to maximum number of threads in the range: [2, 128]. If a programmed value is k, it implies 2(k+1) threads. **Programming Notes** If this field is changed between 3DPRIMITIVE commands, a PIPE_CONTROL command with Stall at Pixel Scoreboard set is required to be issued.
	22	**Reserved** Format: MBZ
	21	**Reserved** Format: MBZ
	20	**Reserved** Format: MBZ
	19:12	**Reserved** Format: MBZ
	11	**Push Constant Enable** Format: Enable This field must be enabled if the sum of the **PS Constant Buffer [3:0] Read Length** fields in 3DSTATE_CONSTANT_PS is nonzero, and must be disabled if the sum is zero.
	10	**Reserved** Format: MBZ
	9	**Reserved** Format: MBZ
	8	**Render Target Fast Clear Enable** Format: Enable This field is set to enable fast clear of the bound render targets. See "Render Target Fast Clear" for restrictions on enabling this field. **Programming Notes** For PoSH based Tiled Rendering, Color Fast clear operation is recommended to be performed outside of tile pass, for performance reasons. After Fast clear, render cache flush is required. When this bit is set, corresponding BTI for the render target that is being cleared must be equal to 0. When this bit is set, RENDER_SURFACE_STATE type must not be NULL.
	7:6	**Render Target Resolve Type** Format: U2 Enumerated Type Specifies what type of Render Target Resolve is needed for the surface to be consumed properly by the end Client. Programming notes below.

Value	Name	Description	Programming Notes
0h	RESOLVE_DISABLED	No Resolve Needed	
1h	RESOLVE_PARTIAL	Partial resolve is for resolving	[] Display engine does not support

3DSTATE_PS_BODY

		RT for clear values i.e. it leaves no cache lines at implied clear value.	unresolved clear values in the display buffer, hence this resolve is required before binding any compressed RT to the display via flip commands.
2h	FAST_CLEAR_0	Fast Clear to 0 during Clear Pass; Used to Initialize CCS Buffer with 0s to support Lossless Compressed Without Clear.	This state has to be programmed only with Render Target Fast Clear Enable described above. If the Render Target Fast Clear = 0, this Field Cannot be programmed to 2h.
3h	RESOLVE_FULL	Full Resolve is for Resolving RT for Clear/Compressed to Uncompressed State	

Programming Notes
When this bit is set, corresponding BTI for the render target that is being resolved must be equal to 0.
When this bit is set, RENDER_SURFACE_STATE type must not be NULL.

5 **Reserved**

Format:	MBZ

4:3 **Position XY Offset Select**

Format:	U2 Enumerated Type

This field specifies if/what Position XY Offset values are passed in the PS payload. Note that these are per-slot (pixel|sample) offsets, and therefore separate from the subspan XY coordinates passed in R1.

Value	Name	Description
0h	POSOFFSET_NONE	No Position XY Offsets are included in the PS payload.
1h	Reserved	
2h	POSOFFSET_CENTROID	Position XY Offsets will be passed in the PS payload, and these will reflect the Centroid position(s).
3h	POSOFFSET_SAMPLE	Position XY Offsets will be passed in the PS payload, and these will reflect the multisample position(s).

Programming Notes
SW Recommendation: If the PS kernel needs the Position Offsets to compute a Position XY value, this field should match Position ZW Interpolation Mode to ensure a consistent position.xyzw computation
If the PS kernel does not need the Position XY Offsets to compute a Position Value, then this field should be programmed to POSOFFSET_NONE, as the PS kernel should be using the various barycentric inputs to evaluate other-than-position attributes. However, this field can be

3DSTATE_PS_BODY

DWord	Bit	Description
		used to pass Centroid or Sample offsets in the payload for special test modes (e.g., where barycentric coordinates are computed in the PS vs. being HW-generated and passed in the payload). MSDISPMODE_PERSAMPLE is required in order to select POSOFFSET_SAMPLE.
	2	**32 Pixel Dispatch Enable** Format: Enable Enables the Windower to dispatch 8 subspans in one payload. Variable Pixel Dispatch in Section: Pixel Grouping (Dispatch size) control for valid pixel dispatch combinations. **Programming Notes** When NUM_MULTISAMPLES = 16 or FORCE_SAMPLE_COUNT = 16, SIMD32 Dispatch must not be enabled for PER_PIXEL dispatch mode.
	1	**16 Pixel Dispatch Enable** Format: Enable Enables the Windower to dispatch 4 subspans in one payload. Variable Pixel Dispatch in Section: Pixel Grouping (Dispatch size) control for valid pixel dispatch combinations.
	0	**8 Pixel Dispatch Enable** Format: Enable Enables the Windower to dispatch 2 subspans from 1 object (polygon) in one payload. Variable Pixel Dispatch in Section: Pixel Grouping (Dispatch size) control for valid pixel dispatch combinations. **Programming Notes** When Render Target Fast Clear Enable is ENABLED or Render Target Resolve Type = RESOLVE_PARTIAL or RESOLVE_FULL, this bit must be DISABLED.
6	31:23	**Reserved** Format: MBZ
	22:16	**Dispatch GRF Start Register For Constant/Setup Data 0** Format: U7 Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[0]. **Value** / **Name** [0,127] /
	15	**Reserved** Format: MBZ
	14:8	**Dispatch GRF Start Register For Constant/Setup Data 1** Format: U7 Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[1]. **Value** / **Name** [0,127] /

3DSTATE_PS_BODY

DWord	Bit	Description
	7	**Reserved** Format: MBZ
	6:0	**Dispatch GRF Start Register For Constant/Setup Data 2** Format: U7 Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[2]. Value: [0,127] — Name:
7..8	63:6	**Kernel Start Pointer 1** Format: InstructionBaseOffset[63:6]Kernel Specifies the 64-byte aligned address offset of the first instruction in kernel[1]. This pointer is relative to the Instruction Base Address.
	5:0	**Reserved** Format: MBZ
9..10	63:6	**Kernel Start Pointer 2** Format: InstructionBaseOffset[63:6]Kernel Specifies the 64-byte aligned address offset of the first instruction in kernel[2]. This pointer is relative to the **Instruction Base Address**.
	5:0	**Reserved** Format: MBZ

3DSTATE_PS_EXTRA_BODY

3DSTATE_PS_EXTRA_BODY

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

DWord	Bit	Description
0	31	**Pixel Shader Valid** Format: Enable When set indicates a valid pixel shader. When this bit clear the rest of this command should also be clear.
	30	**Pixel Shader Does not write to RT** Format: Enable When set indicates the pixel shader does not write to render target. **Programming Notes** When Pixel Shader writes to UAV but does not write to RT, a dummy render target write is required to convey EOT to the PS dispatch function. Hence, this bit must be reset in this case. Whenever, there is a render target write message even to the NULL render target, this bit must be reset. When Pixel Shader Kills Pixel is set, SW must perform a dummy render target write from the shader and not set this bit, so that Occlusion Query is correct.
	29	**oMask Present to Render Target** Format: Enable This bit is inserted in the PS payload header and made available to the DataPort (either via the message header or via header bypass) to indicate that oMask data from the shader (one or two phases) is included in Render Target Write messages. If present, the oMask data is used to mask off samples.
	28	**Pixel Shader Kills Pixel** Format: Enable This bit, if ENABLED, indicates that the PS kernel has the ability to kill (discard) pixels or samples, other than due to depth or stencil testing. This bit is required to be ENABLED in the following situations: • The API pixel shader program contains "killpix" or "discard" instructions, or other code in the pixel shader kernel that can cause the final pixel mask to differ from the pixel mask received on dispatch.
	27:26	**Pixel Shader Computed Depth Mode** Format: U2 Enumerated Type This field specifies the computed depth mode for the pixel shader.

3DSTATE_PS_EXTRA_BODY

		<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>PSCDEPTH_OFF</td><td>Pixel shader does not compute depth</td></tr><tr><td>1h</td><td>PSCDEPTH_ON</td><td>Pixel shader computes depth with no guarantee as to its value</td></tr><tr><td>2h</td><td>PSCDEPTH_ON_GE</td><td>[] Pixel shader computes depth and guarantees that oDepth >= SourceDepth. If the Position ZW interpolation mode in 3DSTATE_WM does not match the DX Multisample Rasterization mode in 3DSTATE_RASTER, HW will internally convert to PSCDEPTH_ON.</td></tr><tr><td>3h</td><td>PSCDEPTH_ON_LE</td><td>Pixel shader computes depth and guarantees that oDepth <= SourceDepth If the Position ZW interpolation mode in 3DSTATE_WM does not match the DX Multisample Rasterization mode in 3DSTATE_RASTER, HW will internally convert to PSCDEPTH_ON.</td></tr></table> <table><tr><th>Programming Notes</th></tr><tr><td>If this field is set to any value other than PSCDEPTH_OFF, a multi-phase shader (i.e. dispatch RATE_COARSE or RATE_PIXEL with pixel/sample loops or sample loop respectively) must output depth and render targets only at the last phase.</td></tr><tr><td>When PS dispatch rate is COARSE_RATE, this field must be programmed to PSCDEPTH_OFF.</td></tr></table>
	25	**Force Computed Depth** <table><tr><td>Format:</td><td>Enable</td></tr></table> <table><tr><th>Programming Notes</th></tr><tr><td>This field should be left DISABLED. This field should not be tested for functional validation.</td></tr></table>
	24	**Pixel Shader Uses Source Depth** <table><tr><td>Format:</td><td>Enable</td></tr></table>This bit, if ENABLED, indicates that the PS kernel requires the source depth value (vPos.z) to be passed in the payload. The source depth value is interpolated according to the Position ZW Interpolation Mode state. <table><tr><th>Programming Notes</th></tr><tr><td>This bit cannot be enabled when dispatch rate is RATE_COARSE</td></tr></table>
	23	**Pixel Shader Uses Source W** <table><tr><td>Format:</td><td>Enable</td></tr></table>This bit, if ENABLED, indicates that the PS kernel requires the interpolated source W value (vPos.w) to be passed in the payload. The W value is interpolated according to the Position ZW Interpolation Mode state.
	22	**Pixel Shader Requires Requested Coarse Pixel Shading Size** <table><tr><td>Format:</td><td>Enable</td></tr></table>This bit, if ENABLED, indicates that the PS kernel requires values of requested coarse pixel shading size to be passed in the payload for each 2x2 coarse pixel quad. Note: Actual coarse

3DSTATE_PS_EXTRA_BODY

		pixel shading rate is always delivered (constant across thread slot). This bit can only be set when dispatch rate is RATE_COARSE.
	21	**Pixel Shader Requires Source Depth and/or W Plane Coefficients** Format: Enable This bit, if ENABLED, indicates that the PS kernel requires the source depth and/or W plane equations to be passed in the payload. Note: both attributes are always delivered in same message phase, even if only one is used.
	20	**Pixel Shader Requires Perspective Bary Plane Coefficients** Format: Enable This bit, if ENABLED, indicates that the PS kernel requires the perspective plane coefficients to be passed in the payload.
	19	**Pixel Shader Requires Non-Perspective Bary Plane Coefficients** Format: Enable This bit, if ENABLED, indicates that the PS kernel requires the non-perspective plane coefficients to be passed in the payload.
	18	**Pixel Shader Requires Subpixel Sample Offsets** Format: Enable This bit, if ENABLED, indicates that the PS kernel requires the sub-pixel sample offsets to be passed in the payload.
	17	**Reserved** Format: MBZ
	16:12	**Reserved** Format: MBZ
	11	**Reserved** Format: MBZ
	10	**Reserved** Format: MBZ
	9	**Reserved**
	8	**Attribute Enable** Format: Enable This field must be enabled if the Number of SF Output Attributes field in 3DSTATE_SBE is nonzero, and must be disabled if that field is zero.
	7	**Pixel Shader Disables Alpha To Coverage** Format: Enable When set indicates the pixel shader AlphaToCoverage should be disabled due to oMask output. The setting of this bit is API dependent.
	6	**Pixel Shader Is Per Sample** Format: Enable

3DSTATE_PS_EXTRA_BODY

This bit, when ENABLED, indicates that the pixel shader is dispatched at the per sample shading rate. If this bit is DISABLED, the dispatch rate is determined by the value of Pixel Shader Is Per Coarse Pixel. If this bit is ENABLED, Pixel Shader Is Per Coarse Pixel bit must be DISABLED.

5 — Pixel Shader Computes Stencil

Format:	Enable

This field when set indicates that the pixel shader computes the stencil reference value.

Programming Notes
If this field is ENABLED, a multi-phase shader (i.e. dispatch RATE_COARSE or RATE_PIXEL with pixel/sample loops or sample loop respectively) must output stencil and render targets only at the last phase.
WhenPixel Shader is at COARSE_RATE, this field must not be set.

4 — Pixel Shader Is Per Coarse Pixel

Format:	Enable

If Pixel Shader Is Per Sample is DISABLED and this bit is ENABLED, the pixel shader is dispatched at the per coarse pixel shading rate. If Pixel Shader Is Per Sample is DISABLED and this bit is DISABLED, the pixel shader is dispatched at the per pixel shading rate. If Pixel Shader Is Per Sample is ENABLED, this bit must be DISABLED.

Restriction
SIMD32 kernel version cannot be configured when this bit is ENABLED.

3 — Pixel Shader Pulls Bary

Format:	Enable

This bit indicates if Pixel Shader uses Pull Bary i.e. PI message. If this bit is reset, PS does not do Pull Bary.

2 — Pixel Shader Has UAV

Format:	Enable
Format:	U1 Enumerated Type

This field when set indicates that the pixel shader has a UAV attached to it.

1:0 — Input Coverage Mask State

Format:	U2

This field indicates the type of input coverage mask that the PS kernel requires to be passed in the payload.

Value	Name	Description	Programming Notes
0h	NONE	Pixel shader does not use input coverage masks.	
1h	NORMAL	Input Coverage masks based on outer conservatism and factors in SAMPLE_MASKs. If Pixel is conservatively covered, all samples are enabled.	
2h	INNER_CONSERVATIVE	Input Coverage masks based on	When PS Dispatch Rate is at

3DSTATE_PS_EXTRA_BODY

				inner conservatism. If Pixel is conservatively fully covered all samples are enabled else none of the samples are covered.	Coarse Pixel, requesting this Input Coverage Mask mode is illegal and not supported by HW. Input converage masks based on inner consevatism incorrectly ANDs SAMPLE_MASK in HW. Therefore, PS must retrieve the INNER coverage mask per pixel by bit-wise OR operation. [] Input converage masks based on inner consevatism incorrectly ANDs SAMPLE_MASK in HW. Therefore, PS must retrieve the INNER coverage mask per pixel by bit-wise OR operation. [] Input converage masks based on inner consevatism incorrectly ANDs SAMPLE_MASK in HW. Therefore, PS must retrieve the INNER coverage mask per pixel by bit-wise OR operation.
		3h	DEPTH_COVERAGE	Input coverage masks are computed after factoring depth/stencil test results, only if Early Depth Stencil Test is enabled. If Early Depth Stencil Test is not enabled, HW uses NORMAL Input Coverage Masks.	

3DSTATE_PTBR_MARKER_BODY

3DSTATE_PTBR_MARKER_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:2	**Reserved** Format: MBZ
	1	**End of Tile** Format: Enable When set, indicates marker stating End of Tile in the command sequence.
	0	**Start of Tile** Format: Enable When set, indicates marker stating Start of Tile in the command sequence.

3DSTATE_PTBR_TILE_SELECT_BODY

<table>
<tr><th colspan="3">3DSTATE_PTBR_TILE_SELECT_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="6">0</td><td>31</td><td>Free Render List Disable
<table><tr><td>Format:</td><td>Disable</td></tr></table>
This bit controls the recycling (Freeing up, add back to the free pool) of the visibility data pages by render pipe.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>Render pipe will free the pages to be recycled after consuming the visibility data for the current tile.</td></tr>
<tr><td>1</td><td></td><td>Render pipe will not free the pages to be recycled after consuming the visibility data for the current tile.</td></tr>
</table></td></tr>
<tr><td>30</td><td>Geometry Statistics Disable
<table><tr><td>Format:</td><td>Disable</td></tr></table>
This bit controls the incrementing statistics counters in geometry units (VF, VS, HS, TE, TDS, GS, SOL, CL, SF).
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>[Default]</td><td>Geometry units (VF, VS, HS, TE, TDS, GS, SOL, CL, SF) will increment their pipeline statistics counters.</td></tr>
<tr><td>1</td><td></td><td>Geometry units (VF, VS, HS, TE, TDS, GS, SOL, CL, SF) will not increment their pipeline statistics counters.</td></tr>
</table></td></tr>
<tr><td>29</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
<table>
<tr><th>Description</th></tr>
<tr><td>Bit 29 is reserved for HW use only.</td></tr>
</table></td></tr>
<tr><td>28</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>27:24</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>23:16</td><td>Render List Index
<table><tr><td>Format:</td><td>U8</td></tr></table>
Specifies the index in to the Render-List for the current Tile. Range [0..127].
HW will fetch the starting page offset for the visibility data of the current tile from below memory location</td></tr>
</table>

<table>
<tr><th colspan="3">3DSTATE_PTBR_TILE_SELECT_BODY</th></tr>
<tr><td></td><td></td><td>[{render_list_base_address[47:12], 12’b0} + {render_list_pointer[31:6], 6’b0} + (Render List Index «1)]
Programming Notes
Render List Index must be set to “0” when 3DSTATE_TILE_PASS_INFO:Tile Count is 0x0 for the corresponding Tile Pass.</td></tr>
<tr><td></td><td>15:10</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>9:0</td><td>Tile Rect Array Index
Format: U10
Specifies the index in to the Tile Rect Array of the current Tile Pass. Rang [0..1023].HW will fetch the RECT_STATE of the current tile from below memory location
[{dynamic_state_base_addres[47:12], 12’b0} + {Tile Rect Array Pointer[31:6], 6’b0} + (Tile Index«3)]
Programming Notes
Tile Rect Array Index must be set to “0” when 3DSTATE_TILE_PASS_INFO:Tile Count is 0x0 for the corresponding Tile Pass.</td></tr>
</table>

3DSTATE_RASTER_BODY

3DSTATE_RASTER_BODY		
Source:	RenderCS	
Size (in bits):	128	
Default Value:	0x00210000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:28	**Reserved** Format: MBZ
	27	**Reserved** Format: MBZ
	26	**Viewport Z Far Clip Test Enable** Format: Enable This field is used to control whether the Viewport Z Far extent is considered in VertexClipTest.
	25	**Reserved** Format: MBZ
	24	**Conservative Rasterization Enable** Format: Enable This field when set enables conservative rasterization rules for all primitives except rectangles, points and lines. For rectangle, points and lines, setting this bit is ignored by hardware. **Programming Notes** This bit must not be set for primitives with poly-stippling enabled. When this bit is set, sampling mode must be set to "Centre" sampling i.e 3DSTATE_MULTISAMPLE::Pixel Location set to CENTER
	23:22	**API Mode** Software sets this field according to the API's version. These bits are set for DX9 or OGL/DX10.0/DX10.1+/DX11.1 per the following values. Value / Name: 0h: DX9/OGL 1h: DX10.0 2h: DX10.1+ 3h: Reserved
	21	**Front Winding** Determines whether a triangle object is considered "front facing" if the screen space vertex positions, when traversed in the order, result in a clockwise (CW) or counter-clockwise (CCW) winding order. Does not apply to points or lines. Value / Name / Description: 0h: Clockwise: FRONTWINDING_CW 1h: Counter Clockwise **[Default]**: FRONTWINDING_CCW

3DSTATE_RASTER_BODY

	20:18	**Forced Sample Count**

Format:	U3 Enumerated Type

This field specifies how many samples/pixel exist for RT Independent Rasterization

Value	Name	Description
0h	NUMRASTSAMPLES_0	No RT Independent Rasterization
1h	NUMRASTSAMPLES_1	1 rast-sample/pixel
2h	NUMRASTSAMPLES_2	2 rast-samples/pixel
3h	NUMRASTSAMPLES_4	4 rast-samples/pixel
4h	NUMRASTSAMPLES_8	8 rast-samples/pixel
5h	NUMRASTSAMPLES_16	16 rast-samples/pixel
6h-7h	Reserved	

Programming Notes
When 3DSTATE_MULTISAMPLE::Number of Multisamples !=NUMSAMPLES_1, this field must be either NUMRASTSAMPLES_0 or NUMRASTSAMPLES_1. When 3DSTATE_MULTISAMPLE::Number of Multisamples == NUMSAMPLES_1, this field must not be NUMRASTSAMPLES_1.

	17:16	**Cull Mode**

Controls removal (culling) of triangle objects based on orientation. The cull mode only applies to triangle objects and does not apply to lines, points or rectangles.

Value	Name	Description
0h	CULLMODE_BOTH	All triangles are discarded (i.e., no triangle objects are drawn)
1h	CULLMODE_NONE **[Default]**	No triangles are discarded due to orientation
2h	CULLMODE_FRONT	Triangles with a front-facing orientation are discarded
3h	CULLMODE_BACK	Triangles with a back-facing orientation are discarded

Programming Notes
Orientation determination is based on the setting of the Front Winding state.

	15	**Reserved**

Format:	MBZ

	14	**Force Multisampling**

This field provides a work around override for the computation of SF_INT::Multisample Rasterization Mode and WM_INT::Multisample Rasterization Mode.

Value	Name	Description
0h	Normal	Multisampling mode is computed by HW according to formula for signal SF_INT::Multisample Rasterization Mode and WM_INT::Multisample Rasterization Mode in 3DSTATE_WM.

3DSTATE_RASTER_BODY

		1h | Force | Forces the DX Multisampling mode to be used directly
	13	**Smooth Point Enable** Format: Enable Software sets this according to API. When OGL and smooth point rasterization is required, this bit must be set. HW ignores this bit for primitives other than points.
	12	**DX Multisample Rasterization Enable** Format: Enable Software sets this according to the API's multisample enable **Programming Notes** This state only effects how the SF_INT/WM_INT::Multisample Rasterization Mode are set depending on some other states. This state mainly modifies the how the line rendering is done by setting SF_INT/WM_INT::Multisample Rasterization Mode to either OFF* or ON* . Please refer to table under SF_INT::Multisample Rasterization Mode.
	11:10	**DX Multisample Rasterization Mode** Format: U2 enumerated type This field determines whether multisample rasterization is turned on/off, and how the pixel sample point(s) are defined. Software sets this according to the API's multisample state setting (if any) Value | Name 0h | MSRASTMODE_ OFF_PIXEL 1h | MSRASTMODE_ OFF_PATTERN 2h | MSRASTMODE_ ON_PIXEL 3h | MSRASTMODE_ ON_PATTERN **Programming Notes** This field is used to directly set the SF_INT/WM_INT::Multisample Rasterization Mode when DX Multisample Rasterization Enable is set. Please refer to equation of SF_INT::Multisample Rasterization Mode.
	9	**Global Depth Offset Enable Solid** Format: Enable Enables computation and application of Global Depth Offset for SOLID objects.
	8	**Global Depth Offset Enable Wireframe** Format: Enable Enables computation and application of Global Depth Offset when triangles are rendered in WIREFRAME mode.
	7	**Global Depth Offset Enable Point** Format: Enable Enables computation and application of Global Depth Offset when triangles are rendered in POINT mode.
	6:5	**Front Face Fill Mode**

<table>
<tr><th colspan="3">3DSTATE_RASTER_BODY</th></tr>
<tr><td></td><td></td><td>
<table><tr><td>Format:</td><td>U2 enumerated type</td></tr></table>
This state controls how front-facing triangle and rectangle objects are rendered.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>SOLID</td><td>Any triangle or rectangle object found to be front-facing is rendered as a solid object. This setting is required when rendering rectangle (RECTLIST) objects.</td></tr>
<tr><td>1h</td><td>WIREFRAME</td><td>Any triangle object found to be front-facing is rendered as a series of lines along the triangle boundaries (as determined by the topology type and controlled by the vertex EdgeFlags).</td></tr>
<tr><td>2h</td><td>POINT</td><td>Any triangle object found to be front-facing is rendered as a set of point primitives at the triangle vertices (as determined by the topology type and controlled by the vertex EdgeFlags).</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>4:3</td><td>
Back Face Fill Mode
<table><tr><td>Format:</td><td>U2 enumerated type</td></tr></table>
This state controls how back-facing triangle and rectangle objects are rendered.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>SOLID</td><td>Any triangle or rectangle object found to be back-facing is rendered as a solid object. This setting is required when rendering rectangle (RECTLIST) objects.</td></tr>
<tr><td>1h</td><td>WIREFRAME</td><td>Any triangle object found to be back-facing is rendered as a series of lines along the triangle boundaries (as determined by the topology type and controlled by the vertex EdgeFlags).</td></tr>
<tr><td>2h</td><td>POINT</td><td>Any triangle object found to be back-facing is rendered as a set of point primitives at the triangle vertices (as determined by the topology type and controlled by the vertex EdgeFlags).</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>2</td><td>
Antialiasing Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
This field enables "alpha-based" line antialiasing.
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This field must be disabled if any of the render targets have integer (UINT or SINT) surface format.</td></tr>
</table>
</td></tr>
<tr><td></td><td>1</td><td>
Scissor Rectangle Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
Enables operation of Scissor Rectangle.
</td></tr>
<tr><td></td><td>0</td><td>
Viewport Z Near Clip Test Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
This field is used to control whether the Viewport Z Near extent is considered in VertexClipTest.
</td></tr>
<tr><td>1</td><td>31:0</td><td>Global Depth Offset Constant</td></tr>
</table>

3DSTATE_RASTER_BODY		
		Format: IEEE_FLOAT32 Specifies the constant term in the Global Depth Offset function.
2	31:0	**Global Depth Offset Scale** Format: IEEE_FLOAT32 Specifies the scale term used in the Global Depth Offset function.
3	31:0	**Global Depth Offset Clamp** Format: IEEE_FLOAT32 Specifies the clamp term used in the Global Depth Offset function.

3DSTATE_SAMPLE_MASK_BODY

<table>
<tr><th colspan="3">3DSTATE_SAMPLE_MASK_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:0</td><td>Sample Mask
Format: Enable[16] Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_MULTISAMPLE)
A per-multisample-position mask state variable that is immediately and unconditionally ANDed with the sample coverage mask as part of the rasterization process. This mask is applied prior to centroid selection. This mask must be ignored for centroid selection when RTIR is enabled i.e. Forced_Sample_Count > 0.
Programming Notes
If Number of Multisamples is NUMSAMPLES_1, bits 15:1 of this field will be zeroed by HW.
If Number of Multisamples is NUMSAMPLES_2, bits 15:2 of this field will be zeroed by HW.
If Number of Multisamples is NUMSAMPLES_4, bits 15:4 of this field will be zeroed by HW.
If Number of Multisamples is NUMSAMPLES_8, bits 15:8 of this field will be zeroed by HW.
When pixel shader writes to UAV but does not have actual render target write (i.e. no RT is bound to pixel shader, eventhough, RT write message is sent for EOT), appropriate SAMPLE_MASK must be all set depending on Number of Multisamples.</td></tr>
</table>

3DSTATE_SAMPLER_STATE_POINTERS_BODY

3DSTATE_SAMPLER_STATE_POINTERS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:5	**Pointer to Sampler State** Format: DynamicStateOffset[31:5]SAMPLER_STATE*16 Specifies the 32-byte aligned address offset of the function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.
	4:0	**Reserved** Format: MBZ

3DSTATE_SBE_BODY

<table>
<tr><th colspan="3">3DSTATE_SBE_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 160
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>30</td><td>Reserved</td></tr>
<tr><td></td><td>29</td><td>Force Vertex URB Entry Read Length
Format: Enable
This field provides a work around override for the computation of SBE_INT::Vertex URB Entry Read Length. If asserted, 3DSTATE_SBE::Vertex URB Entry Read Length is be used directly. Otherwise, SBE_INT::Vertex URB Entry Read Length is computed normally.</td></tr>
<tr><td></td><td>28</td><td>Force Vertex URB Entry Read Offset
Format: Enable
This field provides a work around override for the computation of SBE_INT::Vertex URB Entry Read Offset. If asserted, 3DSTATE_SBE::Vertex URB Entry Read Offset is be used directly. Otherwise, SBE_INT::Vertex URB Entry Read Offset is computed normally.</td></tr>
<tr><td></td><td>27:22</td><td>Number of SF Output Attributes
Format: U6 count of attributes
Specifies the number of vertex attributes passed from the SF stage to the WM stage (does not include Position).<table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,32]</td><td></td></tr></table></td></tr>
<tr><td></td><td>21</td><td>Attribute Swizzle Enable
Format: Enable
Enables the SF to perform swizzling on (up to the first 16) vertex attributes. If DISABLED, all vertex attributes are passed through.</td></tr>
<tr><td></td><td>20</td><td>Point Sprite Texture Coordinate Origin
This state controls how Point Sprite Texture Coordinates are generated (when enabled on a per-attribute basis by Point Sprite Texture Coordinate Enable).<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>UPPERLEFT</td><td>Top Left = (0,0,0,1)Bottom Left = (0,1,0,1)Bottom Right = (1,1,0,1)</td></tr><tr><td>1h</td><td>LOWERLEFT</td><td>Top Left = (0,1,0,1)Bottom Left = (0,0,0,1)Bottom Right = (1,0,0,1)</td></tr></table></td></tr>
</table>

<table>
<tr><th colspan="3">3DSTATE_SBE_BODY</th></tr>
<tr><td></td><td>19</td><td>Primitive ID Override Component W<table><tr><td>Format:</td><td>Enable</td></tr></table>If set, the W component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</td></tr>
<tr><td></td><td>18</td><td>Primitive ID Override Component Z<table><tr><td>Format:</td><td>Enable</td></tr></table>If set, the Z component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</td></tr>
<tr><td></td><td>17</td><td>Primitive ID Override Component Y<table><tr><td>Format:</td><td>Enable</td></tr></table>If set, the Y component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</td></tr>
<tr><td></td><td>16</td><td>Primitive ID Override Component X<table><tr><td>Format:</td><td>Enable</td></tr></table>If set, the X component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</td></tr>
<tr><td></td><td>15:11</td><td>Vertex URB Entry Read Length<table><tr><td>Format:</td><td>U5</td></tr></table>Specifies the amount of URB data read for each Vertex URB entry, in 256-bit register increments.<table><tr><th>Value</th><th>Name</th></tr><tr><td>[1,16]</td><td></td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>It is UNDEFINED to set this field to 0 indicating no Vertex URB data to be read.This field should be set to the minimum length required to read the maximum source attribute. The maximum source attribute is indicated by the maximum value of the enabled Attribute # Source Attribute if Attribute Swizzle Enable is set, Number of Output Attributes-1 if enable is not set.
read_length = ceiling((max_source_attr+1)/2)</td></tr></table></td></tr>
<tr><td></td><td>10:5</td><td>Vertex URB Entry Read Offset
Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB.</td></tr>
<tr><td></td><td>4:0</td><td>Primitive ID Override Attribute Select
Specifies which attribute is overridden w/ the Primitive ID<table><tr><th>Programming Notes</th></tr><tr><td>Set all Primitive ID Override Component Select X/Y/Z/W to 0 to indicate there is no Primitive ID override.</td></tr></table></td></tr>
<tr><td>1</td><td>31:0</td><td>Point Sprite Texture Coordinate Enable</td></tr>
</table>

3DSTATE_SBE_BODY		
		Format: Enable[32] When processing point primitives, the attributes from the incoming point vertex are typically copied to the point object corner vertices. However, if a bit is set in this field, the corresponding Attribute is selected as a Point Sprite Texture Coordinate, in which case each corner vertex is assigned a pre-defined texture coordinate as defined by the Point Sprite Texture Coordinate Origin state bit. Bit 0 corresponds to output Attribute 0.
2	31:0	**Constant Interpolation Enable** Format: Enable[32] This field is a bitmask containing a Constant Interpolation Enable bit for each corresponding attribute. If a bit is set, that attribute will undergo constant interpolation, and the corresponding WrapShortest Enable bits (if defined) will be ignored. If a bit is clear, components which are not enabled for WrapShortest interpolation (if defined) will be linearly interpolated.
3	31:30	**Attribute 15 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 15 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	29:28	**Attribute 14 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 14 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	27:26	**Attribute 13 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 13 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	25:24	**Attribute 12 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 12 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	23:22	**Attribute 11 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 11 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.

3DSTATE_SBE_BODY

	21:20	**Attribute 10 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 10 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	19:18	**Attribute 9 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 9 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	17:16	**Attribute 8 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 8 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	15:14	**Attribute 7 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 7 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	13:12	**Attribute 6 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 6 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	11:10	**Attribute 5 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 5 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	9:8	**Attribute 4 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 4 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.

3DSTATE_SBE_BODY

	7:6	**Attribute 3 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 3 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	5:4	**Attribute 2 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 2 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	3:2	**Attribute 1 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 1 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	1:0	**Attribute 0 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 0 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
4	31:30	**Attribute 31 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 31 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	29:28	**Attribute 30 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 30 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	27:26	**Attribute 29 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 29 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.

3DSTATE_SBE_BODY		
	25:24	**Attribute 28 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 28 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	23:22	**Attribute 27 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 27 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	21:20	**Attribute 26 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 26 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	19:18	**Attribute 25 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 25 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	17:16	**Attribute 24 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 24 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	15:14	**Attribute 23 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 23 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	13:12	**Attribute 22 Active Component Format** Format: Attribute_Component_Format This state indicates which components of Attribute 22 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.

3DSTATE_SBE_BODY		
	11:10	**Attribute 21 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 21 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	9:8	**Attribute 20 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 20 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	7:6	**Attribute 19 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 19 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	5:4	**Attribute 18 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 18 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	3:2	**Attribute 17 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 17 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.
	1:0	**Attribute 16 Active Component Format** Format: **Attribute_Component_Format** This state indicates which components of Attribute 16 are being used by the pixel shader Kernel. SBE will not perform attribute delta calculations for any disabled components. Operation is UNDEFINED if kernel uses attribute vertex delta for any disabled component.

3DSTATE_SBE_SWIZ_BODY

3DSTATE_SBE_SWIZ_BODY		
Source:	RenderCS	
Size (in bits):	320	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..7	255:0	**Attribute** Format: **SF_OUTPUT_ATTRIBUTE_DETAIL[16]**
8..9	63:60	**Attribute 15 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	59:56	**Attribute 14 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	55:52	**Attribute 13 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	51:48	**Attribute 12 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	47:44	**Attribute 11 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	43:40	**Attribute 10 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	39:36	**Attribute 09 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	35:32	**Attribute 08 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	31:28	**Attribute 07 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	27:24	**Attribute 06 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	23:20	**Attribute 05 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	19:16	**Attribute 04 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**
	15:12	**Attribute 03 Wrap Shortest Enables** Format: **WRAP_SHORTEST_ENABLE**

3DSTATE_SBE_SWIZ_BODY		
	11:8	**Attribute 02 Wrap Shortest Enables** Format: WRAP_SHORTEST_ENABLE
	7:4	**Attribute 01 Wrap Shortest Enables** Format: WRAP_SHORTEST_ENABLE
	3:0	**Attribute 00 Wrap Shortest Enables** Format: WRAP_SHORTEST_ENABLE

3DSTATE_SCISSOR_STATE_POINTERS_BODY

3DSTATE_SCISSOR_STATE_POINTERS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:5	**Scissor Rect Pointer** Format: DynamicStateOffset[31:5]SCISSOR_RECT*16 Specifies the 32-byte aligned address offset of the SCISSOR_RECT state. This offset is relative to the **Dynamic State Base Address**.
	4:0	**Reserved** Format: MBZ

3DSTATE_SF_BODY

3DSTATE_SF_BODY		
Source:	RenderCS	
Size (in bits):	96	
Default Value:	0x00000000, 0x00000000, 0x00000800	

DWord	Bit	Description
0	31:30	**Reserved** Format: MBZ
	29:12	**Line Width** Format: U11.7 Range: [0.0, 2047.9921875] Controls width of line primitives. Setting a Line Width of 0.0 specifies the rasterization of the "thinnest" (one-pixel-wide), non-antialiased lines. Note that this effectively overrides the effect of AAEnable (though the AAEnable state variable is not modified). **Programming Notes** Software must not program a value of 0.0 when running in MSRASTMODE_ON_xxx modes - zero-width lines are not available when multisampling rasterization is enabled.
	11	**Legacy Global Depth Bias Enable** Format: Enable Enables the SF to use the Global Depth Offset Constant state unmodified. If this bit is not set, the SF will scale the Global Depth Offset Constant as described in section Error! Reference source not found. of this document. **Programming Notes** This bit should be set whenever non zero depth bias (Slope, Bias) values are used. Setting this bit may have some degradation of performance for some workloads.
	10	**Statistics Enable** Format: Enable If ENABLED, this FF unit will increment CL_PRIMITIVES_COUNT on behalf of the CLIP stage. If DISABLED, CL_PRIMITIVES_COUNT will be left unchanged. **Programming Notes** This bit should be set whenever clipping is enabled and the Statistics Enable bit is set in CLIP_STATE. It should be cleared if clipping is disabled or Statistics Enable in CLIP_STATE is clear.
	9:2	**Reserved** Format: MBZ

<table>
<tr><th colspan="3">3DSTATE_SF_BODY</th></tr>
<tr><td></td><td>1</td><td>Viewport Transform Enable<table><tr><td>Format:</td><td>Enable</td></tr></table>This bit controls the Viewport Transform function.</td></tr>
<tr><td></td><td>0</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>1</td><td>31:29</td><td>Reserved<table><tr><td></td><td></td></tr></table></td></tr>
<tr><td></td><td>28</td><td>Reserved</td></tr>
<tr><td></td><td>27:18</td><td>Reserved<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>17:16</td><td>Line End Cap Antialiasing Region Width<table><tr><td>Format:</td><td>U2</td></tr></table>This field specifies the distances over which the coverage of anti-aliased line end caps are computed.<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>0.5 pixels</td><td>0.5 pixels</td></tr><tr><td>1h</td><td>1.0 pixels</td><td>1.0 pixels</td></tr><tr><td>2h</td><td>2.0 pixels</td><td>2.0 pixels</td></tr><tr><td>3h</td><td>4.0 pixels</td><td>4.0 pixels</td></tr></table></td></tr>
<tr><td></td><td>15</td><td>Reserved<table><tr><td></td><td></td></tr></table></td></tr>
<tr><td></td><td>14</td><td>Reserved<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>13</td><td>Reserved</td></tr>
<tr><td></td><td>12</td><td>Reserved</td></tr>
<tr><td></td><td>11:0</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>2</td><td>31</td><td>Last Pixel Enable<table><tr><td>Format:</td><td>Enable</td></tr></table>If ENABLED, the last pixel of a diamond line will be lit. This state will only affect the rasterization of Diamond lines (will not affect wide lines or anti-aliased lines).<table><tr><th>Programming Notes</th></tr><tr><td>Last pixel is applied to all lines of a LINELIST, and only the last line of a LINESTRIP.</td></tr></table></td></tr>
<tr><td></td><td>30:29</td><td>Triangle Strip/List Provoking Vertex Select</td></tr>
</table>

3DSTATE_SF_BODY

		Format: U2 Selects which vertex of a triangle (in a triangle strip or list primitive) is considered the "provoking vertex". Used for flat shading of primitives.Does current implementation send provoking vertex first?

Value	Name
0h	0
1h	1
2h	2
3h	Reserved

28:27 Line Strip/List Provoking Vertex Select

Format:	U2

Selects which vertex of a line (in a line strip or list primitive) is considered the "provoking vertex".

Value	Name	Description
0h	0	Vertex 0
1h	1	Vertex 1
2h	Reserved	Reserved
3h	Reserved	Reserved

26:25 Triangle Fan Provoking Vertex Select

Format:	U2

Selects which vertex of a triangle (in a triangle fan primitive) is considered the "provoking vertex".

Value	Name
0h	0
1h	1
2h	2
3h	Reserved

24:15 Reserved

Format:	MBZ

14 AA Line Distance Mode

Format:	U1

This bit controls the distance computation for antialiased lines.

Value	Name	Description
1h	AALINEDISTANCE_TRUE	True distance computation. This is the normal setting which should yield WHQL compliance.

13 Smooth Point Enable

Format:	Enable
Double Buffer Armed By:	Enables logic to draw smooth OGL Points

3DSTATE_SF_BODY

		Programming Notes If Enabled, SF will treat points in the same fashion that AA lines are processed
	12	**Vertex Sub Pixel Precision Select** Format: U1 Selects the number of fractional bits maintained in the vertex data

Value	Name	Description
0h	8	8 sub pixel precision bits maintained
1h	4	4 sub pixel precision bits maintained

Programming Notes
When Conservative Rasterization is enabled, this bit must be programmed to 0.

11 — Point Width Source

Controls whether the point width passed on the vertex or from state is used for rendering point primitives.

Value	Name	Description
0h	Vertex	Use Point Width on Vertex
1h	State **[Default]**	Use Point Width from State

10:0 — Point Width

Format:	U8.3

Range: [0.125, 255.875] pixels
This field specifies the size (width) of point primitives in pixels. This field is overridden (though not overwritten) whenever point width information is passed in the FVF

3DSTATE_STREAMOUT_BODY

3DSTATE_STREAMOUT_BODY		
Source:	RenderCS	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31	**SO Function Enable** Format: U1 If set, the SO function is enabled. Vertex data will be streamed out to memory (subject to overflow detection) as controlled by the various SO-related state variables. If clear, the SO function is disabled, and therefore no vertex data will be streamed out to memory. However, the Rendering Disable and Render Stream Select fields will still be used to determine which vertices (if any) are forwarded down the pipeline for (possible) rendering.
	30	**API Rendering Disable** Format: U1 If set, Indicates the API wants the SO stage not to forward any topologies down the pipeline. If clear, Indicates the API wants the SO stage to forward topologies associated with **Render Stream Select** down the pipeline. This bit is used even if **SO Function Enable** is DISABLED. **Programming Notes** The SOL unit generates an SOL_INT::Render_Enable which ultimately controls whether rendering occurs or not.
	29	**Reserved** Format: MBZ
	28:27	**Render Stream Select** Format: U2 **Description** This field specifies which stream has been selected to be forwarded down the pipeline for possible rendering. Topologies from other streams will not be passed down the pipeline. If Rendering Disable is set, this field is ignored, as no topologies are sent down the pipeline. SO Function Enable must also be ENABLED in order for this field to select a stream for rendering. When **SO Function Enable** is DISABLED and Rendering Disable is cleared (i.e., rendering is enabled), StreamID is ignored downstream of the SO stage, allowing any stream to be rendered.
	26	**Reorder Mode** This bit controls how vertices of triangle objects in TRISTRIP[_ADJ] and TRISTRIP_REV are reordered for the purposes of stream-out only (does not impact rendering). See table in Input Buffering. **Value** | **Name** | **Description**

3DSTATE_STREAMOUT_BODY

DWord	Bit	Description
		0h | LEADING | Reorder the vertices of alternating triangles of a TRISTRIP[_ADJ] such that the leading (first) vertices are in consecutive order starting at v0. A similar reordering is performed on alternating triangles in a TRISTRIP_REV. 1h | TRAILING | Reorder the vertices of alternating triangles of a TRISTRIP[_ADJ] such that the trailing (last) vertices are in consecutive order starting at v2. A similar reordering is performed on alternating triangles in a TRISTRIP_REV.
	25	**SO Statistics Enable** Format: Enable This bit controls whether StreamOutput statistics register(s) can be incremented. Value | Name | Description 0h | Disable | SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers cannot increment. 1h | Enable | SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers can increment.
	24:23	**Force Rendering** This field provides a work around override for the computation of SOL_INT::Render_Enable Value | Name | Description 0h | Normal | SOL_INT::Render_Enable is computed normally 1h | Resreved | 2h | Force_Off | Forces the rendering to be disabled. 3h | Force_on | Forces the rendering to be enabled.
	22:21	**Reserved** Format: MBZ
	20:12	**Reserved** Format: MBZ
	11:8	**Reserved** Format: MBZ
	7:0	**Reserved** Format: MBZ
1	31:30	**Reserved** Format: MBZ
	29	**Stream 3 Vertex Read Offset**

3DSTATE_STREAMOUT_BODY		
		Format: U1 count of 256-bit units Specifies amount of data to skip over before reading back Stream 3 vertex data. (See **Stream 0 Vertex Read Offset**)
	28:24	**Stream 3 Vertex Read Length** Format: U5-1 count of 256-bit units (See Stream 0 Vertex Read Length)
	23:22	**Reserved** Format: MBZ
	21	**Stream 2 Vertex Read Offset** Format: U1 count of 256-bit units Specifies amount of data to skip over before reading back Stream 2 vertex data. (See Stream 0 Vertex Read Offset)
	20:16	**Stream 2 Vertex Read Length** Format: U5-1 count of 256-bit units
	15:14	**Reserved** Format: MBZ
	13	**Stream 1 Vertex Read Offset** Format: U1 count of 256-bit units Specifies amount of data to skip over before reading back Stream 1 vertex data. (See Stream 0 Vertex Read Offset)
	12:8	**Stream 1 Vertex Read Length** Format: U5-1 count of 256-bit units (See Stream 0 Vertex Read Length)
	7:6	**Reserved** Format: MBZ
	5	**Stream 0 Vertex Read Offset**

3DSTATE_STREAMOUT_BODY		
		Format: U1 count of 256-bit units Specifies amount of data to skip over before reading back Stream 0 vertex data. Must be zero if the GS is enabled and the Output Vertex Size field in 3DSTATE_GS is programmed to 0 (i.e., one 16B unit).
	4:0	**Stream 0 Vertex Read Length** Format: U5-1 count of 256-bit units Specifies amount of vertex data to read back for Stream 0 vertices, starting at the Stream 0 Vertex Read Offset location. Maximum readback is 17 256-bit units (34 128-bit vertex attributes). Read data past the end of the valid vertex data has undefined contents, and therefore shouldn't be used to source stream out data. Must be zero (i.e., read length = 256b) if the GS is enabled and the Output Vertex Size field in 3DSTATE_GS is programmed to 0 (i.e., one 16B unit).
2	31:28	**Reserved** Format: MBZ
	27:16	**Buffer 1 Surface Pitch**
	15:12	**Reserved** Format: MBZ
	11:0	**Buffer 0 Surface Pitch** Format: U12 pitch in Bytes This field specifies the pitch of the SO buffer in #Bytes. Value: [0,2048] — Name: Must be 0 or a multiple of 4 Bytes. **Programming Notes** A Surface Pitch of 0 indicates an un-bound buffer. No writes are performed. Surface Base Address is ignored.
3	31:28	**Reserved** Format: MBZ
	27:16	**Buffer 3 Surface Pitch** Format: U12

3DSTATE_STREAMOUT_BODY		
	15:12	**Reserved** Format: MBZ
	11:0	**Buffer 2 Surface Pitch** Format: U12

3DSTATE_TE_BODY

3DSTATE_TE_BODY

Source:	RenderCS
Size (in bits):	96
Default Value:	0x00000000, 0x427C0000, 0x42800000

DWord	Bit	Description
0	31:24	**Reserved** Format: MBZ
	23:22	**Reserved** Format: MBZ
	21	**Reserved** Format: MBZ
	20	**Reserved** Format: MBZ
	19	**Reserved** Format: MBZ
	18:17	**Reserved** Format: MBZ
	16	**Reserved** Format: MBZ
	15:14	**Reserved** Format: MBZ
	13:12	**Partitioning** Format: U2 This field specifies how edges are partitioned based on tessellation factor. **Value** / **Name** / **Description** 0h / INTEGER / Outside/inside edges are divided into an integer number of equal-sized segments. 1h / ODD_FRACTIONAL / Outside/inside edges are divided into an odd number of possibly-unequal-sized segments. 2h / EVEN_FRACTIONAL / Outside/inside edges are divided into an even number of possibly-unequal-sized segments.
	11:10	**Reserved** Format: MBZ
	9:8	**Output Topology** Format: U2 This field specifies which primitive types are to be output.

3DSTATE_TE_BODY

Value	Name	Description
0h	POINT	Points are output (as POINTLIST topologies)
1h	LINE	Lines are output (as LINESTRIP topologies). Only valid if ISOLINE domain is selected.
2h	TRI_CW	Clockwise-ordered triangles are output (either as TRISTRIP, TRISTRIP_REV or TRILIST topologies). Not valid if ISOLINE domain is selected.
3h	TRI_CCW	Count-clockwise-ordered triangles are output (either as TRISTRIP, TRISTRIP_REV or TRILIST topologies). Not valid if ISOLINE domain is selected.

7:6 **Reserved**

Format:	MBZ

5:4 **TE Domain**

Format:	U2

This field specifies which type of domain is to be tessellated.

Value	Name	Description
0h	QUAD	2D (U, V) domain is tessellated
1h	TRI	Triangular (U, V, W) domain is tessellated
2h	ISOLINE	2D (U, V) domain is tessellated.

3 **Reserved**

Format:	MBZ

2:1 **TE Mode**

Format:	U2

When TE Enable is ENABLED, this field specifies the overall operation of the TE stage.This field is ignored if TE Enable is DISABLED.

Value	Name	Description
0h	HW_TESS	Normal HW Tessellation Mode. The TessFactors are read from the patch URB entry, and are used to perform fixed-function hardware tessellation of the specified domain.

0 **TE Enable**

Format:	Enable

If ENABLED, the TE stage will perform tessellation processing on incoming patch primitives. The TE Mode field determines how this tessellation operation proceeds.If DISABLED, the TE goes into pass-through mode. All other state fields are ignored.

Programming Notes

3DSTATE_TE_BODY

		The tessellation stages (HS, TE and DS) must be enabled/disabled as a group. I.e., draw commands can only be issued if all three stages are enabled or all three stages are disabled, otherwise the behavior is UNDEFINED.
1	31:0	**Maximum Tessellation Factor Odd**

Format:	IEEE_FLOAT32

This field specifies the maximum TessFactor for ODD_FRACTIONAL partitioning when in HW_TESS mode.

Value	Name	Description
[427c0000h,427c0000h]	63 **[Default]**	Per API Spec, For normal operation software should set this value to 63.0

Programming Notes
Note that ISOLINE's LineDensity TF is always subjected to INTEGER partitioning regardless of the Partitioning state.

2	31:0	**Maximum Tessellation Factor Not Odd**

Format:	IEEE_FLOAT32

This field specifies the maximum TessFactor for EVEN_FRACTIONAL, INTEGER or POW2 partitioning when in HW_TESS mode.

Value	Name	Description
[42800000h,42800000h]	64 **[Default]**	Per API Spec, For normal operation software should set this value to 64.0

Programming Notes
Note that ISOLINE's LineDensity TF is always subjected to INTEGER partitioning regardless of the Partitioning state.
If Partioning is set to POW2, this field must be programmed to a power of 2 number.

intel

3DSTATE_URB_DS_BODY

3DSTATE_URB_DS_BODY		
Source: RenderCS Size (in bits): 32 Default Value: 0x00000000		
DWord	**Bit**	**Description**
0	31:25	**DS URB Starting Address** Format: U7 Offset from the start of the URB memory where DS starts its allocation, specified in multiples of 8 KB. Value: [0,127] — Name: (none) **Programming Notes** If CTXT_SR_CTL::POSH_Enable is set and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 8. If CTXT_SR_CTL::POSH_Enable is clear and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 4. If Push Constants are not required andDevice[SliceCount] == 1, the lower limit is 0.
	24:16	**DS URB Entry Allocation Size** Format: U9-1 Count of 512-bit units Specifies the length of each URB entry owned by DS. This field is always used (even if DS Function Enable is DISABLED). Value: [0,9] — Name: (none)
	15:0	**DS Number of URB Entries** **Description** Specifies the number of URB entries that are used by DS, based on only 1 slice enabled. When multiple slices are enabled, HW will multiply the value programmed by the number of slices in order to determine the total number of entries. SW shall ensure that the total number of entries does not exceed the relevant ValidValue range listed below. This field is always used (even if DS Function Enable is DISABLED). If Domain Shader Thread Dispatch is Enabled then the minimum number of handles that must be allocated is 34 URB entries. **Value** | **Name**

3DSTATE_URB_DS_BODY

		[0,2384]		RenderCS

Programming Notes

DS Number of URB Entries must be divisible by 8 if the DS URB Entry Allocation Size is programmed to a value less than 9, which is 10 512-bit URB entries. "2:0" = reserved "000"

3DSTATE_URB_GS_BODY

3DSTATE_URB_GS_BODY

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

DWord	Bit	Description
0	31:25	**GS URB Starting Address** Format: U7 Offset from the start of the URB memory where GS starts its allocation, specified in multiples of 8 KB. Value: [0,127] — Name: (none) **Programming Notes** If CTXT_SR_CTL::POSH_Enable is set and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 8. If CTXT_SR_CTL::POSH_Enable is clear and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 4. If Push Constants are not required andDevice[SliceCount] == 1, the lower limit is 0.
	24:16	**GS URB Entry Allocation Size** Format: U9-1 512-bit units Specifies the length of each URB entry owned by GS. This field is always used (even if GS Function Enable is DISABLED).
	15:0	**GS Number of URB Entries** Specifies the number of URB entries that are used by GS, based on only 1 slice enabled. When multiple slices are enabled, HW will multiply the value programmed by the number of slices in order to determine the total number of entries. SW shall ensure that the total number of entries does not exceed the relevant ValidValue range listed below. This field is always used (even if GS Function Enable is DISABLED). Value: [0,1032] — Name: (none) — RenderCS **Programming Notes** Only if GS is disabled can this field be programmed to 0. If GS is enabled this field shall be programmed to a value greater than 0. For GS Dispatch Mode "Single", this field shall be programmed to a value greater than or equal to 1. For other GS Dispatch Modes, refer to the definition of Dispatch Mode (3DSTATE_GS) for minimum values of this field. GS Number of URB Entries must be divisible by 8 if the GS URB Entry Allocation Size is less than 9 512-bit URB entries. "2:0" = reserved "000" When 3DSTATE_GS:Enable is true, the minimum number of GS Number of URB Entries must be set to 2.

3DSTATE_URB_HS_BODY

3DSTATE_URB_HS_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:25	**HS URB Starting Address** Format: U7 Offset from the start of the URB memory where HS starts its allocation, specified in multiples of 8 KB. Value: [0,127] — Name: (none) **Programming Notes** If CTXT_SR_CTL::POSH_Enable is set and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 8. If CTXT_SR_CTL::POSH_Enable is clear and Push Constants are required orDevice[SliceCount] GT 1, the lower limit is 4. If Push Constants are not required andDevice[SliceCount] == 1, the lower limit is 0.
	24:16	**HS URB Entry Allocation Size** Format: U9-1 Count of 512-bit units Specifies the length of each URB entry owned by HS. This field is always used (even if HS Function Enable is DISABLED).
	15:0	**HS Number of URB Entries** Specifies the number of URB entries that are used by HS, based on only 1 slice enabled. When multiple slices are enabled, HW will multiply the value programmed by the number of slices in order to determine the total number of entries. SW shall ensure that the total number of entries does not exceed the relevant ValidValue range listed below. This field is always used (even if HS Function Enable is DISABLED). Programming Restriction:HS Number of URB Entries must be divisible by 8 if the HS URB Entry Allocation Size is less than 9 512-bit URB entries."2:0" = reserved "000" Value: [0,1032] — Name: (none) — RenderCS **Programming Notes** When 3DSTATE_HS:Enable is true and 3DSTATE_HS:Dispatch Mode is set to 8 patch, the minimum number of HS Number of URB Entries must be set to 16. When 3DSTATE_HS:Enable is true and 3DSTATE_HS:Dispatch Mode is set to dual patch, the minimum number of HS Number of URB Entries must be set to 4.

3DSTATE_URB_VS_BODY

<table>
<tr><th colspan="3">3DSTATE_URB_VS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:25</td><td>VS URB Starting Address
Format: U7
Offset from the start of the URB memory where VS starts its allocation, specified in multiples of 8 KB.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,127]</td><td></td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>If CTXT_SR_CTL::POSH_Enable is set and Push Constants are required or Device[SliceCount] GT 1, the lower limit is 8.
If CTXT_SR_CTL::POSH_Enable is clear and Push Constants are required or Device[SliceCount] GT 1, the lower limit is 4.
If Push Constants are not required and Device[SliceCount] == 1, the lower limit is 0.</td></tr></table></td></tr>
<tr><td></td><td>24:16</td><td>VS URB Entry Allocation Size
Format: U9-1 count of 512-bit units
Specifies the length of each URB entry owned by VS. This field is always used (even if VS Function Enable is DISABLED).
<table><tr><th>Programming Notes</th></tr><tr><td>Programming Restriction: As the VS URB entry serves as both the per-vertex input and output of the VS shader, the VS URB Allocation Size must be sized to the maximum of the vertex input and output structures.</td></tr></table></td></tr>
<tr><td></td><td>15:0</td><td>VS Number of URB Entries
Format: U16
Specifies the number of URB entries that are used by VS, based on only 1 slice enabled. When multiple slices are enabled, HW will multiply the value programmed by the number of slices in order to determine the total number of entries. SW shall ensure that the total number of entries does not exceed the relevant ValidValue range listed below.
This field is always used (even if VS Function Enable is DISABLED).
<table><tr><th>Value</th><th>Name</th><th></th></tr><tr><td>[64,2384]</td><td></td><td>RenderCS</td></tr><tr><td>[64,1024]</td><td></td><td>PositionCS</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>Programming Restriction: VS Number of URB Entries must be divisible by 8 if the VS URB Entry</td></tr></table></td></tr>
</table>

3DSTATE_URB_VS_BODY		
		Allocation Size is less than 9 512-bit URB entries."2:0" = reserved "000b"

3DSTATE_VF_BODY

3DSTATE_VF_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:0	**Cut Index** This field specifies the index value that is considered the "cut index" which vertex indices are compared to if a Cut Index Enable is set. The Cut Index is compared to the fetched (and possibly-sign-extended) vertex index, and if these values are equal, the current primitive topology is terminated. Note that, for index buffers less than 32bpp, it is possible to set the Cut Index to a (large) value that will never match a sign-extended vertex index.

3DSTATE_VF_COMPONENT_PACKING_BODY

3DSTATE_VF_COMPONENT_PACKING_BODY		
Source:	RenderCS	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0..3	127:0	**Vertex Elements Enables** Format: COMPONENT_ENABLES[32]

3DSTATE_VF_INSTANCING_BODY

3DSTATE_VF_INSTANCING_BODY		
Source:	RenderCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0	31:9	**Reserved** Format: MBZ
	8	**Instancing Enable** Format: Enable Value: 0h; Name: Disabled; Description: This vertex element is not instanced and therefore vertices within instances can each receive different data for this vertex element. Within each instance, the source vertex data for this vertex element is determined according the Vertex Access Type of the 3DPRIMITIVE command. There is no Instance Data Step Rate state defined for this vertex element. Value: 1h; Name: Enabled; Description: This vertex element is instanced and therefore vertices within instances will receive the same data for this vertex element. The source pointer for this particular vertex element will be (a) initialized at the start of 3DPRIMITIVE processing, (b) held constant for all vertices within an instance, and (c) advanced between instances as a function of Instance Data Step Rate.
	7:6	**Reserved** Format: MBZ
	5:0	**Vertex Element Index** Format: U6 This field identifies which vertex element state is to be modified by this command. Value: [0,33]; Name:
1	31:0	**Instance Data Step Rate** If Instancing Enable is ENABLED, this field determines the rate at which data for this particular vertex element is changed between instances. Only after the number of instances specified by this field is generated is new (sequential) vertex element data provided. This process continues for each group of instances defined in the 3DPRIMTIVE command. For example, a value of 1 in this field causes new data to be supplied for this vertex element with each sequential (instance) group of vertices. A value of 2 causes every other instance group of vertices to be provided with new vertex element data. The special value of 0 causes all vertices of all instances generated by the 3DPRIMITIVE command to be provided with the same data for this vertex element. (The same effect can be achieved by setting this field to its maximum value.) If Instancing Enable is DISABLED, this field is ignored.

3DSTATE_VF_SGVS_2_BODY

3DSTATE_VF_SGVS_2_BODY		
Source:	RenderCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0	31	**XP1 Enable**

Format:	Boolean

Value	Name	Description
0h	Disabled	XP1 is not inserted
1h	Enabled	XP1 (as defined by XP1 Source Select) is inserted.

DWord	Bit	Description
	30:29	**XP1 Component Number** If XP1 Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is inserted. If XP1 Enable is DISABLED, this field is ignored.

Value	Name	Description
0	COMP_0	If enabled, XP1 is inserted in component 0 (.x)
1	COMP_1	If enabled, XP1 is inserted in component 1 (.y)
2	COMP_2	If enabled, XP1 is inserted in component 2 (.z)
3	COMP_3	If enabled, XP1 is inserted in component 3 (.w)

DWord	Bit	Description
	28	**XP1 Source Select** If XP1 Enable is ENABLED, this field selects between the available sources for the XP1 SGV to be inserted. If XP1 Enable is DISABLED, this field is ignored.

Value	Name	Description	Programming Notes
1h	Start Instance Location	The XP1 value is sourced from the Start Instance Location Parameter.	Start Instance Location is the only valid value if 3DSTATE_VF::InstanceIDOffsetEnable is set.
0h	XP1_PARAMETER	The XP1 value is sourced from the XP1 parameter as defined by 3DPRIMITIVE.	

DWord	Bit	Description
	27:22	**Reserved**

Format:	MBZ

DWord	Bit	Description
	21:16	**XP1 Element Offset**

3DSTATE_VF_SGVS_2_BODY

Format:	U6 Offset of 128-bit element

If XP1 Enable is ENABLED, this field specifies the VUE element offset of the 128-bit element where it is to be inserted. The XP1 Component Number specifies where in the specified element it is to be inserted.

Value	Name
[0,33]	

15 — XP0 Enable

Format:	Boolean

Value	Name	Description
0h	Disabled	XP0 is not inserted
1h	Enabled	XP0 (as defined by XP0 Source Select) is inserted

14:13 — XP0 Component Number

If XP0 Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is inserted. If XP0 Enable is DISABLED, this field is ignored.

Value	Name	Description
0	COMP_0	If enabled, XP0 is inserted in component 0 (.x)
1	COMP_1	If enabled, XP0 is inserted in component 1 (.y)
2	COMP_2	If enabled, XP0 is inserted in component 2 (.z)
3	COMP_3	If enabled, XP0 is inserted in component 3 (.w)

12 — XP0 Source Select

If XP0 Enable is ENABLED, this field selects between the available sources for the XP0 SGV to be inserted.
If XP0 Enable is DISABLED, this field is ignored.

Value	Name	Description
1h	VERTEX_LOCATION	The XP0 value is sourced from one of the two Vertex Location parameters passed in 3DPRIMITIVE. If Vertex Access Mode is SEQUENTIAL, the Start Vertex Location value is used. If Vertex Access Mode is RANDOM, the Base Vertex Location value is used.
0h	XP0_PARAMETER	The XP0 value is sourced from the XP0 parameter as defined by 3DPRIMITIVE.

11:6 — Reserved

Format:	MBZ

5:0 — XP0 Element Offset

Format:	U6 Offset of 128-bit element

If XP0 Enable is ENABLED, this field specifies the VUE element offset of the 128-bit element where it is to be inserted. The XP0 Component Number specifies where in the specified element it is to be inserted. If XP0 Enable is DISABLED, this field is ignored.

3DSTATE_VF_SGVS_2_BODY		
		<table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,33]</td><td></td></tr></table>
1	31:16	**Reserved** <table><tr><td>Format:</td><td>MBZ</td></tr></table>
	15	**XP2 Enable** <table><tr><td>Format:</td><td>Boolean</td></tr></table> <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>Disabled</td><td>XP2 is not inserted</td></tr><tr><td>1h</td><td>Enabled</td><td>XP2 is inserted, sourced from the XP2 parameter as defined by 3DPRIMITIVE.</td></tr></table>
	14:13	**XP2 Component Number** If XP2 Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is to be inserted. If XP2 Enable is DISABLED, this field is ignored. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>COMP_0</td><td>If enabled, XP2 is inserted in component 0 (.x)</td></tr><tr><td>1</td><td>COMP_1</td><td>If enabled, XP2 is inserted in component 1 (.y)</td></tr><tr><td>2</td><td>COMP_2</td><td>If enabled, XP2 is inserted in component 2 (.z)</td></tr><tr><td>3</td><td>COMP_3</td><td>If enabled, XP2 is inserted in component 3 (.w)</td></tr></table>
	12:6	**Reserved** <table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
	5:0	**XP2 Element Offset** <table><tr><td>Format:</td><td>U6 Offset of 128-bit element</td></tr></table> If XP2 Enable is ENABLED, this field specifies the VUE element offset of the 128-bit element where it is to be inserted. The XP2 Component Number specifies where in the specified element it is to be inserted. If XP2 Enable is DISABLED, this field is ignored. <table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,33]</td><td></td></tr></table>

3DSTATE_VF_SGVS_BODY

3DSTATE_VF_SGVS_BODY

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

DWord	Bit	Description
0	31	**InstanceID Enable** Format: Enable Value | Name | Description 0h | Disabled | InstanceID is not inserted 1h | Enabled | InstanceID is inserted
	30:29	**InstanceID Component Number** If InstanceID Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is inserted. If InstanceID Enable is DISABLED, this field is ignored. Value | Name | Description 0 | COMP_0 | If enabled, InstanceID is inserted in component 0 (.x) 1 | COMP_1 | If enabled, InstanceID is inserted in component 1 (.y) 2 | COMP_2 | If enabled, InstanceID is inserted in component 2 (.z) 3 | COMP_3 | If enabled, InstanceID is inserted in component 3 (.w)
	28:22	**Reserved** Format: MBZ
	21:16	**InstanceID Element Offset** Format: U6 Offset of 128-bit element If InstanceID Enable is ENABLED, this field specifies the VUE element offset of the 128-bit element where it is to be inserted. The InstanceID Component Number specifies where in the specified element it is inserted. Value | Name [0,33] |
	15	**VertexID Enable** Format: Enable

3DSTATE_VF_SGVS_BODY

Value	Name	Description
0h	Disabled	VertexID is not inserted
1h	Enabled	VertexID is inserted

14:13 **VertexID Component Number**

If VertexID Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is inserted.If VertexID Enable is DISABLED, this field is ignored.

Value	Name	Description
0	COMP_0	If enabled, VertexID is inserted in component 0 (.x)
1	COMP_1	If enabled, VertexID is inserted in component 1 (.y)
2	COMP_2	If enabled, VertexID is inserted in component 2 (.z)
3	COMP_3	If enabled, VertexID is inserted in component 3 (.w)

12:6 **Reserved**

Format:	MBZ

5:0 **VertexID Element Offset**

Format:	U6 Offset of 128-bit element

If VertexID Enable is ENABLED, this field specifies the VUE element offset of the 128-bit element where it is to be inserted. The VertexID Component Number specifies where in the specified element it is inserted. This is also the vertex element index. If VertexID Enable is DISABLED, this field is ignored.

Value	Name
[0,33]	

3DSTATE_VF_TOPOLOGY_BODY

3DSTATE_VF_TOPOLOGY_BODY

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

DWord	Bit	Description
0	31:6	**Reserved** Format: MBZ
	5:0	**Primitive Topology Type** Format: 3D_Prim_Topo_Type This field specifies the VF stage's Topology state.

3DSTATE_VIEWPORT_STATE_POINTERS_CC_BODY

3DSTATE_VIEWPORT_STATE_POINTERS_CC_BODY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:5	**CC Viewport Pointer** Format: DynamicStateOffset[31:5]CC_VIEWPORT*16 Specifies the 32-byte aligned address offset of the CC_VIEWPORT state. This offset is relative to the Dynamic State Base Address.
	4:0	**Reserved** Format: MBZ

3DSTATE_VIEWPORT_STATE_POINTERS_SF_CLIP_BODY

3DSTATE_VIEWPORT_STATE_POINTERS_SF_CLIP_BODY	
Source:	RenderCS
Size (in bits):	32
Default Value:	0x00000000

DWord	Bit	Description
0	31:6	**SF Clip Viewport Pointer** Format: DynamicStateOffset[31:6]SF_CLIP_VIEWPORT*16 Specifies the 64-byte aligned address offset of the SF_CLIP_VIEWPORT state. This offset is relative to the Dynamic State Base Address.
	5:0	**Reserved** Format: MBZ

3DSTATE_VS_BODY

<table>
<tr><th colspan="3">3DSTATE_VS_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0..1</td><td>63:6</td><td>Kernel Start Pointer
Format: InstructionBaseOffset[63:6]
This field specifies the starting location of the kernel program run by threads spawned by the VS pipeline stage. It is specified as a 64-byte-granular offset from the Instruction Base Address. This field is ignored if VS Function Enable is DISABLED.</td></tr>
<tr><td></td><td>5:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>2</td><td>31</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>30</td><td>Vector Mask Enable
Format: Enable
Upon subsequent VS thread dispatches, this bit is loaded into the EU’s Vector Mask Enable (VME, cr0.0[3]) thread state. Refer to EU documentation for the definition and use of VME state.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>Dmask</td><td>The EU will use the Dispatch Mask (supplied by the VS stage) for instruction execution.</td></tr>
<tr><td>1h</td><td>Vmask</td><td>The EU will use the Vector Mask (derived from the Dispatch Mask) for instruction execution.</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>Under normal conditions SW shall specify DMask, as the VS stage will provide a Dispatch Mask appropriate to SIMD4x2 or SIMD8 thread execution (as a function of SIMD8 Dispatch Enable). E.g., for SIMD4x2 thread execution, the VS stage will generate a Dispatch Mask that is equal to what the EU would use as the Vector Mask. For SIMD8 execution there is no known usage model for use of Vector Mask (as there is for PS shaders).</td></tr>
</table></td></tr>
<tr><td></td><td>29:27</td><td>Sampler Count</td></tr>
</table>

3DSTATE_VS_BODY

		Format: U3 This field specifies (in multiples of 4) the number of sets of sampler state that will be prefetched for use by the VS kernel. While the prefetching of sampler state is optional and does not impact functionality, it may improve performance. This field is ignored if the Function Enable state is set to DISABLED.

Value	Name	Description
0h	No Samplers	no samplers used
1h	1-4 Samplers	between 1 and 4 samplers used
2h	5-8 Samplers	between 5 and 8 samplers used
3h	9-12 Samplers	between 9 and 12 samplers used
4h	13-16 Samplers	between 13 and 16 samplers used

26 **Reserved**

Format:	MBZ

25:18 **Binding Table Entry Count**

Format:	U8

Description
Specifies how many binding table entries the kernel uses. Used only for prefetching of the binding table entries and associated surface state. Note: 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 if VS Function Enable is DISABLED.
When HW Generated Binding Table bit is enabled: This field indicates which cache lines (512bit units - 32 Binding Table Entry section) should be fetched. Each bit in this field corresponds to a cache line. Only the 1st 4 non-zero Binding Table entries of each 32 Binding Table entry section prefetched will have its surface state prefetched.

Value	Name
[0,255]	

Programming Notes
When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.

17 **Thread Dispatch Priority**

Format:	U1 Enumerated Type

Specifies the priority of the thread for dispatch: This field is ignored if VS Function Enable is DISABLED.

3DSTATE_VS_BODY

		Value / **Name** / **Description**

Value	Name	Description
0h	Normal	Normal Priority
1h	High	High Priority

16 — Floating Point Mode

Format:	U1 Enumerated Type

Specifies the initial floating point mode used by the dispatched thread. This field is ignored if VS Function Enable is DISABLED.

Value	Name	Description
0h	IEEE-754	Use IEEE-754 Rules
1h	Alternate	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. This field is ignored if VS Function Enable is DISABLED.

12 — Accesses UAV

Format:	Enable

This field must be set when VS has a UAV access.

Programming Notes
This field must not be set when VS Function Enable is disabled.
This bit shall not be set when the command is executed in the PCS pipeline.

11:8 — 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. This field is ignored if VS Function Enable is DISABLED.

6:0 — Reserved

Format:	MBZ

3DSTATE_VS_BODY

DWord	Bit	Description
3..4	63:32	**Reserved** Format: MBZ
	31:10	**Scratch Space Base Pointer** Format: GeneralStateOffset[31:10] Specifies the starting location of the scratch space area allocated to this FF unit as a 1K-byte aligned offset from the General State Base Address. If required, each thread spawned by this FF unit will be allocated some portion of this space, as specified by Per-Thread Scratch Space. The computed offset of the thread-specific portion will be passed in the thread payload as Scratch Space Offset. The thread is expected to utilize "stateless" DataPort read/write requests to access scratch space, where the DataPort will cause the General State Base Address to be added to the offset passed in the request header. This field is ignored if VS Function Enable is DISABLED. In 64b OS all pointers need to be seen by SW as 48b. HW does not support a Scratch Space Base Pointer larger than 32b, therefore SW must ensure Bits<63:32> are set to 0's. **Programming Notes** The scratch spaces allocated to the POCS VSR stage and RCS VS stage shall not overlap with each other or the scratch space allocations of any other enabled stage in the RCS pipeline.
	9:4	**Reserved** Format: MBZ
	3:0	**Per-Thread Scratch Space** Format: U4 power of 2 Bytes over 1K Bytes Specifies the amount of scratch space to be allocated to each thread spawned by this FF unit. The driver must allocate enough contiguous scratch space, starting at the Scratch Space Base Pointer, to ensure that the Maximum Number of Threads can each get Per-Thread Scratch Space size without exceeding the driver-allocated scratch space. This field is ignored if VS Function Enable is DISABLED. **Value** | **Name** | **Description** [0,11] | | Indicating [1K Bytes, 2M Bytes] **Programming Notes** This amount is available to the kernel for information only. It will be passed verbatim (if not altered by the kernel) to the Data Port in any scratch space access messages, but the Data Port will ignore it.
5	31:25	**Reserved** Format: MBZ
	24:20	**Dispatch GRF Start Register For URB Data**

3DSTATE_VS_BODY

		Format: U5 Specifies the starting GRF number for the URB portion (URB constants and vertices) of the thread payload. This field is ignored if VS Function Enable is DISABLED. **Value** | **Name** | **Description** [0,31] | | indicating GRF [R0, R31]
	19:17	**Reserved** Format: MBZ
	16:11	**Vertex URB Entry Read Length** Format: U6 Specifies the number of pairs of 128-bit vertex elements to be passed into the payload for each vertex. This field is ignored if VS Function Enable is DISABLED. For SIMD4x2 dispatch, each vertex element requires one GRF of payload data, therefore the number of GRFs with vertex data will be double the value programmed in this field. For SIMD8 dispatch, each vertex element requires 4 GRFs of payload data, therefore the number of GRFs with vertex data will be 8 times the value programmed in this field. The EU limit of 128 GRFs imposes a maximum limit of 30 elements per vertex pushed into the payload, though the practical limit may be lower. If input vertices exceed the practical limit, software must decide between resorting to pulling elements during thread execution or dropping back to SIMD4x2 dispatch. Note that the VUE is used for both input and output, so when using the pull-model software must ensure inputs are not overwritten before last use. **Value** | **Name** | **Description** [1,63] | | if SIMD8 dispatch disabled [0,15] | | if SIMD8 dispatch enabled
	10	**Reserved** Format: MBZ
	9:4	**Vertex URB Entry Read Offset** Format: U6 Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB before being included in the thread payload. This offset applies to all Vertex URB entries passed to the thread. This field is ignored if VS Function Enable is DISABLED. **Value** | **Name** [0,63] |
	3:0	**Reserved** Format: MBZ
6	31:22	**Maximum Number of Threads**

3DSTATE_VS_BODY		
		Format: U10-1 Thread count Specifies the maximum number of simultaneous threads allowed to be active. Used to avoid using up the scratch space. Programming the value of the max threads over the number of threads based off number of threads supported in the execution units may improve performance since the architecture allows threads to be buffered between the check for max threads and the actual dispatch into the EU. Programming the max values to a number less than the number of threads supported in the execution units may reduce performance. This field is ignored if VS Function Enable is DISABLED.

Value	Name	Description	
[0,363]		indicating thread count of [1,364]	
[0,191]		indicating thread count of [1,192]	PositionCS

3DSTATE_VS_BODY		
	21:13	**Reserved** Format: MBZ
	12:11	**Reserved** Format: MBZ
	10	**Statistics Enable** Format: Enable If ENABLED, the VS stage will perform statistics gathering. See the Statistics Gathering subsection. If DISABLED, statistics information associated with the VS stage will be left unchanged. **Programming Notes** When a 3DPRIMITIVE command with POSH Enable set is executed from the RCS command stream, VS statistics gathering is inhibited for that command.
	9	**SIMD8 Single Instance Dispatch Enable** Format: Enable This field is used to specify whether vertices from different instances can be combined in a single SIMD8 dispatch. This bit is <u>ignored</u> if SIMD4x2 dispatches are enabled (i.e., SIMD8 Dispatch Enable is DISABLED). If ENABLED, SIMD8 VS thread dispatches <u>will not</u> combine vertices from different instances. This allows the VS kernel to handle instance-specific operations (e.g., read constants indexed by the InstanceID) in a global fashion, as these operations pertain to all vertices of the dispatch. If DISABLED, SIMD8 VS thread dispatches can combine vertices from different instances. The VS kernel must determine if instance-specific operations can be handled globally (vs. per-vertex). E.g., it can examine the Single Instance payload bit. **Programming Notes**

3DSTATE_VS_BODY

		SIMD8 Single Instance Dispatch Enable is not supported for HPCXTs.
	8:3	**Reserved** Format: MBZ
	2	**SIMD8 Dispatch Enable** Format: Enable This field determines how VS threads are dispatched and how the thread payloads are generated. The setting of this field must agree with how the VS kernel was compiled. If ENABLED, SIMD8 VS thread dispatches are performed. The **Single Vertex Dispatch** field is ignored. If DISABLED, SIMD4x2 thread dispatches are performed. The **Single Vertex Dispatch** field can be used to force single-vertex dispatches. **Programming Notes** The only supported mode is SIMD8 Dispatch Enable set to Enable (1).
	1	**Vertex Cache Disable** Format: Disable This bit controls the operation of the Vertex Cache. This field is always used. If the Vertex Cache is DISABLED and the VS Function is ENABLED, the Vertex Cache is not used and all incoming vertices will be passed to VS threads. If the Vertex Cache is ENABLED and the VS Function is ENABLED, only incoming vertices that do not hit in the Vertex Cache will be passed to VS threads. If the Vertex Cache is ENABLED and the VS Function is DISABLED, input vertices that miss in the Vertex Cache will be assembled and written to the URB (by the VF stage), and subsequently passed through the VS stage unmodified (i.e, no VS threads are spawned). The Vertex Cache is invalidated whenever the Vertex Cache becomes DISABLED, whenever the VS Function Enable toggles, between 3DPRIMITIVE commands and between instances within a 3DPRIMITIVE command.
	0	**Function Enable** Format: Enable This bit determines whether or not the VS stage spawns VS threads, which comprises the bulk of the VS stage functionality. If ENABLED, VS threads may be spawned to process VF-generated vertices before the resulting vertices are passed down the pipeline. If DISABLED, VF-generated vertices will pass thru the VS function and are sent down the pipeline unmodified. The Vertex Cache (if enabled) is still available.
7	31:27	**Reserved**

3DSTATE_VS_BODY

		Format: MBZ
	26:21	**Vertex URB Entry Output Read Offset** Format: U6 Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB by the Setup Back-End (SBE) function. The offset programmed will specify the start of Attribute 0 to be passed in subsequent Pixel Shader thread payloads. Refer to the Attribute Interpolator Setup documentation. **Value** | **Name** [0,63] | **Programming Notes** As the vertex header data located at the start of the Vertex URB entry is typically only used by 3D pipeline FFs (i.e., Clipper, Setup FrontEnd) and not required as interpolated attributes in PIxel Shader threads, it is expected that SW will program this Start Offset skip over the vertex header. This offset value is ignored if SBE's Number of SF Attributes state is programmed to 0 (i.e., no attributes are defined beyond the position read from the Vertex Header)
	20:16	**Vertex URB Entry Output Length** Format: U5 Specifies the amount of Vertex Attribute URB data to be read by the Setup Back-End function for each Vertex URB entry, in 256-bit units. The attribute data will be read starting at the offset specified by the Vertex URB Entry Output Read Offset state. **Value** | **Name** [1,16] | **Programming Notes** This length value is ignored if SBE's Number of SF Attributes state is programmed to 0 (i.e., no attributes are defined beyond the position read from the Vertex Header).
	15:8	**User Clip Distance Clip Test Enable Bitmask** Format: U8 This 8 bit mask field selects which of the 8 Clip Distance Values (if any) are to be included in the Clip stage's trivial reject / trivial accept / must clip determination function. The ClipDistance Values (if present) are located in DW8-15 of the VUE Vertex Header located at the beginning of VUE URB entries. Bit 0 of this field corresponds to Clip Distance Value 0.
	7:0	**User Clip Distance Cull Test Enable Bitmask**

3DSTATE_VS_BODY		
		Format: U8 This 8 bit mask field selects which of the 8 Clip Distance Values (if any) are to be included in the Clip stage's trivial reject / trivial accept determination function. Note that must clip determination is not included in this function. The ClipDistance Values (if present) are located in DW8-15 of the VUE Vertex Header located at the beginning of VUE URB entries. Bit 0 of this field corresponds to Clip Distance Value 0.

3DSTATE_WM_BODY

<table>
<tr><th colspan="3">3DSTATE_WM_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31</td><td>Statistics Enable
Format: Enable
If ENABLED, the Windower and pixel pipeline will engage in statistics gathering. If DISABLED, statistics information associated with this FF stage will be left unchanged. See Statistics Gathering.
Programming Notes
This bit must be disabled if any of these bits is set: 3DSTATE_WM::Legacy Depth Buffer Clear, 3DSTATE_WM::Legacy Hierarchical Depth Buffer Resolve Enable or 3DSTATE_WM::Legacy Depth Buffer Resolve Enable.</td></tr>
<tr><td></td><td>30</td><td>Legacy Depth Buffer Clear Enable
Format: Enable
When set, the depth buffer is initialized as a side-effect of rendering pixels.
Programming Notes
If this field is enabled,
1. the Depth Test Enable field in DEPTH_STENCIL_STATE must be disabled.
2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set.
3. 3DSTATE_DEPTH_BUFFER::Stencil Write Enable must be set if 3DSTATE_STENCIL_BUFFER::Stencil buffer enable is set. Additionally the following must be set to the correct values.
  1. DEPTH_STENCIL_STATE::Stencil Write Mask must be 0xFF
  2. DEPTH_STENCIL_STATE::Stencil Test Mask must be 0xFF
  3. DEPTH_STENCIL_STATE::Back Face Stencil Write Mask must be 0xFF
  4. DEPTH_STENCIL_STATE::Back Face Stencil Test Mask must be 0xFF
Refer to section 0 "Depth Buffer Clear" for additional restrictions when this field is enabled. If this field is enabled,Pixel Shader Kill Pixel must be disabled.</td></tr>
<tr><td></td><td>29</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>28</td><td>Legacy Depth Buffer Resolve Enable</td></tr>
</table>

3DSTATE_WM_BODY

		Format: Enable When set, the depth buffer is made to be consistent with the hierarchical depth buffer as a side-effect of rendering pixels. This is intended to be used when the depth buffer is to be used as a surface outside of the 3D rendering operation. **Programming Notes** If this field is enabled, 1. the **Legacy Depth Buffer Clear** and **Legacy Hierarchical Depth Buffer Resolve Enable** fields must both be disabled. 2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. Refer to section 11.5.4.2 "Depth Buffer Resolve" for additional restrictions when this field is enabled. **If Hierarchical Depth Buffer Enable** is disabled, enabling this field will have no effect.
	27	**Legacy Hierarchical Depth Buffer Resolve Enable** Format: Enable When set, the hierarchical depth buffer is made to be consistent with the depth buffer as a side-effect of rendering pixels. This is intended to be used when the depth buffer has been modified outside of the 3D rendering operation. **Programming Notes** If this field is enabled, 1. the **Legacy Depth Buffer Clear** and **Legacy Depth Buffer Resolve Enable** fields must both be disabled. 2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. Refer to section 11.5.4.3 "Hierarchical Depth Buffer Resolve" for additional restrictions when this field is enabled. If **Hierarchical Depth Buffer Enable** is disabled, enabling this field will have no effect. **Performance Note:** expect the hierarchical depth buffer's impact on performance to be reduced for some period of time after this operation is performed, as the hierarchical depth buffer is initialized to a state that makes it ineffective. Further rendering will tend to bring the hierarchical depth buffer back to a more effective state.
	26	**Legacy Diamond Line Rasterization** Format: Enable This bit, if ENABLED, indicates that the Windower will rasterize zero width lines using the DX9 rasterization rules. If DISABLED, the Windower will rasterize zero width lines using the DX10 rasterization rules (see Strips Fans chapter).
	25:23	**Reserved** Format: MBZ
	22:21	**Early Depth/Stencil Control**

3DSTATE_WM_BODY

		Format: U2 Enumerated Type This field specifies the behavior of early depth/stencil test.

Value	Name	Description
0h	NORMAL	Depth/Stencil Test/Write behaves as if it happens post-shader, however the pixel shader is not necessarily executed if the pixel fails depth or stencil test (this is the legacy behavior)
1h	PSEXEC	Depth/Stencil Test/Write behaves as if it happens post-shader, and the pixel shader is executed if the pixel fails depth or stencil test (although pre-shader actions such as primitive inclusion, stipple, etc. will still cause the shader not to execute)
2h	PREPS	Depth/Stencil Test/Write behaves as if it happens pre-shader. The pixel shader is not executed if the pixel fails depth or stencil test. Depth and stencil writes occur even if the pixel is killed by the shader or post-shader by alpha test, etc. Depth output by the pixel shader is ignored.
3h	Reserved	

Programming Notes
The Early Depth/Stencil Control field cannot be set to PREPS (value = 2h) if ForceKillpix = ForceON or Forced Thread Dispatch = ForceON

20:19 **Force Thread Dispatch Enable**

Value	Name	Description
0h	Normal	WM_INT::ThreadDispatchEnable is computed normally
1h	ForceOff	Forces WM_INT::ThreadDispatchEnable Off
2h	ForceON	Forces WM_INT::ThreadDispatchEnable On
3h	Reserved	

Programming Notes
This should must always be set to Normal. This field should not be tested for functional validation

18:17 **Position ZW Interpolation Mode**

Format:	U2 Enumerated Type

This field elects "interpolation mode" associated with the Position Z (source depth) and W coordinates passed in the PS payload when the PS requires Position as input. This field does not determine whether these coordinates are actually included in the payload (see Pixel Shader Requires Depth, Pixel Shader Requires W).

Value	Name	Description
0h	INTERP_PIXEL	`Evaluate Z & W at the pixel center or UL corner (as`

3DSTATE_WM_BODY

			specified by Pixel Location of 3DSTATE_MULTISAMPLE)
		1h	Reserved
		2h	INTERP_CENTROID
		3h	INTERP_SAMPLE

Programming Notes
WM_INT::RT Independent Rasterization Enable must be disabled in order to select INTERP_SAMPLE.
MSDISPMODE_PERSAMPLE is required in order to select INTERP_SAMPLE.

16:11 **Barycentric Interpolation Mode**

Format:	Enable[6]

Controls which barycentric interpolation terms must be passed into the pixel shader kernel. Bit 0: Perspective Pixel Location barycentric is required Bit 1: Perspective Centroid barycentric is required Bit 2: Perspective Sample barycentric is required Bit 3: Non-perspective Pixel Location barycentric is required Bit 4: Non-perspective Centroid barycentric is required Bit 5: Non-perspective Sample barycentric is required

Programming Notes
If contiguous dispatch modes are enabled, only bit 3 (non-perspective pixel location) can be set, all other bits in this field must be zero. Pixel Location below refers to either the upper left corner or pixel center depending on the **Pixel Location** state of 3DSTATE_MULTISAMPLING). MSDISPMODE_PERSAMPLE is required in order to select Perspective Sample or Non-perspective Sample barycentric coordinates.

10 **Reserved**

Format:	MBZ

9:8 **Line End Cap Antialiasing Region Width**

Format:	U2

This field specifies the distances over which the coverage of anti-aliased line end caps are computed.

Value	Name	Description
0h	0.5 pixels	0.5 pixels
1h	1.0 pixels	1.0 pixels
2h	2.0 pixels	2.0 pixels
3h	4.0 pixels	4.0 pixels

7:6 **Line Antialiasing Region Width**

Format:	U2

3DSTATE_WM_BODY

		This field specifies the distance over which the anti-aliased line coverage is computed. Value | Name | Description 0h | 0.5 pixels | 0.5 pixels 1h | 1.0 pixels | 1.0 pixels 2h | 2.0 pixels | 2.0 pixels 3h | 4.0 pixels | 4.0 pixels
	5	**Reserved** Format: MBZ
	4	**Polygon Stipple Enable** Format: Enable Enables the Polygon Stipple function.
	3	**Line Stipple Enable** Format: Enable Enables the Line Stipple function.
	2	**Point Rasterization Rule** This field specifies the rasterization rules to be applied whenever the edges of a point primitive fall exactly on a pixel sampling point. Value | Name | Description 0h | RASTRULE_UPPER_LEFT | To match "normal" upper left rules for surface primitives 1h | RASTRULE_UPPER_RIGHT | To match OpenGL point rasterization rules (round to + infinity, where this is the upper right direction wrt OpenGL screen origin of lower left).
	1:0	**Force Kill Pixel Enable** Value | Name | Description 0h | Normal | WM_INT:: Pixel Shader Kill Pixel is computed normally 1h | ForceOff | Forces WM_INT:: Pixel Shader Kill Pixel Off 2h | ForceON | Forces WM_INT:: Pixel Shader Kill Pixel On 3h | Reserved | Programming Notes

3DSTATE_WM_BODY		
		This should must always be set to Normal. This field should not be tested for functional validation

3DSTATE_WM_CHROMAKEY_BODY

<table>
<tr><th colspan="3">3DSTATE_WM_CHROMAKEY_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31</td><td>ChromaKey Kill Enable
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>Enable</td></tr></table>If ENABLED, indicates that at least one of the attached samplers has ChromaKeyKill enabled.</td></tr>
<tr><td>30:0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

3DSTATE_WM_DEPTH_STENCIL_BODY

3DSTATE_WM_DEPTH_STENCIL_BODY		
Source:	RenderCS	
Size (in bits):	96	
Default Value:	0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:29	**Stencil Fail Op** Format: **3D_Stencil_Operation** This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test fails. **Programming Notes** 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.
	28:26	**Stencil Pass Depth Fail Op** Format: **3D_Stencil_Operation** This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes but the depth pass fails.
	25:23	**Stencil Pass Depth Pass Op** Format: **3D_Stencil_Operation** This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes but the depth test passes.
	22:20	**Backface Stencil Test Function** Format: **3D_Compare_Function**
	19:17	**Backface Stencil Fail Op** Format: **3D_Stencil_Operation**
	16:14	**Backface Stencil Pass Depth Fail Op** Format: **3D_Stencil_Operation** This field specifies the operation to perform on the Stencil Buffer when the stencil test passes but the depth pass fails.

3DSTATE_WM_DEPTH_STENCIL_BODY		
	13:11	**Backface Stencil Pass Depth Pass Op** Format: 3D_Stencil_Operation This field specifies the operation to perform on the Stencil Buffer when the stencil test passes and the depth pass passes (or is disabled).
	10:8	**Stencil Test Function** Format: 3D_Compare_Function This field specifies the comparison function used in the (front face) StencilTest function.
	7:5	**Depth Test Function** Format: 3D_Compare_Function Specifies the comparison function used in DepthTest function. **Programming Notes** If the Depth Test Function is ALWAYS or NEVER, the depth buffer is not read.
	4	**Double Sided Stencil Enable** Format: Enable Enable doubled sided stencil operations. Value | Name | Description 0h | False | Double Sided Stencil Disabled 1h | True | Double Sided Stencil Enabled **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 state • Back-facing primitives will be rendered, honoring all current device state, as though it were a front-facing primitive with no implicitly overloaded state.
	3	**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.
	2	**Stencil Buffer Write Enable**

3DSTATE_WM_DEPTH_STENCIL_BODY

DWord	Bit	Description
		Format: Enable Enables writes to the Stencil Buffer. **Programming Notes** If this field is enabled, Stencil Test Enable must also be enabled.
	1	**Depth Test Enable** Format: Enable Enables the DepthTest function of the Pixel Processing pipeline. Value / Name: 0h: Disable 1h: Enable **Programming Notes** If any of the render targets are YUV format, this field must be disabled.
	0	**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. This bit must not be set when WM_INT::RT Independent Rasterization Enable is true.
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 This field specifies a bit mask applied to backface stencil test values. Both the stencil reference

3DSTATE_WM_DEPTH_STENCIL_BODY		
		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	31:16	**Reserved** Format: MBZ
	15:8	**Stencil Reference Value** Format: U8 This field specifies the stencil reference value to compare against in the (front face) StencilTest function.
	7:0	**Backface Stencil Reference Value** Format: U8 This field specifies the stencil reference value to compare against in the StencilTest function.

3DSTATE_WM_HZ_OP_BODY

<table>
<tr><th colspan="3">3DSTATE_WM_HZ_OP_BODY</th></tr>
<tr><td colspan="3">Source: RenderCS
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="3">0</td><td>31</td><td>Stencil Buffer Clear Enable
Format: Enable
When set, the stencil buffer is initialized.
Programming Notes
If this field is enabled,
1. the Depth Buffer Resolve Enable (full or partial) and Hierarchical Depth Buffer Resolve Enable fields must both be disabled.
2. 3DSTATE_DEPTH_BUFFER::Stencil Write Enable must be set if 3DSTATE_STENCIL_BUFFER::Stencil buffer enable is set.</td></tr>
<tr><td>30</td><td>Depth Buffer Clear Enable
Format: Enable
When set, the depth buffer is initialized.
Programming Notes
If this field is enabled,
1. the Depth Buffer Resolve Enable(full or partial) and Hierarchical Depth Buffer Resolve Enable fields must both be disabled.
2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set.</td></tr>
<tr><td>29</td><td>Scissor Rectangle Enable
Format: Enable
Enables operation of Scissor Rectangle.
Programming Notes
In order get the functionality right if this bit is disabled, driver must clip the clear rectangle to scissor rectangle if scissor test is enabled before clearing.</td></tr>
</table>

3DSTATE_WM_HZ_OP_BODY		
	28	**Depth Buffer Resolve Enable** Format: Enable When set, the depth buffer is made to be consistent with the hierarchical depth buffer as a side-effect of rendering pixels. This is intended to be used when the depth buffer is to be used as a surface outside of the 3D rendering operation. The Depth buffer will be in uncompressed state after this operation. **Programming Notes** If this field is enabled, 1. the **Depth Buffer Clear** and **Hierarchical Depth Buffer Resolve Enable** fields must both be disabled. 2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. 3. Depth Buffer Partial Resolve Enable field should be disabled.
	27	**Hierarchical Depth Buffer Resolve Enable** Format: Enable When set, the hierarchical depth buffer is made to be consistent with the depth buffer as a side-effect of rendering pixels. This is intended to be used when the depth buffer has been modified outside of the 3D rendering operation. **Programming Notes** If this field is enabled, 1. the **Depth Buffer Clear** and **Depth Buffer Resolve Enable (full or partial)** fields must both be disabled. 2. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. 3. Stencil Buffer Resolve Enable must be disabled. Doing a Hierarchical Depth Buffer resolve (HZ resolve) on a partial HZ buffer is not permitted. The HZ resolve operation must be done on the entire HZ buffer. **Performance Note:** expect the hierarchical depth buffer's impact on performance to be reduced for some period of time after this operation is performed, as the hierarchical depth buffer is initialized to a state that makes it ineffective. Further rendering will tend to bring the hierarchical depth buffer back to a more effective state.
	26	**Pixel Position Offset Enable**

3DSTATE_WM_HZ_OP_BODY

		Format: Enable Enumerated Type Enables the device to offset pixel positions by 0.5 both in horizontal and vertical directions. **Programming Notes** Setting this field along with setting the Pixel Location to upper left and number of multisamples to greater than one will cause the device to offset pixel positions by 0.5 both in horizontal and vertical directions. It is to be noted this is done to adjust the pixel co-ordinate system to DX9 like, so any WM_HZ_OP screen space rectangles (eg: legacy HiZ Clear, Resolve etc) generated internally by driver in this mode needs to be aware of this offset adjustment and send the rectangles according to alignment restriction taking this offset adjustment into consideration. SW can choose to set this bit only for DX9 API. DX10/OGL API's should not have any effect by setting or not setting this bit.
	25	**Full Surface Depth and Stencil Clear** Format: Enable **Programming Notes** Setting this field to "1" along with "Depth buffer clear" will cause all the pixels/samples in an the HZ and Stencil CLs to be cleared. Software must set this only when the APP requires the entire Depth surface to be cleared. Setting this field to "1" for STC-buffer only clear without "depth buffer clear" will cause all the pixels/samples in the STC-CL to get the stc-ref value.
	24	**Reserved** Format: MBZ
	23:16	**Stencil Clear Value** Format: U8.0 This field specifies the stencil clear value.
	15:13	**Number of Multisamples** Format: U3 Enumerated Type This field specifies how many samples/pixel exist in the Depth Buffer and Stencil buffers, as log2(#samples). **Value** | **Name** | **Description**

<table>
<tr><th colspan="3">3DSTATE_WM_HZ_OP_BODY</th></tr>
<tr><td rowspan="2"></td><td></td><td><table>
<tr><td>0h</td><td>1</td><td>1 sample/pixel</td></tr>
<tr><td>1h</td><td>2</td><td>2 samples/pixel</td></tr>
<tr><td>2h</td><td>4</td><td>4 samples/pixel</td></tr>
<tr><td>3h</td><td>8</td><td>8 samples/pixel</td></tr>
<tr><td>5h-7h</td><td>Reserved</td><td></td></tr>
</table></td></tr>
<tr><td>12:0</td><td>Reserved<table><tr><td></td><td></td></tr></table></td></tr>
<tr><td rowspan="2">1
Programming Notes:
The Clear/Resolve rectangle X and Y Min values must be shifted by the LOD level; i.e. the hardware does not include the LOD in this function. Hence to clear any particular X, Y from the base level, to clear the contents at level "LOD" use (X»LOD) and (Y»LOD).
The final X and Y Min values, after LOD adjustment described above, have to be manually 8x4 aligned for Depth and HZ Resolve passes only. For Clears see "Full Surface Depth and Stencil Clear" field in this command instead.
resolve_aligned_y_min = (y_min & ~0x3) //round down to last multiple of 4
resolve_aligned_x_min = (x_min & ~0x7) //round down to last multiple of 8</td><td>31:16</td><td>Clear Rectangle Y Min<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16 in Pixels from Depth Buffer origin (upper left corner)</td></tr></table>Specifies Ymin value of (inclusive) of clear rectangle with the Depth Buffer, used for clipping. Pixels with Y coordinates less than Ymin will not be affected.</td></tr>
<tr><td>15:0</td><td>Clear Rectangle X Min<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16 in Pixels from Depth Buffer origin (upper left corner)</td></tr></table>Specifies Xmin value of (inclusive) of clear rectangle with the Depth Buffer, used for clipping. Pixels with X coordinates less than or equal to Xmin will not be affected.</td></tr>
<tr><td rowspan="2">2
Programming Notes:
See the programming note in the previous DWORD for the Min values.
The Clear/Resolve rectangleX andYMax values must be shifted by the LOD level; i.e. the hardware does not include the LOD in this function. Hence to clear any particular X, Y from the base level, to clear the contents at level "LOD" use (X»LOD) and (Y»LOD).
The final X and Y Max values, after LOD adjustment described above, have to be manually 8x4 aligned for Depth and HZ Resolve passes only. For Clears see "Full Surface Depth and Stencil Clear" field in this command instead.
resolve_aligned_y_max= (y_max & ~0x3)</td><td>31:16</td><td>Clear Rectangle Y Max<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16 in Pixels from Depth Buffer origin (lower right corner)</td></tr></table>Specifies Ymax value of (exclusive) of clear rectangle with the Depth Buffer, used for clipping. Pixels with Y coordinates greater than Ymax will be not be cleared.</td></tr>
<tr><td>15:0</td><td>Clear Rectangle X Max<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16 in Pixels from Depth Buffer origin (lower right corner)</td></tr></table>Specifies Xmax value of (exclusive) of clear rectangle with the Depth Buffer, used for clipping. Pixels with X coordinates greater than or equal to Xmax will be not be affected.</td></tr>
</table>

3DSTATE_WM_HZ_OP_BODY		

+ ((y_max & 0x3 == 0) ? 0 : 4) //round up to next multiple of 4
resolve_aligned_x_max= (x_max & ~0x7) + ((x_max & 0x7 == 0) ? 0 : 8) //round up to next multiple of 8		

3	31:16	**Reserved** Format: MBZ
	15:0	**Sample Mask** Format: `Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_WM_HZ_OP)` A per-multisample-position mask state variable that is immediately and unconditionally ANDed with the sample coverage mask as part of the rasterization process. This mask is applied prior to centroid selection. **Programming Notes** If Number of Multisamples is NUMSAMPLES_1, bits 15:1 of this field will be zeroed by HW.If Number of Multisamples is NUMSAMPLES_2, bits 15:2 of this field will be zeroed by HW.If Number of Multisamples is NUMSAMPLES_4, bits 15:4 of this field will be zeroed by HW.If Number of Multisamples is NUMSAMPLES_8, bits 15:8 of this field will be zeroed by HW.

A32 Buffer Base Address Message Header Control

MHC_A32_BBA - A32 Buffer Base Address Message Header Control

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

DWord	Bit	Description
0	31:10	**Buffer Base Address Offset** Format: GeneralStateOffset[31:10] Specifies the base address offset page [31:10] for A32 stateless messages. **Restriction** When using stateless A32 Data Port messages, General State Base Address[47:12] + Buffer Base Address[31:10] must be less than 2^48. It is illegal for this to be greater or equal than 2^48.
	9:0	**Reserved** Format: MBZ Ignored.

A32 Scaled Header Present Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_A32_MHP - A32 Scaled Header Present Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>Message Header Present
<table><tr><td></td><td></td></tr></table>In combination with the MDC_A32_SSO field, specifies the access type and address calculation. The access is an SLM access when the Sideband Scale Offset is enabled and the Message Header is not present
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>No [Default]</td><td>The Sideband Scale Offset field from the Message Descriptor are used as offsets with the Address Payload.</td></tr></table></td></tr>
</table>

A32 Sideband Scale and Offset Enable Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_A32_SBSO - A32 Sideband Scale and Offset Enable Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 8
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>7</td><td>Sideband Offset Enable
Format: MBO
Must be set for a scaled SLM access.The 16-bit offset from the Sideband is added to all the offsets in the Address Payload for the SLM access. The 16-bit Sideband Offset is specified in the extended function control field in the SEND instruction.</td></tr>
<tr><td>6:0</td><td>Scale
Format: U7
Specifies the scale pitch to be used for SLM messages as (#bytes-1).</td></tr>
</table>

A64 Data Size Message Descriptor Control Field

MDC_A64_DS - A64 Data Size Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	

DWord	Bit	Description
0	1:0	**Data Size**

Specifies the number of data elements to be read or written

Value	Name	Description	Programming Notes
00h	DE1	1 data element (B, DW, QW)	
01h	DE2	2 data elements (B, DW, QW)	
02h	DE4	4 data elements (B, DW, QW)	
03h	DE8	8 data elements (B, DW, QW)	[] This setting is supported for DW and QW but not for B. For bytes, the maximum number of data elements is 4.

Restriction
The number of elements is constrained by SIMD Mode and Data Width. The max data payload limit is 256B: 2 elements SIMD16 QW, 4 elements SIMD16 DW, or 4 elements SIMD8 QW.

A64 Hword Block Message Header

MH_A64_HWB - A64 Hword Block Message Header		
Source:	EuSubFunctionDataPort1	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..1	63:0	**BlockOffset** Format: U64 Specifies the U64 byte offset of Oword block. **Programming Notes** If the BlockOffset is not in the 48-bit canonical address range, the access is Out-of-Bounds. **Restriction** The byte offset must be aligned to the message's data type. Dwords have [1:0] = 0, Qwords have [2:0] = 0, and Hwords have [4:0] = 0.
2..4	95:0	**Reserved** Format: MBZ Ignored
5	31:0	**Reserved** Format: MBZ Ignored
6..7	63:0	**Reserved** Format: MBZ Ignored

A64 Hword Data Blocks Message Descriptor Control Field

MDC_A64_DB_HW - A64 Hword Data Blocks Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	3	
Default Value:	0x00000001	
DWord	**Bit**	**Description**
0	2:0	**Data Blocks** Specifies the number of Hwords to be read or written

Value	Name	Description
01h	HW1 **[Default]**	1 Hword block
02h	HW2	2 Hword blocks
03h	HW4	4 Hword blocks
04h	HW8	8 Hword blocks
Others	Reserved	Ignored

A64 Oword Block Message Header

<table>
<tr><th colspan="3">MH_A64_OWB - A64 Oword Block Message Header</th></tr>
<tr><td colspan="3">Source: EuSubFunctionDataPort1
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0..1</td><td>63:0</td><td>BlockOffset
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U64</td></tr></table>Specifies the U64 byte offset of Oword block.
<table><tr><th>Programming Notes</th></tr><tr><td>If the BlockOffset is not in the 48-bit canonical address range, the access is Out-of-Bounds.</td></tr></table>
<table><tr><th>Restriction</th></tr><tr><td>The byte offset must be aligned to the message's data type. Dwords have [1:0] = 0, Qwords have [2:0] = 0, and Hwords have [4:0] = 0.</td></tr></table></td></tr>
<tr><td>2..7</td><td>191:0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>Ignored</td></tr>
</table>

A64 Oword Data Blocks Message Descriptor Control Field

MDC_A64_DB_OW - A64 Oword Data Blocks Message Descriptor Control Field

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

DWord	Bit	Description
0	2:0	**Data Blocks**

Specifies the number of Oword blocks to be read or written

Value	Name	Description
00h	OW1L	1 Oword, read into or written from the low 128 bits of the destination register
01h	OW1U	1 Oword, read into or written from the high 128 bits of the destination register
02h	OW2	2 Owords
03h	OW4	4 Owords
04h	OW8	8 Owords
Others	Reserved	Ignored

A64 Scaled Header Present Message Descriptor Control Field

MDC_A64_MHP - A64 Scaled Header Present Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**Message Header Present**

Specifies if the message uses the optional message header to modify the A64 address calculation, in combination with MDC_A64_SSO field.

Value	Name	Description
0h	No	Message header is not present
1h	Yes	Message header is present

Programming Notes
The access is Out-of-Bounds if the SideBand Offset is enabled when the Message Header is not present.

AddrSubRegNum

AddrSubRegNum		
Source:	Eulsa	
Size (in bits):	4	
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	3:0	**Address Subregister Number** Value: 0-15 — Name: Address Subregister Number

Value	Name
0-15	Address Subregister Number

Any Binding Table Index Message Descriptor Control Field

MDC_BTS_SLM_A32 - Any Binding Table Index Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	8	
Default Value:	0x00000000	

DWord	Bit	Description
0	7:0	**Binding Table Index**

Specifies the surface for the message, which can be Surface State Model, SLM or Stateless.

Value	Name	Description
00h-0EFh	BTS	Index of Binding Table State Surfaces
F0h-0FBh	Reserved	Reserved
0FCh	SSO	Specifies a Surface State Offset supplied by the extended message descriptor
0FEh	SLM	Specifies an SLM access
0FFh	A32_A64	Specifies a A32 or A64 Stateless access that is locally coherent (coherent within a thread group)
0FDh	A32_A64_NC	Specifies a A32 or A64 Stateless access that is non-coherent (coherent within a thread).

Restriction
When using A32_A64_NC, SW must ensure that 2 threads do not both access the same cache line (64B)

Atomic Float Binary Operation Message Descriptor Control Field

MDC_FOP2 - Atomic Float Binary Operation Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	3	
Default Value:	0x00000001	
DWord	**Bit**	**Description**
0	2:0	**Atomic Float Operation Type** Specifies the atomic float binary operation to be performed

Value	Name	Description	Programming Notes
01h	AOP_FMAX **[Default]**	new_dst = fmax(old_dst, src0)	The fmax operation implements the IEEE specification, which differs slightly from the DX and OCL specifications when a source operand is a sNaN. fmax(x,qNaN) = fmax(qNaN,x) = x fmax(x,sNaN) = fmax(sNaN,x) = quietize(sNaN) fmax(sNaN,sNaN) = fmax(sNaN,qNaN) = fmax(qNan,sNaN) = quietize(sNaN) fmax(qNaN,qNan) = qNaN fmax(-0, +0) = fmax(+0, -0) = +0 [] Fmax with sNaN operand returns sNaN instead of quietize(sNaN) [] Fmax(-0,+0) returns -0. Should be +0, to match EU Fmax instruction.
02h	AOP_FMIN	new_dst = fmin(old_dst, src0)	The fmin operation implements the IEEE specification, which differs slightly from the DX and OCL specifications when a source operand is a sNaN. fmin(x,qNaN) = fmin(qNaN,x) = x fmin(x,sNaN) = fmin(sNaN,x) = quietize(sNaN) fmin(sNaN,sNaN) = fmin(sNaN,qNaN) = fmin(qNan,sNaN) = quietize(sNaN) fmin(qNaN,qNan) = qNaN fmin(+0, -0) = fmin(-0, +0) = -0 [] Fmin with sNaN operand returns sNaN instead of quietize(sNaN) [] Fmin(+0,-0) returns +0. Should be -0, to match EU Fmin instruction.

Atomic Float Ternary Operation Message Descriptor Control Field

MDC_FOP3 - Atomic Float Ternary Operation Message Descriptor Control Field

Source:	BSpec
Size (in bits):	3
Default Value:	0x00000003

DWord	Bit	Description
0	2:0	**Atomic Float Operation Type**

Specifies the atomic float ternary operation to be performed

Value	Name	Description	Programming Notes
03h	AOP_FCMPWR **[Default]**	new_dst = (src0 == old_dst) ? src1 : old_dst	The fcmpwr operation performs the comparison using IEEE specification rules, and performs the store as a raw move (so SNaN is not quietized). fcmpwr(NaN,x,y) = NaN fcmpwr(x, NaN,y) = x fcmpwr(x,x, NaN) = NaN
Others	Reserved	Ignored	

Programming Notes
When Return Data Control is set, old_dst is returned.

Atomic Integer Binary Operation Message Descriptor Control Field

MDC_AOP2 - Atomic Integer Binary Operation Message Descriptor Control Field

Source: BSpec
Size (in bits): 4
Default Value: 0x00000001

DWord	Bit	Description
0	3:0	**Atomic Integer Operation Type**

Specifies the atomic integer binary operation to be performed

Value	Name	Description
01h	AOP_AND **[Default]**	new_dst = old_dst AND src0
02h	AOP_OR	new_dst = old_dst | src0
03h	AOP_XOR	new_dst = old_dst ^ src0
04h	AOP_MOV	new_dst = src0
07h	AOP_ADD	new_dst = old_dst + src0
08h	AOP_SUB	new_dst = old_dst - src0
09h	AOP_REVSUB	new_dst = src0 - old_dst
0Ah	AOP_IMAX	new_dst = imax(old_dst, src0)
0Bh	AOP_IMIN	new_dst = imin(old_dst, src0)
0Ch	AOP_UMAX	new_dst = umax(old_dst, src0)
0Dh	AOP_UMIN	new_dst = umin(old_dst, src0)
Others	Reserved	Ignored

Programming Notes
When Return Data Control is set, old_dst is returned.

Atomic Integer Ternary Operation Message Descriptor Control Field

MDC_AOP3 - Atomic Integer Ternary Operation Message Descriptor Control Field

Source:	BSpec
Size (in bits):	4
Default Value:	0x0000000E

DWord	Bit	Description
0	3:0	**Atomic Integer Operation Type**

Specifies the atomic integer ternary operation to be performed

Value	Name	Description
00h	AOP_CMPWR_2W	new_dst = (src0_2W == old_dst_2W) ? src1_2W : old_dst_2W
0Eh	AOP_CMPWR **[Default]**	new_dst = (src0 == old_dst) ? src1 : old_dst
Others	Reserved	Ignored

Programming Notes
When Return Data Control is set, old_dst is returned.

Atomic Integer Unary Operation Message Descriptor Control Field

MDC_AOP1 - Atomic Integer Unary Operation Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	4	
Default Value:	0x00000005	

DWord	Bit	Description
0	3:0	**Atomic Integer Operation Type** Specifies the atomic integer unary operation to be performed

Value	Name	Description
05h	AOP_INC **[Default]**	new_dst = old_dst + 1
06h	AOP_DEC	new_dst = old_dst - 1
0Fh	AOP_PREDEC	new_dst = old_dst - 1
Others	Reserved	Ignored

Programming Notes
When Return Data Control is set, new_dst is returned by AOP_PREDEC and otherwise old_dst is returned.

Audio Power State Format

Audio Power State Format		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000003	
DWord	**Bit**	**Description**
0	1:0	**Power State**

Value	Name	Description
00b	D0	D0
01b,10b	Unsupported	Unsupported
11b	D3 **[Default]**	D3

AVC CABAC

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	**Reserved** MBZ
	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

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	**Reserved** Reserved
	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	**Mbytype/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.

AVC CAVLC		
	0	**MB Concealment Flag** Each pulse from this flag indicates one MB is concealed by hardware.

Barrier Data Payload

<table>
<tr><th colspan="3">MDP_Barrier - Barrier Data Payload</th></tr>
<tr><td colspan="3">Source: EuSubFunctionGateway
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0..1</td><td>63:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="7">2</td><td>31</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>30:24</td><td>Barrier ID
Format: U7
This field indicates which barrier state is updated.
Range = [0,63]</td></tr>
<tr><td>23</td><td>Predicate Mask Enable
Format: Enable
This bit indicates that the barrier is a predicated barrier and the SIMD channels passing the predicate should be summed. All threads sending this message to the same barrier should have an identical value for this field, and must specify a response length of 1 for the predicate sum response. Note that Global Barriers must not have the Predicate Mask Enable bit set.
Programming Notes
This control is intended only for GPGPU or Media threads. This control must not be set if the barrier is for a Hull Shader thread.</td></tr>
<tr><td>22:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15</td><td>Barrier Count Enable
Format: Enable
Allows the message to reprogram the terminating barrier count. If set, the stored value of the terminating barrier count is set to the value of Barrier Count field (below), and used for this barrier operation. If clear, the stored value of the terminating barrier count is not modified and the stored value is used for this barrier operation.
Programming Notes
This control is intended only for Hull Shader threads. This control must not be set if the barrier is allocated by a GPGPU or Media thread.</td></tr>
<tr><td>14:8</td><td>Barrier Count
Format: U7</td></tr>
</table>

MDP_Barrier - Barrier Data Payload

		If Barrier Count Enable is set, this field specifies the terminating barrier count. Otherwise this field is ignored. All threads that belong to a single barrier must deliver the same value for this field for a particular barrier iteration.
	7:0	**Reserved** Format: MBZ
3	31:0	**Predicate Mask** Format: U32 This field has a bit set per SIMD channel that passes the predicate. For SIMD8 and SIMD16 the rest of the bits must be 0. This field is ignored for non-predicated barriers.
4..7	127:0	**Reserved** Format: MBZ

BaseAddress4KByteAligned

BaseAddress4KByteAligned		
Source:	BSpec	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Specifies a 64-bit (48-bit canonical) 4K-byte aligned memory base address.		
DWord	**Bit**	**Description**
0..1	63:12	**Base Address** Format: GraphicsAddress63-12
	11:0	**Reserved** Format: MBZ

BCS Hardware-Detected Error Bit Definitions

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	**Command Privilege Violation Error** This bit is set if a command classified as privileged is parsed in a non-privileged batch buffer. The command will be converted to a NOOP and parsing will continue.
	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: 1 — Name: (none) — Description: Instruction Error detected **Programming Notes** This error indications cannot be cleared except by reset (i.e., it is a fatal error).

BINDING_TABLE_EDIT_ENTRY

BINDING_TABLE_EDIT_ENTRY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:24	**Reserved** Format: MBZ
	23:16	**Binding Table Index** Format: U8 This field specifies the index of binding table entry that will be updated.
	15:0	**Surface State Pointer** Format: SurfaceStateOffset[21:6]RENDER_SURFACE_STATE [] Surface State Pointer. This address points to a surface state block. This pointer is relative to the Surface State Base Address.

Bit Definition for Interrupt Control Registers - Media

<table>
<tr><th colspan="3">Bit Definition for Interrupt Control Registers - Media</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="12">0</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:12</td><td>Reserved
Format: MBZ
These bits may be assigned to interrupts on future products/steppings.</td></tr>
<tr><td>11</td><td>Wait on Semaphore
Exec-List Scheduling: Set when MI_SEMAPHORE_WAIT command is un-successful and when "Inhibit Synchronous Context Switch" is set. Scheduler can use this interrupt to preempt the context waiting on semaphore wait. Ring Buffer Scheduling: Set when MI_SEMAPHORE_WAIT command is un-successful.</td></tr>
<tr><td>10</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>9</td><td>Reserved</td></tr>
<tr><td>8</td><td>Context Switch Interrupt
Set when a context switch has just occurred. Execlist Enable bit needs to be set for this interrupt to occur.</td></tr>
<tr><td>7</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>6</td><td>Timeout Counter Expired
Set when the VCS timeout counter has reached the timeout thresh-hold value.</td></tr>
<tr><td>5</td><td>Reserved</td></tr>
<tr><td>4</td><td>MI_FLUSH_DW 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.</td></tr>
<tr><td>3</td><td>Video 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 Blitter Instruction Parser encounters an error while parsing an</td></tr>
</table>

Bit Definition for Interrupt Control Registers - Media		
		instruction.
	2:1	**Reserved** Format: MBZ
	0	**Video Command Parser User Interrupt** This status bit is set when an MI_USER_INTERRUPT instruction is executed on the Video 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

BLEND_STATE	
Source:	BSpec
Size (in bits):	544
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

The blend state is stored as a structure containing a common DWORD that applies to all RTs and an array of up to 8 elements, each of which contains the two DWords for each. The start of each element is spaced 2 DWords apart. The blend state is aligned to a 64-byte boundary, which is pointed to by a field in 3DSTATE_BLEND_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	**Alpha To Coverage 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 field is applied to all the RTs in MRT case.
	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.The field is applied to all the RTs in MRT case.
	29	**Alpha To One Enable** Format: Enable If set, Source0 Alpha is set to 1.0f after (possibly) being used to generate the AlphaToCoverage coverage mask.If Dual Source Blending is enabled, this bit must be disabled.The field is applied to all the RTs in MRT case.
	28	**Alpha To Coverage 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. If

BLEND_STATE

		AlphaToCoverage is disabled, AlphaToCoverage Dither does not have any impact.The field is applied to all the RTs in MRT case.
	27	**Alpha Test Enable** Format: Enable Enables the AlphaTest function of the Pixel Processing pipeline.The field is applied to all the RTs in MRT case. **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.
	26:24	**Alpha Test Function** Format: 3D_Compare_Function This field specifies the comparison function used in the AlphaTest functionThe field is applied to all the RTs in MRT case.
	23	**Color Dither Enable** Format: Enable Enables dithering of colors (including any alpha component) before they are written to the Color Buffer. The field is applied to all the RTs in MRT case. **Programming Notes** For YUV render target formats, this field must be programmed to 0.
	22:21	**X Dither Offset** Format: U2 Specifies offset to apply to pixel X coordinate LSBs when accessing dither table.The field is applied to all the RTs in MRT case.
	20:19	**Y Dither Offset** Format: U2 Specifies offset to apply to pixel Y coordinate LSBs when accessing dither table.The field is applied to all the RTs in MRT case.
	18:0	**Reserved** Format: MBZ
1..16	511:0	**Entry** Format: BLEND_STATE_ENTRY[8]

BLEND_STATE_ENTRY

<table>
<tr><th colspan="3">BLEND_STATE_ENTRY</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0..1</td><td>63</td><td>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</td></tr>
<tr><td></td><td>62:59</td><td>Logic Op Function
Format: 3D_Logic_Op_Function
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.</td></tr>
<tr><td></td><td>58:37</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>36</td><td>Pre-Blend Source Only Clamp Enable
Format: Enable
This field specifies whether the source(s) are clamped prior to blending, regardless of whether blending is enabled. If DISABLED, no clamping is performed prior to blending. If ENABLED, only source0 and source 1, if dual source is enabled, are clamped prior to the blend to the range specified by Color Clamp Range.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>Disabled</td><td>No clamping is performed prior to blending.</td></tr>
<tr><td>1</td><td>Enabled</td><td>Only Source(s) are clamped prior to blend function. Other inputs to blend must be clamped according to the behavior specified for "pre-blend color clamp disable" in the pre-blend color clamping table .</td></tr>
</table>
Programming Notes
This field is ignored (treated as DISABLED) for UINT and SINT RT surface formats. Blending is not supported for those RT surface formats. When this bit is enabled Pre-Blend Color Clamp Enable must be disabled.</td></tr>
<tr><td></td><td>35:34</td><td>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</td></tr>
</table>

BLEND_STATE_ENTRY

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). Unsigned Floating Point components are clamped to positive zero.
3	Reserved	Reserved

Programming Notes

If Pre Blend Source Only Clamping is enabled, Color Clamp Range should be programmed to COLORCLAMP_UNORM, else it should always be programmed to the RT range.

33 **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.

32 **Post-Blend Color Clamp Enable**

Format:	Enable

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

This field is ignored (treated as DISABLED) for UINT and SINT RT surface formats. Blending is not supported for those RT surface formats. Post Blend Clamp Enable must be programmed identical to Pre Blend Clamp Enable. The device will automatically clamp source color channels to the respective RT surface range. **When this bit is enabled Pre-Blend Source Only Clamp Enable must be disabled.**

31 **Color Buffer Blend Enable**

Format:	Enable

Enables the ColorBufferBlending (nee "alpha blending") function of the Pixel Processing Pipeline

BLEND_STATE_ENTRY

		for this render target. **Programming Notes** Enabling LogicOp and ColorBufferBlending at the same time is UNDEFINED
	30:26	**Source Blend Factor** Format: **3D_Color_Buffer_Blend_Factor** Controls the "source factor" in the ColorBufferBlending function.Refer to Source Alpha Blend Factor for encodings.
	25:21	**Destination Blend Factor** Format: **3D_Color_Buffer_Blend_Factor** Controls the "destination factor" in the ColorBufferBlending function. Refer to Source Alpha Blend Factor for encodings.
	20:18	**Color Blend Function** Format: **3D_Color_Buffer_Blend_Function** 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.
	17:13	**Source Alpha Blend Factor** Format: **3D_Color_Buffer_Blend_Factor** Controls the "source factor" in alpha Color Buffer Blending stage.Note: For the 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.
	12:8	**Destination Alpha Blend Factor** Format: **3D_Color_Buffer_Blend_Factor** Controls the "destination factor" in alpha Color Buffer Blending stage. Refer to Source Alpha Blend Factor for encodings.
	7:5	**Alpha Blend Function** Format: **3D_Color_Buffer_Blend_Function** 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.
	4	**Reserved** Format: MBZ
	3	**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**

BLEND_STATE_ENTRY

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

2 — 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).

1 — 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).

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

Blitter Interrupt Vector

BLITTER_INTR_VEC - Blitter Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	15	**Catastrophic Error** This interrupt signals that a unrecoverable error (for e.g encountered fault when accessing a page mapped in Global GTT) during the engine processing. When Memory interface signals this error, the Command Streamer will stop parsing any more instructions. Scheduler is expected to reset the engine to evict the context
	14:12	**Reserved**
	11	**BCS Wait On Semaphore**
	10	**Reserved**
	9	**Reserved**
	8	**BCS Context Switch Interrupt**
	7	**Legacy Context Per Process Page Fault Interrupt** Fault interrupt is generated by GA fabric, not by the CS This interrupt is for handling Legacy context PPGTT Page Fault.
	6	**BCS Watchdog Counter Expired**
	5	**Reserved**
	4	**BCS MI Flush DW Notify**
	3	**BCS Error Interrupt**
	2:1	**Reserved**
	0	**BCS MI User Interrupt**

Block Dimensions Message Header Control

MHC_BDIM - Block Dimensions Message Header Control

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

DWord	Bit	Description
0	31:22	**Reserved** Format: MBZ Ignored
	21:20	**Block Height** Height in rows of block being accessed. Range = [0,3] representing 1 to 8 rows.
	19:2	**Reserved** Format: MBZ Ignored
	1:0	**Block Width** Width in Dwords of block being accessed. Range = [0,3] representing 1 to 8 Dwords.

Block Height values:

Value	Name	Description
0h	H1	Block height = 1 row
1h	H2	Block height = 2 rows
2h	H4	Block height = 4 rows
03h	H8	Block height = 8 rows

Block Width values:

Value	Name	Description
0h	W1	Block width = 1 Dword
1h	W2	Block width = 2 Dwords
2h	W4	Block width = 4 Dwords
03h	W8	Block width = 8 Dwords

Block Message Header

MH_BTS_GO - Block Message Header		
Source:	EuSubFunctionDataPort0	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..1	63:0	**Reserved** Format: MBZ Ignored
2	31:0	**Global Offset** Format: U32 Specifies the global element index into the buffer, in units of Hwords, Owords, Dwords, or Bytes (depending on the message). **Programming Notes** The Global Offset for the Aligned Block operations is specified as a Dword-aligned byte offset (offset bits [1:0] = 0), Oword-aligned byte offset (offset bits [3:0]=0), or Hword-aligned byte offset (offset bits [4:0]=0). If the address offset calculated with the Global Offset is greater than the Surface Size, then the access is Out-of-Bounds.
3..7	159:0	**Reserved** Format: MBZ Ignored

BR00 - BLT Opcode and Control

<table>
<tr><th colspan="3">BR00 - BLT Opcode and Control</th></tr>
<tr><td colspan="3">Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31</td><td>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.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>Idle [Default]</td></tr><tr><td>1</td><td>Busy</td></tr></table></td></tr>
<tr><td></td><td>30</td><td>Setup Instruction Instruction
<table><tr><td>Default Value:</td><td>0</td></tr></table>The current instruction performs clipping (1).</td></tr>
<tr><td></td><td>29</td><td>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.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>Color [Default]</td></tr><tr><td>1</td><td>Monochrome</td></tr></table></td></tr>
<tr><td></td><td>28:22</td><td>Instruction Target (Opcode)
<table><tr><td>Default Value:</td><td>0000000b</td></tr></table>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.</td></tr>
<tr><td></td><td>21:20</td><td>32bpp Byte Mask
This field is only used for 32bpp.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>00b</td><td>[Default]</td></tr><tr><td>1xb</td><td>Write Alpha Channel</td></tr><tr><td>x1b</td><td>Write RGB Channel</td></tr></table></td></tr>
<tr><td></td><td>19:17</td><td>Monochrome Source Start
<table><tr><td>Default Value:</td><td>000b</td></tr></table>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.</td></tr>
</table>

BR00 - BLT Opcode and Control

	16	**Bit/Byte Packed** Byte packed is for the NT driver. <table><tr><th>Value</th><th>Name</th></tr><tr><td>0b</td><td>Bit **[Default]**</td></tr><tr><td>1b</td><td>Byte</td></tr></table>
	15	**Src Tiling Enable** <table><tr><th>Value</th><th>Name</th></tr><tr><td>0b</td><td>Tiling Disabled (Linear) **[Default]**</td></tr><tr><td>1b</td><td>Tiling enabled: Tile-X or Tile-Y</td></tr></table>
	14:12	**Horizontal Pattern Seed** <table><tr><td>Default Value:</td><td>0b</td></tr></table>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. Pre-Dev Blitter never executes in Tiled-Y mode, DevGT+ Blitter supports both Tile-X and Tile-Y modes. On reset, this bit will be '0'. This definition applies to only X, Y Blits. <table><tr><th>Value</th><th>Name</th></tr><tr><td>0b</td><td>Tiling Disabled (Linear blit) **[Default]**</td></tr><tr><td>1b</td><td>Tiling enabled: Tile-X or Tile-Y</td></tr></table>
	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. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>xx0b</td><td>**[Default]**</td><td>No color transparency mode enabled. This causes normal operation with regard to writing data to the destination.</td></tr><tr><td>001b</td><td></td><td>[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.</td></tr><tr><td>011b</td><td></td><td>[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,</td></tr></table>

		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."
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.

Bit	Description
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

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. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0b</td><td>**[Default]**</td><td>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.</td></tr><tr><td>1b</td><td></td><td>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.</td></tr></table>
	30	**Clipping Enabled** <table><tr><th>Value</th><th>Name</th></tr><tr><td>0b</td><td>**[Default]**</td></tr><tr><td>1b</td><td></td></tr></table>
	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. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0b</td><td>**[Default]**</td><td>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.</td></tr><tr><td>1b</td><td></td><td>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.</td></tr></table>
	28	**Monochrome Pattern Transparency Mode**

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

		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 x1b / Write RGB Channel
	25:24	**Color Depth** **Value** / **Name** 00b / 8 Bit Color Depth **[Default]** 01b / 16 Bit Color Depth 10b / Alternate 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

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

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

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 Lower Order Address bits

<table>
<tr><th colspan="3">BR07 - Setup Blit Color Pattern Address Lower Order Address bits</th></tr>
<tr><td colspan="3">Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:6</td><td>Setup Blit Color Pattern Address
Format: GraphicsAddress[31:6]
Lower 32bits of the 48bit addressing.
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.
The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</td></tr>
<tr><td>5:0</td><td>Reserved
Format: MBZ</td></tr>
</table>

BR09 - Destination Address Lower Order Address Bits

BR09 - Destination Address Lower Order Address Bits		
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. when tiling is disabled for XY-blits, this base address should be CL (64byte) aligned. These lower 32bits of the 48bit address, which specify the starting pixel address of the destination data. This register is also the working destination address register for the lower 32bits of the address, 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)

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 Lower order Address bits

BR12 - Source Address Lower order Address bits		
Source:	BlitterCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:0	**Source Address Bits** Format: GraphicsAddress[31:0] Lower 32bits of the 48bit addressing. When tiling is enabled for XY-blits with Color source surfaces, this base address should be aligned to 4KB. When tiling is disabled for XY-blits, this base address should be CL (64byte) aligned. Note that for non-XY blit with Color Source (SRC_COPY_BLT), this address points to the first byte to be read. These lower 32bits of the 48bit address, 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. If this Source happens to be a Monosource surface, then this Monosource Base Address programmed, must always be Cache Line (64byte) aligned.

BR13 - BLT Raster OP, Control, and Destination Pitch

<table>
<tr><th colspan="3">BR13 - BLT Raster OP, Control, and Destination Pitch</th></tr>
<tr><td colspan="3">Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31</td><td>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.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>[Default]</td><td>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.</td></tr>
<tr><td>1</td><td></td><td>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.</td></tr>
</table></td></tr>
<tr><td></td><td>30</td><td>Clipping Enabled
<table>
<tr><td>Default Value:</td><td>0</td></tr>
</table></td></tr>
<tr><td></td><td>29</td><td>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.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>[Default]</td><td>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.</td></tr>
<tr><td>1</td><td></td><td>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.</td></tr>
</table></td></tr>
<tr><td></td><td>28</td><td>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</td></tr>
</table>

BR13 - BLT Raster OP, Control, and Destination Pitch

		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 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

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 Lower order Address bits

BR15 - Color Pattern Address Lower order Address bits		
Source:	BlitterCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:6	**Color Pattern Address** Format: GraphicsAddress[31:6] Lower 32bits of the 48bit addressing. 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. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.
	5:0	**Reserved** Format: MBZ

BR16 - Pattern Expansion Background and Solid Pattern Color

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

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

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

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.

BR27 - Destination Higher Order Address

BR27 - Destination Higher Order Address	
Source:	BlitterCS
Size (in bits):	32
Default Value:	0x00000000

Upper 32 bits of the starting pixel address for the destination data. This structure is also the working location for the upper bits of the destination address, and changes as the BLT Engine performs the accesses. See BR09 for the lower 32 bits. 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.
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), the destination address points to the first byte to be written. This structure is always the last location written for a BLT drawing instruction. Writing to BR27 starts the BLT engine execution. Note: Some instructions affect only one scan line (requiring only one coordinate); other instructions affect multiple scan lines and need both coordinates.
GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The uppermost reserved bits are ignored and MBZ.

DWord	Bit	Description
0	31:16	**Reserved** Format: MBZ
	15:0	**Destination Address Upper DWORD** Format: GraphicsAddress[47:32]

BR28 - Source Higher Order Address

<table>
<tr><th colspan="3">BR28 - Source Higher Order Address</th></tr>
<tr><td colspan="3">Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0
Upper 32 bits of the Source address, specifying the starting pixel address of the color or mono source data. 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.
GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The uppermost reserved bits are ignored and MBZ.</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:0</td><td>Source Address Upper DWORD
Format: GraphicsAddress[47:32]</td></tr>
</table>

BR29 - Color Pattern Higher Order Address

BR29 - Color Pattern Higher Order Address		
Source:	BlitterCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0 Upper 32 bits of the Color Pattern address, specifying the starting location of the (8X8) pixel pattern. GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The uppermost reserved bits are ignored and MBZ.	31:16	**Reserved** Format: MBZ
	15:0	**Color Pattern Address Upper DWORD** Format: GraphicsAddress[47:32]

BR30 - Setup Blit Color Pattern Higher Order Address

<table>
<tr><th colspan="3">BR30 - Setup Blit Color Pattern Higher Order Address</th></tr>
<tr><td colspan="3">Source: BlitterCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0
Upper 32 bits of the Color Pattern address, specifying the starting location of the (8X8) pixel pattern.
GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The uppermost reserved bits are ignored and MBZ.</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:0</td><td>Setup Blit Color Pattern Upper DWORD
Format: GraphicsAddress[47:32]</td></tr>
</table>

Byte Masked Media Block Message Header

MH_MBBM - Byte Masked Media Block Message Header

Source:	EuSubFunctionDataPort1
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:0	**X Offset** Format: S31 X offset (in bytes) of the upper left corner of the block into the surface. **Programming Notes** Must be DWord aligned (Bits 1:0 MBZ) for the write form of the message.
1	31:0	**Y Offset** Format: S31 Y offset (in rows) of the upper left corner of the block into the surface.
2	31:0	**Media Block Message Control** Format: MHC_MBBM_CONTROL Specifies the Byte Masked message subtype and its additional input parameters.
3	31:0	**Byte Mask** Format: U32 Specifies the Byte Mask for writes when Message Mode field is BYTE_MASK. **Programming Notes** The Byte mask applies horizontally to each row of output: bit 0 for byte 0, through bit 31 for byte 31.
4	31:0	**FFTID** Format: MHC_FFTID Fixed Function Thread ID
5..7	95:0	**Reserved** Format: MBZ Ignored

Byte Masked Media Block Message Header Control

MHC_MBBM_CONTROL - Byte Masked Media Block Message Header Control

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

DWord	Bit	Description
0	31:30	**Message Mode** Specifies the Media Block Write Message subtype is Byte Masked. Value | Name | Description 02h | BYTE_MASK | The Block Height and Block Width fields are specified in this Dword. The Byte Mask qualifies which bytes are written. Others | Reserved | Reserved.
	29	**Reserved** Format: MBZ Ignored
	28:24	**Sub-Register Offset** Format: U5 This field is ignored (reserved) for Media Block Write message.
	23:22	**Reserved** Format: MBZ Ignored
	21:16	**Block Height** Format: U6 Height in rows of block being accessed. Range = [0,63] representing 1 to 64 rows **Restriction** If Block Width (bytes), then Maximum Block Height (rows) is constrained by (# Dwords width) * (# rows) <= 64 Dwords.
	15:10	**Reserved**

MHC_MBBM_CONTROL - Byte Masked Media Block Message Header Control

		Format: MBZ Ignored
	9:8	**Register Pitch Control** Format: U2 This field is ignored (reserved) for a Media Block Write message.
	7:6	**Reserved** Format: MBZ Ignored
	5:0	**Block Width** Format: U6 Width in bytes of the block being accessed. Range = [0,31] representing 1 to 32 Bytes. **Programming Notes** Must be DWord aligned for Media Block Write message.

CC_VIEWPORT

<table>
<tr><th colspan="3">CC_VIEWPORT</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">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 and Maximum Depth legal value ranges are dependant on the depth buffer format.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:0</td><td>Minimum Depth
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>IEEE_Float</td></tr></table>
Indicates the minimum depth. The interpolated or computed depth is clamped to this value prior to the depth test.
<table><tr><th>Programming Notes</th></tr><tr><td>The Minimum depth value must be less-than-or-equal to the Maximum depth value.
The Minimum depth value cannot be NAN (Not-A-Number).
For All depth formats:Minimum depth value must not be less than 0.0, also it may not be -0.0 (negative zero)</td></tr></table></td></tr>
<tr><td>1</td><td>31:0</td><td>Maximum Depth
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>IEEE_Float</td></tr></table>
Indicates the maximum depth. The interpolated or computed depth is clamped to this value prior to the depth test.
<table><tr><th>Programming Notes</th></tr><tr><td>The Maximum depth value cannot be smaller than Minimum depth value.
The Maximum depth value cannot be NAN (Not-A-Number).
For all depth formats: The Maximum depth value must be between +0.0to +1.0.</td></tr></table></td></tr>
</table>

Channel Mask Message Descriptor Control Field

MDC_CMASK - Channel Mask Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	4	
Default Value:	0x00000000	

DWord	Bit	Description
0	3:0	**Mask** For the read message, indicates that which channels are read from the surface and included in the writeback message. For the write message, indicates which channels are included in the message payload and written to the surface.

Value	Name	Description
00h	RGBA **[Default]**	Red, Green, Blue, and Alpha are included
01h	GBA	Green, Blue, and Alpha are included
02h	RBA	Red, Blue, and Alpha are included
03h	BA	Blue and Alpha are included
04h	RGA	Red, Green, and Alpha are included
05h	GA	Green and Alpha are included
06h	RA	Red and Alpha are included
07h	A	Alpha is included
08h	RGB	Red, Green, and Blue are included
09h	GB	Green and Blue are included
0Ah	RB	Red and Blue are included
0Bh	B	Blue is included
0Ch	RG	Red and Green are included
0Dh	G	Green is included
0Eh	R	Red is included
0Fh	Reserved	Ignored

Channel Mode Message Descriptor Control Field

MDC_CMODE - Channel Mode Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	0	**Channel Mode** Two modes of channel-enable are provided: a SIMD8 or SIMD16 Dword channel serial view of a register, and a SIMD4x2 view of a register. **Value** | **Name** | **Description** 0 | Oword | All 4 Dwords are read or written if one or more of these channels are enabled 1 | Dword | Each Dword is read or written only if its corresponding channel is enabled.

Clear Color

CLEAR_COLOR - Clear Color

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

DWord	Bit	Description
0 **Programming Notes:** Software shall write the Raw Clear Color channels such that the channel order matches the "SURFACE_STATE.Shader Channel Select" programming.	31:0	**Raw Clear Color : Red** Format: F32 Format: U32 Format: S31
1 **Programming Notes:** Software shall write the Raw Clear Color channels such that the channel order matches the "SURFACE_STATE.Shader Channel Select" programming.	31:0	**Raw Clear Color: Blue** Format: F32 Format: U32 Format: S31
2 **Programming Notes:** Software shall write the Raw Clear Color channels such that the channel order matches the "SURFACE_STATE.Shader Channel Select" programming.	31:0	**Raw Clear Color : Green** Format: F32 Format: U32 Format: S31
3 **Programming Notes:** Software shall write the Raw Clear Color channels such that the channel order matches the "SURFACE_STATE.Shader Channel Select" programming.	31:0	**Raw Clear Color : Alpha** Format: F32 Format: U32 Format: S31
4	31:0	**Converted Clear Color and Clear Depth** This DWORD stores the format converted clear color. If bits per pixel are 32, entire pixel's clear value is stored in this DWORD. If bits per pixel are 64, lower DOWRD is stored in this field. If bits per pixel are 128, this field is not used to store clear value. This field is packed according to the RT format
5	31:0	**Converted Clear Color**

<table>
<tr><th colspan="3">CLEAR_COLOR - Clear Color</th></tr>
<tr><td></td><td></td><td>This DWORD stores the format converted clear color.
If bits per pixel are 64, upper DOWRD is stored in this field
If bits per pixel are 32 or 128, this field is not used to store clear value.
The field is packed according to the RT format</td></tr>
<tr><td rowspan="2">6</td><td>31:1</td><td>Reserved<table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>MBZ</td></tr></table></td></tr>
<tr><td>0</td><td>Color Discard Enable<table><tr><td></td><td></td></tr></table><table><tr><th>Description</th></tr><tr><td>When this bit is set for a Render Target Surface, SW indicates HW that this surface's cachelines from on-chip caches do not need to be written back to memory after a complete Render Pass (aka Tile Pass). This bit applies to both the main and the associated AUX surfaces.</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>This bit must be programmed before binding a surface to a Render Pass (Tile Pass). This bit must not be changed during the Tile Pass.</td></tr></table></td></tr>
<tr><td>7</td><td>31:0</td><td>Reserved<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

Clock Gating Disable Format

Clock Gating Disable Format		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**Clock_Gate_Disable**

Value	Name	Description
0b	Enable	Clock gating controlled by unit logic
1b	Disable	Disable clock gating function

COLOR_CALC_STATE

COLOR_CALC_STATE

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

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

DWord	Bit	Description
0	31:16	**Reserved** Format: MBZ
	15	**Round Disable Function Disable** Format: 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	31:8	**Reserved** Exists If: [Alpha Test Format] == 'ALPHATEST_UNORM8' Format: MBZ
	31:0	**Alpha Reference Value As FLOAT32** 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. **Programming Notes**

COLOR_CALC_STATE		
		This field should not be programmed to NaN.
	7:0	**Alpha Reference Value As UNORM8** Exists If: [Alpha Test Format] == 'ALPHATEST_UNORM8' Format: UNORM8 Upper 24 bits MBZ 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

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

COLOR_PROCESSING_STATE - ACE State

		The value of the y_pixel for point 1 in PWL.
	7:0	**Ymin** 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

COLOR_PROCESSING_STATE - ACE State		
		Format: U8 The value of the y_pixel for point 9 in PWL.
	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**

DWord	Bit	Description
		Default Value: 136 Format: U8 The value of the bias for point 6 in PWL.
	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.

COLOR_PROCESSING_STATE - ACE State		
	15:11	**Reserved** 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. The default is 1024/1024.
	15:11	**Reserved** Format: MBZ

COLOR_PROCESSING_STATE - ACE State

	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 - CSC State

COLOR_PROCESSING_STATE - CSC State		
Source:	BSpec	
Size (in bits):	288	
Default Value:	0x00002000, 0x00000000, 0x00000400, 0x00000000, 0x000004B4, 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

COLOR_PROCESSING_STATE - CSC State		
		Transform coefficient.
	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**

COLOR_PROCESSING_STATE - CSC State		
		Format: MBZ
	19:10	**Offset out 1** 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**

COLOR_PROCESSING_STATE - CSC State

		Format: U1 Enumerated Type **Value** | **Name** | **Description** 0 | | Alpha is taken from message 1 | | Alpha is taken from state
	15:0	**Color Pipe Alpha** Format: U16

COLOR_PROCESSING_STATE - PROCAMP State

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

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 / Name: 0: Output Pixels **[Default]** 1: Output STD Decisions
	1	**STE Enable**

COLOR_PROCESSING_STATE - STD/STE State

DWord	Bit	Description
		Format: Enable
	0	**STD Enable** Format: Enable
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**

COLOR_PROCESSING_STATE - STD/STE State		
		Default Value: 0 Format: S6 2's complement
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**

COLOR_PROCESSING_STATE - STD/STE State

DWord	Bit	Description
		Format: U0.16 1/(2* Skin_types_margin) Value: 20 — Name: **[Default]** — Description: 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** Default Value: 236

COLOR_PROCESSING_STATE - STD/STE State		
		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.
	18:11	**P0U** Default Value: 46 Format: U8

COLOR_PROCESSING_STATE - STD/STE State		
		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 Format: U8 V Bias 3 of the upper part of the detection PWLF.
	7:0	**B2U** Default Value: 200

COLOR_PROCESSING_STATE - STD/STE State

DWord	Bit	Description
		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: 0 — Name: **[Default]** — Description: Disable
	10:0	**S3U** Format: S2.8 2's complement 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.

COLOR_PROCESSING_STATE - STD/STE State

DWord	Bit	Description
	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). 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**

COLOR_PROCESSING_STATE - STD/STE State

DWord	Bit	Description
		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.
	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

COLOR_PROCESSING_STATE - STD/STE State		
		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) 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**

COLOR_PROCESSING_STATE - STD/STE State		
		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.
	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**

COLOR_PROCESSING_STATE - STD/STE State		
		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 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

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.

COLOR_PROCESSING_STATE - TCC State

DWord	Bit	Description
	15:8	**SatFactor4** Default Value: 220 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

<table>
<tr><th colspan="3">COLOR_PROCESSING_STATE - TCC State</th></tr>
<tr><td></td><td></td><td>The calculation result of 1 / (BC2 - BC1) [1/57]</td></tr>
<tr><td>5</td><td>31:16</td><td>Color Transit Slope 45
Default Value: 407
Format: U0.16
The calculation result of 1 / (BC5 - BC4) [1/57]</td></tr>
<tr><td></td><td>15:0</td><td>Color Transit Slope 34
Default Value: 1131
Format: U0.16
The calculation result of 1 / (BC4 - BC3) [1/61]</td></tr>
<tr><td>6</td><td>31:16</td><td>Color Transit Slope 61
Default Value: 377
Format: U0.16
The calculation result of 1 / (BC1 - BC6) [1/62]</td></tr>
<tr><td></td><td>15:0</td><td>Color Transit Slope 56
Default Value: 372
Format: U0.16
The calculation result of 1 / (BC6 - BC5) [1/62]</td></tr>
<tr><td>7</td><td>31:22</td><td>Color Bias 3
Default Value: 0
Format: U2.8
Color bias for BaseColor3.</td></tr>
<tr><td></td><td>21:12</td><td>Color Bias 2
Default Value: 150
Format: U2.8
Color bias for BaseColor2.</td></tr>
<tr><td></td><td>11:2</td><td>Color Bias 1
Default Value: 0
Format: U2.8
Color bias for BaseColor1.</td></tr>
<tr><td></td><td>1:0</td><td>Reserved
Format: MBZ</td></tr>
</table>

<table>
<tr><th colspan="3">COLOR_PROCESSING_STATE - TCC State</th></tr>
<tr><td rowspan="4">8</td><td>31:22</td><td>Color Bias 6
Default Value: 0
Format: U2.8
Color bias for BaseColor6.</td></tr>
<tr><td>21:12</td><td>Color Bias 5
Default Value: 0
Format: U2.8
Color bias for BaseColor5.</td></tr>
<tr><td>11:2</td><td>ColorBias4
Default Value: 0
Format: U2.8
Color bias for BaseColor4.</td></tr>
<tr><td>1:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="7">9</td><td>31</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>30:24</td><td>UV Threshold
Default Value: 3
Format: U7
Low UV threshold.</td></tr>
<tr><td>23:19</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>18:16</td><td>UV Threshold Bits
Default Value: 3
Format: U3
Low UV transition width bits.</td></tr>
<tr><td>15:13</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>12:8</td><td>STE Threshold
Default Value: 0
Format: U5
Skin tone pixels enhancement threshold.</td></tr>
<tr><td>7:3</td><td>Reserved</td></tr>
</table>

COLOR_PROCESSING_STATE - TCC State

DWord	Bit	Description
		Format: MBZ
	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.

Color Calculator State Pointer Message Header Control

MHC_RT_CCSP - Color Calculator State Pointer Message Header Control

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

DWord	Bit	Description
0	31:6	**Color Calculator State Pointer** Format: GeneralStateOffset[31:6] Specifies the 64-byte aligned point to the color calculator state. This pointer is relative to the General State Base Address.
	5:0	**Reserved** Format: Ignore Ignored

Color Code Message Header Control

MHC_RT_CC - Color Code Message Header Control

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

DWord	Bit	Description
0	31:10	**Reserved** Format: Ignore Ignored
	9:8	**Color Code** Format: U2 This ID is assigned by the Windower unit and is used to track synchronizing events. Reserved for HW implementation use
	7:0	**FFTID** Format: U8 This ID is assigned by the fixed function unit and is a unique identifier for the thread. It is used to free up resources used by the thread upon thread completion.

Context Descriptor Format

CONTEXT_DESCRIPTOR - Context Descriptor Format

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

This is the format of context descriptors which make up submitted execlists.

DWord	Bit	Description
0..1	63:32	**Context ID** **Description** Context ID is a unique field assigned by GFX driver when a new context is created by which it is identified across all hierarchies of SW and HW. • Context ID is used for semaphore signaling by hardware and software. • Context ID matching is used by hardware to detect Lite Restore. • Context ID is used by hardware for page fault reporting and response with IOMMU. • Context switch reason and the associated Context ID are reported to Context Switch Status Buffer by hardware on a context switch. Context ID[15:0] (bits[47:32] of the context descriptor) are used for comparing during lite restore, semaphore signaling and context specific OA enabling. Context ID which is a 32 bit field is further divided in to following segments described below: • **Bits[63:61] (Bits 31:29 of Context ID)** represents **Engine class**. • **Bits[60:55] (Bits 28:23 of Context ID)** represents **SW Counter** • **Bit[54] (Bit 22 of Context ID)** – MBZ for SW programming; this bit is used by hardware to distinguish between F&H vs F&S page requests and response messages to and from IOMMU. This bit is used by hardware on receiving page response to properly manage the page fault counters • **Bits[53:48] (Bits 21:16 of Context ID)** represents **Engine Instance** (within a Engine class). • **Bits[47:37] (Bits 15:5 of Context ID)**represents **SW Context ID** which is a software assigned unique context ID. (supports 2048 contexts per virtual function) • **Bits[36:32] (Bits 4:0 of Context ID)** represents **Virtual Function Number** (when virtualization is enabled). Set to zero when virtualization is not enabled.This field contains the bits [4:0] of the Virtual Function Number. **Programming Note:** "**Vitrual Function Number"** must be always programmed to value 0x0. Hardware compares the following fields of the ongoing context to that of the incoming context todetect alite restore. Lite restore is detected when the following fields are equal and the incoming context does not have the "Force Restore" bit set. On a lite restore hadware will only sample the tail pointer from memory (LRCA)and keep executing the ongoing context with out initiating any context switch flows (Flush, Context Save, Context Restore).Lite restore is HW detected context switchoptimizaion transparent to SW, Context Switch Status report and

CONTEXT_DESCRIPTOR - Context Descriptor Format

		Context Switch Interrupt generation happens on a lite restore, Hardware Front End may temporarily get stalled from parsing new commands. • **DW1.SW Context ID** • **DW1.Virtual Function Number** • **DW0.Logical Ring Context Address (LRCA)** • **DW0. Reserved Bits[11:9]** Context ID is reported by hardware to OABUFFER along with the performance statistics counters, Context ID is used for filtering the statistics on per context basis.
	31:12	**Logical Ring Context Address (LRCA)** Format: GraphicsAddress[31:12] This field contains the 4 KB-aligned address of the Logical Ring Context associated with this execlist element. LRCA must be always programmed in GGTT memory.
	11	**Reserved** Format: MBZ
	10:9	**Reserved** Format: MBZ
	7:6	**Fault Handling** Source: CommandStreamer
	5	**Reserved** Format: MBZ
	4:3	**Addressing Mode & Legacy Context**

Fault Handling (7:6) values:

Value	Name	Description
0h	Fault and Hang	Fault model is not supported and fault occurrence is treated as catastrophic. GAM indicates Fault Error to Command streamer. Fault Error interrupt is reported to scheduler. Command Streamer will not initiate context switch on occurrence of Fault Error.
Others	Reserved	Reserved

Programming Notes
When execlist mode is set to "Legacy Context mode" Fault Handling mode must be set to "Fault and Hang." For proper programming for Page Fault modes, refer to memory interface section of the Bspec for the corresponding generation.

<table>
<tr><th colspan="3">CONTEXT_DESCRIPTOR - Context Descriptor Format</th></tr>
<tr><td></td><td></td><td>
<table><tr><td>Format:</td><td>U2</td></tr></table>
Legacy context set indicates GPU is operating in legacy context mode of operation and doesn't support any SVM features. Legacy context reset indicates GPU is operating in advanced context mode of operation and support SVM features. Based on the Context mode set Addressing mode is interpreted appropriately. The table below summarizes the combinations supported.
GFX engine always uses 32b virtual addressing mode when translated using GGTT irrespective of below options.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>01b</td><td>Legacy Context with no 64 bit VA support</td><td>GPU is enabled for legacy context mode of operation and DOESN'T support any SVM features. GPU supports 32b PPGTT graphics virtual addressing. PDP*_DESCRIPTOR contains the base address to 4GB of memory space supported.</td></tr>
<tr><td>11b</td><td>Legacy Context with 64 bit VA support</td><td>GPU is enabled for legacy context mode of operation and DOESN'T support any SVM features. GPU supports 64b (48bit canonical) PPGTT graphics virtual addressing and PDP0_DESCRIPTOR contains the base address to PML4 and other PDP Descriptors are ignored.</td></tr>
<tr><td>Others</td><td>Reserved</td><td>Other values are not supported.</td></tr>
</table>
</td></tr>
<tr><td></td><td>2</td><td>Force Restore
Setting this bit will force a context restore operation when switching to this context even if the LRCA in the CCID register (normally the LRCA of the last context from the prior execlist) matches this one.
Note that it is legal (and likely desirable) for the Render Context Restore Inhibit bit (part of the CTXT_SR_CTL register) in the context image being restored to also be set. The "ring" context is being forced to be restored from a newly initialized context despite a possible LRCA match. However, the render context for such a newly initialized context will likely be uninitialized and so should not be restored.</td></tr>
<tr><td></td><td>1</td><td>Reserved
<table><tr><td></td><td></td></tr></table>
</td></tr>
<tr><td></td><td>0</td><td>Valid
Set if this register holds a valid context descriptor. SW should set this bit in the Element registers that it has set up to contain valid context descriptors. Any execlist elements that are not used in a submitted execlist must have this bit clear.</td></tr>
</table>

Context Status

CONTEXT_STATUS - Context Status

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

DWord	Bit	Description
0	63:32	**Context ID** Format: U32 **Description** The format of Context ID (sub-fields) described in "Context Descriptor Format". On “IDLE to ACTIVE” context switch status report, engine populates the “Engine Class” and “Engine Instance” fields of the “Context ID” with its corresponding engine class and engine instance values. Bits[63:61] of context status (Bits 31:29 of Context ID) represents Engine class and bits[53:48] of context status (Bits 21:16 of Context ID) represents Engine Instance (within an Engine class).
	31:30	**Reserved** Format: MBZ
	29	**Preempt To Idle** "Preempt to Idle" request from SQ load has resulted in context switch.
	28	**POSH Context Complete** States the state of the POSH pipeline on a context switch by render pipe. When set indicates POSH pipe has completely processed (Head eqv to Tail)thesubmitted workload. This bit will be set for POSH disabled contexts.
	27:25	**Reserved** Format: MBZ
	24:20	**Display Plane Unified** This indicates the display plane for which Wait on Scanline/V-Blank/Sync Flip has been executed leading to context switch. This field is only valid when one of the "Wait on Scanline" or "Wait on Vblank" or "Wait on sync Flip" is set.

Value	Name	Exists If
0h	Display Plane-1	[Wait on V-blank]==0 AND [Wait on Scanline]==0
0h	Display Pipe-A	[Wait on V-blank]==1 OR [Wait on Scanline]==1
1h	Display Plane-2	[Wait on V-blank]==0 AND [Wait on Scanline]==0

CONTEXT_STATUS - Context Status

1h	Display Pipe-B	[Wait on V-blank]==1 OR [Wait on Scanline]==1
2h	Display Plane-3	[Wait on V-blank]==0 AND [Wait on Scanline]==0
2h	Display Pipe-C	[Wait on V-blank]==1 OR [Wait on Scanline]==1
3h	Display Plane-4	[Wait on V-blank]==0 AND [Wait on Scanline]==0
3h	Display Pipe-D	[Wait on V-blank]==1 OR [Wait on Scanline]==1
4h	Display Plane-5	
5h	Display Plane-6	
6h	Display Plane-7	
7h	Display Plane-8	
8h	Display Plane-9	
9h	Display Plane-10	
Ah	Display Plane-11	
Bh	Display Plane-12	
Ch	Display Plane-13	
Dh	Display Plane-14	
Eh	Display Plane-15	
Fh	Display Plane-16	
10h	Display Plane-17	
11h	Display Plane-18	
12h	Display Plane-19	
13h	Display Plane-20	
14h	Display Plane-21	
15h	Display Plane-22	
16h	Display Plane-23	
17h	Display Plane-24	
18h	Display Plane-25	
19h	Display Plane-26	
1Ah	Display Plane-27	
1Bh	Display Plane-28	
1Ch	Display Plane-29	
1Dh	Display Plane-30	
1Eh	Display Plane-31	
1Fh	Display Plane-32	

19:16 **Display Plane**

This indicates the display plane for which Wait on Scanline/V-Blank/Sync Flip has been executed leading to context switch. This field is only valid when one of the "Wait on Scanline" or "Wait on

CONTEXT_STATUS - Context Status

		Vblnak" or "Wait on sync Flip" is set.

Value	Name
0h	Reserved (Look at field 14:12)
1h	Reserved
2h	Reserved
3h	Display Plane-7
4h	Display Plane-8
5h	Display Plane-9
6h	Display Plane-10
7h	Display Plane-11
8h	Display Plane-12
[9h, Fh]	Reserved

15 **Lite Restore**

Format:	Enable

This bit is only valid only when Preempted bit is set. When set, this bit indicates that a given context got preempted with the same context resulting in Lite Restore in HW.

14:12 **Display Plane Additional**

This indicates the display plane for which Wait on Scanline/V-Blank/Sync Flip has been executed leading to context switch. This field is only valid when one of the "Wait on Scanline" or "Wait on Vblnak" or "Wait on sync Flip" is set and when "Display Plane" is 0

Value	Name	Exists If
0h	Display Plane-1	[Wait on V-blank]==0
0h	Display Pipe-A	[Wait on V-blank]==1
1h	Display Plane-2	[Wait on V-blank]==0
1h	Display Pipe-B	[Wait on V-blank]==1
2h	Display Plane-3	[Wait on V-blank]==0
2h	Display Pipe-C	[Wait on V-blank]==1
3h	Display Plane-4	
4h	Display Plane-5	
5h	Display Plane-6	

11 **Semaphore Wait Mode**

Value	Name
0h	Signal Mode
1h	Poll Mode

10:9 **Reserved**

CONTEXT_STATUS - Context Status

		Format: MBZ
	8	**Wait on Scanline**
	7	**Wait on Semaphore**
	6	**Wait on V-blank**
	5	**Wait on Sync Flip**
	4	**Context Complete** Element is completely processed (Head eqv to Tail) and resulted in a context switch.
	3	**ACTIVE to IDLE** Following this context switch there is no active element available in HW to execute
	2	**Element Switch** Context Switch happened from first element in the current execlist to the second element of the same execlist
	1	**Preempted** Submission of a new execlist has resulted in context switch. The switch is from element in current execlist to element in pending execlist
	0	**IDLE to ACTIVE** **Description** Execlist submitted when HW is IDLE. When this bit is set rest of the fields in CSQ are not valid with exception to Context ID. On "IDLE to ACTIVE" context switch status report, engine populates the "Engine Class" and "Engine Instance" fields of the "Context ID" with its corresponding engine class and engine instance values. Bits[63:61] of context status (Bits 31:29 of Context ID) represents Engine class and bits[53:48] of context status (Bits 21:16 of Context ID) represents Engine Instance (within an Engine class).

CSC COEFFICIENT FORMAT

<table>
<tr><th colspan="3">CSC COEFFICIENT FORMAT</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 16
Default Value: 0x00000000</td></tr>
<tr><td colspan="3">Coefficients for the CSC are stored in sign-exponent-mantissa format. Two CSC coefficients are stored in each dword, the table below show the data packing in each dword.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>15</td><td>Sign
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0b</td><td>Positive</td></tr>
<tr><td>1b</td><td>Negative</td></tr>
</table></td></tr>
<tr><td>14:12</td><td>Exponent_bits
Represented as 2^(-n)
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>110b</td><td>4</td><td>4 or mantissa is bb.bbbbbbb</td></tr>
<tr><td>111b</td><td>2</td><td>2 or mantissa is b.bbbbbbbb</td></tr>
<tr><td>000b</td><td>1</td><td>1 or mantissa is 0.bbbbbbbbb</td></tr>
<tr><td>001b</td><td>0.5</td><td>0.5 or mantissa is 0.0bbbbbbbbb</td></tr>
<tr><td>010b</td><td>0.25</td><td>0.25 or mantissa is 0.00bbbbbbbbb</td></tr>
<tr><td>011b</td><td>0.125</td><td>0.125 or mantissa is 0.000bbbbbbbbb</td></tr>
<tr><td>Others</td><td>Reserved</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>11:3</td><td>Mantissa</td></tr>
<tr><td>2:0</td><td>Reserved</td></tr>
</table>

Data Port 0 Message Types

MT_DP0 - Data Port 0 Message Types

Source:	EuSubFunctionDataPort0
Size (in bits):	5
Default Value:	0x00000000

Lists all the Message Types in a Data Port 0 Message Descriptor [18:14]. The Legacy messages are encoded in Data Port 0 with Bit 18 set to zero. The Message Header is optional for many (but not all) of these operations. The Scratch Block messages are encoded in Data Port 0 with Bit 18 set to one. A Message Header is required.

DWord	Bit	Description
0	4	**Legacy DAP-DC Message** Legacy Message
	3:0	**Message Type** Specifies type of message

Bit 4 values:

Value	Name	Description
0h	No **[Default]**	Legacy DAP-DC Message
1h	Reserved	Scratch Block Message, descriptor uses different Message Type encoding

Bits 3:0 values:

Value	Name	Description
00h	MT0R_B **[Default]**	Block Read message
01h	MT0R_AB	Aligned Block Read message
03h	MT0R_DWS	Dword Scattered Read message
04h	MT0R_BS	Byte Scattered Read message
07h	MT0_MEMFENCE	Memory Fence message
08h	MT0W_B	Block Write message
0Bh	MT0W_DWS	Dword Scattered Write message
0Ch	MT0W_BS	Byte Scattered Write message
Others	Reserved	Ignored

Data Port 1 Message Types

MT_DP1 - Data Port 1 Message Types		
Source:	EuSubFunctionDataPort1	
Size (in bits):	5	
Default Value:	0x00000000	

Lists all the Message Types in a Data Port 1 Message Descriptor [18:14]. Most surface and atomic operations, both typed and untyped, are encoded on Data Port 1. The Message Header is optional for many (but not all) of these operations. Most A64 Stateless operations are also encoded on Data Port 1. The Message Header is forbidden for all A64 messages on Data Port 1.

DWord	Bit	Description
0	4:0	**Message Type** Specifies type of message

Value	Name	Description
00h	MT1R_T	Transpose Read message
01h	MT1R_US	Untyped Surface Read message
02h	MT1A_UI	Untyped Atomic Integer Operation message
04h	MT1R_MB	Media Block Read message
05h	MT1R_TS	Typed Surface Read message
06h	MT1A_TA	Typed Atomic Integer Operation message
08h	Reserved	Ignored
09h	MT1W_US	Untyped Surface Write mesage
0Ah	MT1W_MB	Media Block Write message
0Bh	MT1A_TC	Typed Atomic Counter Operation message
0Dh	MT1W_TS	Typed Surface Write message
0Eh	Reserved	Ignored
10h	MT1R_A64_SB	A64 Scattered Read message
11h	MT1R_A64_US	A64 Untyped Surface Read message
12h	MT1A_A64_UI	A64 Untyped Atomic Integer Operation message
14h	MT1R_A64_B	A64 Block Read message
15h	MT1W_A64_B	A64 Block Write message
18h	Reserved	Ignored
19h	MT1W_A64_US	A64 Untyped Surface Write message
1Ah	MT1W_A64_SB	A64 Scattered Write message
1Bh	MT1A_UF	Untyped Atomic Float Operation message
1Dh	MT1A_A64_UF	A64 Untyped Atomic Float Operation message
Others	Reserved	Ignored

Data Port 2 Extended Message Descriptor

DP2_EXTDESC - Data Port 2 Extended Message Descriptor		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:16	**Sideband Offset** Format: U16 Specifies the 16-bit offset from the Sideband added to all the offsets in the Address Payload for DP2 messages.
	15:11	**Reserved** Format: MBZ Ignored
	10:0	**Execution Unit Extended Message Descriptor Definition** Format: Execution_Unit_Extended_Message_Descriptor EU uses this information as part of the SEND instruction.

Data Port 2 Message Types

MT_DP2 - Data Port 2 Message Types		
Source:	EuSubFunctionDataPort2	
Size (in bits):	5	
Default Value:	0x00000002	

Lists all the Message Types in a Data Port 2 Message Descriptor [18:14]. Scaled operations are on Data Port 2. They provide a pitch-scaled data address calculation for SLM Stateless address models. The Message Header is forbidden for SLM operations.

DWord	Bit	Description
0	4:1	**Message Type** Specifies type of message
	0	**Reserved** Format: MBZ Ignored

Message Type values (bits 4:1):

Value	Name	Description
01h	MT2R_US **[Default]**	Untyped Surface Read message
04h	MT2R_BS	Byte Scattered Read message
09h	MT2W_US	Untyped Surface Write message
0Ch	MT2W_BS	Byte Scattered Write message
Others	Reserved	Ignored

Data Port Bindless Surface Extended Message Descriptor

DP_EXTDESC_BTI252 - Data Port Bindless Surface Extended Message Descriptor

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

DWord	Bit	Description
0	31:12	**Bindless Surface Offset** Format: BindlessSurfaceOffset[25:6] Specifies the bindless surface offset if the Binding Table Index is set to 252. Ignored otherwise. The bindless surface offset is added to the Bindless Surface Base Address as bits 25:6 of the byte-based address. The resulting address is the location of SURFACE_STATE for this message.
	11	**Reserved**
	10:0	**Execution Unit Extended Message Descriptor Definition** Format: Execution_Unit_Extended_Message_Descriptor EU uses this information as part of the SEND instruction.

Data Size Message Descriptor Control Field

MDC_DS - Data Size Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	1:0	**Data Size** Specifies the number of Bytes to be read or written

Value	Name	Description
00h	B	1 Byte
01h	W	2 Bytes
02h	DW	4 Bytes
03h	Reserved	Reserved

Depth Clear Value Format

Depth Clear Value Format		
Source: BSpec		
Size (in bits): 32		
Default Value: 0x00000000		
DWord	**Bit**	**Description**
0 This field defines the clear value that will be applied to the depth buffer if the Depth Buffer Clear field is enabled. It is valid only if Depth Buffer Clear Value Valid is set. **Programming Notes:** The clear value must be between the min and max depth values (inclusive) defined in the CC_VIEWPORT. If the depth buffer format is D32_FLOAT, then values must be limited to the range of +0.0f and 1.0f inclusive; values outside this range are reserved	31:0	**Depth Clear Value** Format: IEEE_FLOAT

Deptrh Clear Value Format

STRUCTURE_TEMPLATE - Deptrh Clear Value Format		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:0	**Address1** Format: IEEE_FLOAT32 Format: UNORM24 Format: UNORM16 When this field contains 24-bit UNORM, the upper 8-bits are reserved (0's) When this field contains 16-bit UNORM the upper 16-bits are reserved (0's)

Display Engine Render Response Message Definition

Display Engine Render Response Message Definition	
Source:	BSpec
Size (in bits):	96
Default Value:	0x00000000, 0x00000000, 0x00000000

The Display Engine Render Response Registers use bit definitions from this table.

Programming Notes
Some events can be sent to CS (Render Command Streamer) or BCS (Blitter Command Streamer). For render response messages sending flip done or scanline events, the destination, CS or BCS, is selected depending on the initiator of the flip or the load scanline command. For render response messages sending vertical blank events, the destinations, CS or BCS, or both CS and BCS,is selected depending on the DE_RR_DEST setting. Command Streamer Plane number to the Display Plane name mapping is available in the **Display Plane Capability and Interoperability** section.
The STEREO3D_EVENT_MASK selects between left eye and right eye reporting of vertical blank and scanline events in stereo 3D modes.

DWord	Bit	Description
0	31	**Spare 31**
	30	**Reserved**
	29	**Reserved**
	28	**Spare 28**
	27	**Spare 27**
	26	**Spare 26**
	25	**Spare 25**
	24	**Spare 24**
	23	**Spare 23**
	22	**Reserved**
	21	**Pipe_C_Start_of_Vertical_Blank_Event** This event is reported on the start of the vertical blank of the transcoder attached to Pipe C.
	20	**Plane_6_Flip_Done_Event** This event is reported on the completion of a flip for Plane 6.
	19	**Plane_12_Flip_Done_Event**

Display Engine Render Response Message Definition		
		This event is reported on the completion of a flip for Plane 12.
	18	**Plane_11_Flip_Done_Event** This event is reported on the completion of a flip for Plane 11.
	17	**Plane_10_Flip_Done_Event** This event is reported on the completion of a flip for Plane 10.
	16	**Plane_9_Flip_Done_Event** This event is reported on the completion of a flip for Plane 9.
	15	**Plane_3_Flip_Done_Event** This event is reported on the completion of a flip for Plane 3.
	14	**Pipe_C_Scanline_Event** This event is reported on the start of the selected scan line for the transcoder attached to Pipe C.
	13	**Reserved**
	12	**Spare 12** Unused
	11	**Pipe_B_Start_of_Vertical_Blank_Event** This event is reported on the start of the vertical blank of the transcoder attached to Pipe B.
	10	**Plane_5_Flip_Done_Event** This event is reported on the completion of a flip for Plane 5.
	9	**Plane_2_Flip_Done_Event** This event is reported on the completion of a flip for Plane 2.
	8	**Pipe_B_Scanline_Event** This event is reported on the start of the selected scan line for the transcoder attached to Pipe B.
	7	**Plane_8_Flip_Done_Event** This event is reported on the completion of a flip for Plane 8.
	6	**Plane_7_Flip_Done_Event** This event is reported on the completion of a flip for Plane 7.
	5	**Reserved**
	4	**Spare 4** Unused
	3	**Pipe_A_Start_of_Vertical_Blank_Event** This event is reported on the start of the vertical blank of the transcoder attached to Pipe A.
	2	**Plane_4_Flip_Done_Event** This event is reported on the completion of a flip for Plane 4.
	1	**Plane_1_Flip_Done_Event** This event is reported on the completion of a flip for Plane 1.
	0	**Pipe_A_Scanline_Event** This event is reported on the start of the selected scan line for the transcoder attached to Pipe A.
1	31	**Spare 31**

Display Engine Render Response Message Definition		
		Unused.
	30	**Spare 30** Unused.
	29	**Spare 29** Unused.
	28	**Spare 28** Unused
	27	**Spare 27** Unused
	26	**Spare 26** Unused
	25	**Spare 25** Unused
	24	**Spare 24** Unused
	23	**Spare 23** Unused
	22	**Spare 22** Unused
	21	**Spare 21** Unused
	20	**Spare 20** Unused
	19	**Spare 19** Unused
	18	**Spare 18** Unused
	17	**Spare 17** Unused
	16	**Spare 16** Unused
	15	**Spare 15** Unused
	14	**Spare 14** Unused
	13	**Spare 13** Unused
	12	**Spare 12** Unused
	11	**Spare 11** Unused

Display Engine Render Response Message Definition		
	10	**Spare 10** Unused
	9	**Spare 9** Unused
	8	**Spare 8** Unused
	7	**Spare 7** Unused
	6	**Spare 6** Unused
	5	**Spare 5** Unused
	4	**Spare 4** Unused
	3	**Spare 3** Unused
	2	**Reserved**
	1	**Pipe_D_Scanline_Event** This event is reported on the start of the selected scan line for the transcoder attached to Pipe D. Some SKUs may not have Pipe D.
	0	**Pipe_D_Start_of_Vertical_Blank_Event** This event is reported on the start of the vertical blank of the transcoder attached to Pipe D. Some SKUs may not have Pipe D.
2	31	**Spare 31** Unused.
	30	**Spare 30** Unused.
	29	**Spare 29** Unused.
	28	**Spare 28** Unused
	27	**Spare 27** Unused
	26	**Spare 26** Unused
	25	**Spare 25** Unused
	24	**Spare 24** Unused
	23	**Spare 23** Unused

Display Engine Render Response Message Definition		
	22	**Spare 22** Unused
	21	**Spare 21** Unused
	20	**Spare 20** Unused
	19	**Plane_32_Flip_Done_Event** This event is reported on the completion of a flip for Plane 32.
	18	**Plane_31_Flip_Done_Event** This event is reported on the completion of a flip for Plane 31.
	17	**Plane_30_Flip_Done_Event** This event is reported on the completion of a flip for Plane 30.
	16	**Plane_29_Flip_Done_Event** This event is reported on the completion of a flip for Plane 29.
	15	**Plane_28_Flip_Done_Event** This event is reported on the completion of a flip for Plane 28.
	14	**Plane_27_Flip_Done_Event** This event is reported on the completion of a flip for Plane 27.
	13	**Plane_26_Flip_Done_Event** This event is reported on the completion of a flip for Plane 26.
	12	**Plane_25_Flip_Done_Event** This event is reported on the completion of a flip for Plane 25.
	11	**Plane_24_Flip_Done_Event** This event is reported on the completion of a flip for Plane 24.
	10	**Plane_23_Flip_Done_Event** This event is reported on the completion of a flip for Plane 23.
	9	**Plane_22_Flip_Done_Event** This event is reported on the completion of a flip for Plane 22.
	8	**Plane_21_Flip_Done_Event** This event is reported on the completion of a flip for Plane 21.
	7	**Plane_20_Flip_Done_Event** This event is reported on the completion of a flip for Plane 20.
	6	**Plane_19_Flip_Done_Event** This event is reported on the completion of a flip for Plane 19.
	5	**Plane_18_Flip_Done_Event** This event is reported on the completion of a flip for Plane 18.
	4	**Plane_17_Flip_Done_Event** This event is reported on the completion of a flip for Plane 17.
	3	**Plane_16_Flip_Done_Event** This event is reported on the completion of a flip for Plane 16.
	2	**Plane_15_Flip_Done_Event**

Display Engine Render Response Message Definition		
		This event is reported on the completion of a flip for Plane 15.
	1	**Plane_14_Flip_Done_Event** This event is reported on the completion of a flip for Plane 14.
	0	**Plane_13_Flip_Done_Event** This event is reported on the completion of a flip for Plane 13.

DstRegNum

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

Description

Register Number The register number for the operand. For a GRF register, is the part of a register address that aligns to a 256-bit (32-byte) boundary. For an ARF register, this field is encoded such that MSBs identify the architecture register type and LSBs provide the register number. An ARF register can only be dst or src0. Any src1 or src2 operands cannot be ARF registers. RegNum and SubRegNum together provide the byte-aligned address for the origin of a register region. RegNum provides bits 12:5 of that address. For one-source and two-source instructions, SubregNum provides bits 4:0. For three-source instructions, the address must be DWord-aligned; SubRegNum provides bits 4:2 of the address and bits 1:0 are zero. This field is present for the direct addressing mode and not present for indirect addressing. This field applies to both source and destination operands.

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 Execution Environment chapter for details.

DstSubRegNum

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

Description
Subregister Number The subregister number for the operand. For a GRF register, is the byte address within a 256-bit (32-byte) register. For an ARF register, determines the sub-register number according to the specified encoding for the given architecture register. RegNum and SubRegNum together provide the byte-aligned address for the origin of a GRF register region. RegNum provides bits 12:5 of that address. For one-source and two-source instructions, SubregNum provides bits 4:0. For three-source instructions, the address must be QWord-aligned; SubRegNum provides bits 4:3 of the address and bits 2:0 are zero.

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 Execution Environment chapter for details.

DUALSUBSLICE_HASH_TABLE_8x8

DUALSUBSLICE_HASH_TABLE_8x8		
Source:	BSpec	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	

8x8 [Y][X] dualsubslice hashing table. Each entry is a single bit that indicates which dualSubSlice(DSS) the indicated xy location maps to. A value of 0 indicates the larger DSS, or DSS=0 if both DSS have are balanced(have same number of enabled lsubslices)

DWord	Bit	Description
0	31:24	**SubSlice Hashing Table Entries[3]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=3 and x=7..0
	23:16	**SubSlice Hashing Table Entries[2]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=2 and x=7..0
	15:8	**SubSlice Hashing Table Entries[1]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=1 and x=7..0
	7:0	**SubSlice Hashing Table Entries[0]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=0 and x=7..0
1	31:24	**SubSlice Hashing Table Entries[7]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=7 and x=7..0
	23:16	**SubSlice Hashing Table Entries[6]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=6 and x=7..0
	15:8	**SubSlice Hashing Table Entries[5]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=5 and x=7..0
	7:0	**SubSlice Hashing Table Entries[4]x[7:0]** Format: U8 Indicates the dualsubslice_id for the pixel block that has y=4 and x=7..0

DUALSUBSLICE_HASH_TABLE_16x8

DUALSUBSLICE_HASH_TABLE_16x8		
Source:	BSpec	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000	

16x8 [Y][X] dualsubslice hashing table. Each entry is a single bit that indicates which dualSubSlice(DSS) the indicated xy location maps to. A value of 0 indicates the larger DSS, or DSS=0 if both DSS have are balanced(have same number of enabled lsubslices)

DWord	Bit	Description
0	31:16	**SubSlice Hashing Table Entries y[1]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=1 and x=15..0
	15:0	**SubSlice Hashing Table Entries y[0]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=0 and x=15..0
1	31:16	**SubSlice Hashing Table Entries y[3]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=3 and x=15..0
	15:0	**SubSlice Hashing Table Entries y[2]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=2 and x=15..0
2	31:16	**SubSlice Hashing Table Entries y[5]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=5 and x=15..0
	15:0	**SubSlice Hashing Table Entries y[4]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=4 and x=15..0
3	31:16	**SubSlice Hashing Table Entries y[7]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=7 and x=15..0
	15:0	**SubSlice Hashing Table Entries y[6]x[15:0]** Format: U16 Indicates the dualsubslice_id for the pixel block that has y=6 and x=15..0

Dword Data Payload Register

MDCR_DW - Dword Data Payload Register		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0	31:0	**Dword0** Format: U32 Specifies the slot 0 data in this payload register
0.1	31:0	**Dword1** Format: U32 Specifies the slot 1 data in this payload register
0.2	31:0	**Dword2** Format: U32 Specifies the slot 2 data in this payload register
0.3	31:0	**Dword3** Format: U32 Specifies the slot 3 data in this payload register
0.4	31:0	**Dword4** Format: U32 Specifies the slot 4 data in this payload register
0.5	31:0	**Dword5** Format: U32 Specifies the slot 5 data in this payload register
0.6	31:0	**Dword6**

MDCR_DW - Dword Data Payload Register		
		Format: U32 Specifies the slot 6 data in this payload register
0.7	31:0	**Dword7** Format: U32 Specifies the slot 7 data in this payload register

Dword SIMD8 Atomic Operation CMPWR Message Data Payload

MDP_AOP8_DW2 - Dword SIMD8 Atomic Operation CMPWR Message 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	**Src0** Format: MDCR_DW Specifies the Slot [7:0] Source 0 data
1.0-1.7	255:0	**Src1** Format: MDCR_DW Specifies the Slot [7:0] Source 1 data

Dword SIMD8 Data Payload

MDP_DW_SIMD8 - Dword SIMD8 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	**Data[7:0]** Format: MDCR_DW Specifies the Slot [7:0] data

Dword SIMD16 Atomic Operation CMPWR Message Data Payload

MDP_AOP16_DW2 - Dword SIMD16 Atomic Operation CMPWR Message Data Payload	
Source:	BSpec
Size (in bits):	1024
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Src0[7:0]** Format: MDCR_DW Specifies the Source 0 data for Slot [7:0]
1.0-1.7	255:0	**Src0[15:8]** Format: MDCR_DW Specifies the Source 0 data for Slot [15:8]
2.0-2.7	255:0	**Src1[7:0]** Format: MDCR_DW Specifies the Source 1 data for Slot [7:0]
3.0-3.7	255:0	**Src1[15:8]** Format: MDCR_DW Specifies the Source 1 data for Slot [15:8]

Dword SIMD16 Data Payload

<table>
<tr><th colspan="3">MDP_DW_SIMD16 - Dword SIMD16 Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 512
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Data[7:0]<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDCR_DW</td></tr></table>Specifies the Slot [7:0] data</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Data[15:8]<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDCR_DW</td></tr></table>Specifies the Slot [15:8] data</td></tr>
</table>

Encoder Base Address Parameters1

Encoder Base Address Parameters1

Source:	BSpec
Size (in bits):	320
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

Please note that DW0-9, correspond to DW10-19 of **WiGig Parameters**.

DWord	Bit	Description
0	31:0	**Reserved** Format: MBZ
1	31:0	**Reserved** Format: MBZ
2	31:0	**Reserved** Format: MBZ
3	31:12	**Display Buffer Surface Base Address[31:12]** Specifies the 4K byte aligned video shared buffer address for display engine to deliver display frame data.
	11:0	**Reserved** Format: MBZ
4	31:16	**Reserved** Format: MBZ
	15:0	**Display Buffer Surface Base Address [47:32]** This field is for the upper range of Display Buffer Surface Base Address.
5	31:15	**Reserved** Format: MBZ
	14:9	**Reserved** Format: MBZ
	8:7	**Arbitration Priority Control for Display Buffer Surface Base Address** This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface. Value / Name: 00b: Highest priority 01b: Second highest priority 10b: Third highest priority 11b: Lowest priority **Programming Notes**

Encoder Base Address Parameters1

		consistent with rest of media.;
	6:5	**Memory Type: LLC/eLLC Cache ability Control (LeLLCCC) for Display Buffer Surface Base Address** This is the field used in GT interface block to determine what type of access need to be generated to uncore. For the cases where the LeLLCCC is set, cacheable transaction are generated to enable LLC usage for particular stream.
	4:3	**Target Cache (TC) for Display Buffer Surface Base Address** This field allows the choice of LLC vs eLLC for caching
	2	**Reserved**
	1:0	**Age for QUADLRU (AGE) for Display Buffer Surface Base Address** This field allows the selection of AGE parameter for a given surface in LLC or eLLC. . If a particular allocation is done at youngest age ("3") it tends to stay longer in the cache as compared to older age allocations ("2", "1", or "0"). This option is given to driver to be able to decide which surfaces are more likely to generate HITs, hence need to be replaced least often in caches.

Bits 6:5 values:

Value	Name	Description
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	
01b	UC	Uncacheable - non-cacheable
10b	WT	Writethrough
11b	WB	Writeback

Programming Notes
This field should be consistant with display capture surface Cachebility.

Bits 4:3 values:

Value	Name
00b	eLLC Only - not snooped in GT
01b	LLC Only
10b	LLC/eLLC Allowed
11b	L3, LLC, eLLC Allowed

Programming Notes
This field should be consistant with display capture surface Cachebility.

Bits 1:0 values:

Value	Name
11b	Good chance of generating hits.
10b	Next good chance of generating hits
01b	Decent chance of generating hits
00b	Poor chance of generating hits

Programming Notes

<table>
<tr><th colspan="3">Encoder Base Address Parameters1</th></tr>
<tr><td></td><td></td><td>This field can be set to 00.</td></tr>
<tr><td rowspan="2">6</td><td>31:12</td><td>Destination TFD Surface Base Address
A.k.A GFX_WNIC_SHARED_DATABUFFER_BASE_ADDRESS, this field specifies the 4K byte aligned shared data buffer for WDBOX to transfer AV mux TS data to WNIC device.</td></tr>
<tr><td>11:0</td><td>Reserved
Format: | MBZ</td></tr>
<tr><td rowspan="2">7</td><td>31:16</td><td>Reserved
Format: | MBZ</td></tr>
<tr><td>15:0</td><td>Destination TFD Surface Base Address [47:32]
This field is for the upper range of Destination TFD Surface Base Address.</td></tr>
<tr><td rowspan="4">8</td><td>31:15</td><td>Reserved
Format: | MBZ</td></tr>
<tr><td>14:9</td><td>Reserved
Format: | MBZ</td></tr>
<tr><td>8:7</td><td>Arbitration Priority Control for Destination TFD Surface Base Address
This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>00b</td><td>Highest priority</td></tr>
<tr><td>01b</td><td>Second highest priority</td></tr>
<tr><td>10b</td><td>Third highest priority</td></tr>
<tr><td>11b</td><td>Lowest priority</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>consistent with rest of media.;</td></tr>
</table></td></tr>
<tr><td>6:5</td><td>Memory Type: LLC/eLLC Cache ability Control (LeLLCCC) for Destination TFD Surface Base Address
This is the field used in GT interface block to determine what type of access need to be generated to uncore. For the cases where the LeLLCCC is set, cacheable transaction are generated to enable LLC usage for particular stream.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>00b</td><td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td><td></td></tr>
<tr><td>01b</td><td>UC</td><td>Uncacheable - non-cacheable</td></tr>
<tr><td>10b</td><td>WT</td><td>Writethrough</td></tr>
<tr><td>11b</td><td>WB</td><td>Writeback</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td></td></tr>
</table></td></tr>
</table>

Encoder Base Address Parameters1		
	4:3	**Target Cache (TC) for Destination TFD Surface Base Address** This field allows the choice of LLC vs eLLC for caching <table><tr><th>Value</th><th>Name</th></tr><tr><td>00b</td><td>eLLC Only - not snooped in GT</td></tr><tr><td>01b</td><td>LLC Only</td></tr><tr><td>10b</td><td>LLC/eLLC Allowed</td></tr><tr><td>11b</td><td>L3, LLC, eLLC Allowed</td></tr></table>
	2	**Reserved**
	1:0	**Age for QUADLRU (AGE) for Destination TFD Surface Base Address** This field allows the selection of AGE parameter for a given surface in LLC or eLLC. . If a particular allocation is done at youngest age ("3") it tends to stay longer in the cache as compared to older age allocations ("2", "1", or "0"). This option is given to driver to be able to decide which surfaces are more likely to generate HITs, hence need to be replaced least often in caches. <table><tr><th>Value</th><th>Name</th></tr><tr><td>11b</td><td>Good chance of generating hits.</td></tr><tr><td>10b</td><td>Next good chance of generating hits</td></tr><tr><td>01b</td><td>Decent chance of generating hits</td></tr><tr><td>00b</td><td>Poor chance of generating hits</td></tr></table>
9	31:0	**Reserved** <table><tr><td>Format:</td><td>MBZ</td></tr></table>

Encoder Base Address Parameters2

<table>
<tr><th colspan="3">Encoder Base Address Parameters2</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 320
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">Please note that DW0-9, correspond to DW20-29 of WiGig Parameters.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>1</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>2</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>3</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>4</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="2">5</td><td>31:12</td><td>Even Reconstructed pixel Reference Surface Base Address
Specifies the 4K byte aligned frame buffer address for outputting the reconstructed YUV picture. This field is ignored if I-frame only mode is set to 0 (disable).</td></tr>
<tr><td>11:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="2">6</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:0</td><td>Even Reconstructed pixel Reference Surface Base Address [47:32]
This field is for the upper range of Even Reconstructed pixel Reference Surface Base Address.</td></tr>
<tr><td rowspan="3">7</td><td>31:11</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>10</td><td>Memory Compression Mode
This distinguishes vertical from horizontal compression.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0h</td><td>Horizontal Compression Mode</td></tr><tr><td>1h</td><td>Vertical Compression Mode</td></tr></table></td></tr>
<tr><td>9</td><td>Memory Compression Enable
Format: Enable
If enabled, memory compression will be attempted on this surface.</td></tr>
</table>

Encoder Base Address Parameters2

	8:7	**Reserved** Format: MBZ
	6:1	**Even Reconstructed Pixel Reference Surface - 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**
8	31:12	**ODD Reconstructed pixel Reference Surface address** Specifies the 4K byte aligned frame buffer address for outputting the reconstructed YUV picture. **Programming Notes** This field is ignored if I-frame only mode is set to 0 (disable).
	11:0	**Reserved** Format: MBZ
9	31:16	**Reserved** Format: MBZ
	15:0	**ODD Reconstructed pixel Reference Surface Base Address [47:32]** This field is for the upper range of ODD Reconstructed pixel Reference Surface Base Address.

Encoder Base Address Parameters3

Encoder Base Address Parameters3		
Source:	BSpec	
Size (in bits):	320	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
Please note that DW0-9, correspond to DW30-39 of **WiGig Parameters**.		
DWord	**Bit**	**Description**
0	31:11	**Reserved** Format: MBZ
	10	**Memory Compression Mode** This distinguishes vertical from horizontal compression. **Value** / **Name** 0h: Horizontal Compression Mode 1h: Vertical Compression Mode
	9	**Memory Compression Enable** Format: Enable If enabled, memory compression will be attempted on this surface.
	8:7	**Reserved** Format: MBZ
	6:1	**Odd Reconstructed Pixel Reference Surface - 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**
1	31:2	**Reserved** Format: MBZ [17:4] WidthMinus1- EVEN/ODD reconstructed pixel reference surfaces 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. [3:2] Reserved
	1:0	**Cr(V)/Cb(U) Pixel Offset V Direction- EVEN/ODD Reconstructed Pixel Reference Surfaces**

Encoder Base Address Parameters3

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.

DWord 2, Bits 31:21 — Reserved

Format:	MBZ

DWord 2, Bits 20:3 — Surface Pitch Minus1 - EVEN/ODD Reconstructed Pixel Reference Surfaces

Format:	U18-1 Pitch in (Bytes - 1)

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

Value	Name	Description	Exists If
[0,2047]		For Pitch of [1B, 2048B]	//[Surfaces Type] = SURFTYPE_BUFFER
[0, 262143]		For Pitch of [1B, 256KB]	//[Surfaces Type] = Linear Surface
[511, 262143]		For Pitch of [512B, 256KB]=[1tile, 512 tiles]	//[Surfaces Type] = X-tiled
[127, 262143]		For Pitch of [128B, 256KB]=[1tile, 2048 tiles]	//[Surfaces Type] = Y-tiles

Programming Notes

If Media Memory Compression is enabled, the following max pitch size restriction must be honored. For larger resolution, Media Memory compression Must be disabled.

Tiling Mode	Pixel Format	Max Frame Width (bytes)	Max Frame Width (pixels)	Max Pitch (bytes)
Legacy 4K	8bpp	16k	16k	16k 127
	16bpp	16k	8k	16k 127
	32bpp	16k	4k	16k 127
	64bpp	16k	2k	16k 127
	128bpp	16k	1k	16k 127

DWord 2, Bits 2:0 — Reserved

Format:	MBZ

DWord 3, Bits 31:14 — Reserved

Format:	MBZ

DWord 3, Bits 13:0 — Y Offset for U(Cb) - EVEN/ODD Reconstructed Pixel Reference Surfaces

Format:	U14 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.

Encoder Base Address Parameters3

DWord	Bit	Description
		Programming Notes For PLANAR_420 and PLANAR_422 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs.
4	31:14	**Reserved** Format: MBZ
	13:0	**Y Offset for V(Cr) - EVEN/ODD reconstructed pixel reference surfaces** Format: U14 Pixel Row Offset This field specifies the vertical offset in rows from the **Surface Base Address** to the start (origin) of the V(Cr) plane. This field is only used for PLANAR surface formats with **Interleave Chroma** disabled. **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must indicate an even number of pixels.
5	31:0	**Reserved** Format: MBZ
6	31:21	**Reserved** Format: MBZ
	20:3	**Surface Pitch Minus1 - EVEN/ODD Reconstructed Pixel Reference Surfaces** Format: U18-1 Pitch in (Bytes - 1) This field specifies the surface pitch in (#Bytes - 1).

Value	Name	Description	Exists If
[0,2047]		For Pitch of [1B, 2048B]	//[Surfaces Type] = SURFTYPE_BUFFER
[0, 262143]		For Pitch of [1B, 256KB]	//[Surfaces Type] = Linear Surface
[511, 262143]		For Pitch of [512B, 256KB]=[1tile, 512 tiles]	//[Surfaces Type] = X-tiled
[127, 262143]		For Pitch of [128B, 256KB]=[1tile, 2048 tiles]	//[Surfaces Type] = Y-tiles

Programming Notes

If Media Memory Compression is enabled, the following max pitch size restriction must be honored. For larger resolution, Media Memory compression Must be disabled.

Tiling Mode	Pixel Format	Max Frame Width (bytes)	Max Frame Width (pixels)	Max Pitch (bytes)
Legacy 4K	8bpp	16k	16k	16k 127
	16bpp	16k	8k	16k 127
	32bpp	16k	4k	16k 127
	64bpp	16k	2k	16k 127
	128bpp	16k	1k	16k 127

Encoder Base Address Parameters3

	2:0	**Reserved**	
		Format:	MBZ
7	31:0	**Reserved**	
		Format:	MBZ
8	31:0	**Reserved**	
		Format:	MBZ
9	31:0	**Reserved**	
		Format:	MBZ

Encoder Base Address Parameters4

Encoder Base Address Parameters4		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

Please note that DW0-9, correspond to DW40-49 of **WiGig Parameters**.

DWord	Bit	Description
0	31:0	**Reserved** Format: MBZ
1	31:0	**Reserved** Format: MBZ
2	31:0	**Reserved** Format: MBZ
3	31:0	**Reserved** Format: MBZ
4	31:0	**Reserved** Format: MBZ
5	31:0	**Reserved** Format: MBZ
6	31:16	**Reserved** Format: MBZ
	15:0	**Max Threshold on the Number of Intra 4x4 Coded MBs Per Frame** Format: U16 Value | Name | Description 0 | Disable | No Limit 1-65535 | Enable | Maximum number of intra 4x4 MBs allowed per frame. **Programming Notes** **Restriction:** When this threshold is on, one or both of the intra16x16 and intra8x8 modes have to be enabled. For I-frame once the intra4x4 threshold is met, the subsequence MBs will be coded as intra16x16 or intra8x8 based on the enable settings. For P-frame, in additional to intra16x16 or intra8x8, the subsequence MBs could be coded with inter type.
7..9	31:0	**Reserved** Format: MBZ

Encoder Control State Parameters0

Encoder Control State Parameters0		
Source:	BSpec	
Size (in bits):	320	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31:30	**Reserved** Format: MBZ
	29	**WDE Packetization enable** Format: Enable Value | Name | Description 0 | WDE packetization is Disabled | Test Mode. 1 | WDE packetization is Enabled | Default mode.
	28:27	**AVC Encoder Chroma Sub-sample type** AVC YUV chroma compression mode. Value | Name 00b | YUV 4:2:0 01b | RGBA 4:4:4:4 10b | YUV 4:4:4 11b | Reserved
	26:5	**Reserved** Format: MBZ
	4	**Reserved**
	3	**Reserved** Format: MBZ
	2	**Conditional Replenishment Enable** Format: U1 If the distortion for a MB is below a fixed threshold, it is coded as Pskip (No Coded Coeff). If above threshold, MB is coded as Intra type. **Programming Notes** This field must be set once per session. If this field is off, all frames are encoded as IDR frames, and frame number as 0. GOP Size parameter must be set to 1. This field should be turned off when IPCM is on.
	1	**Reserved** Format: MBZ

Encoder Control State Parameters0

	0	**Reserved**
1	31:17	**Reserved** Format: MBZ
	16	**GOP-level indirect user-defined NAL data packet - enable** GOP-level user defined NAL packet is inserted during the first frame of a GOP if this field is enabled. Value | Name | Description 0 | Disable | Disable insertion of indirect NAL data packet 1 | Enable | Enable insertion of indirect NAL data packet **Programming Notes** Recommended value is 0. Header contains several NAL units. AVC spec needs AUD NAL to be the first NAL for a frame. If GOP header and Frame headers are enabled and both have AUD NAL it would lead to incorrect bit stream. Multiple NAL data packets can be programmed within the GOP-level indirect surface. A linked list data structure is allocated multiple elements (NAL packets) in contiguous memory. Each element consists of link descriptor fields and datum. The first four bytes of each element contain the link descriptor fields: Byte 0: length_in_bytes[7:0] -- LSB length of data element/NAL packet. Byte 1: length_in_bytes[15:8] -- MSB length of data element/NAL packet. Byte 2: <{7'b000_0000, last_NAL} -- Bit 0 indicates the current element is the last element of the link list. Byte 3: Reserved -- Reserved- MBZ Datum starts at Byte 4 of each element: Byte 4: NAL_START Byte 5: ... Byte N: NAL_END Case 1: last_NAL =1 all the bytes through ByteN would be sent to BSP. Case 2: last_NAL =0 and ByteN is CL alighned (N mod 64 =0). Next NAL descriptor starts at N 1 Byte. Case 3: last_NAL = 0 and ByteN is not CL aligned zero bytes stuffed till next NAL descriptor (63 – (N mod 64) bytes of zeros). Note: empty/zero datum is not allowed See the figure at "WiGig Parameters" for more information.
	15:0	**GOP-level indirect user-defined NAL data packet - Offset** Specifies the Cache line aligned address of the user-defined NAL data packet(s) relatives to the WiDi state base address.
2	31:17	**Reserved** Format: MBZ
	16	**Frame-level indirect user-defined NAL data packet - enable** Frame-level user defined NAL packet is inserted during the first frame of a frame if this field is enabled.

Encoder Control State Parameters0

		Value | **Name** | **Description** 0 | Disable | Disable insertion of indirect NAL data packet 1 | Enable | Enable insertion of indirect NAL data packet **Programming Notes** The recommended value is 1. As mentioned in GOP header enable, enabling GOP header and frame header causes problem. Enabling only frame header is safe and sufficient. Multiple NAL data packets can be programmed within the frame-level indirect surface. A linked list data structure is allocated multiple elements (NAL packets) in contiguous memory. Each element consists of link descriptor fields and datum. The first four bytes of each element contain the link descriptor fields: Byte 0: length_in_bytes[7:0] -- LSB length of data element/NAL packet. Byte 1: length_in_bytes[15:8] -- MSB length of data element/NAL packet. Byte 2: <{7'b000_0000, last_NAL} -- Bit 0 indicates the current element is the last element of the link list. Byte 3: Reserved -- Reserved- MBZ Datum starts at Byte 4 of each element: Byte 4: NAL_START Byte 5: ... Byte N: NAL_END Note: empty/zero datum is not allowed
	15:0	**Frame-level indirect user-defined NAL data packet - Offset** Specifies the Cache line aligned address of the user-defined NAL data packet(s) relatives to the WiDi state base address.
3	31:0	**Reserved** Format: | MBZ
4	31:0	**Reserved** Format: | MBZ
5	31:16	**GOP size parameter** Number of (P-Frames + 1) per GOP structure. **Value** | **Name** | **Description** 0 | | Infinite number of P-frames followed by an initial I-frame(IDR) 1 | | GOP structure will contain one IDR frame. HME and IME inter predictions are disabled. 2 | | One IDR + one P-frame per GOP 3-FFFFh | | This parameter indicates the number of frames within the GOP structure. Each GOP will have one IDR frame follows by the value of the parameter minus one P-frames.
	15:4	**Reserved** Format: | MBZ
	3:0	**log2 Maximum Frame Number Minus4** This field specifies the value of the variable MaxFrameNum that is used in Frame_Num count for every frame and must match the definition of log2_max_frame_num_minus field in SPS NAL packet. **Value** | **Name** 0-12 |

<table>
<tr><th colspan="3">Encoder Control State Parameters0</th></tr>
<tr><td></td><td></td><td><table><tr><th>Programming Notes</th></tr><tr><td>The same value should be set for log2_max_pic_order_cnt_lsb_minus4 field in SPS NAL packet.</td></tr></table></td></tr>
<tr><td>6</td><td>31:0</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>7</td><td>31</td><td>Transform 8x8 Flag
This field indicates that 8x8 transform can be used within the frame.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>4x4 Integer Transform</td><td>The MB must be set to 4x4 transform.</td></tr>
<tr><td>1</td><td>8x8 Integer Transform</td><td>The MB could be coded with 8x8 transform.</td></tr></table>
<table><tr><th>Programming Notes</th></tr>
<tr><td>When this field is set to 1, The transform_size_8x8_flag syntax element, if present in the output bit stream, is the same as this field; However, whether transform_szie_8x8_flag is present or not in the output bit stream depends on several other conditions.
Hardware set MB level transform8x8 flag to 1 for two conditions:
It might be 1 if IntraMbFlag = INTRA and IntraMbMode = INTRA_8x8
It must be 1 if IntraMbFlag = INTER and there is no sub partition size less than 8x8
Otherwise, this field must be set to 0.
For Intra MB if transform8x8 = 0, hardware will always honor it, thus intra8x8 will be the winner.</td></tr>
<tr><td>If transform8x8 = 0, Intra16x16 prediction Enable or Intra4x4 prediction Enable MUST be true.</td></tr></table></td></tr>
<tr><td></td><td>30:29</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>28:24</td><td>Second Chroma QP Offset, Chroma_qp_offset [9:5]<table><tr><td>Format:</td><td>S4</td></tr></table>
<table><tr><td>Range: -12 to +12 According to AVC Spec.</td></tr>
<tr><td>It specifies the offset for determining QP Cr from QP Y. It is set to the upper 5 bits of the value of the syntax element (Chroma_qp_offset[9:0]) read from the current active PPS.
Chroma_qp_offset [9:5] - second_chroma_qp_offset_bits</td></tr></table>
<table><tr><th>Programming Notes</th></tr>
<tr><td>To ensure that the MB size doesn't exceed 3200 bits, Cr/Cb QP cannot go below 10. The Value of MinQp for Luma and Chroma Offset is programmed in such a way to ensure this.
E.g. If chroma offset = -5, MinQp should be >= 15. This would ensure that the Final Chroma QP >= (-5+15 = 10).</td></tr></table></td></tr>
<tr><td></td><td>23:21</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

Encoder Control State Parameters0

	20:16	**Chroma QP Offset, Chroma_qp_offset[4:0]** Format: S4 **Range:** -12 to +12 According to AVC Spec. It specifies the offset for determining QP Cb from QP Y.It is set to the lower 5 bits of the value of the syntax element (Chroma_qp_offset[9:0]) read from the current active PPS. • Chroma_qp_offset [4:0] - chroma_qp_offset_bits (from the current active PPS) **Programming Notes** To ensure that the MB size doesn't exceed 3200 bits, Cr/Cb QP cannot go below 10. The Value of MinQp for Luma and Chroma Offset is programmed in such a way to ensure this. E.g. If chroma offset = -5, MinQp should be >= 15. This would ensure that the Final Chroma QP >= (-5+15 = 10).
	15:8	**Reserved** Format: MBZ
	7	**Round Inter Enable** Format: Enable **Programming Notes** Recommended driver setting is 0. When rounding Inter is disabled, a value of 2/8 is used for rounding inter coefficients.
	6:4	**Rounding Inter (N)** Format: RoundingPrecisionTable_3_Bits **Programming Notes** Hardware default this field to 2 if "Round Inter Enable" is disable.
	3	**Round Intra Enable** Format: Enable **Programming Notes** Recommended driver setting is 0. When rounding Intra is disabled, a value of 4/8 is used for rounding intra coefficients.
	2:0	**Rounding Intra (N)** Format: RoundingPrecisionTable_3_Bits **Programming Notes** Hardware default this field to 4 if "Round Intra Enable" is disable.
8	31:0	**Reserved** Format: MBZ

Encoder Control State Parameters0		
9	31:19	**Reserved** Format: MBZ
	18:16	**Slice Pattern Per MB Row** The field should be set according to the following table and the MB row size. In cases where row cannot be divided evenly, round up to the nearest MB to achieve the indicated number of slices per row. For purposes of this clause, the rate of macroblocks/second is the rate that applies to the video being transmitted, according to the definition in the AVC standard, That is: the macroblock rate = ceiling(frame width, 16)/16 * ceiling(frame height, 16)/16 * (frames/second). Slice size Must be larger than 4MB.
	15:0	**Reserved** Format: MBZ

Slice Pattern Per MB Row (bits 18:16):

Macroblocks/second Rate	Slice Pattern
rate < 250,000	1 row = 1 slice
250,000 <= rate < 500,000	1 row = 2 equal slices
500,000 <= rate < 1,000,000	1 row = 4 equal slices
1,000,000 <= rate < 2,100,000	1 row = 8 equal slices
2,100,000 <= rate	1 row = 16 equal slices

Value	Name	Description
0		1 slice per MB row
1		2 slice per MB row
2		4 slice per MB row
3		8 slice per MB row
4		16 slice per MB row

Encoder Statistics Format

Encoder Statistics Format

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

Description
The per block data is intended for use by the video encoder and consists of 16 bytes of Denoise block data and FMD variances. Much of the data is encoded as an 8-bit mantissa with the leading 1 removed and a 4-bit shift. To recover the original 17-bit integer this code can be used: If (exp != 0) Number = ((0x100 | Mantissa) « exp) » 7; else Number = mantissa;
The values for STAD, SHCM and SVCM for each 4x4 are shited down by 2 bits to make 14-bit values before being summed for the 16x4 block to make a 16-bit value. The result is then converted into the mantissa/exp format.

DWord	Bit	Description
0	31:24	**Tearing_Count 1 (FMD Variance[8])** Format: U8 Number of pixels that have (diff_cTcB > diff_cTcT + diff_cBcB) Value: 0 | Name: | Description: DI is Disabled
	23:16	**Tearing_Count 2** Format: U8 If the frame is Deinterlaced with Top First in the DN/DI state then this is (FMD Variance[9]) = Number of pixels that have (diff_cTpB > diff_cTcT + diff_pBpB) If the frame is bottom first then this is (FMD Variance[10]) = Number of pixels that have (diff_cBpT > diff_pTpT + diff_cBcB) Value: 0 | Name: | Description: DI is Disabled
	15:8	**Motion_Count (FMD Variance[7])** Format: U8 Number of pixels that are moving (different above a threshold) Value: 0 | Name: | Description: DI is Disabled
	7:0	**Reserved** Format: MBZ
1	31:28	**sSTAD** Format: U4

<table>
<tr><th colspan="3">Encoder Statistics Format</th></tr>
<tr><td></td><td></td><td>Shift for the Sum in time of absolute differences for 16x4.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>Temporal Denoise Filtering is Disabled.</td></tr></table></td></tr>
<tr><td></td><td>27:24</td><td>sSHCM
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the Sum horizontaly of absolute differences.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DN is Disabled</td></tr></table></td></tr>
<tr><td></td><td>23:20</td><td>sSVCM
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the Sum vertically of absolute differences.</td></tr>
<tr><td></td><td>19:16</td><td>sDiff_cTpT
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the sum of differences in top fields of current and previous frame.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table></td></tr>
<tr><td></td><td>15:12</td><td>sDiff_cBpB
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the sum of differences in bottom field of current and previous frame.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table></td></tr>
<tr><td></td><td>11:8</td><td>sDiff_cTcB
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the sum of differences between top and bottom field in current frame.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table></td></tr>
<tr><td></td><td>7:4</td><td>sDiff_cTpB
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the sum of differences between current top and previous bottom.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table></td></tr>
<tr><td></td><td>3:0</td><td>sDiff_cBpT
<table><tr><td>Format:</td><td>U4</td></tr></table>
Shift for the sum of differences between current bottom and previous top.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table></td></tr>
<tr><td>2</td><td>31:24</td><td>mDiff_cBpB (FMD Variance[1])</td></tr>
</table>

Encoder Statistics Format

DWord	Bit	Description
		Format: U8 Mantissa of sum of differences in bottom field of current and previous frame. **Value** | **Name** | **Description** 0 | | DI is Disabled
	23:16	**mDiff_cTcB (FMD Variance[2])** Format: U8 Mantissa of sum of differences between top and bottom field in current frame. **Value** | **Name** | **Description** 0 | | DI is Disabled
	15:8	**mDiff_cTpB (FMD Variance[3])** Format: U8 Mantissa of sum of differences between current top and previous bottom. **Value** | **Name** | **Description** 0 | | DI is Disabled
	7:0	**mDiff_cBpT (FMD Variance[4])** Format: U8 Mantissa of sum of differences between current bottom and previous top. **Value** | **Name** | **Description** 0 | | DI is Disabled
3	31:24	**mSTAD** Format: U8 Mantissa of Sum in time of absolute differences for 16x4. **Value** | **Name** | **Description** 0 | | Temporal Denoise Filtering is disabled.
	23:16	**mSHCM** Format: U8 Mantissa of Sum horizontaly of absolute differences. **Value** | **Name** | **Description** 0 | | DN is Disabled
	15:8	**mSVCM** Format: U8 Mantissa of Sum vertically of absolute differences. **Value** | **Name** | **Description** 0 | | DN is Disabled
	7:0	**mDiff_cTpT (FMD Variance[0])** Format: U8 Mantissa of sum of differences in top fields of current and previous frame.

Encoder Statistics Format		
		<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DI is Disabled</td></tr></table>

Engine ID Definition

Engine ID Definition			
Source:	BSpec		
Size (in bits):	9		
Default Value:	0x00000000		
Defines the values used for Engine IDs for interrupt processing and Context IDs.			
DWord	**Bit**	**Description**	
0	8:3	**Instance ID**	
		Format:	U6

Value	Name	Description	Exists If
0h	RCS		[Class ID] == 'Render'
0h	VCS0		[Class ID] == 'Video Decode'
1h	VCS1		[Class ID] == 'Video Decode'
2h	VCS2		[Class ID] == 'Video Decode'
3h	VCS3		[Class ID] == 'Video Decode'
4h	VCS4		[Class ID] == 'Video Decode'
5h	VCS5		[Class ID] == 'Video Decode'
6h	VCS6		[Class ID] == 'Video Decode'
7h	VCS7		[Class ID] == 'Video Decode'
0h	VECS0		[Class ID] == 'Video Enhancement'
1h	VECS1		[Class ID] == 'Video Enhancement'
2h	VECS2		[Class ID] == 'Video Enhancement'
3h	VECS3		[Class ID] == 'Video Enhancement'
0h	BCS		[Class ID] == 'Copy Engine'
0h	Reserved		[Class ID] == 'Other'
1h	GTPM	Power Management for GT	[Class ID] == 'Other'
2h	WD OA Perf	Wireless Display/Observabilitiy	[Class ID] == 'Other'
3h	Reserved		[Class ID] == 'Other'
4h	Reserved		[Class ID] == 'Other'
5h	GUNIT		[Class ID] == 'Other'
6h	CSME	Manageability Engine	[Class ID] == 'Other'
0h	CCS0		[Class ID] == 'Compute'
1h	CCS1		[Class ID] == 'Compute'
2h	CCS2		[Class ID] == 'Compute'
3h	CCS3		[Class ID] == 'Compute'

Engine ID Definition		
		5h-3fh | Reserved
	2:0	**Class ID**
		Format: U3

Value	Name
0h	Render
1h	Video Decode
2h	Video Enhancement
3h	Copy Engine
4h	Other
5h	Compute
6h-7h	Reserved

EU_INSTRUCTION_ALIGN1_THREE_SRC

EU_INSTRUCTION_ALIGN1_THREE_SRC

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

DWord	Bit	Description
0..3	127:126	**Reserved** Format: MBZ
	125	**Reserved** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='IMM') Format: MBZ
	125:118	**Source 2 Register number** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='GRF') Format: SrcRegNum
	124:109	**Source 2 Immediate Value** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='IMM')
	117:113	**Source 2 Subregister number** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='GRF') Format: SrcSubRegNum
	112:111	**Source 2 Horizontal Stride** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='GRF') Format: TernaryAlign1HorzStride
	110:109	**Reserved** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 2 Register File]=='GRF') Format: MBZ
	108:106	**Source 2 Datatype** Format: TernaryAlign1DataType Selects source 2 datatype.
	105	**Reserved** Format: MBZ
	104:97	**Source 1 Register number** Format: SrcRegNum
	96:92	**Source 1 Subregister number** Format: SrcSubRegNum
	91:90	**Source 1 Horizontal Stride**

EU_INSTRUCTION_ALIGN1_THREE_SRC

Bits	Description
	Format: **TernaryAlign1HorzStride**
89:88	**Source 1 Vertical Stride** Format: **TernaryAlign1VertStride**
87:85	**Source 1 Datatype** Format: **TernaryAlign1DataType** Selects source 1 datatype.
84	**Reserved** Format: MBZ
83	**Reserved** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='IMM') Format: MBZ
83:76	**Source 0 Register number** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='GRF') Format: **SrcRegNum**
82:67	**Source 0 Immediate Value** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='IMM')
75:71	**Source 0 Subregister number** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='GRF') Format: **SrcSubRegNum**
70:69	**Source 0 Horizontal Stride** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='GRF') Format: **TernaryAlign1HorzStride**
68:67	**Source 0 Vertical Stride** Exists If: (Structure[EU_INSTRUCTION_ALIGN1_THREE_SRC][Source 0 Register File]=='GRF') Format: **TernaryAlign1VertStride**
66:64	**Source 0 Datatype** Format: **TernaryAlign1DataType** Selects source 0 datatype.
63:56	**Destination Register Number** Format: **DstRegNum**
55:54	**Destination Subregister Number** Format: (DstSubRegNum[4:3])
53:50	**Reserved** Format: MBZ

EU_INSTRUCTION_ALIGN1_THREE_SRC

	49	**Destination Horizontal Stride** Selects destination horizontal stride.Destination horizontal stride is required for striding based on execution size or packing the destination datatype. <table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>1 element</td></tr><tr><td>1</td><td>2 element</td></tr></table>
	48:46	**Destination Datatype** <table><tr><td>Format:</td><td>**TernaryAlign1DataType**</td></tr></table>Selects destination datatype.
	45	**Source 2 Register File** Selects source 2 register file. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>GRF</td><td>Selects General Register File as source 2.</td></tr><tr><td>1</td><td>IMM</td><td>Selects Immediate Register File as source 2.</td></tr></table>
	44	**Source 1 Register File** Selects source 1 register file. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>GRF</td><td>Selects General Register File as source 1.</td></tr><tr><td>1</td><td>ARF</td><td>Selects Architectural Register File as source 1. Only Accumulator is allowed.</td></tr></table>
	43	**Source 0 Register File** Selects source 0 register file. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>GRF</td><td>Selects General Register File as source 0.</td></tr><tr><td>1</td><td>IMM</td><td>Selects Immediate Register File as source 0.</td></tr></table>
	42:41	**Source 2 Modifier** <table><tr><td>Exists If:</td><td>(Property[Source Modifier]=='true')</td></tr><tr><td>Format:</td><td>**SrcMod**</td></tr></table>
	42:37	**Reserved** <table><tr><td>Exists If:</td><td>(Property[Source Modifier]=='false')</td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
	40:39	**Source 1 Modifier** <table><tr><td>Exists If:</td><td>(Property[Source Modifier]=='true')</td></tr><tr><td>Format:</td><td>**SrcMod**</td></tr></table>
	38:37	**Source 0 Modifier** <table><tr><td>Exists If:</td><td>(Property[Source Modifier]=='true')</td></tr><tr><td>Format:</td><td>**SrcMod**</td></tr></table>
	36	**Destination Register File**

EU_INSTRUCTION_ALIGN1_THREE_SRC

		Selects destination register file. **Value** | **Name** | **Description** 0 | GRF | Selects General Register File as Destination. 1 | ARF | Selects Architectural Register File as Destination. Only Accumulator is allowed.
	35	**Execution Datatype** This field defines common data type for all sources and destination operands. **Value** | **Name** | **Description** 0 | Integer | Integer datatypes. 1 | Float | Floating point datatype.
	34	**MaskCtrl** This flag disables the normal write enables; it should normally be 0. **Value** | **Name** | **Description** 0 | Normal | Use the normal write enables in Dst.ChanEn (normal setting). 1 | NoMask | Write all channels except those disabled by predication or by other masks besides the write enables. **Programming Notes** MaskCtrl = NoMask also skips the check for PcIP[n] == ExIP before enabling a channel, as described in the Evaluate Write Enable section.
	33	**Flag Register Number** This field contains the flag register number for instructions with a non-zero Conditional Modifier.
	32	**Flag Subregister Number** This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BASIC_ONE_SRC

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

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

EU_INSTRUCTION_BASIC_THREE_SRC

EU_INSTRUCTION_BASIC_THREE_SRC

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

DWord	Bit	Description
0..3	127	**Reserved** Format: MBZ
	126:106	**Source 2** Format: **EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC**
	105:85	**Source 1** Format: **EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC**
	84:64	**Source 0** Format: **EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC**
	63:56	**Destination Register Number** Format: **DstRegNum**
	55:53	**Destination Subregister Number**
	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
	48:46	**Destination Data Type**

Value	Name	Description
000b	:f	single precision Float (32-bit)
001b	:d	signed Doubleword integer
010b	:ud	Unsigned Doubleword integer
011b	:df	Double precision Float (64-bit)
100b	:hf	Half Float (16-bit)

EU_INSTRUCTION_BASIC_THREE_SRC

Bits	Field
	101b-111b / Reserved
45:43	**Source Data Type**
42:41	**Source 2 Modifier**
42:37	**Reserved**
40:39	**Source 1 Modifier**
38:37	**Source 0 Modifier**
36	**Source 1 Type**
35	**Source 2 Type**

101b-111b	Reserved	

45:43 Source Data Type

Value	Name	Description
000b	:f	single precision Float (32-bit)
001b	:d	signed Doubleword integer
010b	:ud	Unsigned Doubleword integer
011b	:df	Double precision Float (64-bit)
100b	:hf	Half Float (16-bit)
101b-111b	Reserved	

42:41 Source 2 Modifier

Exists If:	(Property[Source Modifier]=='true')
Format:	**SrcMod**

42:37 Reserved

Exists If:	(Property[Source Modifier]=='false')
Format:	MBZ

40:39 Source 1 Modifier

Exists If:	(Property[Source Modifier]=='true')
Format:	**SrcMod**

38:37 Source 0 Modifier

Exists If:	(Property[Source Modifier]=='true')
Format:	**SrcMod**

36 Source 1 Type

Format:	U1

Only used if Source Data Type is :f or :hf, else Source 1 Data Type matches Source 0 type and this bit is ignored.

Value	Name	Description
0b	:f	single precision Float (32-bit)
1b	:hf	Half Float (16-bit)

35 Source 2 Type

Format:	U1

Only used if Source Data Type is :f or :hf, else Source 2 Data Type matches Source 0 type and this bit is ignored.

Value	Name	Description
0b	:f	single precision Float (32-bit)
1b	:hf	Half Float (16-bit)

EU_INSTRUCTION_BASIC_THREE_SRC

	34	**MaskCtrl** (formerly WECtrl/Write Enable Control). This flag disables the normal write enables; it should normally be 0. **Value** | **Name** | **Description** 0 | Normal | Use the normal write enables in Dst.ChanEn (normal setting). 1 | NoMask | Write all channels except those disabled by predication or by other masks besides the write enables. **Programming Notes** MaskCtrl = NoMask also skips the check for PcIP[n] == ExIP before enabling a channel, as described in the Evaluate Write Enable section.
	33	**Flag Register Number** This field contains the flag register number for instructions with a non-zero Conditional Modifier.
	32	**Flag Subregister Number** This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_BASIC_TWO_SRC

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

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

EU_INSTRUCTION_BRANCH_CONDITIONAL

EU_INSTRUCTION_BRANCH_CONDITIONAL		
Source:	Eulsa	
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_REG
	127:64	**Sources** Exists If: ([Src1.RegFile]=='IMM') Format: EU_INSTRUCTION_SOURCES_REG_IMM
	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**

EU_INSTRUCTION_BRANCH_CONDITIONAL		
		Format: **RegFile**
	36:34	**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.
	33:32	**Destination Register File** Format: **RegFile**
	31:0	**Header** Format: **EU_INSTRUCTION_HEADER**

EU_INSTRUCTION_BRANCH_ONE_SRC

EU_INSTRUCTION_BRANCH_ONE_SRC

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

DWord	Bit	Description
0..3	127:96	**JIP** Format: S31 Jump Target Offset. The relative offset in bytes if a jump is taken for the instruction.
	95	**Source 0 Address Immediate [9] Sign Bit**
	94:91	**Src1.SrcType** Format: **SrcType**
	90:89	**Src1.RegFile** Format: **RegFile**
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16**
	88:64	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1**
	63:32	**Operand Control** Format: **EU_INSTRUCTION_OPERAND_CONTROLS**
	31:0	**Header** Format: **EU_INSTRUCTION_HEADER**

EU_INSTRUCTION_BRANCH_TWO_SRC

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

DWord	Bit	Description
0..3	127:96	**JIP** Format: S31 The byte-aligned jump distance if a jump is taken for the channel.
	95:64	**UIP** Format: S31 The byte aligned jump distance if a jump is taken for the instruction.
	63:32	**Operand Control** Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	**Header** Format: EU_INSTRUCTION_HEADER

EU_INSTRUCTION_COMPACT_THREE_SRC

EU_INSTRUCTION_COMPACT_THREE_SRC		
Source:	Eulsa	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0..1	63:57	**Src2.RegNum[6:0]** Format: **SrcRegNum[6:0]** Src2.RegNum[6:0]. The SourceIndex field in the compact instruction determines Src2.RegNum[7]. Maps to 124:118
	56:50	**Src1.RegNum[6:0]** Format: **SrcRegNum[6:0]** Src1.RegNum[6:0]. The SourceIndex field in the compact instruction determines Src1.RegNum[7]. Maps to 103:97
	49:43	**Src0.RegNum[6:0]** Format: **SrcRegNum[6:0]** Src0.RegNum[6:0]. The SourceIndex field in the compact instruction determines Src0.RegNum[7]. Maps to 82:76
	42:40	**Src2.SubRegNum** Format: **SrcSubRegNum[4:2]** Maps to 117:115
	39:37	**Src1.SubRegNum** Format: **SrcSubRegNum[4:2]** Maps to 96:94
	36:34	**Src0.SubRegNum** Format: **SrcSubRegNum[4:2]** Maps to 75:73
	33	**Src2.RepCtrl** Format: RepCtrl Maps to 106

EU_INSTRUCTION_COMPACT_THREE_SRC

	32	**Src1.RepCtrl** Format: RepCtrl Maps to 85
	31	**Reserved** Exists If: (Property[Saturation]=='false') Format: MBZ
	31	**Saturate** Exists If: (Property[Saturation]=='true') Maps to 31
	30	**Reserved**
	29	**Compaction Control** Format: CmptCtrl
	28	**Src0.RepCtrl** Format: RepCtrl Maps to 64
	27:19	**Reserved** Format: MBZ
	18:12	**Dst.RegNum[6:0]** Format: DstRegNum[6:0] Dst.RegNum[7:0] with MSB of zero and [6:0] from the compact instruction Maps to 63:56 (Dst.RegNum)
	11:10	**SourceIndex** Lookup one of four 46-bit values. That value is used (from MSB to LSB) for the Src2.RegNum[7], Src1.RegNum[7], Src0.RegNum[7], Src2.ChanSel, Src1.ChanSel, Src0.ChanSel, Dst.SubRegNum, Dst.ChanEnable, Dst.DstType, SrcType, Src2.Modifier, Src1.Modifier, and Src0.Modifier bit fields. Maps to 125, 104, 83, 114:107, 93:86, 72:65, 55:49, 48:43, 42:37

Value	Name	Description
0	0001110010011100100111001000001111000000000000	No Negation
1	0001110010011100100111001000001111000000000010	Negate Src0
2	0001110010011100100111001000001111000000001000	Negate Src1
3	0001110010011100100111001000001111000000100000	Negate Src2

EU_INSTRUCTION_COMPACT_THREE_SRC

	9:8	**ControlIndex**

Lookup one of four 24-bit values. That value is used (from MSB to LSB) for the MaskCtrl, FlagRegNum/FlagSubRegNum, AccWrCtrl, CondModifier, ExecSize, PredInv, PredCtrl, ThreadCtrl, QtrCtrl, NibCtrl, DepCtrl, and AccessMode bit fields.
Maps to 34, 33:32, 28:8

Value	Name	Description
0	100000000110000000000001	(8) Q1 NoMask Align16
1	000000000110000000000001	(8) Q1 Align16
2	000000001000000000000001	(16) H1 Align16
3	000000001000000000100001	(16) H2 Align16

	7	**Reserved**

Format:	MBZ

	6:0	**Opcode**

EU_INSTRUCTION_COMPACT_TWO_SRC

EU_INSTRUCTION_COMPACT_TWO_SRC

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

The following table describes the EU compact instruction format. The compact instruction format for 1 or 2-source instructions is essentially identical to the compact instruction format for earlier generations, but the compact fields expand to somewhat different fields in the native instruction format, as the native instruction format changed.

DWord	Bit	Description
0..1	63:56	**Src1.RegNum** Exists If: ([DataTypeIndex][Src1.RegFile]!='IMM') Format: **SrcRegNum** Maps to 108:101 (Src1.RegNum)
	63:56	**Src1.RegNum** Exists If: ([DataTypeIndex][Src1.RegFile]=='IMM') Maps to 103:96 (Imm32[7:0])
	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') Format: **SrcIndex** If not an immediate operand, lookup one of 32 12-bit values that maps to bits 120:109. That value is used (from MSB to LSB) for the Src1.VertStride, various Src1 bit fields based on AccessMode (Src1.ChanSel[7:4], Src1.Width, Src1.HorzStride), Src1.AddrMode, and Src1.SrcMod bit fields Maps to 120:109
	39:35	**Src1Index** Exists If: ([DataTypeIndex][Src1.RegFile]=='IMM') If an immediate operand, there is no lookup. Determines bits 127:104 (Imm32[31:8]) as follows: map bits 39:35 directly to bits 108:104. Sign extend to fill bits 127:109. Compact format bit 39 is thus copied to all of bits 127:108 for an immediate operand. Maps to 127:104

EU_INSTRUCTION_COMPACT_TWO_SRC

	34:30	**Src0Index** Format: **SrcIndex** Lookup one of 32 12-bit values. That value is used (from MSB to LSB) for the Src0.VertStride, various Src0 bit fields based on AccessMode (Src0.ChanSel[7:4], Src0.Width, Src0.HorzStride), Src0.AddrMode, and Src0.SrcMod bit fields. Note that this field spans a DWord boundary within the QWord compacted instruction. Maps to 88:77
	29	**Compaction Control** Format: CmptCtrl
	28	**Reserved** Format: MBZ
	27:24	**Reserved** Exists If: (Property[Conditional Modifier]=='false') Format: MBZ
	27:24	**Conditional Modifier** Exists If: (Property[Conditional Modifier]=='true') Format: **CondModifier**
	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 |

EU_INSTRUCTION_COMPACT_TWO_SRC

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
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 21-bit values. That value is used (from MSB to LSB) for the Dst.AddrMode, Dst.HorzStride, Src1.SrcType, Src1.RegFile, Src0.SrcType, Src0.RegFile, Dst.DstType, and Dst.RegFile bit fields.

Maps to 63:61, 94:89, 46:35

Value	Name	Description
0	001000000000000000001	r:ud | a:ud | a:ud | <1> | dir |
1	001000000000001000000	a:ud | r:ud | a:ud | <1> | dir |
2	001000000000001000001	r:ud | r:ud | a:ud | <1> | dir |
3	001000000000011000001	r:ud | i:ud | a:ud | <1> | dir |
4	001000000000101011101	r:f | r:d | a:ud | <1> | dir |
5	001000000010111011101	r:f | i:vf | a:ud | <1> | dir |
6	001000000011101000001	r:ud | r:f | a:ud | <1> | dir |
7	001000000011101000101	r:d | r:f | a:ud | <1> | dir |
8	001000000011101011101	r:f | r:f | a:ud | <1> | dir |

EU_INSTRUCTION_COMPACT_TWO_SRC

9	0010000001000001000001	r:ud | r:ud | r:ud | <1> | dir |
10	0010000011000001000000	a:ud | r:ud | i:ud | <1> | dir |
11	0010000011000001000001	r:ud | r:ud | i:ud | <1> | dir |
12	0010001010001010000101	r:d | r:d | r:d | <1> | dir |
13	0010001110001010000100	a:d | r:d | i:d | <1> | dir |
14	0010001110001010000101	r:d | r:d | i:d | <1> | dir |
15	0010111000111010111101	r:f | r:f | a:f | <1> | dir |
16	0010111010111000111101	r:f | a:f | r:f | <1> | dir |
17	0010111010111010111100	a:f | r:f | r:f | <1> | dir |
18	0010111010111010111101	r:f | r:f | r:f | <1> | dir |
19	0010111110111010111100	a:f | r:f | i:f | <1> | dir |
20	0000000000100000001100	a:w | a:ub | a:ud | <0> | dir |
21	0010000000000001011101	r:f | r:ud | a:ud | <1> | dir |
22	0010000000000101000101	r:d | r:d | a:ud | <1> | dir |
23	0010000001000001000000	a:ud | r:ud | r:ud | <1> | dir |
24	0010001010001010000100	a:d | r:d | r:d | <1> | dir |
25	0010001110001000000100	a:d | a:d | i:d | <1> | dir |
26	0010010010010000001001	r:uw | a:uw | r:uw | <1> | dir |
27	0010101110111010111101	r:f | r:f | i:vf | <1> | dir |
28	0010111110111010111101	r:f | r:f | i:f | <1> | dir |
29	0010011110011010011100	a:w | r:w | i:w | <1> | dir |
30	0010010010010010010000	a:uw | r:uw | r:uw | <1> | dir |
31	0010010110010010010000	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 33:32, 31, 23:12, 10:9, 34, 8

Value	Name	Description
0	0000000000000000010	Align1 | We | (1) | f0.0
1	0000100000000000000	Align1 | (4) | f0.0
2	0000100000000000001	Align16 | (4) | f0.0
3	0000100000000000010	Align1 | We | (4) | f0.0
4	0000100000000000011	Align16 | We | (4) | f0.0
5	0000100000000000100	Align1 | NoDDClr | (4) | f0.0
6	0000100000000000101	Align16 | NoDDClr | (4) | f0.0

EU_INSTRUCTION_COMPACT_TWO_SRC

		7	0000100000000000111	Align16 | We | NoDDClr | (4) | f0.0
		8	0000100000000001000	Align1 | NoDDChk | (4) | f0.0
		9	0000100000000001001	Align16 | NoDDChk | (4) | f0.0
		10	0000100000000001101	Align16 | NoDDClr, NoDDChk | (4) | f0.0
		11	0000110000000000000	Align1 | Q1 | (8) | f0.0
		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	**Reserved**		
	6:0	**Opcode**		

EU_INSTRUCTION_CONTROLS_A

EU_INSTRUCTION_CONTROLS_A

Source:	Eulsa
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. PMask is the final predication mask produced by the effects of both fields. Value | Name | Description 0 | Positive **[Default]** | Positive polarity of predication. Use the predication mask produced by PredCtrl 1 | Negative | Negative polarity of predication. If PredCtrl is nonzero, invert the predication mask.
	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

EU_INSTRUCTION_CONTROLS_A

Thread Control. This field provides explicit control for thread switching. If this field is set to 00b, it is up to the execution units to manage thread switching. This is the normal (and unnamed) mode. In this mode, for example, if the current instruction cannot proceed due to operand dependencies, the EU switches to the next available thread to fill the compute pipe. In another example, if the current instruction is ready to go, however, there is another thread with higher priority that also has an instruction ready, the EU switches to that thread. If this field is set to Switch, a forced thread switch occurs after the current instruction is executed and before the next instruction. In addition, a long delay (longer than the execution pipe latency) is introduced for the current thread. Particularly, the instruction queue of the current thread is flushed after the current instruction is dispatched for execution. Switch is designed primarily as a safety feature in case there are race conditions for certain instructions.

5:4 **QtrCtrl**

Format:	QtrCtrl

Quarter Control.
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 **NibCtrl**

Nibble Control. This field is used in some instructions along with QtrCtrl. See the description of QtrCtrl below. NibCtrl is only used for SIMD4 instructions with a DF (Double Float) source or destination.

Value	Name	Description
0	Odd	Use an odd 1/8th for DMask/VMask and ARF (first, third, fifth, or seventh depending on QtrCtrl).
1	Even	Use an even 1/8th for DMask/VMask and ARF (second, fourth, sixth, or eighth depending on QtrCtrl).

Programming Notes
Note that if eighths are given zero-based indices from 0 to 7, then NibCtrl = 0 indicates even indices and NibCtrl = 1 indicates odd indices.

2:1 **DepCtrl**

Format:	DepCtrl

Destination Dependency Control. This field selectively disables destination dependency check and clear for this instruction. When it is set to 00, normal destination dependency control is performed for the instruction - hardware checks for destination hazards to ensure data integrity. Specifically, destination register dependency check is conducted before the instruction is made ready for execution. After the instruction is executed, the destination register scoreboard will be cleared when the destination operands retire. When bit 10 is set (NoDDClr), the destination register scoreboard will NOT be cleared when the destination operands retire. When bit 11 is set (NoDDChk), hardware does not check for destination register dependency before the instruction is made ready for execution. NoDDClr and NoDDChk are not mutual exclusive. When this field is not all-zero, hardware does not protect against destination hazards for the instruction. This is typically used to assemble data in a fine grained fashion (e.g. matrix-vector compute with dot-

EU_INSTRUCTION_CONTROLS_A		
		product instructions), where the data integrity is guaranteed by software based on the intended usage of instruction sequences.
	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

<table>
<tr><th colspan="3">EU_INSTRUCTION_CONTROLS_B</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 4
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>3</td><td>Reserved
<table><tr><td>Exists If:</td><td>(Property[Saturation]=='false')</td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>3</td><td>Saturate
<table><tr><td>Exists If:</td><td>(Property[Saturation]=='true')</td></tr></table>
Enables or disables destination saturation. When it is set, output values to the destination register are saturated. The saturation operation depends on the destination data type. Saturation is the operation that converts any value outside the saturation target range for the data type to the closest value in the target range. For a floating-point destination type, the saturation target range is [0.0, 1.0]. For a floating-point NaN, there is no closest value; any NaN saturates to 0.0. Note that enabling Saturate overrides all of the NaN propagation behaviors described for various numeric instructions. Any floating-point number greater than 1.0, including +INF, saturates to 1.0. Any negative floating-point number, including -INF, saturates to 0.0. Any floating-point number in the range 0.0 to 1.0 is not changed by saturation. For an integer destination type, the maximum range for that type is the saturation target range. For example, the saturation range for B (Signed Byte Integer) is [-128, 127]. When Saturate is clear, destination values are not saturated. For example, a wrapped result (modulo) is output to the destination for an overflowed integer value. See the Numeric Data Typessection for information about data types and their ranges.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>No destination modification [Default]</td><td></td></tr>
<tr><td>1</td><td>sat</td><td>Saturate the output</td></tr></table></td></tr>
<tr><td>2</td><td>Reserved</td></tr>
<tr><td>1</td><td>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. See EU Compact Instruction Format for more information.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>NoCompaction</td><td>No compaction. 128-bit native instruction supporting all instruction options.</td></tr>
<tr><td>1</td><td>Compacted</td><td>Compaction is enabled. 64-bit compact instruction supporting only some instruction variations.</td></tr></table></td></tr>
</table>

EU_INSTRUCTION_CONTROLS_B

	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

EU_INSTRUCTION_CONTROLS		
Source:	Eulsa	
Size (in bits):	24	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	23:20	**Controls B** Format: EU_INSTRUCTION_CONTROLS_B
	19:16	**Reserved** Exists If: (Property[Conditional Modifier]=='false') Format: MBZ
	19:16	**CondModifier** Exists If: (Property[Conditional Modifier]=='true') Format: CondModifier Does not exist for send/sendc/math/branch/break-continue opcodes
	15:0	**Controls A** Format: EU_INSTRUCTION_CONTROLS_A

EU_INSTRUCTION_HEADER

EU_INSTRUCTION_HEADER		
Source:	Eulsa	
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

<table>
<tr><th colspan="3">EU_INSTRUCTION_ILLEGAL</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 128
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0..3</td><td>127:7</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>6:0</td><td>Opcode
Format: EU_OPCODE</td></tr>
</table>

EU_INSTRUCTION_MATH

EU_INSTRUCTION_MATH		
Source:	Eulsa	
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

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

DWord	Bit	Description
0..3	127:31	**Reserved** Format: MBZ
	30	**Reserved**
	29:7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_OPERAND_CONTROLS

EU_INSTRUCTION_OPERAND_CONTROLS		
Source:	Eulsa	
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	**Reserved** Exists If: ([Destination Register Region][Destination Addressing Mode]=='Direct') Format: MBZ
	15	**Destination Address Immediate[9:9]** Exists If: ([Destination Register Region][Destination Addressing Mode]=='Indirect') Format: U1
	14:11	**Src0.SrcType** Exists If: ([Src0.RegFile]!='IMM') Format: **SrcType**
	14:11	**Src0.SrcType** Exists If: ([Src0.RegFile]=='IMM') Format: **SrcImmType**
	10:9	**Src0.RegFile** Format: **RegFile**
	8:5	**Destination Data Type** Format: **DstType** 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.
	4:3	**Destination Register File** Format: **RegFile**
	2	**MaskCtrl** Mask Control (formerly Write Enable Control). This field determines if the the per channel write

EU_INSTRUCTION_OPERAND_CONTROLS		
		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.

	1:0	**Flag Register Number/Subregister Number**

EU_INSTRUCTION_OPERAND_DST_ALIGN1

<table>
<tr><th colspan="3">EU_INSTRUCTION_OPERAND_DST_ALIGN1</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 16
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>15</td><td>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.)</td></tr>
<tr><td></td><td>14:13</td><td>Destination Horizontal Stride
Format: HorzStride
For a send instruction, this field applies to CurrDst. PostDst only uses the register number.</td></tr>
<tr><td></td><td>12:9</td><td>Destination Address Subregister Number
Exists If: ([Destination Addressing Mode]=='Indirect')
Format: AddrSubRegNum
For a send instruction, this field applies to PostDst</td></tr>
<tr><td></td><td>12:5</td><td>Destination Register Number
Exists If: ([Destination Addressing Mode]=='Direct')
Format: DstRegNum
For a send instruction, this field applies to PostDst.</td></tr>
<tr><td></td><td>8:0</td><td>Destination Address Immediate
Exists If: ([Destination Addressing Mode]=='Indirect')
Format: S8
For a send instruction, this field applies to PostDst.</td></tr>
<tr><td></td><td>4:0</td><td>Destination Subregister Number
Exists If: ([Destination Addressing Mode]=='Direct')
Format: DstSubRegNum
For a send instruction, this field applies to CurrDst.</td></tr>
</table>

EU_INSTRUCTION_OPERAND_DST_ALIGN16

EU_INSTRUCTION_OPERAND_DST_ALIGN16		
Source:	Eulsa	
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** Format: MBZ
	12:9	**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.
	8:4	**Destination Address Immediate[8:4]** Exists If: ([Destination Addressing Mode]=='Indirect') Format: S8[8:4] 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.
	3:0	**Destination Channel Enable**

EU_INSTRUCTION_OPERAND_DST_ALIGN16

		Format: ChanEn[4] For a send instruction, this field applies to the CurrDst

EU_INSTRUCTION_OPERAND_SEND_MSG

<table>
<tr><th colspan="3">EU_INSTRUCTION_OPERAND_SEND_MSG</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="3">0</td><td>31</td><td>EOT
<table>
<tr><th>Description</th></tr>
<tr><td>This field controls the termination of the thread. For a send instruction, if this field is set, EU will terminate the thread and also set the EOT bit in the message sideband. This field only applies to the send instruction. It is not present for other instructions.</td></tr>
</table>
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0</td><td>Thread is not terminated</td></tr>
<tr><td>1</td><td>EOT</td></tr>
</table>
</td></tr>
<tr><td>30:0</td><td>Message Descriptor
<table>
<tr><td></td><td></td></tr>
<tr><td>Exists If:</td><td>[SelReg32Desc]=='IMM'</td></tr>
<tr><td>Format:</td><td>MsgDescpt31</td></tr>
</table>
</td></tr>
<tr><td>30:0</td><td>Reg32
<table>
<tr><td></td><td></td></tr>
<tr><td>Exists If:</td><td>[SelReg32Desc]!='IMM'</td></tr>
</table>
In a send or sendc instruction refers to the option of providing the message descriptor field DWord, of which bits 30:0 are used, in the first two words of the Address Register rather than as an immediate operand.
</td></tr>
</table>

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1		
Source:	Eulsa	
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**
	12:9	**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**
	8:0	**Source Address Immediate [8:0]** Exists If: ([Source Addressing Mode]=='Indirect') Format: S9[8:0]
	4:0	**Source Subregister Number** Exists If: ([Source Addressing Mode]=='Direct') Format: **SrcSubRegNum**

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16

EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16		
Source:	Eulsa	
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**
	12:9	**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**
	8:4	**Source Address Immediate[8:4]** Exists If: ([Source Addressing Mode]=='Indirect') Format: S9[8:4]
	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

EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC		
Source: Eulsa Size (in bits): 21 Default Value: 0x00000000		
DWord	**Bit**	**Description**
0	20	**Source Subregister Number [1]** Format: **SrcSubRegNum[1:1]**
	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

EU_INSTRUCTION_SEND

Source:	Eulsa
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	**Reserved** Format: MBZ
	94:91	**ExDesc[31:28]** Format: **ExtMsgDescpt[31:28]**
	90:89	**Reserved** Format: MBZ
	88:85	**ExDesc[27:24]** Format: **ExtMsgDescpt[27:24]**
	84	**Reserved** Format: MBZ
	83:80	**ExDesc[23:20]** Format: **ExtMsgDescpt[23:20]**
	79:68	**Reserved** Format: MBZ
	67:64	**ExDesc[19:16]** Format: ExMsgDescpt[19:16]
	63:32	**Operand Control** Format: **EU_INSTRUCTION_OPERAND_CONTROLS**
	31:28	**Controls B** Format: **EU_INSTRUCTION_CONTROLS_B**

EU_INSTRUCTION_SEND

	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_SENDS

EU_INSTRUCTION_SENDS

Source: Eulsa

Size (in bits): 128

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

DWord	Bit	Description
0..3	127:96	**Message** Format: **EU_INSTRUCTION_OPERAND_SEND_MSG**
	95:80	**ExDesc[31:16]** Format: **ExtMsgDescpt[31:16]**
	79	**Source 0 Addressing Mode** Format: **AddrMode**
	78	**Reserved** Exists If: ([Source 0 Addressing Mode]=='Direct') Format: MBZ
	78	**Source 0 Address Immediate Sign [9]** Exists If: ([Source 0 Addressing Mode]=='Indirect') Format: S9[9:9]
	77	**SelReg32Desc** Indicate the source of Message Descriptor. Immediate value from instruction of indirect value from address registers. Value: 0 — Name: IMM Value: 1 — Name: REG32
	76:73	**Source 0 Address Subregister Number** Exists If: ([Source 0 Addressing Mode]=='Indirect')
	76:69	**Source 0 Register Number** Exists If: ([Source 0 Addressing Mode]=='Direct')
	72:68	**Source 0 Address Immediate [8:4]** Exists If: ([Source 0 Addressing Mode]=='Indirect') Format: S9[8:4]
	68	**Source 0 Subregister Number** Exists If: ([Source 0 Addressing Mode]=='Direct')
	67:64	**ExDesc[9:6]** Format: **ExtMsgDescpt[9:6]**
	63	**Destination Addressing Mode**

EU_INSTRUCTION_SENDS

Bits	Description
	Format: **AddrMode**
62	**Destination Address Immediate Sign [9]** Exists If: ([Destination Addressing Mode]=='Indirect') Format: S9[9:9]
62	**Reserved** Exists If: ([Destination Addressing Mode]=='Direct') Format: MBZ
61	**SelReg32ExDesc** Indicate the source of Extended Message Descriptor. Immediate value from instruction of indirect value from address registers. Value 0 = IMM Value 1 = REG32
60:57	**Destination Address Subregister Number** Exists If: ([Destination Addressing Mode]=='Indirect')
60:53	**Destination Register Number** Exists If: ([Destination Addressing Mode]=='Direct')
56:52	**Destination Address Immediate [8:4]** Exists If: ([Destination Addressing Mode]=='Indirect') Format: S9[8:4]
52	**Destination Subregister Number [4]** Exists If: ([Destination Addressing Mode]=='Direct')
51:44	**Source 1 Register Number**
43:41	**Reserved** Format: MBZ
40:37	**Destination Type**
36	**Source 1 Register File** Format: **RegFile[0:0]**
35	**Destination Register File** Format: **RegFile[0:0]**
34	**MaskCtrl**
33:32	**Flag Register Number/Subregister Number**
31:28	**Controls B** Format: **EU_INSTRUCTION_CONTROLS_B**
27:24	**Shared Function ID (SFID)** Format: **SFID**

EU_INSTRUCTION_SENDS

	23:8	**Controls A** Format: EU_INSTRUCTION_CONTROLS_A
	7	**Reserved** Format: MBZ
	6:0	**Opcode** Format: EU_OPCODE

EU_INSTRUCTION_SOURCES_IMM32

<table>
<tr><th colspan="3">EU_INSTRUCTION_SOURCES_IMM32</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 64
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">Single source, immediate</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0..1</td><td>63:32</td><td>Source 0 Immediate</td></tr>
<tr><td>31:25</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>24:0</td><td>Source 0
Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM')
Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16</td></tr>
<tr><td>24:0</td><td>Source 0
Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM')
Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1</td></tr>
</table>

EU_INSTRUCTION_SOURCES_REG

EU_INSTRUCTION_SOURCES_REG		
Source:	Eulsa	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Single source, register		
DWord	**Bit**	**Description**
0..1	63:25	**Reserved** Format: MBZ
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16**
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1**

EU_INSTRUCTION_SOURCES_REG_IMM

EU_INSTRUCTION_SOURCES_REG_IMM		
Source:	Eulsa	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Dual source, register and immediate		
DWord	**Bit**	**Description**
0..1	63:32	**Source 1 Immediate**
	31	**Reserved** Exists If: ([Source 0][Source Addressing Mode]=='Direct') Format: MBZ
	31	**Source 0 Address Immediate [9] (Sign Bit)** Exists If: ([Source 0][Source Addressing Mode]=='Indirect') Format: S9[9:9]
	30:27	**Src1.SrcType** Format: **SrcImmType**
	26:25	**Src1.RegFile** Format: **RegFile**
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16**
	24:0	**Source 0** Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM') Format: **EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1**

EU_INSTRUCTION_SOURCES_REG_REG

EU_INSTRUCTION_SOURCES_REG_REG

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

Dual source, both registers

DWord	Bit	Description
0..1	63:58	**Reserved** Format: MBZ
	57	**Reserved** Exists If: ([Source 1][Source Addressing Mode]=='Direct') Format: MBZ
	57	**Source 1 Address Immediate [9] (Sign Bit)** Exists If: ([Source 1][Source Addressing Mode]=='Indirect') Format: S9[9:9]
	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	**Reserved** Exists If: ([Source 0][Source Addressing Mode]=='Direct') Format: MBZ
	31	**Source 0 Address Immediate [9] (Sign Bit)** Exists If: ([Source 0][Source Addressing Mode]=='Indirect') Format: S9[9:9]
	30:27	**Src1.SrcType** Format: SrcType 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. **Value** | **Name** 11b | Reserved

<table>
<tr><th colspan="3">EU_INSTRUCTION_SOURCES_REG_REG</th></tr>
<tr><td></td><td></td><td><table><tr><th>Programming Notes</th></tr><tr><td>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.</td></tr><tr><td>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.</td></tr></table></td></tr>
<tr><td></td><td>26:25</td><td>Src1.RegFile<table><tr><td>Format:</td><td>RegFile</td></tr></table></td></tr>
<tr><td></td><td>24:0</td><td>Source 0<table><tr><td>Exists If:</td><td>(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM')</td></tr><tr><td>Format:</td><td>EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16</td></tr></table></td></tr>
<tr><td></td><td>24:0</td><td>Source 0<table><tr><td>Exists If:</td><td>(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1') AND (Structure[EU_INSTRUCTION_OPERAND_CONTROLS][Src0.RegFile]!='IMM')</td></tr><tr><td>Format:</td><td>EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1</td></tr></table></td></tr>
</table>

Event Data Payload

MDP_EVENT - Event Data Payload		
Source:	EuSubFunctionGateway	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31:24	**Reserved** Format: MBZ
	23:0	**Event ID** Format: U24 Indicates the ID of the event to be signalled.
1..7	223:0	**Reserved** Format: MBZ

Execution_Unit_Extended_Message_Descriptor

Execution_Unit_Extended_Message_Descriptor	
Source:	BSpec
Size (in bits):	11
Default Value:	0x00000000

DWord	Bit	Description
0	10	**Reserved** Format: MBZ
	9:6	**Extended Message Length** Exists If: //(Structure[EU_INSTRUCTION_HEADER][Opcode]=='Sends' OR Structure[EU_INSTRUCTION_HEADER][Opcode]=='Sendsc') Format: U4 This field specifies the number of 256-bit GRF registers starting from <src1> to be sent out on the request message payload. Valid value ranges from 0 to 15. Must be 0 when <src1> is null register. Value: [0,15] — Name:
	9:6	**Reserved** Exists If: //(Structure[EU_INSTRUCTION_HEADER][Opcode]=='Send' OR Structure[EU_INSTRUCTION_HEADER][Opcode]=='Sendc') Format: MBZ
	5	**End Of Thread** This field, if set, indicates that this is the final message of the thread and the thread's resources can be reclaimed. This bit maps to bit 127 of the send/sends instruction. Value 0 — Name: No Termination Value 1 — Name: EOT
	4	**Reserved** Format: MBZ
	3:0	**Target Function ID** This field indicates the function unit for which the message is intended. **Refer to "GPU Overview" document for the mapping of Shared Function IDs**

Extended Message Descriptor - Execution Unit

<table>
<tr><th colspan="3">Extended Message Descriptor - Execution Unit</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="5">0</td><td>31:16</td><td>Extended Function Control
Exists If: (Structure[EU_INSTRUCTION_SENDS][SelReg32ExDesc]=='IMM')
Format: U16</td></tr>
<tr><td>31:12</td><td>Extended Function Control
Exists If: (Structure[EU_INSTRUCTION_SENDS][SelReg32ExDesc]!='IMM')
Format: U20
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.</td></tr>
<tr><td>15:12</td><td>Reserved
Exists If: (Structure[EU_INSTRUCTION_SENDS][SelReg32ExDesc]=='IMM')
Format: MBZ</td></tr>
<tr><td>11</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>10:0</td><td>Execution Unit Extended Message Descriptor Definition
Format: Execution_Unit_Extended_Message_Descriptor</td></tr>
</table>

Extended Message Descriptor Render Target

Extended Message Descriptor Render Target

Extended Message Descriptor Render Target

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

DWord	Bit	Description
0	31:25	**Reserved** Format: MBZ
	24:21	**Reserved** Format: MBZ
	20	**Null Render Target** **Description** When this bit is set, RT write or read message is considered to be a dummy message and as if it is directed to the NULL render target. Setting this bit in the descriptor, allows SW to not use any entry from the Binding Table to convey NULL RT. **Programming Notes** SW must set this bit for Render Target Write just to clear the Pixel Scoreboard without allocating an entry in the Binding Table.
	19:16	**Pixel shading phase for CPS+PS inner loop** Format: U4 The loop counter value of a PS phase within CPS+PS(+S) monolithic shader; this value is same as value delivered to Pixel Interpolator when requesting input data for a new PS loop phase. Data Port uses this index to match pixel XY positions delivered by bypass path from PI hardware when a new phase started. **Programming Notes** The SIMD width of a render target read/write message with PS phase counter must match SIMD width of the Pixel Interpolator Pull message which returns PS phase counter.
	15	**Src0 Alpha Present** **Description** Setting this bit indicates that Src0 Alpha is present in the Render Target Write Message. **Programming Notes** SW must not send a header to send Src0 Alpha present, but instead, it must set this bit and avoid sending the header for RT write messages.
	14:12	**Render Target Array Index**

Extended Message Descriptor Render Target

		Format: U3 **Description** This bit-field is used to set the Render Target Index for MRT messages. **Programming Notes** SW must not send a header to send Render Target Array Index, but instead, it must set this bit-field approproately and avoid sending the header for RT messages.
	11:10	**Reserved** Format: MBZ
	9:6	**Extended Message Length** Format: U4 This field specifies the number of 256-bit GRF registers starting from <src1> to be sent out on the request message payload. Valid value ranges from 0 to 15. Must be 0 when <src1> is null register.
	5	**End of Thread** This field, if set, indicates that this is the final message of the thread and the thread's resources can be reclaimed.
	4	**Reserved** Format: MBZ
	3:0	**Target Function ID** This field indicates the function unit for which the message is intended. *Refer to "GPU Overview" document for the mapping of Shared Function IDs*

Extended Message Descriptor - Sampling Engine

Extended Message Descriptor - Sampling Engine		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:12	**Bindless Surface Offset** Format: BindlessSurfaceOffset[25:6] Specifies the bindless surface offset if the **Binding Table Index** is set to 252. Ignored otherwise. The bindless surface offset is added to the **Bindless Surface Base Address** as bits 25:6 of the byte-based address. The resulting address is the location of SURFACE_STATE for this message.
	11	**CPS Message LOD Compensation Enable** Format: Enable Specifies whether LOD Compenstation is enabled for this message. See **CPS LOD Compensation Enable** in SAMPLER_STATE for more details. **Programming Notes** This field must be disabled if the response length of the message is zero. This field must be disabled if the messages is from a 32-pixel dispatch thread. This field must be disabled unless **SIMD Mode** is SIMD8* or SIMD16*.
	10:0	**Execution Unit Extended Message Descriptor Definition** Format: Execution_Unit_Extended_Message_Descriptor

ExtMsgDescpt

ExtMsgDescpt

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

DWord	Bit	Description
0 Extended Message Descriptor Definition for SendS (Immediate)	31:12	**Extended Function Control** Format: U20 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.
	11	**Reserved** Format: MBZ
	10:6	**Extended Message Length** Format: U5 This field specifies the number of 256-bit GRF registers starting from <src1> to be sent out on the request message payload. Value: [0,15] — Name: (none) **Programming Notes**: Must be 0 when <src1> is null register.
	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

<table>
<tr><th colspan="3">ExtMsgDescpt</th></tr>
<tr><td></td><td></td><td>to the specific shared function section for details.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0000b</td><td>Null</td></tr><tr><td>0001b</td><td>Reserved</td></tr><tr><td>0010b</td><td>SamplingEngine</td></tr><tr><td>0011b</td><td>MessageGateway</td></tr><tr><td>0100b</td><td>DataCacheDataPort2</td></tr><tr><td>0101b</td><td>DataPortRenderCache</td></tr><tr><td>0110b</td><td>URB</td></tr><tr><td>0111b</td><td>ThreadSpawner</td></tr><tr><td>1000b</td><td>VideoMotionEstimation</td></tr><tr><td>1001b</td><td>DataCacheReadOnlyDataPort</td></tr><tr><td>1010b</td><td>DataCacheDataPort</td></tr><tr><td>1011b</td><td>PixelInterpolator</td></tr><tr><td>1100b</td><td>DataCacheDataPort 1</td></tr><tr><td>1101b</td><td>CheckandRefinementEngine</td></tr><tr><td>[1110b,1111b]</td><td>Reserved</td></tr></table></td></tr>
</table>

ExtMsgDescptImmediate

ExtMsgDescptImmediate		
Source:	Eulsa	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0 Extended Message Descriptor Definition for SendS (Immediate)	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.
	15:12	**Reserved** Format: MBZ
	11	**Reserved** Format: MBZ
	10	**Reserved** Format: MBZ
	9:6	**Extended Message Length** Format: U4 This field specifies the number of 256-bit GRF registers starting from <src1> to be sent out on the request message payload. Value: [0,15] — Name: (none) **Programming Notes** Must be 0 when <src1> is null register.
	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

ExtMsgDescptImmediate		
	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	Name
0000b	Null
0001b	Reserved
0010b	SamplingEngine
0011b	MessageGateway
0100b	DataCacheDataPort2
0101b	DataPortRenderCache
0110b	URB
0111b	ThreadSpawner
1000b	VideoMotionEstimation
1001b	DataCacheReadOnlyDataPort
1010b	DataCacheDataPort
1011b	PixelInterpolator
1100b	DataCacheDataPort 1
1101b	CheckandRefinementEngine
[1110b,1111b]	Reserved

FFTID Message Header Control

MHC_FFTID - FFTID Message Header Control

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

DWord	Bit	Description
0	31:8	**Reserved** Format: MBZ Ignored
	7:0	**FFTID** Format: U8 Fixed function thread ID, used to free up resources by the thread on thread completion.

Filter_Coefficient

Filter_Coefficient		
Source:	BSpec	
Size (in bits):	8	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	7:0	**Filter Coefficient** Format: S1.6 2's Complement Range : [-1 63/64, +1 63/64]

Filter_Coefficients

Filter_Coefficients		
Source:	BSpec	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	63:56	**Filter Coefficient Offset 7** Format: Filter_Coefficient
	55:48	**Filter Coefficient Offset 6** Format: Filter_Coefficient
	47:40	**Filter Coefficient Offset 5** Format: Filter_Coefficient
	39:32	**Filter Coefficient Offset 4** Format: Filter_Coefficient
	31:24	**Filter Coefficient Offset 3** Format: Filter_Coefficient
	23:16	**Filter Coefficient Offset 2** Format: Filter_Coefficient
	15:8	**Filter Coefficient Offset 1** Format: Filter_Coefficient
	7:0	**Filter Coefficient Offset 0** Format: Filter_Coefficient

FrameDeltaQp

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

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

FunctionControl		
Source:	Eulsa	
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	INVM
1111b	RSQRTM

Gamut_Expansion_Gamma_Correction

Gamut_Expansion_Gamma_Correction	
Source:	VideoEnhancementCS
Size (in bits):	32768
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x01000100, 0x01000100, 0x01000100, 0x01000100, 0x02000200, 0x02000200, 0x02000200, 0x02000200, 0x03000300, 0x03000300, 0x03000300, 0x03000300, 0x04000400, 0x04000400, 0x04000400, 0x04000400, 0x05000500, 0x05000500, 0x05000500, 0x05000500, 0x06000600, 0x06000600, 0x06000600, 0x06000600, 0x07000700, 0x07000700, 0x07000700, 0x07000700, 0x08000800, 0x08000800, 0x08000800, 0x08000800, 0x09000900, 0x09000900, 0x09000900, 0x09000900, 0x0A000A00, 0x0A000A00, 0x0A000A00, 0x0A000A00, 0x0B000B00, 0x0B000B00, 0x0B000B00, 0x0B000B00, 0x0C000C00, 0x0C000C00, 0x0C000C00, 0x0C000C00, 0x0D000D00, 0x0D000D00, 0x0D000D00, 0x0D000D00, 0x0E000E00, 0x0E000E00, 0x0E000E00, 0x0E000E00, 0x0F000F00, 0x0F000F00, 0x0F000F00, 0x0F000F00, 0x10001000, 0x10001000, 0x10001000, 0x10001000, 0x11001100, 0x11001100, 0x11001100, 0x11001100, 0x12001200, 0x12001200, 0x12001200, 0x12001200, 0x13001300, 0x13001300, 0x13001300, 0x13001300, 0x14001400, 0x14001400, 0x14001400, 0x14001400, 0x15001500, 0x15001500, 0x15001500, 0x15001500, 0x16001600, 0x16001600, 0x16001600, 0x16001600, 0x17001700, 0x17001700, 0x17001700, 0x17001700, 0x18001800, 0x18001800, 0x18001800, 0x18001800, 0x19001900, 0x19001900, 0x19001900, 0x19001900, 0x1A001A00, 0x1A001A00, 0x1A001A00, 0x1A001A00, 0x1B001B00, 0x1B001B00, 0x1B001B00, 0x1B001B00, 0x1C001C00, 0x1C001C00, 0x1C001C00, 0x1C001C00, 0x1D001D00, 0x1D001D00, 0x1D001D00, 0x1D001D00, 0x1E001E00, 0x1E001E00, 0x1E001E00, 0x1E001E00, 0x1F001F00, 0x1F001F00, 0x1F001F00, 0x1F001F00, 0x20002000, 0x20002000, 0x20002000, 0x20002000, 0x21002100, 0x21002100, 0x21002100, 0x21002100, 0x22002200, 0x22002200, 0x22002200, 0x22002200, 0x23002300, 0x23002300, 0x23002300, 0x23002300, 0x24002400, 0x24002400, 0x24002400, 0x24002400, 0x25002500, 0x25002500, 0x25002500, 0x25002500, 0x26002600, 0x26002600, 0x26002600, 0x26002600, 0x27002700, 0x27002700, 0x27002700, 0x27002700, 0x28002800, 0x28002800, 0x28002800, 0x28002800, 0x29002900, 0x29002900, 0x29002900, 0x29002900, 0x2A002A00, 0x2A002A00, 0x2A002A00, 0x2A002A00, 0x2B002B00, 0x2B002B00, 0x2B002B00, 0x2B002B00, 0x2C002C00, 0x2C002C00, 0x2C002C00, 0x2C002C00, 0x2D002D00, 0x2D002D00, 0x2D002D00, 0x2D002D00, 0x2E002E00, 0x2E002E00, 0x2E002E00, 0x2E002E00, 0x2F002F00, 0x2F002F00, 0x2F002F00, 0x2F002F00, 0x30003000, 0x30003000, 0x30003000, 0x30003000, 0x31003100, 0x31003100, 0x31003100, 0x31003100, 0x32003200, 0x32003200, 0x32003200, 0x32003200, 0x33003300, 0x33003300, 0x33003300, 0x33003300, 0x34003400, 0x34003400, 0x34003400, 0x34003400, 0x35003500, 0x35003500, 0x35003500, 0x35003500, 0x36003600, 0x36003600, 0x36003600, 0x36003600, 0x37003700, 0x37003700, 0x37003700, 0x37003700, 0x38003800, 0x38003800, 0x38003800, 0x38003800, 0x39003900, 0x39003900, 0x39003900, 0x39003900, 0x3A003A00, 0x3A003A00, 0x3A003A00, 0x3A003A00, 0x3B003B00, 0x3B003B00, 0x3B003B00, 0x3B003B00, 0x3C003C00, 0x3C003C00, 0x3C003C00, 0x3C003C00, 0x3D003D00, 0x3D003D00, 0x3D003D00,

Gamut_Expansion_Gamma_Correction	
	0x3D003D00, 0x3E003E00, 0x3E003E00, 0x3E003E00, 0x3E003E00, 0x3F003F00, 0x3F003F00, 0x3F003F00, 0x3F003F00, 0x40004000, 0x40004000, 0x40004000, 0x40004000, 0x41004100, 0x41004100, 0x41004100, 0x41004100, 0x42004200, 0x42004200, 0x42004200, 0x42004200, 0x43004300, 0x43004300, 0x43004300, 0x43004300, 0x44004400, 0x44004400, 0x44004400, 0x44004400, 0x45004500, 0x45004500, 0x45004500, 0x45004500, 0x46004600, 0x46004600, 0x46004600, 0x46004600, 0x47004700, 0x47004700, 0x47004700, 0x47004700, 0x48004800, 0x48004800, 0x48004800, 0x48004800, 0x49004900, 0x49004900, 0x49004900, 0x49004900, 0x4A004A00, 0x4A004A00, 0x4A004A00, 0x4A004A00, 0x4B004B00, 0x4B004B00, 0x4B004B00, 0x4B004B00, 0x4C004C00, 0x4C004C00, 0x4C004C00, 0x4C004C00, 0x4D004D00, 0x4D004D00, 0x4D004D00, 0x4D004D00, 0x4E004E00, 0x4E004E00, 0x4E004E00, 0x4E004E00, 0x4F004F00, 0x4F004F00, 0x4F004F00, 0x4F004F00, 0x50005000, 0x50005000, 0x50005000, 0x50005000, 0x51005100, 0x51005100, 0x51005100, 0x51005100, 0x52005200, 0x52005200, 0x52005200, 0x52005200, 0x53005300, 0x53005300, 0x53005300, 0x53005300, 0x54005400, 0x54005400, 0x54005400, 0x54005400, 0x55005500, 0x55005500, 0x55005500, 0x55005500, 0x56005600, 0x56005600, 0x56005600, 0x56005600, 0x57005700, 0x57005700, 0x57005700, 0x57005700, 0x58005800, 0x58005800, 0x58005800, 0x58005800, 0x59005900, 0x59005900, 0x59005900, 0x59005900, 0x5A005A00, 0x5A005A00, 0x5A005A00, 0x5A005A00, 0x5B005B00, 0x5B005B00, 0x5B005B00, 0x5B005B00, 0x5C005C00, 0x5C005C00, 0x5C005C00, 0x5C005C00, 0x5D005D00, 0x5D005D00, 0x5D005D00, 0x5D005D00, 0x5E005E00, 0x5E005E00, 0x5E005E00, 0x5E005E00, 0x5F005F00, 0x5F005F00, 0x5F005F00, 0x5F005F00, 0x60006000, 0x60006000, 0x60006000, 0x60006000, 0x61006100, 0x61006100, 0x61006100, 0x61006100, 0x62006200, 0x62006200, 0x62006200, 0x62006200, 0x63006300, 0x63006300, 0x63006300, 0x63006300, 0x64006400, 0x64006400, 0x64006400, 0x64006400, 0x65006500, 0x65006500, 0x65006500, 0x65006500, 0x66006600, 0x66006600, 0x66006600, 0x66006600, 0x67006700, 0x67006700, 0x67006700, 0x67006700, 0x68006800, 0x68006800, 0x68006800, 0x68006800, 0x69006900, 0x69006900, 0x69006900, 0x69006900, 0x6A006A00, 0x6A006A00, 0x6A006A00, 0x6A006A00, 0x6B006B00, 0x6B006B00, 0x6B006B00, 0x6B006B00, 0x6C006C00, 0x6C006C00, 0x6C006C00, 0x6C006C00, 0x6D006D00, 0x6D006D00, 0x6D006D00, 0x6D006D00, 0x6E006E00, 0x6E006E00, 0x6E006E00, 0x6E006E00, 0x6F006F00, 0x6F006F00, 0x6F006F00, 0x6F006F00, 0x70007000, 0x70007000, 0x70007000, 0x70007000, 0x71007100, 0x71007100, 0x71007100, 0x71007100, 0x72007200, 0x72007200, 0x72007200, 0x72007200, 0x73007300, 0x73007300, 0x73007300, 0x73007300, 0x74007400, 0x74007400, 0x74007400, 0x74007400, 0x75007500, 0x75007500, 0x75007500, 0x75007500, 0x76007600, 0x76007600, 0x76007600, 0x76007600, 0x77007700, 0x77007700, 0x77007700, 0x77007700, 0x78007800, 0x78007800, 0x78007800, 0x78007800, 0x79007900, 0x79007900, 0x79007900, 0x79007900, 0x7A007A00, 0x7A007A00, 0x7A007A00, 0x7A007A00, 0x7B007B00, 0x7B007B00, 0x7B007B00, 0x7B007B00, 0x7C007C00, 0x7C007C00, 0x7C007C00, 0x7C007C00, 0x7D007D00, 0x7D007D00, 0x7D007D00, 0x7D007D00, 0x7E007E00, 0x7E007E00, 0x7E007E00, 0x7E007E00, 0x7F007F00, 0x7F007F00, 0x7F007F00, 0x7F007F00, 0x80008000, 0x80008000, 0x80008000, 0x80008000, 0x81008100, 0x81008100, 0x81008100, 0x81008100, 0x82008200, 0x82008200, 0x82008200, 0x82008200, 0x83008300,

Gamut_Expansion_Gamma_Correction
0x83008300, 0x83008300, 0x83008300, 0x84008400, 0x84008400, 0x84008400, 0x84008400, 0x85008500, 0x85008500, 0x85008500, 0x85008500, 0x86008600, 0x86008600, 0x86008600, 0x86008600, 0x87008700, 0x87008700, 0x87008700, 0x87008700, 0x88008800, 0x88008800, 0x88008800, 0x88008800, 0x89008900, 0x89008900, 0x89008900, 0x89008900, 0x8A008A00, 0x8A008A00, 0x8A008A00, 0x8A008A00, 0x8B008B00, 0x8B008B00, 0x8B008B00, 0x8B008B00, 0x8C008C00, 0x8C008C00, 0x8C008C00, 0x8C008C00, 0x8D008D00, 0x8D008D00, 0x8D008D00, 0x8D008D00, 0x8E008E00, 0x8E008E00, 0x8E008E00, 0x8E008E00, 0x8F008F00, 0x8F008F00, 0x8F008F00, 0x8F008F00, 0x90009000, 0x90009000, 0x90009000, 0x90009000, 0x91009100, 0x91009100, 0x91009100, 0x91009100, 0x92009200, 0x92009200, 0x92009200, 0x92009200, 0x93009300, 0x93009300, 0x93009300, 0x93009300, 0x94009400, 0x94009400, 0x94009400, 0x94009400, 0x95009500, 0x95009500, 0x95009500, 0x95009500, 0x96009600, 0x96009600, 0x96009600, 0x96009600, 0x97009700, 0x97009700, 0x97009700, 0x97009700, 0x98009800, 0x98009800, 0x98009800, 0x98009800, 0x99009900, 0x99009900, 0x99009900, 0x99009900, 0x9A009A00, 0x9A009A00, 0x9A009A00, 0x9A009A00, 0x9B009B00, 0x9B009B00, 0x9B009B00, 0x9B009B00, 0x9C009C00, 0x9C009C00, 0x9C009C00, 0x9C009C00, 0x9D009D00, 0x9D009D00, 0x9D009D00, 0x9D009D00, 0x9E009E00, 0x9E009E00, 0x9E009E00, 0x9E009E00, 0x9F009F00, 0x9F009F00, 0x9F009F00, 0x9F009F00, 0xA000A000, 0xA000A000, 0xA000A000, 0xA000A000, 0xA100A100, 0xA100A100, 0xA100A100, 0xA100A100, 0xA200A200, 0xA200A200, 0xA200A200, 0xA200A200, 0xA300A300, 0xA300A300, 0xA300A300, 0xA300A300, 0xA400A400, 0xA400A400, 0xA400A400, 0xA400A400, 0xA500A500, 0xA500A500, 0xA500A500, 0xA500A500, 0xA600A600, 0xA600A600, 0xA600A600, 0xA600A600, 0xA700A700, 0xA700A700, 0xA700A700, 0xA700A700, 0xA800A800, 0xA800A800, 0xA800A800, 0xA800A800, 0xA900A900, 0xA900A900, 0xA900A900, 0xA900A900, 0xAA00AA00, 0xAA00AA00, 0xAA00AA00, 0xAA00AA00, 0xAB00AB00, 0xAB00AB00, 0xAB00AB00, 0xAB00AB00, 0xAC00AC00, 0xAC00AC00, 0xAC00AC00, 0xAC00AC00, 0xAD00AD00, 0xAD00AD00, 0xAD00AD00, 0xAD00AD00, 0xAE00AE00, 0xAE00AE00, 0xAE00AE00, 0xAE00AE00, 0xAF00AF00, 0xAF00AF00, 0xAF00AF00, 0xAF00AF00, 0xB000B000, 0xB000B000, 0xB000B000, 0xB000B000, 0xB100B100, 0xB100B100, 0xB100B100, 0xB100B100, 0xB200B200, 0xB200B200, 0xB200B200, 0xB200B200, 0xB300B300, 0xB300B300, 0xB300B300, 0xB300B300, 0xB400B400, 0xB400B400, 0xB400B400, 0xB400B400, 0xB500B500, 0xB500B500, 0xB500B500, 0xB500B500, 0xB600B600, 0xB600B600, 0xB600B600, 0xB600B600, 0xB700B700, 0xB700B700, 0xB700B700, 0xB700B700, 0xB800B800, 0xB800B800, 0xB800B800, 0xB800B800, 0xB900B900, 0xB900B900, 0xB900B900, 0xB900B900, 0xBA00BA00, 0xBA00BA00, 0xBA00BA00, 0xBA00BA00, 0xBB00BB00, 0xBB00BB00, 0xBB00BB00, 0xBB00BB00, 0xBC00BC00, 0xBC00BC00, 0xBC00BC00, 0xBC00BC00, 0xBD00BD00, 0xBD00BD00, 0xBD00BD00, 0xBD00BD00, 0xBE00BE00, 0xBE00BE00, 0xBE00BE00, 0xBE00BE00, 0xBF00BF00, 0xBF00BF00, 0xBF00BF00, 0xBF00BF00, 0xC000C000, 0xC000C000, 0xC000C000, 0xC000C000, 0xC100C100, 0xC100C100, 0xC100C100, 0xC100C100, 0xC200C200, 0xC200C200, 0xC200C200, 0xC200C200, 0xC300C300, 0xC300C300, 0xC300C300, 0xC300C300, 0xC400C400, 0xC400C400, 0xC400C400, 0xC400C400, 0xC500C500, 0xC500C500, 0xC500C500, 0xC500C500, 0xC600C600, 0xC600C600, 0xC600C600, 0xC600C600, 0xC700C700, 0xC700C700, 0xC700C700, 0xC700C700, 0xC800C800, 0xC800C800,

Gamut_Expansion_Gamma_Correction	
	0xC800C800, 0xC800C800, 0xC900C900, 0xC900C900, 0xC900C900, 0xC900C900, 0xCA00CA00, 0xCA00CA00, 0xCA00CA00, 0xCA00CA00, 0xCB00CB00, 0xCB00CB00, 0xCB00CB00, 0xCB00CB00, 0xCC00CC00, 0xCC00CC00, 0xCC00CC00, 0xCC00CC00, 0xCD00CD00, 0xCD00CD00, 0xCD00CD00, 0xCD00CD00, 0xCE00CE00, 0xCE00CE00, 0xCE00CE00, 0xCE00CE00, 0xCF00CF00, 0xCF00CF00, 0xCF00CF00, 0xCF00CF00, 0xD000D000, 0xD000D000, 0xD000D000, 0xD000D000, 0xD100D100, 0xD100D100, 0xD100D100, 0xD100D100, 0xD200D200, 0xD200D200, 0xD200D200, 0xD200D200, 0xD300D300, 0xD300D300, 0xD300D300, 0xD300D300, 0xD400D400, 0xD400D400, 0xD400D400, 0xD400D400, 0xD500D500, 0xD500D500, 0xD500D500, 0xD500D500, 0xD600D600, 0xD600D600, 0xD600D600, 0xD600D600, 0xD700D700, 0xD700D700, 0xD700D700, 0xD700D700, 0xD800D800, 0xD800D800, 0xD800D800, 0xD800D800, 0xD900D900, 0xD900D900, 0xD900D900, 0xD900D900, 0xDA00DA00, 0xDA00DA00, 0xDA00DA00, 0xDA00DA00, 0xDB00DB00, 0xDB00DB00, 0xDB00DB00, 0xDB00DB00, 0xDC00DC00, 0xDC00DC00, 0xDC00DC00, 0xDC00DC00, 0xDD00DD00, 0xDD00DD00, 0xDD00DD00, 0xDD00DD00, 0xDE00DE00, 0xDE00DE00, 0xDE00DE00, 0xDE00DE00, 0xDF00DF00, 0xDF00DF00, 0xDF00DF00, 0xDF00DF00, 0xE000E000, 0xE000E000, 0xE000E000, 0xE000E000, 0xE100E100, 0xE100E100, 0xE100E100, 0xE100E100, 0xE200E200, 0xE200E200, 0xE200E200, 0xE200E200, 0xE300E300, 0xE300E300, 0xE300E300, 0xE300E300, 0xE400E400, 0xE400E400, 0xE400E400, 0xE400E400, 0xE500E500, 0xE500E500, 0xE500E500, 0xE500E500, 0xE600E600, 0xE600E600, 0xE600E600, 0xE600E600, 0xE700E700, 0xE700E700, 0xE700E700, 0xE700E700, 0xE800E800, 0xE800E800, 0xE800E800, 0xE800E800, 0xE900E900, 0xE900E900, 0xE900E900, 0xE900E900, 0xEA00EA00, 0xEA00EA00, 0xEA00EA00, 0xEA00EA00, 0xEB00EB00, 0xEB00EB00, 0xEB00EB00, 0xEB00EB00, 0xEC00EC00, 0xEC00EC00, 0xEC00EC00, 0xEC00EC00, 0xED00ED00, 0xED00ED00, 0xED00ED00, 0xED00ED00, 0xEE00EE00, 0xEE00EE00, 0xEE00EE00, 0xEE00EE00, 0xEF00EF00, 0xEF00EF00, 0xEF00EF00, 0xEF00EF00, 0xF000F000, 0xF000F000, 0xF000F000, 0xF000F000, 0xF100F100, 0xF100F100, 0xF100F100, 0xF100F100, 0xF200F200, 0xF200F200, 0xF200F200, 0xF200F200, 0xF300F300, 0xF300F300, 0xF300F300, 0xF300F300, 0xF400F400, 0xF400F400, 0xF400F400, 0xF400F400, 0xF500F500, 0xF500F500, 0xF500F500, 0xF500F500, 0xF600F600, 0xF600F600, 0xF600F600, 0xF600F600, 0xF700F700, 0xF700F700, 0xF700F700, 0xF700F700, 0xF800F800, 0xF800F800, 0xF800F800, 0xF800F800, 0xF900F900, 0xF900F900, 0xF900F900, 0xF900F900, 0xFA00FA00, 0xFA00FA00, 0xFA00FA00, 0xFA00FA00, 0xFB00FB00, 0xFB00FB00, 0xFB00FB00, 0xFB00FB00, 0xFC00FC00, 0xFC00FC00, 0xFC00FC00, 0xFC00FC00, 0xFD00FD00, 0xFD00FD00, 0xFD00FD00, 0xFD00FD00, 0xFE00FE00, 0xFE00FE00, 0xFE00FE00, 0xFE00FE00, 0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF

Programming Notes

The default values follow the pattern suggested by incomplete table below.

DWords	DWord 0	DWord 1	DWord 2	DWord 3
0..3 : Point[0]	00000000h	00000000h	00000000h	00000000h
4..7 : Point[1]	01000100h	01000100h	01000100h	01000100h
8..11 : Point[2]	02000200h	02000200h	02000200h	02000200h
12..15 : Point[3]	03000300h	03000300h	03000300h	03000300h
...				

Gamut_Expansion_Gamma_Correction

1016..1019 : Point[254]	fe00fe00h	fe00fe00h	fe00fe00h	fe00fe00h
1020..1023 : Point[255]	ffffffffh	ffffffffh	ffffffffh	ffffffffh

DWord	Bit	Description
0..1	63:48	**Inverse R-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
	47:32	**Inverse Pixel Value 0** Default Value: 0000h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
2..3	63:48	**Forward R-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
	47:32	**Forward Pixel Value 0** Default Value: 0000h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 0** Default Value: 0000h Format: U16
4..5	63:48	**Inverse R-ch Gamma Corrected Value 1** Default Value: 0100h Format: U16
	47:32	**Inverse Pixel Value 1** Default Value: 0100h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 1** Default Value: 0100h

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 1**	
		Default Value:	0100h
		Format:	U16
6..7	63:48	**Forward R-ch Gamma Corrected Value 1**	
		Default Value:	0100h
		Format:	U16
	47:32	**Forward Pixel Value 1**	
		Default Value:	0100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 1**	
		Default Value:	0100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 1**	
		Default Value:	0100h
		Format:	U16
8..9	63:48	**Inverse R-ch Gamma Corrected Value 2**	
		Default Value:	0200h
		Format:	U16
	47:32	**Inverse Pixel Value 2**	
		Default Value:	0200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 2**	
		Default Value:	0200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 2**	
		Default Value:	0200h
		Format:	U16
10..11	63:48	**Forward R-ch Gamma Corrected Value 2**	
		Default Value:	0200h
		Format:	U16
	47:32	**Forward Pixel Value 2**	
		Default Value:	0200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 2**	
		Default Value:	0200h

![intel]

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 2**	
		Default Value:	0200h
		Format:	U16
12..13	63:48	**Inverse R-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
	47:32	**Inverse Pixel Value 3**	
		Default Value:	0300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
14..15	63:48	**Forward R-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
	47:32	**Forward Pixel Value 3**	
		Default Value:	0300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 3**	
		Default Value:	0300h
		Format:	U16
16..17	63:48	**Inverse R-ch Gamma Corrected Value 4**	
		Default Value:	0400h
		Format:	U16
	47:32	**Inverse Pixel Value 4**	
		Default Value:	0400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 4**	
		Default Value:	0400h

Gamut_Expansion_Gamma_Correction			
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 4**	
		Default Value:	0400h
		Format:	U16
18..19	63:48	**Forward R-ch Gamma Corrected Value 4**	
		Default Value:	0400h
		Format:	U16
	47:32	**Forward Pixel Value 4**	
		Default Value:	0400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 4**	
		Default Value:	0400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 4**	
		Default Value:	0400h
		Format:	U16
20..21	63:48	**Inverse R-ch Gamma Corrected Value 5**	
		Default Value:	0500h
		Format:	U16
	47:32	**Inverse Pixel Value 5**	
		Default Value:	0500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 5**	
		Default Value:	0500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 5**	
		Default Value:	0500h
		Format:	U16
22..23	63:48	**Forward R-ch Gamma Corrected Value 5**	
		Default Value:	0500h
		Format:	U16
	47:32	**Forward Pixel Value 5**	
		Default Value:	0500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 5**	
		Default Value:	0500h

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 5**	
		Default Value:	0500h
		Format:	U16
24..25	63:48	**Inverse R-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
	47:32	**Inverse Pixel Value 6**	
		Default Value:	0600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
26..27	63:48	**Forward R-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
	47:32	**Forward Pixel Value 6**	
		Default Value:	0600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 6**	
		Default Value:	0600h
		Format:	U16
28..29	63:48	**Inverse R-ch Gamma Corrected Value 7**	
		Default Value:	0700h
		Format:	U16
	47:32	**Inverse Pixel Value 7**	
		Default Value:	0700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 7**	
		Default Value:	0700h

Gamut_Expansion_Gamma_Correction			
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 7**	
		Default Value:	0700h
		Format:	U16
30..31	63:48	**Forward R-ch Gamma Corrected Value 7**	
		Default Value:	0700h
		Format:	U16
	47:32	**Forward Pixel Value 7**	
		Default Value:	0700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 7**	
		Default Value:	0700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 7**	
		Default Value:	0700h
		Format:	U16
32..33	63:48	**Inverse R-ch Gamma Corrected Value 8**	
		Default Value:	0800h
		Format:	U16
	47:32	**Inverse Pixel Value 8**	
		Default Value:	0800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 8**	
		Default Value:	0800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 8**	
		Default Value:	0800h
		Format:	U16
34..35	63:48	**Forward R-ch Gamma Corrected Value 8**	
		Default Value:	0800h
		Format:	U16
	47:32	**Forward Pixel Value 8**	
		Default Value:	0800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 8**	
		Default Value:	0800h

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 8**	
		Default Value:	0800h
		Format:	U16
36..37	63:48	**Inverse R-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
	47:32	**Inverse Pixel Value 9**	
		Default Value:	0900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
38..39	63:48	**Forward R-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
	47:32	**Forward Pixel Value 9**	
		Default Value:	0900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 9**	
		Default Value:	0900h
		Format:	U16
40..41	63:48	**Inverse R-ch Gamma Corrected Value 10**	
		Default Value:	0a00h
		Format:	U16
	47:32	**Inverse Pixel Value 10**	
		Default Value:	0a00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 10**	
		Default Value:	0a00h

Gamut_Expansion_Gamma_Correction			
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 10**	
		Default Value:	0a00h
		Format:	U16
42..43	63:48	**Forward R-ch Gamma Corrected Value 10**	
		Default Value:	0a00h
		Format:	U16
	47:32	**Forward Pixel Value 10**	
		Default Value:	0a00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 10**	
		Default Value:	0a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 10**	
		Default Value:	0a00h
		Format:	U16
44..45	63:48	**Inverse R-ch Gamma Corrected Value 11**	
		Default Value:	0b00h
		Format:	U16
	47:32	**Inverse Pixel Value 11**	
		Default Value:	0b00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 11**	
		Default Value:	0b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 11**	
		Default Value:	0b00h
		Format:	U16
46..47	63:48	**Forward R-ch Gamma Corrected Value 11**	
		Default Value:	0b00h
		Format:	U16
	47:32	**Forward Pixel Value 11**	
		Default Value:	0b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 11**	
		Default Value:	0b00h

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 11**	
		Default Value:	0b00h
		Format:	U16
48..49	63:48	**Inverse R-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
	47:32	**Inverse Pixel Value 12**	
		Default Value:	0c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
50..51	63:48	**Forward R-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
	47:32	**Forward Pixel Value 12**	
		Default Value:	0c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 12**	
		Default Value:	0c00h
		Format:	U16
52..53	63:48	**Inverse R-ch Gamma Corrected Value 13**	
		Default Value:	0d00h
		Format:	U16
	47:32	**Inverse Pixel Value 13**	
		Default Value:	0d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 13**	
		Default Value:	0d00h

Gamut_Expansion_Gamma_Correction			
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 13**	
		Default Value:	0d00h
		Format:	U16
54..55	63:48	**Forward R-ch Gamma Corrected Value 13**	
		Default Value:	0d00h
		Format:	U16
	47:32	**Forward Pixel Value 13**	
		Default Value:	0d00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 13**	
		Default Value:	0d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 13**	
		Default Value:	0d00h
		Format:	U16
56..57	63:48	**Inverse R-ch Gamma Corrected Value 14**	
		Default Value:	0e00h
		Format:	U16
	47:32	**Inverse Pixel Value 14**	
		Default Value:	0e00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 14**	
		Default Value:	0e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 14**	
		Default Value:	0e00h
		Format:	U16
58..59	63:48	**Forward R-ch Gamma Corrected Value 14**	
		Default Value:	0e00h
		Format:	U16
	47:32	**Forward Pixel Value 14**	
		Default Value:	0e00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 14**	
		Default Value:	0e00h

Gamut_Expansion_Gamma_Correction

		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 14**	
		Default Value:	0e00h
		Format:	U16
60..61	63:48	**Inverse R-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
	47:32	**Inverse Pixel Value 15**	
		Default Value:	0f00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
62..63	63:48	**Forward R-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
	47:32	**Forward Pixel Value 15**	
		Default Value:	0f00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 15**	
		Default Value:	0f00h
		Format:	U16
64..65	63:48	**Inverse R-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
	47:32	**Inverse Pixel Value 16**	
		Default Value:	1000h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
66..67	63:48	**Forward R-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
	47:32	**Forward Pixel Value 16**	
		Default Value:	1000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 16**	
		Default Value:	1000h
		Format:	U16
68..69	63:48	**Inverse R-ch Gamma Corrected Value 17**	
		Default Value:	1100h
		Format:	U16
	47:32	**Inverse Pixel Value 17**	
		Default Value:	1100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 17**	
		Default Value:	1100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 17**	
		Default Value:	1100h
		Format:	U16
70..71	63:48	**Forward R-ch Gamma Corrected Value 17**	
		Default Value:	1100h
		Format:	U16
	47:32	**Forward Pixel Value 17**	
		Default Value:	1100h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 17** Default Value: 1100h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 17** Default Value: 1100h Format: U16
72..73	63:48	**Inverse R-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
	47:32	**Inverse Pixel Value 18** Default Value: 1200h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
74..75	63:48	**Forward R-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
	47:32	**Forward Pixel Value 18** Default Value: 1200h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 18** Default Value: 1200h Format: U16
76..77	63:48	**Inverse R-ch Gamma Corrected Value 19** Default Value: 1300h Format: U16
	47:32	**Inverse Pixel Value 19** Default Value: 1300h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 19**	
		Default Value:	1300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 19**	
		Default Value:	1300h
		Format:	U16
78..79	63:48	**Forward R-ch Gamma Corrected Value 19**	
		Default Value:	1300h
		Format:	U16
	47:32	**Forward Pixel Value 19**	
		Default Value:	1300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 19**	
		Default Value:	1300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 19**	
		Default Value:	1300h
		Format:	U16
80..81	63:48	**Inverse R-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
	47:32	**Inverse Pixel Value 20**	
		Default Value:	1400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
82..83	63:48	**Forward R-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
	47:32	**Forward Pixel Value 20**	
		Default Value:	1400h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 20**	
		Default Value:	1400h
		Format:	U16
84..85	63:48	**Inverse R-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
	47:32	**Inverse Pixel Value 21**	
		Default Value:	1500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
86..87	63:48	**Forward R-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
	47:32	**Forward Pixel Value 21**	
		Default Value:	1500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 21**	
		Default Value:	1500h
		Format:	U16
88..89	63:48	**Inverse R-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
	47:32	**Inverse Pixel Value 22**	
		Default Value:	1600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
90..91	63:48	**Forward R-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
	47:32	**Forward Pixel Value 22**	
		Default Value:	1600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 22**	
		Default Value:	1600h
		Format:	U16
92..93	63:48	**Inverse R-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
	47:32	**Inverse Pixel Value 23**	
		Default Value:	1700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
94..95	63:48	**Forward R-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
	47:32	**Forward Pixel Value 23**	
		Default Value:	1700h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 23**	
		Default Value:	1700h
		Format:	U16
96..97	63:48	**Inverse R-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
	47:32	**Inverse Pixel Value 24**	
		Default Value:	1800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
98..99	63:48	**Forward R-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
	47:32	**Forward Pixel Value 24**	
		Default Value:	1800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 24**	
		Default Value:	1800h
		Format:	U16
100..101	63:48	**Inverse R-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
	47:32	**Inverse Pixel Value 25**	
		Default Value:	1900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
102..103	63:48	**Forward R-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
	47:32	**Forward Pixel Value 25**	
		Default Value:	1900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 25**	
		Default Value:	1900h
		Format:	U16
104..105	63:48	**Inverse R-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
	47:32	**Inverse Pixel Value 26**	
		Default Value:	1a00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
106..107	63:48	**Forward R-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
	47:32	**Forward Pixel Value 26**	
		Default Value:	1a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 26**	
		Default Value:	1a00h
		Format:	U16
108..109	63:48	**Inverse R-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
	47:32	**Inverse Pixel Value 27**	
		Default Value:	1b00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
110..111	63:48	**Forward R-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
	47:32	**Forward Pixel Value 27**	
		Default Value:	1b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 27**	
		Default Value:	1b00h
		Format:	U16
112..113	63:48	**Inverse R-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
	47:32	**Inverse Pixel Value 28**	
		Default Value:	1c00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
114..115	63:48	**Forward R-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
	47:32	**Forward Pixel Value 28**	
		Default Value:	1c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 28**	
		Default Value:	1c00h
		Format:	U16
116..117	63:48	**Inverse R-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
	47:32	**Inverse Pixel Value 29**	
		Default Value:	1d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
118..119	63:48	**Forward R-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
	47:32	**Forward Pixel Value 29**	
		Default Value:	1d00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 29**	
		Default Value:	1d00h
		Format:	U16
120..121	63:48	**Inverse R-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
	47:32	**Inverse Pixel Value 30**	
		Default Value:	1e00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
122..123	63:48	**Forward R-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
	47:32	**Forward Pixel Value 30**	
		Default Value:	1e00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 30**	
		Default Value:	1e00h
		Format:	U16
124..125	63:48	**Inverse R-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
	47:32	**Inverse Pixel Value 31**	
		Default Value:	1f00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
126..127	63:48	**Forward R-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
	47:32	**Forward Pixel Value 31**	
		Default Value:	1f00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 31**	
		Default Value:	1f00h
		Format:	U16
128..129	63:48	**Inverse R-ch Gamma Corrected Value 32**	
		Default Value:	2000h
		Format:	U16
	47:32	**Inverse Pixel Value 32**	
		Default Value:	2000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 32**	
		Default Value:	2000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 32**	
		Default Value:	2000h
		Format:	U16
130..131	63:48	**Forward R-ch Gamma Corrected Value 32**	
		Default Value:	2000h
		Format:	U16
	47:32	**Forward Pixel Value 32**	
		Default Value:	2000h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 32** Default Value: 2000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 32** Default Value: 2000h Format: U16
132..133	63:48	**Inverse R-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
	47:32	**Inverse Pixel Value 33** Default Value: 2100h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
134..135	63:48	**Forward R-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
	47:32	**Forward Pixel Value 33** Default Value: 2100h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 33** Default Value: 2100h Format: U16
136..137	63:48	**Inverse R-ch Gamma Corrected Value 34** Default Value: 2200h Format: U16
	47:32	**Inverse Pixel Value 34** Default Value: 2200h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 34**	
		Default Value:	2200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 34**	
		Default Value:	2200h
		Format:	U16
138..139	63:48	**Forward R-ch Gamma Corrected Value 34**	
		Default Value:	2200h
		Format:	U16
	47:32	**Forward Pixel Value 34**	
		Default Value:	2200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 34**	
		Default Value:	2200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 34**	
		Default Value:	2200h
		Format:	U16
140..141	63:48	**Inverse R-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
	47:32	**Inverse Pixel Value 35**	
		Default Value:	2300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
142..143	63:48	**Forward R-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
	47:32	**Forward Pixel Value 35**	
		Default Value:	2300h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 35**	
		Default Value:	2300h
		Format:	U16
144..145	63:48	**Inverse R-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
	47:32	**Inverse Pixel Value 36**	
		Default Value:	2400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
146..147	63:48	**Forward R-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
	47:32	**Forward Pixel Value 36**	
		Default Value:	2400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 36**	
		Default Value:	2400h
		Format:	U16
148..149	63:48	**Inverse R-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
	47:32	**Inverse Pixel Value 37**	
		Default Value:	2500h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
150..151	63:48	**Forward R-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
	47:32	**Forward Pixel Value 37**	
		Default Value:	2500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 37**	
		Default Value:	2500h
		Format:	U16
152..153	63:48	**Inverse R-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
	47:32	**Inverse Pixel Value 38**	
		Default Value:	2600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
154..155	63:48	**Forward R-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
	47:32	**Forward Pixel Value 38**	
		Default Value:	2600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 38**	
		Default Value:	2600h
		Format:	U16
156..157	63:48	**Inverse R-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
	47:32	**Inverse Pixel Value 39**	
		Default Value:	2700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
158..159	63:48	**Forward R-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
	47:32	**Forward Pixel Value 39**	
		Default Value:	2700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 39**	
		Default Value:	2700h
		Format:	U16
160..161	63:48	**Inverse R-ch Gamma Corrected Value 40**	
		Default Value:	2800h
		Format:	U16
	47:32	**Inverse Pixel Value 40**	
		Default Value:	2800h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 40** Default Value: 2800h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 40** Default Value: 2800h Format: U16
162..163	63:48	**Forward R-ch Gamma Corrected Value 40** Default Value: 2800h Format: U16
	47:32	**Forward Pixel Value 40** Default Value: 2800h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 40** Default Value: 2800h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 40** Default Value: 2800h Format: U16
164..165	63:48	**Inverse R-ch Gamma Corrected Value 41** Default Value: 2900h Format: U16
	47:32	**Inverse Pixel Value 41** Default Value: 2900h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 41** Default Value: 2900h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 41** Default Value: 2900h Format: U16
166..167	63:48	**Forward R-ch Gamma Corrected Value 41** Default Value: 2900h Format: U16
	47:32	**Forward Pixel Value 41** Default Value: 2900h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 41**	
		Default Value:	2900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 41**	
		Default Value:	2900h
		Format:	U16
168..169	63:48	**Inverse R-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
	47:32	**Inverse Pixel Value 42**	
		Default Value:	2a00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
170..171	63:48	**Forward R-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
	47:32	**Forward Pixel Value 42**	
		Default Value:	2a00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 42**	
		Default Value:	2a00h
		Format:	U16
172..173	63:48	**Inverse R-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
	47:32	**Inverse Pixel Value 43**	
		Default Value:	2b00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
174..175	63:48	**Forward R-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
	47:32	**Forward Pixel Value 43**	
		Default Value:	2b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 43**	
		Default Value:	2b00h
		Format:	U16
176..177	63:48	**Inverse R-ch Gamma Corrected Value 44**	
		Default Value:	2c00h
		Format:	U16
	47:32	**Inverse Pixel Value 44**	
		Default Value:	2c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 44**	
		Default Value:	2c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 44**	
		Default Value:	2c00h
		Format:	U16
178..179	63:48	**Forward R-ch Gamma Corrected Value 44**	
		Default Value:	2c00h
		Format:	U16
	47:32	**Forward Pixel Value 44**	
		Default Value:	2c00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 44** Default Value: 2c00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 44** Default Value: 2c00h Format: U16
180..181	63:48	**Inverse R-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
	47:32	**Inverse Pixel Value 45** Default Value: 2d00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
182..183	63:48	**Forward R-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
	47:32	**Forward Pixel Value 45** Default Value: 2d00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 45** Default Value: 2d00h Format: U16
184..185	63:48	**Inverse R-ch Gamma Corrected Value 46** Default Value: 2e00h Format: U16
	47:32	**Inverse Pixel Value 46** Default Value: 2e00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 46**	
		Default Value:	2e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 46**	
		Default Value:	2e00h
		Format:	U16
186..187	63:48	**Forward R-ch Gamma Corrected Value 46**	
		Default Value:	2e00h
		Format:	U16
	47:32	**Forward Pixel Value 46**	
		Default Value:	2e00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 46**	
		Default Value:	2e00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 46**	
		Default Value:	2e00h
		Format:	U16
188..189	63:48	**Inverse R-ch Gamma Corrected Value 47**	
		Default Value:	2f00h
		Format:	U16
	47:32	**Inverse Pixel Value 47**	
		Default Value:	2f00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 47**	
		Default Value:	2f00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 47**	
		Default Value:	2f00h
		Format:	U16
190..191	63:48	**Forward R-ch Gamma Corrected Value 47**	
		Default Value:	2f00h
		Format:	U16
	47:32	**Forward Pixel Value 47**	
		Default Value:	2f00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 47** Default Value: 2f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 47** Default Value: 2f00h Format: U16
192..193	63:48	**Inverse R-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
	47:32	**Inverse Pixel Value 48** Default Value: 3000h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
194..195	63:48	**Forward R-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
	47:32	**Forward Pixel Value 48** Default Value: 3000h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 48** Default Value: 3000h Format: U16
196..197	63:48	**Inverse R-ch Gamma Corrected Value 49** Default Value: 3100h Format: U16
	47:32	**Inverse Pixel Value 49** Default Value: 3100h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 49**	
		Default Value:	3100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 49**	
		Default Value:	3100h
		Format:	U16
198..199	63:48	**Forward R-ch Gamma Corrected Value 49**	
		Default Value:	3100h
		Format:	U16
	47:32	**Forward Pixel Value 49**	
		Default Value:	3100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 49**	
		Default Value:	3100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 49**	
		Default Value:	3100h
		Format:	U16
200..201	63:48	**Inverse R-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
	47:32	**Inverse Pixel Value 50**	
		Default Value:	3200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
202..203	63:48	**Forward R-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
	47:32	**Forward Pixel Value 50**	
		Default Value:	3200h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 50**	
		Default Value:	3200h
		Format:	U16
204..205	63:48	**Inverse R-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
	47:32	**Inverse Pixel Value 51**	
		Default Value:	3300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
206..207	63:48	**Forward R-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
	47:32	**Forward Pixel Value 51**	
		Default Value:	3300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 51**	
		Default Value:	3300h
		Format:	U16
208..209	63:48	**Inverse R-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
	47:32	**Inverse Pixel Value 52**	
		Default Value:	3400h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
210..211	63:48	**Forward R-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
	47:32	**Forward Pixel Value 52**	
		Default Value:	3400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 52**	
		Default Value:	3400h
		Format:	U16
212..213	63:48	**Inverse R-ch Gamma Corrected Value 53**	
		Default Value:	3500h
		Format:	U16
	47:32	**Inverse Pixel Value 53**	
		Default Value:	3500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 53**	
		Default Value:	3500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 53**	
		Default Value:	3500h
		Format:	U16
214..215	63:48	**Forward R-ch Gamma Corrected Value 53**	
		Default Value:	3500h
		Format:	U16
	47:32	**Forward Pixel Value 53**	
		Default Value:	3500h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 53**	Default Value:	3500h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 53**	Default Value:	3500h
			Format:	U16
216..217	63:48	**Inverse R-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
	47:32	**Inverse Pixel Value 54**	Default Value:	3600h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
218..219	63:48	**Forward R-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
	47:32	**Forward Pixel Value 54**	Default Value:	3600h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 54**	Default Value:	3600h
			Format:	U16
220..221	63:48	**Inverse R-ch Gamma Corrected Value 55**	Default Value:	3700h
			Format:	U16
	47:32	**Inverse Pixel Value 55**	Default Value:	3700h
			Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 55**		
			Default Value:	3700h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 55**		
			Default Value:	3700h
			Format:	U16
222..223	63:48	**Forward R-ch Gamma Corrected Value 55**		
			Default Value:	3700h
			Format:	U16
	47:32	**Forward Pixel Value 55**		
			Default Value:	3700h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 55**		
			Default Value:	3700h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 55**		
			Default Value:	3700h
			Format:	U16
224..225	63:48	**Inverse R-ch Gamma Corrected Value 56**		
			Default Value:	3800h
			Format:	U16
	47:32	**Inverse Pixel Value 56**		
			Default Value:	3800h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 56**		
			Default Value:	3800h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 56**		
			Default Value:	3800h
			Format:	U16
226..227	63:48	**Forward R-ch Gamma Corrected Value 56**		
			Default Value:	3800h
			Format:	U16
	47:32	**Forward Pixel Value 56**		
			Default Value:	3800h
			Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 56**	
		Default Value:	3800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 56**	
		Default Value:	3800h
		Format:	U16
228..229	63:48	**Inverse R-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
	47:32	**Inverse Pixel Value 57**	
		Default Value:	3900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
230..231	63:48	**Forward R-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
	47:32	**Forward Pixel Value 57**	
		Default Value:	3900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 57**	
		Default Value:	3900h
		Format:	U16
232..233	63:48	**Inverse R-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
	47:32	**Inverse Pixel Value 58**	
		Default Value:	3a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
234..235	63:48	**Forward R-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
	47:32	**Forward Pixel Value 58**	
		Default Value:	3a00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 58**	
		Default Value:	3a00h
		Format:	U16
236..237	63:48	**Inverse R-ch Gamma Corrected Value 59**	
		Default Value:	3b00h
		Format:	U16
	47:32	**Inverse Pixel Value 59**	
		Default Value:	3b00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 59**	
		Default Value:	3b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 59**	
		Default Value:	3b00h
		Format:	U16
238..239	63:48	**Forward R-ch Gamma Corrected Value 59**	
		Default Value:	3b00h
		Format:	U16
	47:32	**Forward Pixel Value 59**	
		Default Value:	3b00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 59** Default Value: 3b00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 59** Default Value: 3b00h Format: U16
240..241	63:48	**Inverse R-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
	47:32	**Inverse Pixel Value 60** Default Value: 3c00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
242..243	63:48	**Forward R-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
	47:32	**Forward Pixel Value 60** Default Value: 3c00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 60** Default Value: 3c00h Format: U16
244..245	63:48	**Inverse R-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
	47:32	**Inverse Pixel Value 61** Default Value: 3d00h Format: U16

Gamut_Expansion_Gamma_Correction

DWord	Bit	Description
	31:16	**Inverse B-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
246..247	63:48	**Forward R-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
	47:32	**Forward Pixel Value 61** Default Value: 3d00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 61** Default Value: 3d00h Format: U16
248..249	63:48	**Inverse R-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
	47:32	**Inverse Pixel Value 62** Default Value: 3e00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
250..251	63:48	**Forward R-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
	47:32	**Forward Pixel Value 62** Default Value: 3e00h Format: U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 62** Default Value: 3e00h Format: U16
252..253	63:48	**Inverse R-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
	47:32	**Inverse Pixel Value 63** Default Value: 3f00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
254..255	63:48	**Forward R-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
	47:32	**Forward Pixel Value 63** Default Value: 3f00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 63** Default Value: 3f00h Format: U16
256..257	63:48	**Inverse R-ch Gamma Corrected Value 64** Default Value: 4000h Format: U16
	47:32	**Inverse Pixel Value 64** Default Value: 4000h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 64**	
		Default Value:	4000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 64**	
		Default Value:	4000h
		Format:	U16
258..259	63:48	**Forward R-ch Gamma Corrected Value 64**	
		Default Value:	4000h
		Format:	U16
	47:32	**Forward Pixel Value 64**	
		Default Value:	4000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 64**	
		Default Value:	4000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 64**	
		Default Value:	4000h
		Format:	U16
260..261	63:48	**Inverse R-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
	47:32	**Inverse Pixel Value 65**	
		Default Value:	4100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
262..263	63:48	**Forward R-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
	47:32	**Forward Pixel Value 65**	
		Default Value:	4100h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 65**	
		Default Value:	4100h
		Format:	U16
264..265	63:48	**Inverse R-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
	47:32	**Inverse Pixel Value 66**	
		Default Value:	4200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
266..267	63:48	**Forward R-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
	47:32	**Forward Pixel Value 66**	
		Default Value:	4200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 66**	
		Default Value:	4200h
		Format:	U16
268..269	63:48	**Inverse R-ch Gamma Corrected Value 67**	
		Default Value:	4300h
		Format:	U16
	47:32	**Inverse Pixel Value 67**	
		Default Value:	4300h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 67** Default Value: 4300h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 67** Default Value: 4300h Format: U16
270..271	63:48	**Forward R-ch Gamma Corrected Value 67** Default Value: 4300h Format: U16
	47:32	**Forward Pixel Value 67** Default Value: 4300h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 67** Default Value: 4300h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 67** Default Value: 4300h Format: U16
272..273	63:48	**Inverse R-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
	47:32	**Inverse Pixel Value 68** Default Value: 4400h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
274..275	63:48	**Forward R-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
	47:32	**Forward Pixel Value 68** Default Value: 4400h Format: U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 68** Default Value: 4400h Format: U16
276..277	63:48	**Inverse R-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
	47:32	**Inverse Pixel Value 69** Default Value: 4500h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
278..279	63:48	**Forward R-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
	47:32	**Forward Pixel Value 69** Default Value: 4500h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 69** Default Value: 4500h Format: U16
280..281	63:48	**Inverse R-ch Gamma Corrected Value 70** Default Value: 4600h Format: U16
	47:32	**Inverse Pixel Value 70** Default Value: 4600h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 70**	
		Default Value:	4600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 70**	
		Default Value:	4600h
		Format:	U16
282..283	63:48	**Forward R-ch Gamma Corrected Value 70**	
		Default Value:	4600h
		Format:	U16
	47:32	**Forward Pixel Value 70**	
		Default Value:	4600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 70**	
		Default Value:	4600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 70**	
		Default Value:	4600h
		Format:	U16
284..285	63:48	**Inverse R-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
	47:32	**Inverse Pixel Value 71**	
		Default Value:	4700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
286..287	63:48	**Forward R-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
	47:32	**Forward Pixel Value 71**	
		Default Value:	4700h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 71**	
		Default Value:	4700h
		Format:	U16
288..289	63:48	**Inverse R-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
	47:32	**Inverse Pixel Value 72**	
		Default Value:	4800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
290..291	63:48	**Forward R-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
	47:32	**Forward Pixel Value 72**	
		Default Value:	4800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 72**	
		Default Value:	4800h
		Format:	U16
292..293	63:48	**Inverse R-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
	47:32	**Inverse Pixel Value 73**	
		Default Value:	4900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
294..295	63:48	**Forward R-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
	47:32	**Forward Pixel Value 73**	
		Default Value:	4900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 73**	
		Default Value:	4900h
		Format:	U16
296..297	63:48	**Inverse R-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
	47:32	**Inverse Pixel Value 74**	
		Default Value:	4a00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
298..299	63:48	**Forward R-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
	47:32	**Forward Pixel Value 74**	
		Default Value:	4a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 74**	
		Default Value:	4a00h
		Format:	U16
300..301	63:48	**Inverse R-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
	47:32	**Inverse Pixel Value 75**	
		Default Value:	4b00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
302..303	63:48	**Forward R-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
	47:32	**Forward Pixel Value 75**	
		Default Value:	4b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 75**	
		Default Value:	4b00h
		Format:	U16
304..305	63:48	**Inverse R-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
	47:32	**Inverse Pixel Value 76**	
		Default Value:	4c00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
306..307	63:48	**Forward R-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
	47:32	**Forward Pixel Value 76**	
		Default Value:	4c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 76**	
		Default Value:	4c00h
		Format:	U16
308..309	63:48	**Inverse R-ch Gamma Corrected Value 77**	
		Default Value:	4d00h
		Format:	U16
	47:32	**Inverse Pixel Value 77**	
		Default Value:	4d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 77**	
		Default Value:	4d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 77**	
		Default Value:	4d00h
		Format:	U16
310..311	63:48	**Forward R-ch Gamma Corrected Value 77**	
		Default Value:	4d00h
		Format:	U16
	47:32	**Forward Pixel Value 77**	
		Default Value:	4d00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 77** Default Value: 4d00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 77** Default Value: 4d00h Format: U16
312..313	63:48	**Inverse R-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
	47:32	**Inverse Pixel Value 78** Default Value: 4e00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
314..315	63:48	**Forward R-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
	47:32	**Forward Pixel Value 78** Default Value: 4e00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 78** Default Value: 4e00h Format: U16
316..317	63:48	**Inverse R-ch Gamma Corrected Value 79** Default Value: 4f00h Format: U16
	47:32	**Inverse Pixel Value 79** Default Value: 4f00h Format: U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 79**		
			Default Value:	4f00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 79**		
			Default Value:	4f00h
			Format:	U16
318..319	63:48	**Forward R-ch Gamma Corrected Value 79**		
			Default Value:	4f00h
			Format:	U16
	47:32	**Forward Pixel Value 79**		
			Default Value:	4f00h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 79**		
			Default Value:	4f00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 79**		
			Default Value:	4f00h
			Format:	U16
320..321	63:48	**Inverse R-ch Gamma Corrected Value 80**		
			Default Value:	5000h
			Format:	U16
	47:32	**Inverse Pixel Value 80**		
			Default Value:	5000h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 80**		
			Default Value:	5000h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 80**		
			Default Value:	5000h
			Format:	U16
322..323	63:48	**Forward R-ch Gamma Corrected Value 80**		
			Default Value:	5000h
			Format:	U16
	47:32	**Forward Pixel Value 80**		
			Default Value:	5000h
			Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 80** Default Value: 5000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 80** Default Value: 5000h Format: U16
324..325	63:48	**Inverse R-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
	47:32	**Inverse Pixel Value 81** Default Value: 5100h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
326..327	63:48	**Forward R-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
	47:32	**Forward Pixel Value 81** Default Value: 5100h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 81** Default Value: 5100h Format: U16
328..329	63:48	**Inverse R-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
	47:32	**Inverse Pixel Value 82** Default Value: 5200h Format: U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
330..331	63:48	**Forward R-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
	47:32	**Forward Pixel Value 82** Default Value: 5200h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 82** Default Value: 5200h Format: U16
332..333	63:48	**Inverse R-ch Gamma Corrected Value 83** Default Value: 5300h Format: U16
	47:32	**Inverse Pixel Value 83** Default Value: 5300h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 83** Default Value: 5300h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 83** Default Value: 5300h Format: U16
334..335	63:48	**Forward R-ch Gamma Corrected Value 83** Default Value: 5300h Format: U16
	47:32	**Forward Pixel Value 83** Default Value: 5300h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 83**	
		Default Value:	5300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 83**	
		Default Value:	5300h
		Format:	U16
336..337	63:48	**Inverse R-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
	47:32	**Inverse Pixel Value 84**	
		Default Value:	5400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
338..339	63:48	**Forward R-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
	47:32	**Forward Pixel Value 84**	
		Default Value:	5400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 84**	
		Default Value:	5400h
		Format:	U16
340..341	63:48	**Inverse R-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
	47:32	**Inverse Pixel Value 85**	
		Default Value:	5500h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
342..343	63:48	**Forward R-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
	47:32	**Forward Pixel Value 85**	
		Default Value:	5500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 85**	
		Default Value:	5500h
		Format:	U16
344..345	63:48	**Inverse R-ch Gamma Corrected Value 86**	
		Default Value:	5600h
		Format:	U16
	47:32	**Inverse Pixel Value 86**	
		Default Value:	5600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 86**	
		Default Value:	5600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 86**	
		Default Value:	5600h
		Format:	U16
346..347	63:48	**Forward R-ch Gamma Corrected Value 86**	
		Default Value:	5600h
		Format:	U16
	47:32	**Forward Pixel Value 86**	
		Default Value:	5600h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 86** Default Value: 5600h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 86** Default Value: 5600h Format: U16
348..349	63:48	**Inverse R-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
	47:32	**Inverse Pixel Value 87** Default Value: 5700h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
350..351	63:48	**Forward R-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
	47:32	**Forward Pixel Value 87** Default Value: 5700h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 87** Default Value: 5700h Format: U16
352..353	63:48	**Inverse R-ch Gamma Corrected Value 88** Default Value: 5800h Format: U16
	47:32	**Inverse Pixel Value 88** Default Value: 5800h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 88**	
		Default Value:	5800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 88**	
		Default Value:	5800h
		Format:	U16
354..355	63:48	**Forward R-ch Gamma Corrected Value 88**	
		Default Value:	5800h
		Format:	U16
	47:32	**Forward Pixel Value 88**	
		Default Value:	5800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 88**	
		Default Value:	5800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 88**	
		Default Value:	5800h
		Format:	U16
356..357	63:48	**Inverse R-ch Gamma Corrected Value 89**	
		Default Value:	5900h
		Format:	U16
	47:32	**Inverse Pixel Value 89**	
		Default Value:	5900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 89**	
		Default Value:	5900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 89**	
		Default Value:	5900h
		Format:	U16
358..359	63:48	**Forward R-ch Gamma Corrected Value 89**	
		Default Value:	5900h
		Format:	U16
	47:32	**Forward Pixel Value 89**	
		Default Value:	5900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 89**		
			Default Value:	5900h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 89**		
			Default Value:	5900h
			Format:	U16
360..361	63:48	**Inverse R-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
	47:32	**Inverse Pixel Value 90**		
			Default Value:	5a00h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
362..363	63:48	**Forward R-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
	47:32	**Forward Pixel Value 90**		
			Default Value:	5a00h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 90**		
			Default Value:	5a00h
			Format:	U16
364..365	63:48	**Inverse R-ch Gamma Corrected Value 91**		
			Default Value:	5b00h
			Format:	U16
	47:32	**Inverse Pixel Value 91**		
			Default Value:	5b00h
			Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 91**	
		Default Value:	5b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 91**	
		Default Value:	5b00h
		Format:	U16
366..367	63:48	**Forward R-ch Gamma Corrected Value 91**	
		Default Value:	5b00h
		Format:	U16
	47:32	**Forward Pixel Value 91**	
		Default Value:	5b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 91**	
		Default Value:	5b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 91**	
		Default Value:	5b00h
		Format:	U16
368..369	63:48	**Inverse R-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
	47:32	**Inverse Pixel Value 92**	
		Default Value:	5c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
370..371	63:48	**Forward R-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
	47:32	**Forward Pixel Value 92**	
		Default Value:	5c00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 92**	
		Default Value:	5c00h
		Format:	U16
372..373	63:48	**Inverse R-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
	47:32	**Inverse Pixel Value 93**	
		Default Value:	5d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
374..375	63:48	**Forward R-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
	47:32	**Forward Pixel Value 93**	
		Default Value:	5d00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 93**	
		Default Value:	5d00h
		Format:	U16
376..377	63:48	**Inverse R-ch Gamma Corrected Value 94**	
		Default Value:	5e00h
		Format:	U16
	47:32	**Inverse Pixel Value 94**	
		Default Value:	5e00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 94** Default Value: 5e00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 94** Default Value: 5e00h Format: U16
378..379	63:48	**Forward R-ch Gamma Corrected Value 94** Default Value: 5e00h Format: U16
	47:32	**Forward Pixel Value 94** Default Value: 5e00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 94** Default Value: 5e00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 94** Default Value: 5e00h Format: U16
380..381	63:48	**Inverse R-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
	47:32	**Inverse Pixel Value 95** Default Value: 5f00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
382..383	63:48	**Forward R-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
	47:32	**Forward Pixel Value 95** Default Value: 5f00h Format: U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 95** Default Value: 5f00h Format: U16
384..385	63:48	**Inverse R-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
	47:32	**Inverse Pixel Value 96** Default Value: 6000h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
386..387	63:48	**Forward R-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
	47:32	**Forward Pixel Value 96** Default Value: 6000h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 96** Default Value: 6000h Format: U16
388..389	63:48	**Inverse R-ch Gamma Corrected Value 97** Default Value: 6100h Format: U16
	47:32	**Inverse Pixel Value 97** Default Value: 6100h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 97**	
		Default Value:	6100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 97**	
		Default Value:	6100h
		Format:	U16
390..391	63:48	**Forward R-ch Gamma Corrected Value 97**	
		Default Value:	6100h
		Format:	U16
	47:32	**Forward Pixel Value 97**	
		Default Value:	6100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 97**	
		Default Value:	6100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 97**	
		Default Value:	6100h
		Format:	U16
392..393	63:48	**Inverse R-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
	47:32	**Inverse Pixel Value 98**	
		Default Value:	6200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
394..395	63:48	**Forward R-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
	47:32	**Forward Pixel Value 98**	
		Default Value:	6200h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 98**	
		Default Value:	6200h
		Format:	U16
396..397	63:48	**Inverse R-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
	47:32	**Inverse Pixel Value 99**	
		Default Value:	6300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
398..399	63:48	**Forward R-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
	47:32	**Forward Pixel Value 99**	
		Default Value:	6300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 99**	
		Default Value:	6300h
		Format:	U16
400..401	63:48	**Inverse R-ch Gamma Corrected Value 100**	
		Default Value:	6400h
		Format:	U16
	47:32	**Inverse Pixel Value 100**	
		Default Value:	6400h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 100** Default Value: 6400h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 100** Default Value: 6400h Format: U16
402..403	63:48	**Forward R-ch Gamma Corrected Value 100** Default Value: 6400h Format: U16
	47:32	**Forward Pixel Value 100** Default Value: 6400h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 100** Default Value: 6400h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 100** Default Value: 6400h Format: U16
404..405	63:48	**Inverse R-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
	47:32	**Inverse Pixel Value 101** Default Value: 6500h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
406..407	63:48	**Forward R-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
	47:32	**Forward Pixel Value 101** Default Value: 6500h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 101** Default Value: 6500h Format: U16
408..409	63:48	**Inverse R-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
	47:32	**Inverse Pixel Value 102** Default Value: 6600h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
410..411	63:48	**Forward R-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
	47:32	**Forward Pixel Value 102** Default Value: 6600h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 102** Default Value: 6600h Format: U16
412..413	63:48	**Inverse R-ch Gamma Corrected Value 103** Default Value: 6700h Format: U16
	47:32	**Inverse Pixel Value 103** Default Value: 6700h Format: U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 103**		
			Default Value:	6700h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 103**		
			Default Value:	6700h
			Format:	U16
414..415	63:48	**Forward R-ch Gamma Corrected Value 103**		
			Default Value:	6700h
			Format:	U16
	47:32	**Forward Pixel Value 103**		
			Default Value:	6700h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 103**		
			Default Value:	6700h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 103**		
			Default Value:	6700h
			Format:	U16
416..417	63:48	**Inverse R-ch Gamma Corrected Value 104**		
			Default Value:	6800h
			Format:	U16
	47:32	**Inverse Pixel Value 104**		
			Default Value:	6800h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 104**		
			Default Value:	6800h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 104**		
			Default Value:	6800h
			Format:	U16
418..419	63:48	**Forward R-ch Gamma Corrected Value 104**		
			Default Value:	6800h
			Format:	U16
	47:32	**Forward Pixel Value 104**		
			Default Value:	6800h
			Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 104**	
		Default Value:	6800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 104**	
		Default Value:	6800h
		Format:	U16
420..421	63:48	**Inverse R-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
	47:32	**Inverse Pixel Value 105**	
		Default Value:	6900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
422..423	63:48	**Forward R-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
	47:32	**Forward Pixel Value 105**	
		Default Value:	6900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 105**	
		Default Value:	6900h
		Format:	U16
424..425	63:48	**Inverse R-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
	47:32	**Inverse Pixel Value 106**	
		Default Value:	6a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
426..427	63:48	**Forward R-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
	47:32	**Forward Pixel Value 106**	
		Default Value:	6a00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 106**	
		Default Value:	6a00h
		Format:	U16
428..429	63:48	**Inverse R-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
	47:32	**Inverse Pixel Value 107**	
		Default Value:	6b00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
430..431	63:48	**Forward R-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
	47:32	**Forward Pixel Value 107**	
		Default Value:	6b00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 107**	
		Default Value:	6b00h
		Format:	U16
432..433	63:48	**Inverse R-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
	47:32	**Inverse Pixel Value 108**	
		Default Value:	6c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
434..435	63:48	**Forward R-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
	47:32	**Forward Pixel Value 108**	
		Default Value:	6c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 108**	
		Default Value:	6c00h
		Format:	U16
436..437	63:48	**Inverse R-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
	47:32	**Inverse Pixel Value 109**	
		Default Value:	6d00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
438..439	63:48	**Forward R-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
	47:32	**Forward Pixel Value 109**	
		Default Value:	6d00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 109**	
		Default Value:	6d00h
		Format:	U16
440..441	63:48	**Inverse R-ch Gamma Corrected Value 110**	
		Default Value:	6e00h
		Format:	U16
	47:32	**Inverse Pixel Value 110**	
		Default Value:	6e00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 110**	
		Default Value:	6e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 110**	
		Default Value:	6e00h
		Format:	U16
442..443	63:48	**Forward R-ch Gamma Corrected Value 110**	
		Default Value:	6e00h
		Format:	U16
	47:32	**Forward Pixel Value 110**	
		Default Value:	6e00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 110** Default Value: 6e00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 110** Default Value: 6e00h Format: U16
444..445	63:48	**Inverse R-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
	47:32	**Inverse Pixel Value 111** Default Value: 6f00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
446..447	63:48	**Forward R-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
	47:32	**Forward Pixel Value 111** Default Value: 6f00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 111** Default Value: 6f00h Format: U16
448..449	63:48	**Inverse R-ch Gamma Corrected Value 112** Default Value: 7000h Format: U16
	47:32	**Inverse Pixel Value 112** Default Value: 7000h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 112**	
		Default Value:	7000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 112**	
		Default Value:	7000h
		Format:	U16
450..451	63:48	**Forward R-ch Gamma Corrected Value 112**	
		Default Value:	7000h
		Format:	U16
	47:32	**Forward Pixel Value 112**	
		Default Value:	7000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 112**	
		Default Value:	7000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 112**	
		Default Value:	7000h
		Format:	U16
452..453	63:48	**Inverse R-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
	47:32	**Inverse Pixel Value 113**	
		Default Value:	7100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
454..455	63:48	**Forward R-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
	47:32	**Forward Pixel Value 113**	
		Default Value:	7100h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 113**	
		Default Value:	7100h
		Format:	U16
456..457	63:48	**Inverse R-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
	47:32	**Inverse Pixel Value 114**	
		Default Value:	7200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
458..459	63:48	**Forward R-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
	47:32	**Forward Pixel Value 114**	
		Default Value:	7200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 114**	
		Default Value:	7200h
		Format:	U16
460..461	63:48	**Inverse R-ch Gamma Corrected Value 115**	
		Default Value:	7300h
		Format:	U16
	47:32	**Inverse Pixel Value 115**	
		Default Value:	7300h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 115**		
			Default Value:	7300h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 115**		
			Default Value:	7300h
			Format:	U16
462..463	63:48	**Forward R-ch Gamma Corrected Value 115**		
			Default Value:	7300h
			Format:	U16
	47:32	**Forward Pixel Value 115**		
			Default Value:	7300h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 115**		
			Default Value:	7300h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 115**		
			Default Value:	7300h
			Format:	U16
464..465	63:48	**Inverse R-ch Gamma Corrected Value 116**		
			Default Value:	7400h
			Format:	U16
	47:32	**Inverse Pixel Value 116**		
			Default Value:	7400h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 116**		
			Default Value:	7400h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 116**		
			Default Value:	7400h
			Format:	U16
466..467	63:48	**Forward R-ch Gamma Corrected Value 116**		
			Default Value:	7400h
			Format:	U16
	47:32	**Forward Pixel Value 116**		
			Default Value:	7400h
			Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 116**	
		Default Value:	7400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 116**	
		Default Value:	7400h
		Format:	U16
468..469	63:48	**Inverse R-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
	47:32	**Inverse Pixel Value 117**	
		Default Value:	7500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
470..471	63:48	**Forward R-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
	47:32	**Forward Pixel Value 117**	
		Default Value:	7500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 117**	
		Default Value:	7500h
		Format:	U16
472..473	63:48	**Inverse R-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
	47:32	**Inverse Pixel Value 118**	
		Default Value:	7600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
474..475	63:48	**Forward R-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
	47:32	**Forward Pixel Value 118**	
		Default Value:	7600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 118**	
		Default Value:	7600h
		Format:	U16
476..477	63:48	**Inverse R-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
	47:32	**Inverse Pixel Value 119**	
		Default Value:	7700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
478..479	63:48	**Forward R-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
	47:32	**Forward Pixel Value 119**	
		Default Value:	7700h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 119**	
		Default Value:	7700h
		Format:	U16
480..481	63:48	**Inverse R-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
	47:32	**Inverse Pixel Value 120**	
		Default Value:	7800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
482..483	63:48	**Forward R-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
	47:32	**Forward Pixel Value 120**	
		Default Value:	7800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 120**	
		Default Value:	7800h
		Format:	U16
484..485	63:48	**Inverse R-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
	47:32	**Inverse Pixel Value 121**	
		Default Value:	7900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
486..487	63:48	**Forward R-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
	47:32	**Forward Pixel Value 121**	
		Default Value:	7900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 121**	
		Default Value:	7900h
		Format:	U16
488..489	63:48	**Inverse R-ch Gamma Corrected Value 122**	
		Default Value:	7a00h
		Format:	U16
	47:32	**Inverse Pixel Value 122**	
		Default Value:	7a00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 122**	
		Default Value:	7a00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 122**	
		Default Value:	7a00h
		Format:	U16
490..491	63:48	**Forward R-ch Gamma Corrected Value 122**	
		Default Value:	7a00h
		Format:	U16
	47:32	**Forward Pixel Value 122**	
		Default Value:	7a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 122** Default Value: 7a00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 122** Default Value: 7a00h Format: U16
492..493	63:48	**Inverse R-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
	47:32	**Inverse Pixel Value 123** Default Value: 7b00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
494..495	63:48	**Forward R-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
	47:32	**Forward Pixel Value 123** Default Value: 7b00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 123** Default Value: 7b00h Format: U16
496..497	63:48	**Inverse R-ch Gamma Corrected Value 124** Default Value: 7c00h Format: U16
	47:32	**Inverse Pixel Value 124** Default Value: 7c00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 124**	
		Default Value:	7c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 124**	
		Default Value:	7c00h
		Format:	U16
498..499	63:48	**Forward R-ch Gamma Corrected Value 124**	
		Default Value:	7c00h
		Format:	U16
	47:32	**Forward Pixel Value 124**	
		Default Value:	7c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 124**	
		Default Value:	7c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 124**	
		Default Value:	7c00h
		Format:	U16
500..501	63:48	**Inverse R-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
	47:32	**Inverse Pixel Value 125**	
		Default Value:	7d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
502..503	63:48	**Forward R-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
	47:32	**Forward Pixel Value 125**	
		Default Value:	7d00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 125**	
		Default Value:	7d00h
		Format:	U16
504..505	63:48	**Inverse R-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
	47:32	**Inverse Pixel Value 126**	
		Default Value:	7e00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
506..507	63:48	**Forward R-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
	47:32	**Forward Pixel Value 126**	
		Default Value:	7e00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 126**	
		Default Value:	7e00h
		Format:	U16
508..509	63:48	**Inverse R-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
	47:32	**Inverse Pixel Value 127**	
		Default Value:	7f00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
510..511	63:48	**Forward R-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
	47:32	**Forward Pixel Value 127**	
		Default Value:	7f00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 127**	
		Default Value:	7f00h
		Format:	U16
512..513	63:48	**Inverse R-ch Gamma Corrected Value 128**	
		Default Value:	8000h
		Format:	U16
	47:32	**Inverse Pixel Value 128**	
		Default Value:	8000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 128**	
		Default Value:	8000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 128**	
		Default Value:	8000h
		Format:	U16
514..515	63:48	**Forward R-ch Gamma Corrected Value 128**	
		Default Value:	8000h
		Format:	U16
	47:32	**Forward Pixel Value 128**	
		Default Value:	8000h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 128** Default Value: 8000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 128** Default Value: 8000h Format: U16
516..517	63:48	**Inverse R-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
	47:32	**Inverse Pixel Value 129** Default Value: 8100h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
518..519	63:48	**Forward R-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
	47:32	**Forward Pixel Value 129** Default Value: 8100h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 129** Default Value: 8100h Format: U16
520..521	63:48	**Inverse R-ch Gamma Corrected Value 130** Default Value: 8200h Format: U16
	47:32	**Inverse Pixel Value 130** Default Value: 8200h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 130**	
		Default Value:	8200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 130**	
		Default Value:	8200h
		Format:	U16
522..523	63:48	**Forward R-ch Gamma Corrected Value 130**	
		Default Value:	8200h
		Format:	U16
	47:32	**Forward Pixel Value 130**	
		Default Value:	8200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 130**	
		Default Value:	8200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 130**	
		Default Value:	8200h
		Format:	U16
524..525	63:48	**Inverse R-ch Gamma Corrected Value 131**	
		Default Value:	8300h
		Format:	U16
	47:32	**Inverse Pixel Value 131**	
		Default Value:	8300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 131**	
		Default Value:	8300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 131**	
		Default Value:	8300h
		Format:	U16
526..527	63:48	**Forward R-ch Gamma Corrected Value 131**	
		Default Value:	8300h
		Format:	U16
	47:32	**Forward Pixel Value 131**	
		Default Value:	8300h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 131** Default Value: 8300h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 131** Default Value: 8300h Format: U16
528..529	63:48	**Inverse R-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
	47:32	**Inverse Pixel Value 132** Default Value: 8400h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
530..531	63:48	**Forward R-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
	47:32	**Forward Pixel Value 132** Default Value: 8400h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 132** Default Value: 8400h Format: U16
532..533	63:48	**Inverse R-ch Gamma Corrected Value 133** Default Value: 8500h Format: U16
	47:32	**Inverse Pixel Value 133** Default Value: 8500h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 133**	
		Default Value:	8500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 133**	
		Default Value:	8500h
		Format:	U16
534..535	63:48	**Forward R-ch Gamma Corrected Value 133**	
		Default Value:	8500h
		Format:	U16
	47:32	**Forward Pixel Value 133**	
		Default Value:	8500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 133**	
		Default Value:	8500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 133**	
		Default Value:	8500h
		Format:	U16
536..537	63:48	**Inverse R-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
	47:32	**Inverse Pixel Value 134**	
		Default Value:	8600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
538..539	63:48	**Forward R-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
	47:32	**Forward Pixel Value 134**	
		Default Value:	8600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 134**	
		Default Value:	8600h
		Format:	U16
540..541	63:48	**Inverse R-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
	47:32	**Inverse Pixel Value 135**	
		Default Value:	8700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
542..543	63:48	**Forward R-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
	47:32	**Forward Pixel Value 135**	
		Default Value:	8700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 135**	
		Default Value:	8700h
		Format:	U16
544..545	63:48	**Inverse R-ch Gamma Corrected Value 136**	
		Default Value:	8800h
		Format:	U16
	47:32	**Inverse Pixel Value 136**	
		Default Value:	8800h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 136**		
			Default Value:	8800h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 136**		
			Default Value:	8800h
			Format:	U16
546..547	63:48	**Forward R-ch Gamma Corrected Value 136**		
			Default Value:	8800h
			Format:	U16
	47:32	**Forward Pixel Value 136**		
			Default Value:	8800h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 136**		
			Default Value:	8800h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 136**		
			Default Value:	8800h
			Format:	U16
548..549	63:48	**Inverse R-ch Gamma Corrected Value 137**		
			Default Value:	8900h
			Format:	U16
	47:32	**Inverse Pixel Value 137**		
			Default Value:	8900h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 137**		
			Default Value:	8900h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 137**		
			Default Value:	8900h
			Format:	U16
550..551	63:48	**Forward R-ch Gamma Corrected Value 137**		
			Default Value:	8900h
			Format:	U16
	47:32	**Forward Pixel Value 137**		
			Default Value:	8900h
			Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 137** Default Value: 8900h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 137** Default Value: 8900h Format: U16
552..553	63:48	**Inverse R-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
	47:32	**Inverse Pixel Value 138** Default Value: 8a00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
554..555	63:48	**Forward R-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
	47:32	**Forward Pixel Value 138** Default Value: 8a00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 138** Default Value: 8a00h Format: U16
556..557	63:48	**Inverse R-ch Gamma Corrected Value 139** Default Value: 8b00h Format: U16
	47:32	**Inverse Pixel Value 139** Default Value: 8b00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 139**	
		Default Value:	8b00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 139**	
		Default Value:	8b00h
		Format:	U16
558..559	63:48	**Forward R-ch Gamma Corrected Value 139**	
		Default Value:	8b00h
		Format:	U16
	47:32	**Forward Pixel Value 139**	
		Default Value:	8b00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 139**	
		Default Value:	8b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 139**	
		Default Value:	8b00h
		Format:	U16
560..561	63:48	**Inverse R-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
	47:32	**Inverse Pixel Value 140**	
		Default Value:	8c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
562..563	63:48	**Forward R-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
	47:32	**Forward Pixel Value 140**	
		Default Value:	8c00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 140**	
		Default Value:	8c00h
		Format:	U16
564..565	63:48	**Inverse R-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
	47:32	**Inverse Pixel Value 141**	
		Default Value:	8d00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
566..567	63:48	**Forward R-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
	47:32	**Forward Pixel Value 141**	
		Default Value:	8d00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 141**	
		Default Value:	8d00h
		Format:	U16
568..569	63:48	**Inverse R-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
	47:32	**Inverse Pixel Value 142**	
		Default Value:	8e00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
570..571	63:48	**Forward R-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
	47:32	**Forward Pixel Value 142**	
		Default Value:	8e00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 142**	
		Default Value:	8e00h
		Format:	U16
572..573	63:48	**Inverse R-ch Gamma Corrected Value 143**	
		Default Value:	8f00h
		Format:	U16
	47:32	**Inverse Pixel Value 143**	
		Default Value:	8f00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 143**	
		Default Value:	8f00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 143**	
		Default Value:	8f00h
		Format:	U16
574..575	63:48	**Forward R-ch Gamma Corrected Value 143**	
		Default Value:	8f00h
		Format:	U16
	47:32	**Forward Pixel Value 143**	
		Default Value:	8f00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 143** Default Value: 8f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 143** Default Value: 8f00h Format: U16
576..577	63:48	**Inverse R-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
	47:32	**Inverse Pixel Value 144** Default Value: 9000h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
578..579	63:48	**Forward R-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
	47:32	**Forward Pixel Value 144** Default Value: 9000h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 144** Default Value: 9000h Format: U16
580..581	63:48	**Inverse R-ch Gamma Corrected Value 145** Default Value: 9100h Format: U16
	47:32	**Inverse Pixel Value 145** Default Value: 9100h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 145**	
		Default Value:	9100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 145**	
		Default Value:	9100h
		Format:	U16
582..583	63:48	**Forward R-ch Gamma Corrected Value 145**	
		Default Value:	9100h
		Format:	U16
	47:32	**Forward Pixel Value 145**	
		Default Value:	9100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 145**	
		Default Value:	9100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 145**	
		Default Value:	9100h
		Format:	U16
584..585	63:48	**Inverse R-ch Gamma Corrected Value 146**	
		Default Value:	9200h
		Format:	U16
	47:32	**Inverse Pixel Value 146**	
		Default Value:	9200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 146**	
		Default Value:	9200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 146**	
		Default Value:	9200h
		Format:	U16
586..587	63:48	**Forward R-ch Gamma Corrected Value 146**	
		Default Value:	9200h
		Format:	U16
	47:32	**Forward Pixel Value 146**	
		Default Value:	9200h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 146** Default Value: 9200h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 146** Default Value: 9200h Format: U16
588..589	63:48	**Inverse R-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
	47:32	**Inverse Pixel Value 147** Default Value: 9300h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
590..591	63:48	**Forward R-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
	47:32	**Forward Pixel Value 147** Default Value: 9300h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 147** Default Value: 9300h Format: U16
592..593	63:48	**Inverse R-ch Gamma Corrected Value 148** Default Value: 9400h Format: U16
	47:32	**Inverse Pixel Value 148** Default Value: 9400h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 148**	
		Default Value:	9400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 148**	
		Default Value:	9400h
		Format:	U16
594..595	63:48	**Forward R-ch Gamma Corrected Value 148**	
		Default Value:	9400h
		Format:	U16
	47:32	**Forward Pixel Value 148**	
		Default Value:	9400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 148**	
		Default Value:	9400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 148**	
		Default Value:	9400h
		Format:	U16
596..597	63:48	**Inverse R-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
	47:32	**Inverse Pixel Value 149**	
		Default Value:	9500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
598..599	63:48	**Forward R-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
	47:32	**Forward Pixel Value 149**	
		Default Value:	9500h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 149**	
		Default Value:	9500h
		Format:	U16
600..601	63:48	**Inverse R-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
	47:32	**Inverse Pixel Value 150**	
		Default Value:	9600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
602..603	63:48	**Forward R-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
	47:32	**Forward Pixel Value 150**	
		Default Value:	9600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 150**	
		Default Value:	9600h
		Format:	U16
604..605	63:48	**Inverse R-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
	47:32	**Inverse Pixel Value 151**	
		Default Value:	9700h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
606..607	63:48	**Forward R-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
	47:32	**Forward Pixel Value 151**	
		Default Value:	9700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 151**	
		Default Value:	9700h
		Format:	U16
608..609	63:48	**Inverse R-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
	47:32	**Inverse Pixel Value 152**	
		Default Value:	9800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
610..611	63:48	**Forward R-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
	47:32	**Forward Pixel Value 152**	
		Default Value:	9800h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 152**	
		Default Value:	9800h
		Format:	U16
612..613	63:48	**Inverse R-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
	47:32	**Inverse Pixel Value 153**	
		Default Value:	9900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
614..615	63:48	**Forward R-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
	47:32	**Forward Pixel Value 153**	
		Default Value:	9900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 153**	
		Default Value:	9900h
		Format:	U16
616..617	63:48	**Inverse R-ch Gamma Corrected Value 154**	
		Default Value:	9a00h
		Format:	U16
	47:32	**Inverse Pixel Value 154**	
		Default Value:	9a00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 154** Default Value: 9a00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 154** Default Value: 9a00h Format: U16
618..619	63:48	**Forward R-ch Gamma Corrected Value 154** Default Value: 9a00h Format: U16
	47:32	**Forward Pixel Value 154** Default Value: 9a00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 154** Default Value: 9a00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 154** Default Value: 9a00h Format: U16
620..621	63:48	**Inverse R-ch Gamma Corrected Value 155** Default Value: 9b00h Format: U16
	47:32	**Inverse Pixel Value 155** Default Value: 9b00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 155** Default Value: 9b00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 155** Default Value: 9b00h Format: U16
622..623	63:48	**Forward R-ch Gamma Corrected Value 155** Default Value: 9b00h Format: U16
	47:32	**Forward Pixel Value 155** Default Value: 9b00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 155**	
		Default Value:	9b00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 155**	
		Default Value:	9b00h
		Format:	U16
624..625	63:48	**Inverse R-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
	47:32	**Inverse Pixel Value 156**	
		Default Value:	9c00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
626..627	63:48	**Forward R-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
	47:32	**Forward Pixel Value 156**	
		Default Value:	9c00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 156**	
		Default Value:	9c00h
		Format:	U16
628..629	63:48	**Inverse R-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
	47:32	**Inverse Pixel Value 157**	
		Default Value:	9d00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
630..631	63:48	**Forward R-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
	47:32	**Forward Pixel Value 157**	
		Default Value:	9d00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 157**	
		Default Value:	9d00h
		Format:	U16
632..633	63:48	**Inverse R-ch Gamma Corrected Value 158**	
		Default Value:	9e00h
		Format:	U16
	47:32	**Inverse Pixel Value 158**	
		Default Value:	9e00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 158**	
		Default Value:	9e00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 158**	
		Default Value:	9e00h
		Format:	U16
634..635	63:48	**Forward R-ch Gamma Corrected Value 158**	
		Default Value:	9e00h
		Format:	U16
	47:32	**Forward Pixel Value 158**	
		Default Value:	9e00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 158** Default Value: 9e00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 158** Default Value: 9e00h Format: U16
636..637	63:48	**Inverse R-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
	47:32	**Inverse Pixel Value 159** Default Value: 9f00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
638..639	63:48	**Forward R-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
	47:32	**Forward Pixel Value 159** Default Value: 9f00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 159** Default Value: 9f00h Format: U16
640..641	63:48	**Inverse R-ch Gamma Corrected Value 160** Default Value: a000h Format: U16
	47:32	**Inverse Pixel Value 160** Default Value: a000h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 160**	
		Default Value:	a000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 160**	
		Default Value:	a000h
		Format:	U16
642..643	63:48	**Forward R-ch Gamma Corrected Value 160**	
		Default Value:	a000h
		Format:	U16
	47:32	**Forward Pixel Value 160**	
		Default Value:	a000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 160**	
		Default Value:	a000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 160**	
		Default Value:	a000h
		Format:	U16
644..645	63:48	**Inverse R-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
	47:32	**Inverse Pixel Value 161**	
		Default Value:	a100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
646..647	63:48	**Forward R-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
	47:32	**Forward Pixel Value 161**	
		Default Value:	a100h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 161**	
		Default Value:	a100h
		Format:	U16
648..649	63:48	**Inverse R-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
	47:32	**Inverse Pixel Value 162**	
		Default Value:	a200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
650..651	63:48	**Forward R-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
	47:32	**Forward Pixel Value 162**	
		Default Value:	a200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 162**	
		Default Value:	a200h
		Format:	U16
652..653	63:48	**Inverse R-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
	47:32	**Inverse Pixel Value 163**	
		Default Value:	a300h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
654..655	63:48	**Forward R-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
	47:32	**Forward Pixel Value 163**	
		Default Value:	a300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 163**	
		Default Value:	a300h
		Format:	U16
656..657	63:48	**Inverse R-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
	47:32	**Inverse Pixel Value 164**	
		Default Value:	a400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
658..659	63:48	**Forward R-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
	47:32	**Forward Pixel Value 164**	
		Default Value:	a400h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 164**	
		Default Value:	a400h
		Format:	U16
660..661	63:48	**Inverse R-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
	47:32	**Inverse Pixel Value 165**	
		Default Value:	a500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
662..663	63:48	**Forward R-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
	47:32	**Forward Pixel Value 165**	
		Default Value:	a500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 165**	
		Default Value:	a500h
		Format:	U16
664..665	63:48	**Inverse R-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
	47:32	**Inverse Pixel Value 166**	
		Default Value:	a600h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
666..667	63:48	**Forward R-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
	47:32	**Forward Pixel Value 166**	
		Default Value:	a600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 166**	
		Default Value:	a600h
		Format:	U16
668..669	63:48	**Inverse R-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
	47:32	**Inverse Pixel Value 167**	
		Default Value:	a700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
670..671	63:48	**Forward R-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
	47:32	**Forward Pixel Value 167**	
		Default Value:	a700h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 167**	
		Default Value:	a700h
		Format:	U16
672..673	63:48	**Inverse R-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
	47:32	**Inverse Pixel Value 168**	
		Default Value:	a800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
674..675	63:48	**Forward R-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
	47:32	**Forward Pixel Value 168**	
		Default Value:	a800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 168**	
		Default Value:	a800h
		Format:	U16
676..677	63:48	**Inverse R-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
	47:32	**Inverse Pixel Value 169**	
		Default Value:	a900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
678..679	63:48	**Forward R-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
	47:32	**Forward Pixel Value 169**	
		Default Value:	a900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 169**	
		Default Value:	a900h
		Format:	U16
680..681	63:48	**Inverse R-ch Gamma Corrected Value 170**	
		Default Value:	aa00h
		Format:	U16
	47:32	**Inverse Pixel Value 170**	
		Default Value:	aa00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 170**	
		Default Value:	aa00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 170**	
		Default Value:	aa00h
		Format:	U16
682..683	63:48	**Forward R-ch Gamma Corrected Value 170**	
		Default Value:	aa00h
		Format:	U16
	47:32	**Forward Pixel Value 170**	
		Default Value:	aa00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 170** Default Value: aa00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 170** Default Value: aa00h Format: U16
684..685	63:48	**Inverse R-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
	47:32	**Inverse Pixel Value 171** Default Value: ab00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
686..687	63:48	**Forward R-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
	47:32	**Forward Pixel Value 171** Default Value: ab00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 171** Default Value: ab00h Format: U16
688..689	63:48	**Inverse R-ch Gamma Corrected Value 172** Default Value: ac00h Format: U16
	47:32	**Inverse Pixel Value 172** Default Value: ac00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 172**	
		Default Value:	ac00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 172**	
		Default Value:	ac00h
		Format:	U16
690..691	63:48	**Forward R-ch Gamma Corrected Value 172**	
		Default Value:	ac00h
		Format:	U16
	47:32	**Forward Pixel Value 172**	
		Default Value:	ac00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 172**	
		Default Value:	ac00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 172**	
		Default Value:	ac00h
		Format:	U16
692..693	63:48	**Inverse R-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
	47:32	**Inverse Pixel Value 173**	
		Default Value:	ad00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
694..695	63:48	**Forward R-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
	47:32	**Forward Pixel Value 173**	
		Default Value:	ad00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 173**	
		Default Value:	ad00h
		Format:	U16
696..697	63:48	**Inverse R-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
	47:32	**Inverse Pixel Value 174**	
		Default Value:	ae00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
698..699	63:48	**Forward R-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
	47:32	**Forward Pixel Value 174**	
		Default Value:	ae00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 174**	
		Default Value:	ae00h
		Format:	U16
700..701	63:48	**Inverse R-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
	47:32	**Inverse Pixel Value 175**	
		Default Value:	af00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
702..703	63:48	**Forward R-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
	47:32	**Forward Pixel Value 175**	
		Default Value:	af00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 175**	
		Default Value:	af00h
		Format:	U16
704..705	63:48	**Inverse R-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
	47:32	**Inverse Pixel Value 176**	
		Default Value:	b000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
706..707	63:48	**Forward R-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
	47:32	**Forward Pixel Value 176**	
		Default Value:	b000h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 176**	
		Default Value:	b000h
		Format:	U16
708..709	63:48	**Inverse R-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
	47:32	**Inverse Pixel Value 177**	
		Default Value:	b100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
710..711	63:48	**Forward R-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
	47:32	**Forward Pixel Value 177**	
		Default Value:	b100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 177**	
		Default Value:	b100h
		Format:	U16
712..713	63:48	**Inverse R-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
	47:32	**Inverse Pixel Value 178**	
		Default Value:	b200h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
714..715	63:48	**Forward R-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
	47:32	**Forward Pixel Value 178**	
		Default Value:	b200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 178**	
		Default Value:	b200h
		Format:	U16
716..717	63:48	**Inverse R-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
	47:32	**Inverse Pixel Value 179**	
		Default Value:	b300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
718..719	63:48	**Forward R-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
	47:32	**Forward Pixel Value 179**	
		Default Value:	b300h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 179**	
		Default Value:	b300h
		Format:	U16
720..721	63:48	**Inverse R-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
	47:32	**Inverse Pixel Value 180**	
		Default Value:	b400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
722..723	63:48	**Forward R-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
	47:32	**Forward Pixel Value 180**	
		Default Value:	b400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 180**	
		Default Value:	b400h
		Format:	U16
724..725	63:48	**Inverse R-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
	47:32	**Inverse Pixel Value 181**	
		Default Value:	b500h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
726..727	63:48	**Forward R-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
	47:32	**Forward Pixel Value 181**	
		Default Value:	b500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 181**	
		Default Value:	b500h
		Format:	U16
728..729	63:48	**Inverse R-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
	47:32	**Inverse Pixel Value 182**	
		Default Value:	b600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
730..731	63:48	**Forward R-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
	47:32	**Forward Pixel Value 182**	
		Default Value:	b600h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 182**	
		Default Value:	b600h
		Format:	U16
732..733	63:48	**Inverse R-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
	47:32	**Inverse Pixel Value 183**	
		Default Value:	b700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
734..735	63:48	**Forward R-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
	47:32	**Forward Pixel Value 183**	
		Default Value:	b700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 183**	
		Default Value:	b700h
		Format:	U16
736..737	63:48	**Inverse R-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
	47:32	**Inverse Pixel Value 184**	
		Default Value:	b800h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
738..739	63:48	**Forward R-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
	47:32	**Forward Pixel Value 184**	
		Default Value:	b800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 184**	
		Default Value:	b800h
		Format:	U16
740..741	63:48	**Inverse R-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
	47:32	**Inverse Pixel Value 185**	
		Default Value:	b900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
742..743	63:48	**Forward R-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
	47:32	**Forward Pixel Value 185**	
		Default Value:	b900h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 185**	
		Default Value:	b900h
		Format:	U16
744..745	63:48	**Inverse R-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
	47:32	**Inverse Pixel Value 186**	
		Default Value:	ba00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
746..747	63:48	**Forward R-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
	47:32	**Forward Pixel Value 186**	
		Default Value:	ba00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 186**	
		Default Value:	ba00h
		Format:	U16
748..749	63:48	**Inverse R-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
	47:32	**Inverse Pixel Value 187**	
		Default Value:	bb00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
750..751	63:48	**Forward R-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
	47:32	**Forward Pixel Value 187**	
		Default Value:	bb00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 187**	
		Default Value:	bb00h
		Format:	U16
752..753	63:48	**Inverse R-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
	47:32	**Inverse Pixel Value 188**	
		Default Value:	bc00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
754..755	63:48	**Forward R-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
	47:32	**Forward Pixel Value 188**	
		Default Value:	bc00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 188**	
		Default Value:	bc00h
		Format:	U16
756..757	63:48	**Inverse R-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
	47:32	**Inverse Pixel Value 189**	
		Default Value:	bd00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
758..759	63:48	**Forward R-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
	47:32	**Forward Pixel Value 189**	
		Default Value:	bd00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 189**	
		Default Value:	bd00h
		Format:	U16
760..761	63:48	**Inverse R-ch Gamma Corrected Value 190**	
		Default Value:	be00h
		Format:	U16
	47:32	**Inverse Pixel Value 190**	
		Default Value:	be00h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 190**		
			Default Value:	be00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 190**		
			Default Value:	be00h
			Format:	U16
762..763	63:48	**Forward R-ch Gamma Corrected Value 190**		
			Default Value:	be00h
			Format:	U16
	47:32	**Forward Pixel Value 190**		
			Default Value:	be00h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 190**		
			Default Value:	be00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 190**		
			Default Value:	be00h
			Format:	U16
764..765	63:48	**Inverse R-ch Gamma Corrected Value 191**		
			Default Value:	bf00h
			Format:	U16
	47:32	**Inverse Pixel Value 191**		
			Default Value:	bf00h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 191**		
			Default Value:	bf00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 191**		
			Default Value:	bf00h
			Format:	U16
766..767	63:48	**Forward R-ch Gamma Corrected Value 191**		
			Default Value:	bf00h
			Format:	U16
	47:32	**Forward Pixel Value 191**		
			Default Value:	bf00h
			Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 191**	
		Default Value:	bf00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 191**	
		Default Value:	bf00h
		Format:	U16
768..769	63:48	**Inverse R-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
	47:32	**Inverse Pixel Value 192**	
		Default Value:	c000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
770..771	63:48	**Forward R-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
	47:32	**Forward Pixel Value 192**	
		Default Value:	c000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 192**	
		Default Value:	c000h
		Format:	U16
772..773	63:48	**Inverse R-ch Gamma Corrected Value 193**	
		Default Value:	c100h
		Format:	U16
	47:32	**Inverse Pixel Value 193**	
		Default Value:	c100h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 193**		
			Default Value:	c100h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 193**		
			Default Value:	c100h
			Format:	U16
774..775	63:48	**Forward R-ch Gamma Corrected Value 193**		
			Default Value:	c100h
			Format:	U16
	47:32	**Forward Pixel Value 193**		
			Default Value:	c100h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 193**		
			Default Value:	c100h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 193**		
			Default Value:	c100h
			Format:	U16
776..777	63:48	**Inverse R-ch Gamma Corrected Value 194**		
			Default Value:	c200h
			Format:	U16
	47:32	**Inverse Pixel Value 194**		
			Default Value:	c200h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 194**		
			Default Value:	c200h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 194**		
			Default Value:	c200h
			Format:	U16
778..779	63:48	**Forward R-ch Gamma Corrected Value 194**		
			Default Value:	c200h
			Format:	U16
	47:32	**Forward Pixel Value 194**		
			Default Value:	c200h
			Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 194** Default Value: c200h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 194** Default Value: c200h Format: U16
780..781	63:48	**Inverse R-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
	47:32	**Inverse Pixel Value 195** Default Value: c300h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
782..783	63:48	**Forward R-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
	47:32	**Forward Pixel Value 195** Default Value: c300h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 195** Default Value: c300h Format: U16
784..785	63:48	**Inverse R-ch Gamma Corrected Value 196** Default Value: c400h Format: U16
	47:32	**Inverse Pixel Value 196** Default Value: c400h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 196**	
		Default Value:	c400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 196**	
		Default Value:	c400h
		Format:	U16
786..787	63:48	**Forward R-ch Gamma Corrected Value 196**	
		Default Value:	c400h
		Format:	U16
	47:32	**Forward Pixel Value 196**	
		Default Value:	c400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 196**	
		Default Value:	c400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 196**	
		Default Value:	c400h
		Format:	U16
788..789	63:48	**Inverse R-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
	47:32	**Inverse Pixel Value 197**	
		Default Value:	c500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
790..791	63:48	**Forward R-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
	47:32	**Forward Pixel Value 197**	
		Default Value:	c500h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 197**	
		Default Value:	c500h
		Format:	U16
792..793	63:48	**Inverse R-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
	47:32	**Inverse Pixel Value 198**	
		Default Value:	c600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
794..795	63:48	**Forward R-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
	47:32	**Forward Pixel Value 198**	
		Default Value:	c600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 198**	
		Default Value:	c600h
		Format:	U16
796..797	63:48	**Inverse R-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
	47:32	**Inverse Pixel Value 199**	
		Default Value:	c700h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
798..799	63:48	**Forward R-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
	47:32	**Forward Pixel Value 199**	
		Default Value:	c700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 199**	
		Default Value:	c700h
		Format:	U16
800..801	63:48	**Inverse R-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
	47:32	**Inverse Pixel Value 200**	
		Default Value:	c800h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
802..803	63:48	**Forward R-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
	47:32	**Forward Pixel Value 200**	
		Default Value:	c800h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 200**	
		Default Value:	c800h
		Format:	U16
804..805	63:48	**Inverse R-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
	47:32	**Inverse Pixel Value 201**	
		Default Value:	c900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
806..807	63:48	**Forward R-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
	47:32	**Forward Pixel Value 201**	
		Default Value:	c900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 201**	
		Default Value:	c900h
		Format:	U16
808..809	63:48	**Inverse R-ch Gamma Corrected Value 202**	
		Default Value:	ca00h
		Format:	U16
	47:32	**Inverse Pixel Value 202**	
		Default Value:	ca00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 202** Default Value: ca00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 202** Default Value: ca00h Format: U16
810..811	63:48	**Forward R-ch Gamma Corrected Value 202** Default Value: ca00h Format: U16
	47:32	**Forward Pixel Value 202** Default Value: ca00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 202** Default Value: ca00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 202** Default Value: ca00h Format: U16
812..813	63:48	**Inverse R-ch Gamma Corrected Value 203** Default Value: cb00h Format: U16
	47:32	**Inverse Pixel Value 203** Default Value: cb00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 203** Default Value: cb00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 203** Default Value: cb00h Format: U16
814..815	63:48	**Forward R-ch Gamma Corrected Value 203** Default Value: cb00h Format: U16
	47:32	**Forward Pixel Value 203** Default Value: cb00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 203**	
		Default Value:	cb00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 203**	
		Default Value:	cb00h
		Format:	U16
816..817	63:48	**Inverse R-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
	47:32	**Inverse Pixel Value 204**	
		Default Value:	cc00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
818..819	63:48	**Forward R-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
	47:32	**Forward Pixel Value 204**	
		Default Value:	cc00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 204**	
		Default Value:	cc00h
		Format:	U16
820..821	63:48	**Inverse R-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
	47:32	**Inverse Pixel Value 205**	
		Default Value:	cd00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
822..823	63:48	**Forward R-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
	47:32	**Forward Pixel Value 205**	
		Default Value:	cd00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 205**	
		Default Value:	cd00h
		Format:	U16
824..825	63:48	**Inverse R-ch Gamma Corrected Value 206**	
		Default Value:	ce00h
		Format:	U16
	47:32	**Inverse Pixel Value 206**	
		Default Value:	ce00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 206**	
		Default Value:	ce00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 206**	
		Default Value:	ce00h
		Format:	U16
826..827	63:48	**Forward R-ch Gamma Corrected Value 206**	
		Default Value:	ce00h
		Format:	U16
	47:32	**Forward Pixel Value 206**	
		Default Value:	ce00h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Forward B-ch Gamma Corrected Value 206**		
			Default Value:	ce00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 206**		
			Default Value:	ce00h
			Format:	U16
828..829	63:48	**Inverse R-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
	47:32	**Inverse Pixel Value 207**		
			Default Value:	cf00h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
830..831	63:48	**Forward R-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
	47:32	**Forward Pixel Value 207**		
			Default Value:	cf00h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 207**		
			Default Value:	cf00h
			Format:	U16
832..833	63:48	**Inverse R-ch Gamma Corrected Value 208**		
			Default Value:	d000h
			Format:	U16
	47:32	**Inverse Pixel Value 208**		
			Default Value:	d000h
			Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 208** Default Value: d000h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 208** Default Value: d000h Format: U16
834..835	63:48	**Forward R-ch Gamma Corrected Value 208** Default Value: d000h Format: U16
	47:32	**Forward Pixel Value 208** Default Value: d000h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 208** Default Value: d000h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 208** Default Value: d000h Format: U16
836..837	63:48	**Inverse R-ch Gamma Corrected Value 209** Default Value: d100h Format: U16
	47:32	**Inverse Pixel Value 209** Default Value: d100h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 209** Default Value: d100h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 209** Default Value: d100h Format: U16
838..839	63:48	**Forward R-ch Gamma Corrected Value 209** Default Value: d100h Format: U16
	47:32	**Forward Pixel Value 209** Default Value: d100h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 209**	
		Default Value:	d100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 209**	
		Default Value:	d100h
		Format:	U16
840..841	63:48	**Inverse R-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
	47:32	**Inverse Pixel Value 210**	
		Default Value:	d200h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
842..843	63:48	**Forward R-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
	47:32	**Forward Pixel Value 210**	
		Default Value:	d200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 210**	
		Default Value:	d200h
		Format:	U16
844..845	63:48	**Inverse R-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
	47:32	**Inverse Pixel Value 211**	
		Default Value:	d300h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
846..847	63:48	**Forward R-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
	47:32	**Forward Pixel Value 211**	
		Default Value:	d300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 211**	
		Default Value:	d300h
		Format:	U16
848..849	63:48	**Inverse R-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
	47:32	**Inverse Pixel Value 212**	
		Default Value:	d400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
850..851	63:48	**Forward R-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
	47:32	**Forward Pixel Value 212**	
		Default Value:	d400h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 212**	
		Default Value:	d400h
		Format:	U16
852..853	63:48	**Inverse R-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
	47:32	**Inverse Pixel Value 213**	
		Default Value:	d500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
854..855	63:48	**Forward R-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
	47:32	**Forward Pixel Value 213**	
		Default Value:	d500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 213**	
		Default Value:	d500h
		Format:	U16
856..857	63:48	**Inverse R-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
	47:32	**Inverse Pixel Value 214**	
		Default Value:	d600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
858..859	63:48	**Forward R-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
	47:32	**Forward Pixel Value 214**	
		Default Value:	d600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 214**	
		Default Value:	d600h
		Format:	U16
860..861	63:48	**Inverse R-ch Gamma Corrected Value 215**	
		Default Value:	d700h
		Format:	U16
	47:32	**Inverse Pixel Value 215**	
		Default Value:	d700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 215**	
		Default Value:	d700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 215**	
		Default Value:	d700h
		Format:	U16
862..863	63:48	**Forward R-ch Gamma Corrected Value 215**	
		Default Value:	d700h
		Format:	U16
	47:32	**Forward Pixel Value 215**	
		Default Value:	d700h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 215**
		Default Value: d700h
		Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 215**
		Default Value: d700h
		Format: U16
864..865	63:48	**Inverse R-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
	47:32	**Inverse Pixel Value 216**
		Default Value: d800h
		Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
866..867	63:48	**Forward R-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
	47:32	**Forward Pixel Value 216**
		Default Value: d800h
		Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 216**
		Default Value: d800h
		Format: U16
868..869	63:48	**Inverse R-ch Gamma Corrected Value 217**
		Default Value: d900h
		Format: U16
	47:32	**Inverse Pixel Value 217**
		Default Value: d900h
		Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 217**	
		Default Value:	d900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 217**	
		Default Value:	d900h
		Format:	U16
870..871	63:48	**Forward R-ch Gamma Corrected Value 217**	
		Default Value:	d900h
		Format:	U16
	47:32	**Forward Pixel Value 217**	
		Default Value:	d900h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 217**	
		Default Value:	d900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 217**	
		Default Value:	d900h
		Format:	U16
872..873	63:48	**Inverse R-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
	47:32	**Inverse Pixel Value 218**	
		Default Value:	da00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
874..875	63:48	**Forward R-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
	47:32	**Forward Pixel Value 218**	
		Default Value:	da00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 218**	
		Default Value:	da00h
		Format:	U16
876..877	63:48	**Inverse R-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
	47:32	**Inverse Pixel Value 219**	
		Default Value:	db00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
878..879	63:48	**Forward R-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
	47:32	**Forward Pixel Value 219**	
		Default Value:	db00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 219**	
		Default Value:	db00h
		Format:	U16
880..881	63:48	**Inverse R-ch Gamma Corrected Value 220**	
		Default Value:	dc00h
		Format:	U16
	47:32	**Inverse Pixel Value 220**	
		Default Value:	dc00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Inverse B-ch Gamma Corrected Value 220** Default Value: dc00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 220** Default Value: dc00h Format: U16
882..883	63:48	**Forward R-ch Gamma Corrected Value 220** Default Value: dc00h Format: U16
	47:32	**Forward Pixel Value 220** Default Value: dc00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 220** Default Value: dc00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 220** Default Value: dc00h Format: U16
884..885	63:48	**Inverse R-ch Gamma Corrected Value 221** Default Value: dd00h Format: U16
	47:32	**Inverse Pixel Value 221** Default Value: dd00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 221** Default Value: dd00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 221** Default Value: dd00h Format: U16
886..887	63:48	**Forward R-ch Gamma Corrected Value 221** Default Value: dd00h Format: U16
	47:32	**Forward Pixel Value 221** Default Value: dd00h Format: U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Forward B-ch Gamma Corrected Value 221**		
			Default Value:	dd00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 221**		
			Default Value:	dd00h
			Format:	U16
888..889	63:48	**Inverse R-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
	47:32	**Inverse Pixel Value 222**		
			Default Value:	de00h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
890..891	63:48	**Forward R-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
	47:32	**Forward Pixel Value 222**		
			Default Value:	de00h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 222**		
			Default Value:	de00h
			Format:	U16
892..893	63:48	**Inverse R-ch Gamma Corrected Value 223**		
			Default Value:	df00h
			Format:	U16
	47:32	**Inverse Pixel Value 223**		
			Default Value:	df00h
			Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 223**	
		Default Value:	df00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 223**	
		Default Value:	df00h
		Format:	U16
894..895	63:48	**Forward R-ch Gamma Corrected Value 223**	
		Default Value:	df00h
		Format:	U16
	47:32	**Forward Pixel Value 223**	
		Default Value:	df00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 223**	
		Default Value:	df00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 223**	
		Default Value:	df00h
		Format:	U16
896..897	63:48	**Inverse R-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
	47:32	**Inverse Pixel Value 224**	
		Default Value:	e000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
898..899	63:48	**Forward R-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
	47:32	**Forward Pixel Value 224**	
		Default Value:	e000h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 224**	
		Default Value:	e000h
		Format:	U16
900..901	63:48	**Inverse R-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
	47:32	**Inverse Pixel Value 225**	
		Default Value:	e100h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
902..903	63:48	**Forward R-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
	47:32	**Forward Pixel Value 225**	
		Default Value:	e100h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 225**	
		Default Value:	e100h
		Format:	U16
904..905	63:48	**Inverse R-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
	47:32	**Inverse Pixel Value 226**	
		Default Value:	e200h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
906..907	63:48	**Forward R-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
	47:32	**Forward Pixel Value 226**	
		Default Value:	e200h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 226**	
		Default Value:	e200h
		Format:	U16
908..909	63:48	**Inverse R-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
	47:32	**Inverse Pixel Value 227**	
		Default Value:	e300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
910..911	63:48	**Forward R-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
	47:32	**Forward Pixel Value 227**	
		Default Value:	e300h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 227**	
		Default Value:	e300h
		Format:	U16
912..913	63:48	**Inverse R-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
	47:32	**Inverse Pixel Value 228**	
		Default Value:	e400h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
914..915	63:48	**Forward R-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
	47:32	**Forward Pixel Value 228**	
		Default Value:	e400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 228**	
		Default Value:	e400h
		Format:	U16
916..917	63:48	**Inverse R-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
	47:32	**Inverse Pixel Value 229**	
		Default Value:	e500h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
918..919	63:48	**Forward R-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
	47:32	**Forward Pixel Value 229**	
		Default Value:	e500h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 229**	
		Default Value:	e500h
		Format:	U16
920..921	63:48	**Inverse R-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
	47:32	**Inverse Pixel Value 230**	
		Default Value:	e600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
922..923	63:48	**Forward R-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
	47:32	**Forward Pixel Value 230**	
		Default Value:	e600h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 230**	
		Default Value:	e600h
		Format:	U16
924..925	63:48	**Inverse R-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
	47:32	**Inverse Pixel Value 231**	
		Default Value:	e700h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
926..927	63:48	**Forward R-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
	47:32	**Forward Pixel Value 231**	
		Default Value:	e700h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 231**	
		Default Value:	e700h
		Format:	U16
928..929	63:48	**Inverse R-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
	47:32	**Inverse Pixel Value 232**	
		Default Value:	e800h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
930..931	63:48	**Forward R-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
	47:32	**Forward Pixel Value 232**	
		Default Value:	e800h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 232**	
		Default Value:	e800h
		Format:	U16
932..933	63:48	**Inverse R-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
	47:32	**Inverse Pixel Value 233**	
		Default Value:	e900h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
934..935	63:48	**Forward R-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
	47:32	**Forward Pixel Value 233**	
		Default Value:	e900h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 233**	
		Default Value:	e900h
		Format:	U16
936..937	63:48	**Inverse R-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
	47:32	**Inverse Pixel Value 234**	
		Default Value:	ea00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
938..939	63:48	**Forward R-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
	47:32	**Forward Pixel Value 234**	
		Default Value:	ea00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 234**	
		Default Value:	ea00h
		Format:	U16
940..941	63:48	**Inverse R-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
	47:32	**Inverse Pixel Value 235**	
		Default Value:	eb00h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
942..943	63:48	**Forward R-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
	47:32	**Forward Pixel Value 235**	
		Default Value:	eb00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 235**	
		Default Value:	eb00h
		Format:	U16
944..945	63:48	**Inverse R-ch Gamma Corrected Value 236**	
		Default Value:	ec00h
		Format:	U16
	47:32	**Inverse Pixel Value 236**	
		Default Value:	ec00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 236**	
		Default Value:	ec00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 236**	
		Default Value:	ec00h
		Format:	U16
946..947	63:48	**Forward R-ch Gamma Corrected Value 236**	
		Default Value:	ec00h
		Format:	U16
	47:32	**Forward Pixel Value 236**	
		Default Value:	ec00h
		Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 236** Default Value: ec00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 236** Default Value: ec00h Format: U16
948..949	63:48	**Inverse R-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
	47:32	**Inverse Pixel Value 237** Default Value: ed00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
950..951	63:48	**Forward R-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
	47:32	**Forward Pixel Value 237** Default Value: ed00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 237** Default Value: ed00h Format: U16
952..953	63:48	**Inverse R-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
	47:32	**Inverse Pixel Value 238** Default Value: ee00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Inverse B-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
954..955	63:48	**Forward R-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
	47:32	**Forward Pixel Value 238** Default Value: ee00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 238** Default Value: ee00h Format: U16
956..957	63:48	**Inverse R-ch Gamma Corrected Value 239** Default Value: ef00h Format: U16
	47:32	**Inverse Pixel Value 239** Default Value: ef00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 239** Default Value: ef00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 239** Default Value: ef00h Format: U16
958..959	63:48	**Forward R-ch Gamma Corrected Value 239** Default Value: ef00h Format: U16
	47:32	**Forward Pixel Value 239** Default Value: ef00h Format: U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 239**	
		Default Value:	ef00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 239**	
		Default Value:	ef00h
		Format:	U16
960..961	63:48	**Inverse R-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
	47:32	**Inverse Pixel Value 240**	
		Default Value:	f000h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
962..963	63:48	**Forward R-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
	47:32	**Forward Pixel Value 240**	
		Default Value:	f000h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 240**	
		Default Value:	f000h
		Format:	U16
964..965	63:48	**Inverse R-ch Gamma Corrected Value 241**	
		Default Value:	f100h
		Format:	U16
	47:32	**Inverse Pixel Value 241**	
		Default Value:	f100h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 241**		
			Default Value:	f100h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 241**		
			Default Value:	f100h
			Format:	U16
966..967	63:48	**Forward R-ch Gamma Corrected Value 241**		
			Default Value:	f100h
			Format:	U16
	47:32	**Forward Pixel Value 241**		
			Default Value:	f100h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 241**		
			Default Value:	f100h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 241**		
			Default Value:	f100h
			Format:	U16
968..969	63:48	**Inverse R-ch Gamma Corrected Value 242**		
			Default Value:	f200h
			Format:	U16
	47:32	**Inverse Pixel Value 242**		
			Default Value:	f200h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 242**		
			Default Value:	f200h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 242**		
			Default Value:	f200h
			Format:	U16
970..971	63:48	**Forward R-ch Gamma Corrected Value 242**		
			Default Value:	f200h
			Format:	U16
	47:32	**Forward Pixel Value 242**		
			Default Value:	f200h
			Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 242**	
		Default Value:	f200h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 242**	
		Default Value:	f200h
		Format:	U16
972..973	63:48	**Inverse R-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
	47:32	**Inverse Pixel Value 243**	
		Default Value:	f300h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
974..975	63:48	**Forward R-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
	47:32	**Forward Pixel Value 243**	
		Default Value:	f300h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 243**	
		Default Value:	f300h
		Format:	U16
976..977	63:48	**Inverse R-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
	47:32	**Inverse Pixel Value 244**	
		Default Value:	f400h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
978..979	63:48	**Forward R-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
	47:32	**Forward Pixel Value 244**	
		Default Value:	f400h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 244**	
		Default Value:	f400h
		Format:	U16
980..981	63:48	**Inverse R-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
	47:32	**Inverse Pixel Value 245**	
		Default Value:	f500h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
982..983	63:48	**Forward R-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
	47:32	**Forward Pixel Value 245**	
		Default Value:	f500h
		Format:	U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Forward B-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 245**	
		Default Value:	f500h
		Format:	U16
984..985	63:48	**Inverse R-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
	47:32	**Inverse Pixel Value 246**	
		Default Value:	f600h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
986..987	63:48	**Forward R-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
	47:32	**Forward Pixel Value 246**	
		Default Value:	f600h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 246**	
		Default Value:	f600h
		Format:	U16
988..989	63:48	**Inverse R-ch Gamma Corrected Value 247**	
		Default Value:	f700h
		Format:	U16
	47:32	**Inverse Pixel Value 247**	
		Default Value:	f700h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Inverse B-ch Gamma Corrected Value 247**		
			Default Value:	f700h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 247**		
			Default Value:	f700h
			Format:	U16
990..991	63:48	**Forward R-ch Gamma Corrected Value 247**		
			Default Value:	f700h
			Format:	U16
	47:32	**Forward Pixel Value 247**		
			Default Value:	f700h
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 247**		
			Default Value:	f700h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 247**		
			Default Value:	f700h
			Format:	U16
992..993	63:48	**Inverse R-ch Gamma Corrected Value 248**		
			Default Value:	f800h
			Format:	U16
	47:32	**Inverse Pixel Value 248**		
			Default Value:	f800h
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 248**		
			Default Value:	f800h
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 248**		
			Default Value:	f800h
			Format:	U16
994..995	63:48	**Forward R-ch Gamma Corrected Value 248**		
			Default Value:	f800h
			Format:	U16
	47:32	**Forward Pixel Value 248**		
			Default Value:	f800h
			Format:	U16

Gamut_Expansion_Gamma_Correction		
	31:16	**Forward B-ch Gamma Corrected Value 248** Default Value: f800h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 248** Default Value: f800h Format: U16
996..997	63:48	**Inverse R-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
	47:32	**Inverse Pixel Value 249** Default Value: f900h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
998..999	63:48	**Forward R-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
	47:32	**Forward Pixel Value 249** Default Value: f900h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 249** Default Value: f900h Format: U16
1000..1001	63:48	**Inverse R-ch Gamma Corrected Value 250** Default Value: fa00h Format: U16
	47:32	**Inverse Pixel Value 250** Default Value: fa00h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 250**	
		Default Value:	fa00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 250**	
		Default Value:	fa00h
		Format:	U16
1002..1003	63:48	**Forward R-ch Gamma Corrected Value 250**	
		Default Value:	fa00h
		Format:	U16
	47:32	**Forward Pixel Value 250**	
		Default Value:	fa00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 250**	
		Default Value:	fa00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 250**	
		Default Value:	fa00h
		Format:	U16
1004..1005	63:48	**Inverse R-ch Gamma Corrected Value 251**	
		Default Value:	fb00h
		Format:	U16
	47:32	**Inverse Pixel Value 251**	
		Default Value:	fb00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 251**	
		Default Value:	fb00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 251**	
		Default Value:	fb00h
		Format:	U16
1006..1007	63:48	**Forward R-ch Gamma Corrected Value 251**	
		Default Value:	fb00h
		Format:	U16
	47:32	**Forward Pixel Value 251**	
		Default Value:	fb00h
		Format:	U16

Gamut_Expansion_Gamma_Correction

	31:16	**Forward B-ch Gamma Corrected Value 251** Default Value: fb00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 251** Default Value: fb00h Format: U16
1008..1009	63:48	**Inverse R-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
	47:32	**Inverse Pixel Value 252** Default Value: fc00h Format: U16
	31:16	**Inverse B-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
	15:0	**Inverse G-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
1010..1011	63:48	**Forward R-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
	47:32	**Forward Pixel Value 252** Default Value: fc00h Format: U16
	31:16	**Forward B-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
	15:0	**Forward G-ch Gamma Corrected Value 252** Default Value: fc00h Format: U16
1012..1013	63:48	**Inverse R-ch Gamma Corrected Value 253** Default Value: fd00h Format: U16
	47:32	**Inverse Pixel Value 253** Default Value: fd00h Format: U16

Gamut_Expansion_Gamma_Correction			
	31:16	**Inverse B-ch Gamma Corrected Value 253**	
		Default Value:	fd00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 253**	
		Default Value:	fd00h
		Format:	U16
1014..1015	63:48	**Forward R-ch Gamma Corrected Value 253**	
		Default Value:	fd00h
		Format:	U16
	47:32	**Forward Pixel Value 253**	
		Default Value:	fd00h
		Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 253**	
		Default Value:	fd00h
		Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 253**	
		Default Value:	fd00h
		Format:	U16
1016..1017	63:48	**Inverse R-ch Gamma Corrected Value 254**	
		Default Value:	fe00h
		Format:	U16
	47:32	**Inverse Pixel Value 254**	
		Default Value:	fe00h
		Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 254**	
		Default Value:	fe00h
		Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 254**	
		Default Value:	fe00h
		Format:	U16
1018..1019	63:48	**Forward R-ch Gamma Corrected Value 254**	
		Default Value:	fe00h
		Format:	U16
	47:32	**Forward Pixel Value 254**	
		Default Value:	fe00h
		Format:	U16

Gamut_Expansion_Gamma_Correction				
	31:16	**Forward B-ch Gamma Corrected Value 254**		
			Default Value:	fe00h
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 254**		
			Default Value:	fe00h
			Format:	U16
1020..1021	63:48	**Inverse R-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16
	47:32	**Inverse Pixel Value 255**		
			Default Value:	ffffh
			Format:	U16
	31:16	**Inverse B-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16
	15:0	**Inverse G-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16
1022..1023	63:48	**Forward R-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16
	47:32	**Forward Pixel Value 255**		
			Default Value:	ffffh
			Format:	U16
	31:16	**Forward B-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16
	15:0	**Forward G-ch Gamma Corrected Value 255**		
			Default Value:	ffffh
			Format:	U16

GraphicsAddress63-1

GA63-1 - GraphicsAddress63-1		
Source:	BSpec	
Size (in bits):	63	
Default Value:	0x00000000, 0x00000000	

This structure is intended to define the upper bits of the GraphicsAddress, when bit 0 is already defined in the referring register. So bit 0 of this structure should correspond to bit 1 of the full GraphicsAddress.

DWord	Bit	Description
0..1 GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The upper reserved bits are ignored and must be zero. Some GraphicsAddress fields only specify the upper address bits. For example GraphicsAddress[47:12] would be a 4KB page address.	62:47	**Reserved** Format: MBZ
	46:0	**GraphicsAddress47-1** Format: GraphicsAddress[47:1] Bits 47:1 of a 48-bit GraphicsAddress. Look for bit 0 definition in the referring register.

GraphicsAddress63-12

<table>
<tr><th colspan="3">GA63-12 - GraphicsAddress63-12</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 52
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">This structure is intended to define the upper bits of the GraphicsAddress, when bits 11:0 are already defined in the referring register. So bit 0 of this structure should correspond to bit 12 of the full GraphicsAddress.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0..1

GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The upper reserved bits are ignored and must be zero.
Some GraphicsAddress fields only specify the upper address bits. For example GraphicsAddress[47:12] would be a 4KB page address.</td><td>51:36</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>35:0</td><td>GraphicsAddress47-12
<table><tr><td>Format:</td><td>GraphicsAddress[47:12]</td></tr></table>Bits 47:12 of a 48-bit GraphicsAddress. Look for the definition of bits 11:0 in the referring register.</td></tr>
</table>

GTC Interrupt Bit Definition

GTC Interrupt Bit Definition		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	
The GTC Interrupt Registers all share the same bit definitions from this table.		
DWord	**Bit**	**Description**
0	31	**GTC Lock Loss** GTC has lost lock with a remote GTC sink. The difference between the local and remote GTC has exceeded programmed threshold.
	30:22	**Reserved** Format: MBZ
	21	**GTC Aux Rx Error USBC6** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	20	**GTC Update Complete USBC6** A hardware initiated GTC update has completed with a sink attached to this port.
	19	**GTC Aux Rx Error USBC5** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	18	**GTC Update Complete USBC5** A hardware initiated GTC update has completed with a sink attached to this port.
	17	**GTC Aux Rx Error USBC4** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	16	**GTC Update Complete USBC4** A hardware initiated GTC update has completed with a sink attached to this port.
	15	**GTC Aux Rx Error USBC3** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	14	**GTC Update Complete USBC3** A hardware initiated GTC update has completed with a sink attached to this port.
	13	**GTC Aux Rx Error USBC2** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	12	**GTC Update Complete USBC2** A hardware initiated GTC update has completed with a sink attached to this port.
	11	**GTC Aux Rx Error USBC1** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	10	**GTC Update Complete USBC1** A hardware initiated GTC update has completed with a sink attached to this port.
	9:6	**Reserved** Format: MBZ

GTC Interrupt Bit Definition		
	5	**GTC Aux Rx Error DDIC** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	4	**GTC Update Complete DDIC** A hardware initiated GTC update has completed with a sink attached to this port.
	3	**GTC Aux Rx Error DDIB** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	2	**GTC Update Complete DDIB** A hardware initiated GTC update has completed with a sink attached to this port.
	1	**GTC Aux Rx Error DDIA** An aux channel error occurred during GTC transfer with remote GTC sink attached to this port.
	0	**GTC Update Complete DDIA** A hardware initiated GTC update has completed with a sink attached to this port.

GTPM Interrupt Vector

GTPM_INTR_VEC - GTPM Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	15:14	**Reserved**
	13	**Unslice Frequency Control Up Interrupt**
	12	**Unslice Frequency Control Down Interrupt**
	11	**NFADFL Frequency Up Interrupt**
	10	**NFADFL Frequency Down Interrupt**
	9	**Reserved**
	8	**GTPM Engines Idle Interrupt**
	7	**GTPM Uncore to Core Trap Interrupt**
	6	**GTPM Render Frequency Downwards Timeout During RC6 Interrupt**
	5	**GTPM Render P-State Up Threshold Interrupt**
	4	**GTPM Render P-State Down Threshold Interrupt**
	3	**Spare 3**
	2	**GTPM Render Geyserville Up Evaluation Interval Interrupt**
	1	**GTPM Render Geyserville Down Evaluation Interval Interrupt**
	0	**Reserved**

Half Precision Dual Source SIMD8 Message Data Payload Register

MDPR_DSH_SIMD8 - Half Precision Dual Source SIMD8 Message Data Payload Register

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

DWord	Bit	Description
0	31:16	**Src0 Data1** Format: F16 Specifies the source 0 slot 1 data in this payload register
	15:0	**Src0 Data0** Format: F16 Specifies the source 0 slot 0 data in this payload register
1	31:16	**Src0 Data3** Format: F16 Specifies the source 0 slot 3 data in this payload register
	15:0	**Src0 Data2** Format: F16 Specifies the source 0 slot 2 data in this payload register
2	31:16	**Src0 Data5** Format: F16 Specifies the source 0 slot 5 data in this payload register
	15:0	**Src0 Data4** Format: F16 Specifies the source 0 slot 4 data in this payload register
3	31:16	**Src0 Data7**

MDPR_DSH_SIMD8 - Half Precision Dual Source SIMD8 Message Data Payload Register

		Format: F16 Specifies the source 0 slot 7 data in this payload register
	15:0	**Src0 Data6** Format: F16 Specifies the source 0 slot 6 data in this payload register
4	31:16	**Src1 Data1** Format: F16 Specifies the source 1 slot 1 data in this payload register
	15:0	**Src1 Data0** Format: F16 Specifies the source 1 slot 0 data in this payload register
5	31:16	**Src1 Data3** Format: F16 Specifies the source 1 slot 3 data in this payload register
	15:0	**Src1 Data2** Format: F16 Specifies the source 1 slot 2 data in this payload register
6	31:16	**Src1 Data5** Format: F16 Specifies the source 1 slot 5 data in this payload register
	15:0	**Src1 Data4** Format: F16 Specifies the source 1 slot 4 data in this payload register
7	31:16	**Src1 Data7**

MDPR_DSH_SIMD8 - Half Precision Dual Source SIMD8 Message Data Payload Register

		Format: F16 Specifies the source 1 slot 7 data in this payload register
	15:0	**Src1 Data6** Format: F16 Specifies the source 1 slot 6 data in this payload register

Half Precision OM Replicated SIMD16 Render Target Data Payload

MDP_RTWH_M16REP - Half Precision 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_H_RGBA RGBA for all slots [15:0]

Half Precision OM S0A SIMD8 Render Target Data Payload

MDP_RTWH_MA8 - Half Precision 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: **MDPR_H_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: **MDPR_H_SIMD8** Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: **MDPR_H_SIMD8** Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: **MDPR_H_SIMD8** Slots [7:0] Blue
5.0-5.7	255:0	**Alpha**

MDP_RTWH_MA8 - Half Precision OM S0A SIMD8 Render Target Data Payload

<table>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDPR_H_SIMD8</td></tr></table>Slots [7:0] Alpha</td></tr>
</table>

Half Precision OM S0A SIMD16 Render Target Data Payload

MDP_RTWH_MA16 - Half Precision OM S0A SIMD16 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[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Source 0 Alpha
1.0-1.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
2.0-2.7	255:0	**Red[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Red
3.0-3.7	255:0	**Green[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Green
4.0-4.7	255:0	**Blue[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Blue
5.0-5.7	255:0	**Alpha[15:0]**

<table>
<tr><th colspan="3">MDP_RTWH_MA16 - Half Precision OM S0A SIMD16 Render Target Data Payload</th></tr>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDPR_H_SIMD16</td></tr></table>Slots [15:0] Alpha</td></tr>
</table>

Half Precision OM SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_M8DS - Half Precision OM SIMD8 Dual Source 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** oMask for slots [7:0] and [15:8]. Operation selects upper or lower half.
1.0-1.7	255:0	**Red** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Red
2.0-2.7	255:0	**Green** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Green
3.0-3.7	255:0	**Blue** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Blue
4.0-4.7	255:0	**Alpha** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Alpha

Half Precision OM SIMD8 Render Target Data Payload

<table>
<tr><th colspan="3">MDP_RTWH_M8 - Half Precision OM SIMD8 Render Target Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1280
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>oMask
Format: MDPR_OMASK
Slots [7:0] oMask. Upper half ignored.</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Red
Format: MDPR_H_SIMD8
Slots [7:0] Red</td></tr>
<tr><td>2.0-2.7</td><td>255:0</td><td>Green
Format: MDPR_H_SIMD8
Slots [7:0] Green</td></tr>
<tr><td>3.0-3.7</td><td>255:0</td><td>Blue
Format: MDPR_H_SIMD8
Slots [7:0] Blue</td></tr>
<tr><td>4.0-4.7</td><td>255:0</td><td>Alpha
Format: MDPR_H_SIMD8
Slots [7:0] Alpha</td></tr>
</table>

Half Precision OM SIMD16 Render Target Data Payload

MDP_RTWH_M16 - Half Precision OM SIMD16 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 [15:0] oMask
1.0-1.7	255:0	**Red[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Red
2.0-2.7	255:0	**Green[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Green
3.0-3.7	255:0	**Blue[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Blue
4.0-4.7	255:0	**Alpha[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Alpha

Half Precision OS OM S0A SIMD8 Render Target Data Payload

MDP_RTWH_SMA8 - Half Precision OS 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: MDPR_H_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: MDPR_H_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue

MDP_RTWH_SMA8 - Half Precision OS OM S0A SIMD8 Render Target Data Payload		
5.0-5.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
6.0-6.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

Half Precision OS OM SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_SM8DS - Half Precision OS OM SIMD8 Dual Source 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 oMask for slots [7:0] and [15:8]. Operation selects upper or lower half.
1.0-1.7	255:0	**Red** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
2.0-2.7	255:0	**Green** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
3.0-3.7	255:0	**Blue** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha
5.0-5.7	255:0	**Stencil**

MDP_RTWH_SM8DS - Half Precision OS OM SIMD8 Dual Source Render Target Data Payload		
		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDPR_STENCIL</td></tr></table>Slots [7:0] or [15:8] of Stencil

Half Precision OS OM SIMD8 Render Target Data Payload

MDP_RTWH_SM8 - Half Precision OS 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: MDPR_H_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
5.0-5.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

Half Precision OS S0A SIMD8 Render Target Data Payload

MDP_RTWH_SA8 - Half Precision OS 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: MDPR_H_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Red** Format: MDPR_H_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
5.0-5.7	255:0	**Stencil**

MDP_RTWH_SA8 - Half Precision OS S0A SIMD8 Render Target Data Payload

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDPR_STENCIL</td></tr></table>Slots [7:0] Stencil

Half Precision OS SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_S8DS - Half Precision OS SIMD8 Dual Source 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: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
1.0-1.7	255:0	**Green** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
2.0-2.7	255:0	**Blue** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha
4.0-4.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] or [15:8] of Stencil

Half Precision OS SIMD8 Render Target Data Payload

MDP_RTWH_S8 - Half Precision OS 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: MDPR_H_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
2.0-2.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
4.0-4.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

Half Precision OS SZ OM S0A SIMD8 Render Target Data Payload

MDP_RTWH_SZMA8 - Half Precision OS SZ OM S0A SIMD8 Render Target Data Payload

Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDPR_H_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: MDPR_H_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Alpha**

MDP_RTWH_SZMA8 - Half Precision OS SZ OM S0A SIMD8 Render Target Data Payload		
		Format: **MDPR_H_SIMD8** Slots [7:0] Alpha
6.0-6.7	255:0	**Source Depth** Format: **MDP_DW_SIMD8** Slots [7:0] Source Depth
7.0-7.7	255:0	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] Stencil

Half Precision OS SZ OM SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_SZM8DS - Half Precision OS SZ OM SIMD8 Dual Source Render Target Data Payload

Source:	BSpec
Size (in bits):	1792
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	**Red** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
2.0-2.7	255:0	**Green** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
3.0-3.7	255:0	**Blue** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha

MDP_RTWH_SZM8DS - Half Precision OS SZ OM SIMD8 Dual Source Render Target Data Payload		
5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] or [15:8] of Source Depth
6.0-6.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] or [15:8] of Stencil

Half Precision OS SZ OM SIMD8 Render Target Data Payload

MDP_RTWH_SZM8 - Half Precision OS SZ OM SIMD8 Render Target Data Payload		
Source:	BSpec	
Size (in bits):	1792	
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: **MDPR_H_SIMD8** Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: **MDPR_H_SIMD8** Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: **MDPR_H_SIMD8** Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: **MDPR_H_SIMD8** Slots [7:0] Alpha
5.0-5.7	255:0	**Source Depth**

		Format: **MDP_DW_SIMD8** Slots [7:0] Source Depth
6.0-6.7	255:0	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] Stencil

Half Precision OS SZ S0A SIMD8 Render Target Data Payload

<table>
<tr><th colspan="3">MDP_RTWH_SZA8 - Half Precision OS SZ S0A SIMD8 Render Target Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1792
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Source 0 Alpha
Format: MDPR_H_SIMD8
Slots [7:0] Source 0 Alpha</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Red
Format: MDPR_H_SIMD8
Slots [7:0] Red</td></tr>
<tr><td>2.0-2.7</td><td>255:0</td><td>Green
Format: MDPR_H_SIMD8
Slots [7:0] Green</td></tr>
<tr><td>3.0-3.7</td><td>255:0</td><td>Blue
Format: MDPR_H_SIMD8
Slots [7:0] Blue</td></tr>
<tr><td>4.0-4.7</td><td>255:0</td><td>Alpha
Format: MDPR_H_SIMD8
Slots [7:0] Alpha</td></tr>
<tr><td>5.0-5.7</td><td>255:0</td><td>Source Depth
Format: MDP_DW_SIMD8
Slots [7:0] Source Depth</td></tr>
<tr><td>6.0-6.7</td><td>255:0</td><td>Stencil</td></tr>
</table>

MDP_RTWH_SZA8 - Half Precision OS SZ S0A SIMD8 Render Target Data Payload

		Format:	MDPR_STENCIL
		Slots [7:0] Stencil	

Half Precision OS SZ SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_SZ8DS - Half Precision OS SZ SIMD8 Dual Source Render Target Data Payload

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

DWord	Bit	Description
0.0-0.7	255:0	**Red** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Red
1.0-1.7	255:0	**Green** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Green
2.0-2.7	255:0	**Blue** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Blue
3.0-3.7	255:0	**Alpha** Format: **MDPR_DSH_SIMD8** Slots[7:0] or [15:8] of Src0 and Src1 Alpha
4.0-4.7	255:0	**Source Depth** Format: **MDP_DW_SIMD8** Slots [7:0] or [15:8] of Source Depth
5.0-5.7	255:0	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] or [15:8] of Stencil

Half Precision OS SZ SIMD8 Render Target Data Payload

<table>
<tr><th colspan="3">MDP_RTWH_SZ8 - Half Precision OS SZ SIMD8 Render Target Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1536
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Red
Format: MDPR_H_SIMD8
Slots [7:0] Red</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Green
Format: MDPR_H_SIMD8
Slots [7:0] Green</td></tr>
<tr><td>2.0-2.7</td><td>255:0</td><td>Blue
Format: MDPR_H_SIMD8
Slots [7:0] Blue</td></tr>
<tr><td>3.0-3.7</td><td>255:0</td><td>Alpha
Format: MDPR_H_SIMD8
Slots [7:0] Alpha</td></tr>
<tr><td>4.0-4.7</td><td>255:0</td><td>Source Depth
Format: MDP_DW_SIMD8
Slots [7:0] Source Depth</td></tr>
<tr><td>5.0-5.7</td><td>255:0</td><td>Stencil</td></tr>
</table>

MDP_RTWH_SZ8 - Half Precision OS SZ SIMD8 Render Target Data Payload		
		Format: MDPR_STENCIL Slots [7:0] Stencil

Half Precision Replicated Pixel Render Target Data Payload Register

MDPR_H_RGBA - Half Precision Replicated Pixel Render Target Data Payload Register

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

DWord	Bit	Description
0	31:16	**Green** Format: U16 Specifies the value of all slots' green channel.
	15:0	**Red** Format: U16 Specifies the value of all slots' red channel.
1	31:16	**Alpha** Format: U16 Specifies the value of all slots' alpha channel.
	15:0	**Blue** Format: U16 Specifies the value of all slots' blue channel.
2..7	191:0	**Reserved** Format: Ignore Ignored

Half Precision Replicated SIMD16 Render Target Data Payload

MDP_RTWH_16REP - Half Precision 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_H_RGBA RGBA for all slots [15:0]

Half Precision S0A SIMD8 Render Target Data Payload

MDP_RTWH_A8 - Half Precision 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: MDPR_H_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Red** Format: MDPR_H_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha

Half Precision S0A SIMD16 Render Target Data Payload

MDP_RTWH_A16 - Half Precision S0A SIMD16 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[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Source 0 Alpha
1.0-1.7	255:0	**Red[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Red
2.0-2.7	255:0	**Green[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Green
3.0-3.7	255:0	**Blue[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Blue
4.0-4.7	255:0	**Alpha[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Alpha

Half Precision SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_8DS - Half Precision SIMD8 Dual Source 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: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
1.0-1.7	255:0	**Green** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
2.0-2.7	255:0	**Blue** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha

Half Precision SIMD8 Message Data Payload Register

MDPR_H_SIMD8 - Half Precision SIMD8 Message Data Payload Register

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

DWord	Bit	Description
0	31:16	**Data1** Format: F16 Specifies the slot 1 data in this payload register
	15:0	**Data0** Format: F16 Specifies the slot 0 data in this payload register
1	31:16	**Data3** Format: F16 Specifies the slot 3 data in this payload register
	15:0	**Data2** Format: F16 Specifies the slot 2 data in this payload register
2	31:16	**Data5** Format: F16 Specifies the slot 5 data in this payload register
	15:0	**Data4** Format: F16 Specifies the slot 4 data in this payload register
3	31:16	**Data7**

MDPR_H_SIMD8 - Half Precision SIMD8 Message Data Payload Register

		Format: F16 Specifies the slot 7 data in this payload register
	15:0	**Data6** Format: F16 Specifies the slot 6 data in this payload register
4..7	127:0	**Reserved** Format: Ignore Ignored

Half Precision SIMD8 Render Target Data Payload

MDP_RTWH_8 - Half Precision 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: MDPR_H_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
2.0-2.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha

Half Precision SIMD16 Message Data Payload Register

MDPR_H_SIMD16 - Half Precision SIMD16 Message Data Payload Register		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31:16	**Data1** Format: F16 Specifies the slot 1 data in this payload register
	15:0	**Data0** Format: F16 Specifies the slot 0 data in this payload register
1	31:16	**Data3** Format: F16 Specifies the slot 3 data in this payload register
	15:0	**Data2** Format: F16 Specifies the slot 2 data in this payload register
2	31:16	**Data5** Format: F16 Specifies the slot 5 data in this payload register
	15:0	**Data4** Format: F16 Specifies the slot 4 data in this payload register
3	31:16	**Data7**

MDPR_H_SIMD16 - Half Precision SIMD16 Message Data Payload Register

<table>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 7 data in this payload register</td></tr>
<tr><td></td><td>15:0</td><td>Data6<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 6 data in this payload register</td></tr>
<tr><td>4</td><td>31:16</td><td>Data9<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 9 data in this payload register</td></tr>
<tr><td></td><td>15:0</td><td>Data8<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 8 data in this payload register</td></tr>
<tr><td>5</td><td>31:16</td><td>Data11<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 11 data in this payload register</td></tr>
<tr><td></td><td>15:0</td><td>Data10<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 10 data in this payload register</td></tr>
<tr><td>6</td><td>31:16</td><td>Data13<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 13 data in this payload register</td></tr>
<tr><td></td><td>15:0</td><td>Data12<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 12 data in this payload register</td></tr>
<tr><td>7</td><td>31:16</td><td>Data15</td></tr>
</table>

MDPR_H_SIMD16 - Half Precision SIMD16 Message Data Payload Register

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 15 data in this payload register
	15:0	**Data14**<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>F16</td></tr></table>Specifies the slot 14 data in this payload register

Half Precision SIMD16 Render Target Data Payload

<table>
<tr><th colspan="3">MDP_RTWH_16 - Half Precision SIMD16 Render Target Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1024
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Red[15:0]
Format: MDPR_H_SIMD16
Slots [15:0] Red</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Green[15:0]
Format: MDPR_H_SIMD16
Slots [15:0] Green</td></tr>
<tr><td>2.0-2.7</td><td>255:0</td><td>Blue[15:0]
Format: MDPR_H_SIMD16
Slots [15:0] Blue</td></tr>
<tr><td>3.0-3.7</td><td>255:0</td><td>Alpha[15:0]
Format: MDPR_H_SIMD16
Slots [15:0] Alpha</td></tr>
</table>

Half Precision SZ OM S0A SIMD8 Render Target Data Payload

MDP_RTWH_ZMA8 - Half Precision 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: MDPR_H_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: MDPR_H_SIMD8 Slots [7:0] Red
3.0-3.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
4.0-4.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
5.0-5.7	255:0	**Alpha**

<table>
<tr><th colspan="3">MDP_RTWH_ZMA8 - Half Precision SZ OM S0A SIMD8 Render Target Data Payload</th></tr>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDPR_H_SIMD8</td></tr></table>Slots [7:0] Alpha</td></tr>
<tr><td>6.0-6.7</td><td>255:0</td><td>Source Depth<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDP_DW_SIMD8</td></tr></table>Slots [7:0] Source Depth</td></tr>
</table>

Half Precision SZ OM S0A SIMD16 Render Target Data Payload

MDP_RTWH_ZMA16 - Half Precision SZ OM S0A SIMD16 Render Target Data Payload

Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Source 0 Alpha** Format: MDPR_H_SIMD16 Slots [15:0] Source 0 Alpha
1.0-1.7	255:0	**oMask** Format: MDPR_OMASK Slots [15:0] oMask
2.0-2.7	255:0	**Red** Format: MDPR_H_SIMD16 Slots [15:0] Red
3.0-3.7	255:0	**Green** Format: MDPR_H_SIMD16 Slots [15:0] Green
4.0-4.7	255:0	**Blue** Format: MDPR_H_SIMD16 Slots [15:0] Blue
5.0-5.7	255:0	**Alpha**

MDP_RTWH_ZMA16 - Half Precision SZ OM S0A SIMD16 Render Target Data Payload		
		Format: MDPR_H_SIMD16 Slots [15:0] Alpha
6.0-6.7	255:0	**Source Depth[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
7.0-7.7	255:0	**Source Depth[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

Half Precision SZ OM SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_ZM8DS - Half Precision SZ OM SIMD8 Dual Source 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 oMask for slots [7:0] and [15:8]. Operation selects upper or lower half.
1.0-1.7	255:0	**Red** Format: MDPR_DSHSIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
2.0-2.7	255:0	**Green** Format: MDPR_DSHSIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
3.0-3.7	255:0	**Blue** Format: MDPR_DSHSIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_DSHSIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha
5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] or [15:8] of Source Depth

Half Precision SZ OM SIMD8 Render Target Data Payload

MDP_RTWH_ZM8 - Half Precision 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: **MDPR_H_SIMD8** Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: **MDPR_H_SIMD8** Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: **MDPR_H_SIMD8** Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: **MDPR_H_SIMD8** Slots [7:0] Alpha
5.0-5.7	255:0	**Source Depth**

MDP_RTWH_ZM8 - Half Precision SZ OM SIMD8 Render Target Data Payload

<table>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MDP_DW_SIMD8</td></tr></table>Slots [7:0] Source Depth</td></tr>
</table>

Half Precision SZ OM SIMD16 Render Target Data Payload

MDP_RTWH_ZM16 - Half Precision SZ OM SIMD16 Render Target Data Payload		
Source:	BSpec	
Size (in bits):	1792	
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[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Red
2.0-2.7	255:0	**Green[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Green
3.0-3.7	255:0	**Blue[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Blue
4.0-4.7	255:0	**Alpha[15:0]** Format: MDPR_H_SIMD16 Slots [15:0] Alpha
5.0-5.7	255:0	**Source Depth[7:0]** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
6.0-6.7	255:0	**Source Depth[15:8]**

MDP_RTWH_ZM16 - Half Precision SZ OM SIMD16 Render Target Data Payload

		Format:	**MDP_DW_SIMD8**
		Slots [15:8] Source Depth	

Half Precision SZ S0A SIMD8 Render Target Data Payload

MDP_RTWH_ZA8 - Half Precision 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: MDPR_H_SIMD8 Slots [7:0] Source 0 Alpha
1.0-1.7	255:0	**Red** Format: MDPR_H_SIMD8 Slots [7:0] Red
2.0-2.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
3.0-3.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
4.0-4.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
5.0-5.7	255:0	**Source Depth**

MDP_RTWH_ZA8 - Half Precision SZ S0A SIMD8 Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

Half Precision SZ S0A SIMD16 Render Target Data Payload

MDP_RTWH_ZA16 - Half Precision SZ S0A SIMD16 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[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Source 0 Alpha
1.0-1.7	255:0	**Red[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Red
2.0-2.7	255:0	**Green[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Green
3.0-3.7	255:0	**Blue[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Blue
4.0-4.7	255:0	**Alpha[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Alpha
5.0-5.7	255:0	**Source Depth[7:0]**

MDP_RTWH_ZA16 - Half Precision SZ S0A SIMD16 Render Target Data Payload		
		Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
6.0-6.7	255:0	**Source Depth[15:8]** Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

Half Precision SZ SIMD8 Dual Source Render Target Data Payload

MDP_RTWH_Z8DS - Half Precision SZ SIMD8 Dual Source 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: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Red
1.0-1.7	255:0	**Green** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Green
2.0-2.7	255:0	**Blue** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_DSH_SIMD8 Slots[7:0] or [15:8] of Src0 and Src1 Alpha
4.0-4.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] or [15:8] of Source Depth

Half Precision SZ SIMD8 Render Target Data Payload

MDP_RTWH_Z8 - Half Precision 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: MDPR_H_SIMD8 Slots [7:0] Red
1.0-1.7	255:0	**Green** Format: MDPR_H_SIMD8 Slots [7:0] Green
2.0-2.7	255:0	**Blue** Format: MDPR_H_SIMD8 Slots [7:0] Blue
3.0-3.7	255:0	**Alpha** Format: MDPR_H_SIMD8 Slots [7:0] Alpha
4.0-4.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth

Half Precision SZ SIMD16 Render Target Data Payload

MDP_RTWH_Z16 - Half Precision SZ SIMD16 Render Target Data Payload

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

DWord	Bit	Description
0.0-0.7	255:0	**Red[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Red
1.0-1.7	255:0	**Green[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Green
2.0-2.7	255:0	**Blue[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Blue
3.0-3.7	255:0	**Alpha[15:0]** Format: **MDPR_H_SIMD16** Slots [15:0] Alpha
4.0-4.7	255:0	**Source Depth[7:0]** Format: **MDP_DW_SIMD8** Slots [7:0] Source Depth
5.0-5.7	255:0	**Source Depth[15:8]**

MDP_RTWH_Z16 - Half Precision SZ SIMD16 Render Target Data Payload		
		Format: MDP_DW_SIMD8 Slots [15:8] Source Depth

Hardware-Detected Error Bit Definitions

Hardware-Detected Error Bit Definitions		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:8	**Reserved** Format: MBZ
	7	**Reserved**
	6:3	**Reserved**
	2	**Command Privilege Violation Error** This bit is set if a command classified as privileged is parsed in a non-privileged batch buffer. The command will be converted to a NOOP and parsing will continue.
	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

Hardware Status Page Layout	
Source:	RenderCS
Size (in bits):	640
Default Value:	0x00000000, 0x00000000,

Hardware Status Page Layout
0x00000000, 0x00000000,

Hardware Status Page Layout
0x00000000, 0x00000000,

Hardware Status Page Layout

0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000

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.

Hardware Status Page Layout		
4	31:0	**Ring Head Pointer Storage** **Description** 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	383:0	**Context Status DWords** Format: CONTEXT_STATUS[12]
28..39	31:0	**Reserved**
40..46	31:0	**Reserved**
47	31:0	**Last Written Status Offset**
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.

HCP_PAK_INSERT_OBJECT_INDIRECT_PAYLOAD

HCP_PAK_INSERT_OBJECT_INDIRECT_PAYLOAD

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

DWord	Bit	Description
0	31:0	**Indirect Payload Data Size in bits** Format: U32 Number of bits to be inserted. Not including those skipped bytes in the beginning. For VP9: the Data is always valid from start of cache-line, no offset is allowed.
1..2	63:0	**Indirect Payload Base Address** Format: SplitBaseAddress64ByteAligned 48-bit address of the indirect payload data in memory buffer. **Programming Notes** Payload must begin in a byte position, but the payload can be ended in a bit position.
3	31:0	**Indirect Payload Base Address** Format: MemoryAddressAttributes

HCP_REF_LIST_ENTRY

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

DWord	Bit	Description
0	31:16	**Reserved** Format: MBZ
	15	**bottom_field_flag** Format: U1 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15. Value / Name: 0 — Bottom Field 1 — Top Field **Programming Notes** Not supported in encoder mode.
	14	**field_pic_flag** Format: U1 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15. Value / Name: 0 — Video Frame 1 — Video Field **Programming Notes** Not supported in encoder mode.
	13	**LongTermReference** Format: U1 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15. Value / Name: 0 — Short term reference 1 — Long term reference
	12	**luma_weight_lX_flag** Format: U1 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to

<table>
<tr><th colspan="3">HCP_REF_LIST_ENTRY</th></tr>
<tr><td></td><td></td><td>i=0, DW17 corresponds to i=15.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0</td><td>Default weighted prediction for luma</td></tr>
<tr><td>1</td><td>Explicit weighted prediction for Luma</td></tr>
</table></td></tr>
<tr><td></td><td>11</td><td>chroma_weight_lX_flag
<table>
<tr><td>Format:</td><td>U1</td></tr>
</table>
Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0</td><td>Default weighted prediction for Chroma</td></tr>
<tr><td>1</td><td>Explicit weighted prediction for Chroma</td></tr>
</table></td></tr>
<tr><td></td><td>10:8</td><td>list_entry_lX: Reference Picture Frame ID (RefAddr[0-7])
<table>
<tr><td>Format:</td><td>U3</td></tr>
</table>
Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15.
The reference picture frame ID identifies the reference picture associated with the base address defined in Reference Picture Address (RefAddr[0-7]) in the HCP_PIPE_BUF_ADDR_STATE command.</td></tr>
<tr><td></td><td>7:0</td><td>Reference Picture tb Value
<table>
<tr><td>Format:</td><td>U8</td></tr>
</table>
Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15.
clip(-128,127, CurrentPOC - RefPOC), where RefPOC is the POC value of the reference picture. 8-bit signed.
See the "Derivation process for temporal luma motion vector prediction" in the HEVC standard.</td></tr>
</table>

HCP_TILE_POSITION_IN_CTB

HCP_TILE_POSITION_IN_CTB		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:24	**CtbPos3+i** Format: U8
	23:16	**CtbPos2+i** Format: U8
	15:8	**CtbPos1+i** Format: U8
	7:0	**CtbPos0+i** Format: U8

HCP_TILE_POSITION_IN_CTB_MSB

HCP_TILE_POSITION_IN_CTB_MSB		
Source:	BSpec	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Added to support 16k picture size.		
DWord	**Bit**	**Description**
0..1	63:44	**Reserved**
	43:42	**Ctb position of tile 21 [9:8]** MSB 2 bits of CTB row position of tile row 21. **Programming Notes** Please note that this field is MBZ for columns
	41:40	**Ctb row position of tile column 20 [9:8]** MSB 2 bits of CTB row position of tile row 20. **Programming Notes** Please note that this field is MBZ for columns
	39:38	**Ctb row position of tile column 19 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 19.
	37:36	**Ctb row position of tile column 18 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 18.
	35:34	**Ctb row position of tile column 17 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 17.
	33:32	**Ctb row position of tile column 16 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 16.
	31:30	**Ctb row position of tile column 15 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 15.
	29:28	**Ctb row position of tile column 14 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 14.
	27:26	**Ctb row position of tile column 13 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 13.
	25:24	**Ctb row position of tile column 12 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 12.
	23:22	**Ctb row position of tile column 11 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 11.
	21:20	**Ctb row position of tile column 10 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 10.
	19:18	**Ctb row position of tile column 9 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 9.
	17:16	**Ctb row position of tile column 8 [9:8]**

HCP_TILE_POSITION_IN_CTB_MSB		
		MSB 2 bits of CTB row or column position of tile row or column 8.
	15:14	**Ctb row position of tile column 7 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 7.
	13:12	**Ctb row position of tile column 6 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 6.
	11:10	**Ctb row position of tile column 5 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 5.
	9:8	**Ctb row position of tile column 4 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 4.
	7:6	**Ctb row position of tile column 3 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 3.
	5:4	**Ctb row position of tile column 2 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 2.
	3:2	**Ctb row position of tile column 1 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 1.
	1:0	**Ctb row position of tile column 0 [9:8]** MSB 2 bits of CTB row or column position of tile row or column 0.

HCP_WEIGHTOFFSET_CHROMA_ENTRY

HCP_WEIGHTOFFSET_CHROMA_ENTRY		
Source:	VideoCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:24	**ChromaOffsetLX [i][1]** Where X is the RefPicListNum and i is the list entry number 0 through 15. DW 18corresponds to i=0, DW 33 corresponds to i=15. Valid only if explicit weighted prediction for chroma is enabled, otherwise must be zero. **Programming Notes** This (combined with its MSbyte below) shall be in the range of -WpOffsetHalfRangeC to (WpOffsetHalfRangeC - 1), inclusive WpOffsetHalfRangeC = 1 « (high_precision_offsets_enabled_flag ? (BitDepthC - 1) : 7)
	23:16	**delta_chroma_weight_lX[i][1]** Format: S7 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW 18 corresponds to i=0, DW 33 corresponds to i=15. Valid only if explicit weighted prediction for chroma is enabled, otherwise must be zero. **Programming Notes** This shall be in the range of −128 to 127, inclusive
	15:8	**ChromaOffsetLX[i][0]** Where X is the RefPicListNum and i is the list entry number 0 through 15. DW 18 corresponds to i=0, DW 33 corresponds to i=15. Valid only if explicit weighted prediction for chroma is enabled, otherwise must be zero. **Programming Notes** This (combined with its MSbyte below) shall be in the range of -WpOffsetHalfRangeC to (WpOffsetHalfRangeC - 1), inclusive WpOffsetHalfRangeC = 1 « (high_precision_offsets_enabled_flag ? (BitDepthC - 1) : 7)
	7:0	**delta_chroma_weight_lX[i][0]** Format: S7 Where X is the RefPicListNum and i is the list entry number 0 through 15. DW 18 corresponds to i=0, DW 33 corresponds to i=15. Valid only if explicit weighted prediction for chroma is enabled, otherwise must be zero.

HCP_WEIGHTOFFSET_CHROMA_ENTRY

		Programming Notes
		This shall be in the range of −128 to 127, inclusive

HCP_WEIGHTOFFSET_CHROMA_EXT_ENTRY

<table>
<tr><th colspan="3">HCP_WEIGHTOFFSET_CHROMA_EXT_ENTRY</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>31:24</td><td>ChromaOffsetLX[i+1][1] MSByte

Description
To support 4:4:4, the chroma offset is extended into 16-bit.
In order to keep SW back compatible, the most significant byte is programmed here.

Programming Notes
This is only MSByte portion of ChromaOffsetLX. Please refer to LSB section for available range.</td></tr>
<tr><td>23:16</td><td>ChromaOffsetLX[i][1] MSByte

Description
To support 4:4:4, the chroma offset is extended into 16-bit.
In order to keep SW back compatible, the most significant byte is programmed here.

Programming Notes
This is only MSByte portion of ChromaOffsetLX. Please refer to LSB section for available range.</td></tr>
<tr><td>15:8</td><td>ChromaOffsetLX[i+1][0] MSByte
To support 4:4:4, the chroma offset is extended into 16-bit.
In order to keep SW back compatible, the most significant byte is programmed here.
Programming Notes
This is only MSByte portion of ChromaOffsetLX. Please refer to LSB section for available range.</td></tr>
<tr><td>7:0</td><td>ChromaOffsetLX[i][0] MSByte

Description
To support 4:4:4, the chroma offset is extended into 16-bit.
In order to keep SW back compatible, the most significant byte is programmed here.

Programming Notes
This is only MSByte portion of ChromaOffsetLX. Please refer to LSB section for available range.</td></tr>
</table>

HCP_WEIGHTOFFSET_LUMA_ENTRY

<table>
<tr><th colspan="3">HCP_WEIGHTOFFSET_LUMA_ENTRY</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>31:24</td><td>luma_offset_lX[i] MSByte
To support 4:4:4, the luma offset is extended into 16-bit.
In order to keep SW back compatible, the most significant byte is programmed here.
Programming Notes
This is only MSByte portion of luma_offset_lX. Please refer to LSB section for available range.</td></tr>
<tr><td>23:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:8</td><td>luma_offset_lX[i]
Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15.
Valid only if explicit weighted prediction for luma is enabled, otherwise must be zero.
Programming Notes
This (combined with it MSbyte above) shall be in the range of $-\text{WpOffsetHalfRange}_Y$ to $\text{WpOffsetHalfRange}_Y - 1$, where
$\text{WpOffsetHalfRange}_Y = 1 \ll (\text{high\_precision\_offsets\_enabled\_flag} ? (\text{BitDepth}_Y - 1) : 7)$</td></tr>
<tr><td>7:0</td><td>delta_luma_weight_lX[i]
Format: S7
Where X is the RefPicListNum and i is the list entry number 0 through 15. DW2 corresponds to i=0, DW17 corresponds to i=15.
Valid only if explicit weighted prediction for luma is enabled, otherwise must be zero.
Programming Notes
When luma_weight_l0_flag[i] is equal to 1, the value of delta_luma_weight_l0[i] shall be in the range of −128 to 127, inclusive.</td></tr>
</table>

Header Forbidden Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_MHF - Header Forbidden Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>Message Header Present
Indicates the message forbids a message header.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>No [Default]</td><td>Message header is not present</td></tr><tr><td>1h</td><td>Reserved</td><td>Not used</td></tr></table></td></tr>
</table>

Header Present Message Descriptor Control Field

MDC_MHP - Header Present Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**Message Header Present** Specifies if the message uses the optional message header. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>No</td><td>Message header is not present</td></tr><tr><td>1h</td><td>Yes</td><td>Message header is present</td></tr></table>

Header Required Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_MHR - Header Required Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1
Default Value: 0x00000001</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>Message Header Present
Indicates the message requires a message header.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>Reserved</td><td>Not used</td></tr><tr><td>1h</td><td>Yes [Default]</td><td>Message header is present</td></tr></table></td></tr>
</table>

HEVC_ARBITRATION_PRIORITY

HEVC_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

HEVC_VP9_RDOQ_LAMBDA_FIELDS

HEVC_VP9_RDOQ_LAMBDA_FIELDS		
Source:	VideoCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:16	**LambdaValue1** Lambda value for Intra Luma component of QP=1, 3, .. 9, 11, , 61, 63, ... 73, 75 (odd number) For 12-bit video, the QP range has extended to include 65 to 75. for HEVC
	15:0	**LambdaValue0** Lambda value for Intra Luma component of QP=0, 2, ...8, 10, , 60, 62, ... 72, 74 (even number) For 12-bit video, the QP range has extended to include 64to 74. for HEVC

HW Generated BINDING_TABLE_STATE

HW Generated BINDING_TABLE_STATE	
Source:	BSpec
Size (in bits):	16
Default Value:	0x00000000

Description
The binding table binds surfaces to logical resource indices used by shaders and other compute engine kernels. The HW generated Binding_Table_State have different format than the SW generated Binding_Table_State. The HW generated Binding_Table_State is stored as an array of 256 elements, each of which contains one word as defined here. The start of each element is spaced one word apart. The first element of the binding table is aligned to a 64-byte boundary. Binding table indexes beyond 256 will automatically be mapped to entry 0 by the HW, w/ the exception of any messages which support the special indexes 240 through 255, inclusive.

DWord	Bit	Description
0	15:0	**Surface State Pointer** Format: SurfaceStateOffset[21:6] []

Hword 1 Block Data Payload

MDP_HW1 - Hword 1 Block 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	**Hword** Format: U256 Specifies the Hword data

Hword 2 Block Data Payload

MDP_HW2 - Hword 2 Block 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	**Hword0** Format: U256 Specifies the Hword data for element 0
1.0-1.7	255:0	**Hword1** Format: U256 Specifies the Hword data for element 1

Hword 4 Block Data Payload

<table>
<tr><th colspan="3">MDP_HW4 - Hword 4 Block Data Payload</th></tr>
<tr><td>Source:</td><td colspan="2">BSpec</td></tr>
<tr><td>Size (in bits):</td><td colspan="2">1024</td></tr>
<tr><td>Default Value:</td><td colspan="2">0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Hword0
Format: U256
Specifies the Hword data for element 0</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Hword1
Format: U256
Specifies the Hword data for element 1</td></tr>
<tr><td>2.0-2.7</td><td>255:0</td><td>Hword2
Format: U256
Specifies the Hword data for element 2</td></tr>
<tr><td>3.0-3.7</td><td>255:0</td><td>Hword3
Format: U256
Specifies the Hword data for element 3</td></tr>
</table>

Hword 8 Block Data Payload

MDP_HW8 - Hword 8 Block Data Payload

Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Hword0** Format: U256 Specifies the Hword data for element 0
1.0-1.7	255:0	**Hword1** Format: U256 Specifies the Hword data for element 1
2.0-2.7	255:0	**Hword2** Format: U256 Specifies the Hword data for element 2
3.0-3.7	255:0	**Hword3** Format: U256 Specifies the Hword data for element 3
4.0-4.7	255:0	**Hword4** Format: U256 Specifies the Hword data for element 4
5.0-5.7	255:0	**Hword5** Format: U256 Specifies the Hword data for element 5
6.0-6.7	255:0	**Hword6**

MDP_HW8 - Hword 8 Block Data Payload		
		Format: U256 Specifies the Hword data for element 6
7.0-7.7	255:0	**Hword7** Format: U256 Specifies the Hword data for element 7

Hword Channel Mode Message Header Control

MHC_A64_CMODE - Hword Channel Mode Message Header Control		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31	**Reserved** Format: MDC_CMODE Specifies whether the read or write operation occurs on all 4 Dwords if any of those channel enables are set, or else only on the dwords whose corresponding channel enable is set.
	30:0	**Reserved** Format: MBZ Ignored

Hword Register Blocks Message Descriptor Control Field

MDC_DB_HW - Hword Register Blocks Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	1:0	**Register Blocks** Specifies the number of Hword blocks to be read or written

Value	Name	Description
00h	HW1	1 Hword register
01h	HW2	2 Hword registers
02h	HW4	4 Hword registers
03h	HW8	8 Hword registers

Ignored Message Header

MH_IGNORE - Ignored Message Header	
Source:	EuSubFunctionDataPort0
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

Some messages require a message header or have an optional message header, but do not use any information in the header.

DWord	Bit	Description
0..7	255:0	**Reserved** Format: MBZ Ignored

Inline Data Description for MFD_AVC_BSD_Object

<table>
<tr><th colspan="3">Inline Data Description for MFD_AVC_BSD_Object</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 96
Default Value: 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">This structure includes all the required Slice Header parameters and error handling settings for AVC_BSD_OBJECT Command (DW3..DW5).</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="6">0</td><td>31</td><td>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.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>Intra 16x16 Prediction</td></tr>
<tr><td>1</td><td></td><td>Inter P Copy</td></tr></table></td></tr>
<tr><td>30</td><td>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.</td></tr>
<tr><td>29</td><td>Intra PredMode (4x4/8x8 Luma) Error Control Bit
This field controls if AVC decoder will fix Intra Prediction Mode if the decoded value is incorrect according to MB position
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>AVC decoder will detect and fix IntraPredMode (4x4/8x8 Luma) Errors.</td></tr>
<tr><td>1</td><td></td><td>AVC decoder will NOT detect IntraPredMode (4x4/8x8 Luma) Errors. The wrong IntraPredMode value will be retaind.</td></tr></table></td></tr>
<tr><td>28:27</td><td>MB Error Concealment B Temporal Prediction mode
These two bits control how the reference L0/L1 are overridden in B temporal slice.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>00b</td><td>[Default]</td><td>Both Reference Indexes L0/L1 are forced to 0 during Concealment</td></tr>
<tr><td>01b</td><td></td><td>Only Reference Index L1 is forced to 0; Reference Index L0 is forced to -1</td></tr>
<tr><td>10b</td><td></td><td>Only Reference Index L0 is forced to 0; Reference Index L1 is forced to -1</td></tr>
<tr><td>11b</td><td>Reserved</td><td>Invalid</td></tr></table></td></tr>
<tr><td>26</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>25</td><td>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.</td></tr>
</table>

Inline Data Description for MFD_AVC_BSD_Object

Value	Name	Description
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	All	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

Inline Data Description for MFD_AVC_BSD_Object

Bit 10

Entropy 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 handling.

Bit 9

Reserved

Format:	MBZ

Bit 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.

Bits 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

Bit 5

Reserved

Format:	MBZ

Bit 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

Bit 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. 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.

Bit 2

Reserved

<table>
<tr><th colspan="3">Inline Data Description for MFD_AVC_BSD_Object</th></tr>
<tr><td></td><td></td><td><table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>1</td><td>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.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>[Default]</td><td>Motion Vectors are Overridden to 0 during MB Concealment</td></tr>
<tr><td>1</td><td></td><td>Predicted Motion Vectors are used during MB Concealment</td></tr></table></td></tr>
<tr><td></td><td>0</td><td>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.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>[Default]</td><td>Weight Prediction is Disabled during MB Concealment.</td></tr>
<tr><td>1</td><td></td><td>Weight Prediction will not be overridden during MB Concealment.</td></tr></table></td></tr>
<tr><td>1</td><td>31:16</td><td>First MB Byte Offset of Slice Data or Slice Header
<table><tr><th>Programming Notes</th></tr><tr><td>MFX supports only DXVA2 Long and Short Format.</td></tr></table></td></tr>
<tr><td></td><td>15:8</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>7</td><td>Fix Prev Mb Skipped
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.</td></tr>
<tr><td></td><td>6:5</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
<table><tr><th>Programming Notes</th></tr><tr><td>Please note that the field MUST be set to '0' at this time.</td></tr></table></td></tr>
<tr><td></td><td>4</td><td>Emulation Prevention Byte Present
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>H/W needs to perform Emulation Byte Removal</td></tr>
<tr><td>1</td><td></td><td>H/W does not need to perform Emulation Byte Removal</td></tr></table></td></tr>
<tr><td></td><td>3</td><td>LastSlice Flag
It is needed for both error concealment at the end of a picture (so, no more phantom slice. It is also needed to know to set the last MB in a picture correctly.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td></td><td>If the current Slice to be decoded is the very last slice of the current picture.</td></tr>
<tr><td>0</td><td></td><td>If the current Slice to be decoded is any slice other than the very last slice of the current picture</td></tr></table></td></tr>
</table>

Inline Data Description for MFD_AVC_BSD_Object

DWord	Bit	Description
	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.
2	31	**I Slice Concealment Mode** This field controls how AVC decoder handle MB concealment in I Slice Value / Name: 1 — Intra Concealment 0 — Inter Concealment **Programming Notes** If this field is set to "0" (Inter Concealment), driver must provide a valid reference picture (programmed using "Concealment Reference Picture" field) for concealment reference picture. In this mode, weight prediction is disabled, and motion vectors are forced to 0 as well.
	30	**Reserved** Format: MBZ
	29:24	**Concealment Reference Picture + Field Bit** Format: U6 This field provides the concealment reference picture for hardware to conceal in case driver wants to specify one concealment picture. This field matches with the DPB order sent to hardware. This field applies to all I/P/B slices Bit Filed / Value / Defenition: 29 — MBZ — Reserved 28:25 — All — Reference PIcture Number 24 — All — Field Bit(if the current picture is a field picture [Frame picture must be 0]
	23	**P Slice Concealment Mode** This field controls how AVC decoder handle MB concealment in P Slice Value / Name: 1 — Intra Concealment 0 — Inter Concealment
	22:19	**Reserved** Format: MBZ
	18:16	**P Slice Inter Concealment Mode** This field controls how AVC decoder select reference picture for Concealment in P Slice. Value / Name / Description

Inline Data Description for MFD_AVC_BSD_Object

000b		Top of Reference List L0 (Use top entry of Reference List L0)
001b		Driver Specified Concealment Reference
010b		Predicted Reference (Use reference picture predicted using P-Skip Algorithm)
011b		Temporal Closest (Using POC to select the closest forward picture) [For L0: Closest POC smaller than current POC]
100b		First Long Term Picture in Reference List L0 (If no long term picture available, use Temporal Closest Picture)
101b-111b	Reserved	

15 — **B Slice Concealment Mode**

This field controls how AVC decoder handle MB concealment in B Slice

Value	Name
1	Intra Concealment
0	Inter Concealment

14 — **Reserved**

Format:	MBZ

13:12 — **B Slice Inter Direct Type Concealment Mode**

AVC decoder can use Spatial or Temporal Direct for B Skip/Direct. This field determine can override the mode on how AVC decoder handles MB concealment in B slice.

Value	Name	Description
00b		Use Default Direct Type (slice programmed direct type)
01b		Forced to Spatial Direct Only
10b		Forced to Temporal Direct Only
11b		Spatial Direct without Temporal Componenet (MovingBlock information)

11 — **Reserved**

Format:	MBZ

10:8 — **B Slice Spatial Inter Concealment Mode**

This field controls how AVC decoder select reference picture for Spatial Inter Concealment in B Slice.

Value	Name	Description
000b		Top of Reference List L0/L1 (Use top entry of Reference List L0/L1).
001b		Driver Specified Concealment Reference
011b		Temporal Closest (Using POC to select the closest forward picture) [For L0: Closest POC smaller than current POC] [For L1: Closest POC larger than current POC]
100b		" First Long Term Picture in Reference List L0/L1 (If no long term picture available, use Temporal Closest Picture)

Inline Data Description for MFD_AVC_BSD_Object

		101b-111b | Reserved |
	7	**Reserved** Format: MBZ
	6:4	**B Slice Temporal Inter Concealment Mode** This field controls how AVC decoder select reference picture for Temporal Inter Concealment in B Slice

Value	Name	Description
000b		Top of Reference List L0/L1 (Use top entry of Reference List L0/L1)
001b		Driver Specified Concealment Reference
010b		Predicted Reference (Use reference picture predicted using B-Skip Algorithm)
011b		" Temporal Closest (Using POC to select the closest forward picture) [For L0: Closest POC smaller than current POC] [For L1: Closest POC larger than current POC]
100b		First Long Term Picture in Reference List L0/L1 (If no long term picture available, use Temporal Closest Picture)
101b-111b	Reserved	

	3:2	**Reserved** Format: MBZ
	1	**Intra 8x8/4x4 Prediction Error Concealment Control Bit** This field controls if AVC goes into MB concealment mode (next MB) when an error is detected on Intra8x8/4x4 Prediction Mode (these 2 modes have fixed coding so it may not affect the bitstream.

Value	Name	Description
0		AVC decoder will NOT go into MB concealment when Intra8x8/4x4 Prediction mode is incorrect.
1		AVC decoder will go into MB concealment when Intra8x8/4x4 Prediction mode is incorrect.

	0	**Intra Prediction Error Control Bit (applied to Intra16x16/Intra8x8/Intra4x4 Luma and Chroma)** This field controls if AVC decoder will fix Intra Prediction Mode if the decoded value is incorrect according to MB position.

Value	Name	Description
0		AVC decoder will detect and fix Intra Prediction Mode Errors.
1		AVC decoder will retain the Intra Prediction value decoded from bitstream.

Inline Data Description in MPEG2-IT Mode

<table>
<tr><th colspan="3">Inline Data Description in MPEG2-IT Mode</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 192
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">The content in this command is similar to that in the MEDIA_OBJECT command in IS mode described in the Media Chapter.
Each MFD_IT_OBJECT command corresponds to the processing of one macroblock. Macroblock parameters are passed in as inline data and the non-zero DCT coefficient data for the macroblock is passed in as indirect data. Inline data starts at dword 7 of MFD_IT_OBJECT command. There are 7 dwords total.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="5">0</td><td>31:28</td><td>Motion Vertical Field Select
A bit-wise representation of a long [2][2] array as defined in #167;6.3.17.2 of the ISO/IEC 13818-2 (see also #167;7.6.4).
<table>
<tr><th>Bit</th><th>MVector[r]</th><th>MVector[s]</th><th>MotionVerticalFieldSelect Index</th></tr>
<tr><td>28</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>29</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>30</td><td>1</td><td>0</td><td>2</td></tr>
<tr><td>31</td><td>1</td><td>1</td><td>3</td></tr>
</table>
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>Top Field</td><td>The prediction is taken from the top reference field.</td></tr>
<tr><td>1</td><td>Bottom Field</td><td>The prediction is taken from the bottom reference field.</td></tr>
</table></td></tr>
<tr><td>27</td><td>Reserved
was Second Field</td></tr>
<tr><td>26</td><td>Reserved
HWMC mode</td></tr>
<tr><td>25:24</td><td>Motion Type
When combined with the destination picture type (field or frame) this Motion Type field indicates the type of motion to be applied to the macroblock. See ISO/IEC 13818-2 #167;6.3.17.1, Tables 6-17, 6-18. In particular, the device supports dual-prime motion prediction (11) in both frame and field picture type.
<table>
<tr><th>Value</th><th>Destination = Frame
Picture_Structure = 11</th><th>Destination = Field
Picture_Structure != 11</th></tr>
<tr><td>'00'</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>'01'</td><td>Field</td><td>Field</td></tr>
<tr><td>'10'</td><td>Frame</td><td>16x8</td></tr>
<tr><td>'11'</td><td>Dual-Prime</td><td>Dual-Prime</td></tr>
</table></td></tr>
<tr><td>23:22</td><td>Reserved</td></tr>
</table>

Inline Data Description in MPEG2-IT Mode

		Scan method
	21	**DCT Type** This field specifies the DCT type of the current macroblock. The kernel should ignore this field when processing Cb/Cr data. See ISO/IEC 13818-2 #167;6.3.17.1. This field is zero if Coded Block Pattern is also zero (no coded blocks present). **Value** | **Name** | **Description** 0 | MC_FRAME_DCT | Macroblock is frame DCT coded 1 | MC_FIELD_DCT | Macroblock is field DCT coded
	20	**Reserved** Was Overlap Transform - H261 Loop Filter
	19	**Reserved**
	18	**Macroblock Motion Backward** This field specifies if the backward motion vector is active. See ISO/IEC 13818-2 Tables B-2 through B-4. **Value** | **Name** 0 | No backward motion vector 1 | Use backward motion vector(s)
	17	**Macroblock Motion Forward** This field specifies if the forward motion vector is active. See ISO/IEC 13818-2 Tables B-2 through B-4. **Value** | **Name** 0 | No forward motion vector 1 | Use forward motion vector(s)
	16	**Macroblock Intra Type** This field specifies if the current macroblock is intra-coded. When set, Coded Block Pattern is ignored and no prediction is performed (i.e., no motion vectors are used). See ISO/IEC 13818-2 Tables B-2 through B-4. **Value** | **Name** 0 | Non-intra macroblock 1 | Intra macroblock
	15:12	**Reserved** Format: | MBZ
	11:6	**Coded Block Pattern** Bit 11: Y0 Bit 10: Y1 Bit 9: Y2 Bit 8: Y3 Bit 7: Cb4 Bit 6: Cr5
	5:4	**Reserved**

Inline Data Description in MPEG2-IT Mode		
		Quantization Scale Code
	3	**LastMBInRow** This field indicates the last MB in each row
	2:0	**Reserved** Format: MBZ
1	31:16	**Reserved** Format: MBZ
	15:8	**VertOrigin** Vertical Origin In unit of macroblocks relative to the current picture (frame or field).
	7:0	**HorzOrigin** Horizontal Origin in unit of macroblocks.
2	31:16	**Motion Vectors - Field 0, Forward, Vertical Component** Each vector component is a 16-bit two's-complement value. The vector is relative to the current macroblock location. According to ISO/IEC 13818-2 Table 7-8, the valid range of each vector component is [-2048, +2047.5], implying a format of s11.1. However, it should be noted that motion vector values are sign extended to 16 bits.
	15:0	**Motion Vectors - Field 0, Forward, Horizontal Component**
3	31:16	**Motion Vectors - Field 0, Backward, Vertical Component**
	15:0	**Motion Vectors - Field 0, Backward, Horizontal Component**
4	31:16	**Motion Vectors - Field 1, Forward, Vertical Component**
	15:0	**Motion Vectors - Field 1, Forward, Horizontal Component**
5	31:16	**Motion Vectors - Field 1, Backward, Vertical Component**
	15:0	**Motion Vectors - Field 1, Backward, Horizontal Component**

Inline Data Description - VP8 PAK OBJECT

<table>
<tr><th colspan="3">Inline Data Description - VP8 PAK OBJECT</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 384
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">This structure corresponds to Dw3..6 of MFX_VP8_PAK_OBJECT Command.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:23</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>22:20</td><td>MV Format(Motion Vector Size)
Exists If: //IntraMbFlag = 0
This field specifies the size and format of the output motion vectors.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>000b</td><td>Intra MB</td><td>No Motion vectors</td></tr>
<tr><td>100b</td><td>Inter Predict MB (Unpacked Motion Vector Mode)</td><td>Sixteen Motion Vectors Per MacroBlock</td></tr>
<tr><td>Others</td><td>Reserved</td><td></td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This field MBZ, when the IntraMbFlag = 1.</td></tr>
</table></td></tr>
<tr><td></td><td>19:18</td><td>SegmentID
Format: U2
Segment number 0-3</td></tr>
<tr><td></td><td>17</td><td>Enable Coeff Clamp
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td></td><td>Magnitude of coefficients of the current MB is clamped based on the clamping matrix after quantization</td></tr>
<tr><td>0</td><td></td><td>No Clamping</td></tr>
</table></td></tr>
<tr><td></td><td>16:14</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>13</td><td>Intra MB Flag
This field specifies whether the current macroblock is an Intra (I) Macroblock. For Key pictures (IsKyeFrameFlag DW2, bit[5] of MFX_VP8_PIC_STATE), this field must be set to 1.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0h</td><td>INTER (Inter MacroBlock)</td></tr>
<tr><td>1h</td><td>INTRA (Intra MacroBlock)</td></tr>
</table></td></tr>
</table>

Inline Data Description - VP8 PAK OBJECT

		Programming Notes For I-picture MB (Intra MB Flag =1), this field must be set to 1.
	12:11	**RefPicSelect** This field specifies which reference pic (among Last Frame, Golden Frame and Alt Frame) is selected for the current macroblock when Intra MB Flag = 0 .
	10:8	**MB Type 3-Bits - Inter/Intra MB** MB Type 3 Bits [10:8] specifies InterMB MV mode configurations: 16x16 or 2 16x8 or 4 8x8 or 16 4x4 when Intra MB Flag = 0 and bit [8] = IntraMB mode configurations: 4x4 or 16x16 when Intra MB Flag = 1

RefPicSelect values:

Value	Name
00b	Last Frame
01b	Golden Frame
10b	Alt Frame

MB Type 3-Bits values:

Value	Name	Description
000b	16x16	**Inter MB** Only DW 6 bits 3:0 are used to indicate MVMode, MVMode can't be split
001b	2 16x8 (mv_Top Bottom)	**Inter MB [10:8]** Split MV is inferred. DW5 bits[3:0] are used for MVMode for first 16x8 partition, DW6 bits[3:0] are used for MVMode for second 16x8 partition.
010b	2 8 x16 (mv_left_right)	**Inter MB [10:8]** Split MV is inferred. DW5 bits[3:0] are used for MVMode for first 8x16 partition, DW5 bits[11:8] are used for MVMode for second 8x16 partition.
011b	4 8x8 (mv_quarters)	**Inter MB [10:8]** Split MV is inferred. DW5 bits[3:0] are used for MVMode for first 8x8 partition. DW5 bits[11:8] are used for MvMode for second 8x8 partition. DW6 bits[3:0] are used for MVMode for third 8x8 partition. DW6 bits[11:8] are used for MVMode for fourth 8x8 partition.
100b	16 4x4 (mv_16)	**Inter MB [10:8]** Split MV is inferred. There are 16 partitions. Each Sub-block uses 4 bits in DW6 and DW7.
0b	16x16	**Intra MB [8]** Only DW5, bits[3:0] are used for Y mode. For B_PRED, "16 4x4" should be used which implies B_PRED mode.
1b	16 4x4	**Intra MB [8]** All bits in DW5 and DW6 are used to represent B_PRED modes (Bmodes) in each sub-blocks.

7:6 Reserved

Format:	MBZ

5:4 MB UV Mode

Value	Name
0	DC_PRED
1	V_PRED
2	H_PRED

Inline Data Description - VP8 PAK OBJECT

		3 | TM_PRED
	3	**Reserved** Format: | MBZ
	2	**Skip MB Flag** This field is equivalent to mb_skip_flag in VP8 spec. **Programming Notes** By setting this field to 1, it forces an Inter MacroBlock to be encoded as a skipped MacroBlock
	1:0	**Reserved** Format: | MBZ
1	31:24	**Reserved** Format: | MBZ
	23:16	**MbYCnt (Vertical Origin)** Format: | U8 Unit of MacroBlock This field specifies the vertical origin of current macroblock in the destination picture in units of macroblocks.
	15:8	**Reserved** Format: | MBZ
	7:0	**MbXCnt (Horizontal Origin)** Format: | U8 Unit of MacroBlock This field specifies the horizontal origin of current macroblock in the destination picture in units of macroblocks.
2	31:28	**B Mode for SubBlock7 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	27:24	**B Mode for SubBlock6 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	23:20	**B Mode for SubBlock5 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	19:16	**B Mode for SubBlock4 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	15:12	**B Mode for SubBlock3 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	11:8	**B Mode for SubBlock2 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	7:4	**B Mode for SubBlock1 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	3:0	**B Mode for SubBlock0 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
3	31:28	**B Mode for SubBlock15 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.

Inline Data Description - VP8 PAK OBJECT

	27:24	**B Mode for SubBlock14(Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	23:20	**B Mode for SubBlock13(Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	19:16	**B Mode for SubBlock12(Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	15:12	**B Mode for SubBlock11(Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	11:8	**B Mode for SubBlock10 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	7:4	**B Mode for SubBlock9 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
	3:0	**B Mode for SubBlock8 (Y mode for the macroblock in non-B mode)** For Y-Mode and B-Mode Assignments refer to the assignment lists below this table.
4	31:30	**Reserved** Format: MBZ
	29:16	**MV Y FWD 0** Format: S13 The value of the y component of this motion vector for FWD block 0. Max value +/-1024 full pel (+/- 8192 1/8th pel) precision
	15:14	**Reserved** Format: MBZ
	13:0	**MV X FWD 0** Format: S13 The value of the x component of this motion vector for FWD block 0. Max value +/-1024 full pel (+/- 8192 1/8th pel) precision
5	31:30	**Reserved**
	29:16	**MV Y FWD 1** Format: S13 The value of the y component of this motion vector for FWD block 1.
	15:14	**Reserved**
	13:0	**MV X FWD 1** Format: S13 The value of the x component of this motion vector for FWD block 1.
6	31:30	**Reserved**
	29:16	**MV Y FWD 2** Format: S13 The value of the y component of this motion vector for FWD block 2.

Inline Data Description - VP8 PAK OBJECT		
	15:14	**Reserved**
	13:0	**MV X FWD 2** Format: S13 The value of the x component of this motion vector for FWD block 2.
7	31:30	**Reserved**
	29:16	**MV Y FWD 3** Format: S13 The value of the y component of this motion vector for FWD block 3.
	15:14	**Reserved**
	13:0	**MV X FWD 3** Format: S13 The value of the x component of this motion vector for FWD block 3.
8	31:30	**Reserved**
	29:16	**MV Y BWD 0** Format: S13 The value of the y component of this motion vector for BWD block 0.
	15:14	**Reserved**
	13:0	**MV X BWD 0** Format: S13 The value of the x component of this motion vector for BWD block 0.
9	31:30	**Reserved**
	29:16	**MV Y BWD 1**

DWord	Bit	Description	
		Format:	S13
		The value of the y component of this motion vector for BWD block 1.	
	15:14	**Reserved**	
	13:0	**MV X BWD 1**	
		Format:	S13
		The value of the x component of this motion vector for BWD block 1.	
10	31:30	**Reserved**	
	29:16	**MV Y BWD 2**	
		Format:	S13
		The value of the y component of this motion vector for BWD block 2.	
	15:14	**Reserved**	
	13:0	**MV X BWD 2**	
		Format:	S13
		The value of the x component of this motion vector for BWD block 2.	
11	31:30	**Reserved**	
	29:16	**MV Y BWD 3**	
		Format:	S13
		The value of the y component of this motion vector for BWD block 3.	
	15:14	**Reserved**	
	13:0	**MV X BWD 3**	
		Format:	S13
		The value of the x component of this motion vector for BWD block 3.	

INTERFACE_DESCRIPTOR_DATA

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	31:16	**Reserved** Format: MBZ
	15:0	**Kernel Start Pointer High** Format: InstructionBaseOffset[47:32]Kernel This field specifies the high 16 bits of starting address of the Kernel Pointer.
2	31:21	**Reserved** Format: MBZ
	20	**Thread Preemption disable** This field specifies whether, when dispatched, the thread is allowed to stop in middle on receiving mid-thread pre-emption request. Value | Name | Description 0h | Disable **[Default]** | Thread is pre-empted on receiving pre-emption indication. 1h | Enable | Thread is preempted only in case of page-fault.
	19	**Denorm Mode** This field specifies how Float denormalized numbers are handles in the dispatched thread. Value | Name | Description 0h | Ftz | Float denorms will be flushed to zero when appearing as inputs; denorms will never come out of instructions. Double precision float and half precision float numbers are not flushed to zero. 1h | SetByKernel | Denorms will be handled in by kernel.
	18	**Single Program Flow**

<table>
<tr><th colspan="3">INTERFACE_DESCRIPTOR_DATA</th></tr>
<tr><td></td><td></td><td>Specifies whether the kernel program has a single program flow (SIMDnxm with m = 1) or multiple program flows (SIMDnxm with m > 1).
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0h</td><td>Multiple</td></tr>
<tr><td>1h</td><td>Single</td></tr>
</table></td></tr>
<tr><td></td><td>17</td><td>Thread Priority
Specifies the priority of the thread for dispatch.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0h</td><td>Normal Priority</td></tr>
<tr><td>1h</td><td>High Priority</td></tr>
</table></td></tr>
<tr><td></td><td>16</td><td>Floating Point Mode
Specifies the floating point mode used by the dispatched thread.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0h</td><td>IEEE-754</td></tr>
<tr><td>1h</td><td>Alternate</td></tr>
</table></td></tr>
<tr><td></td><td>15:14</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>13</td><td>Illegal Opcode Exception Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.</td></tr>
<tr><td></td><td>12</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>11</td><td>Mask Stack Exception Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
This bit gets loaded into EU CR0.1[11]. See Exceptions and ISA Execution Environment.</td></tr>
<tr><td></td><td>10:8</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>7</td><td>Software Exception Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
This bit gets loaded into EU CR0.1[13] (note the bit # difference). See Exceptions and ISA Execution Environment.</td></tr>
<tr><td></td><td>6:0</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>3</td><td>31:5</td><td>Sampler State Pointer</td></tr>
</table>

<table>
<tr><th colspan="3">INTERFACE_DESCRIPTOR_DATA</th></tr>
<tr><td></td><td></td><td>
<table><tr><td>Format:</td><td>DynamicStateOffset[31:5]SAMPLER_STATE</td></tr></table>
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.
</td></tr>
<tr><td></td><td>4:2</td><td>
Sampler Count
<table><tr><td>Format:</td><td>U3</td></tr></table>
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.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>[0,4]</td><td></td></tr>
<tr><td>0h</td><td>No samplers used</td></tr>
<tr><td>1h</td><td>Between 1 and 4 samplers used</td></tr>
<tr><td>2h</td><td>Between 5 and 8 samplers used</td></tr>
<tr><td>3h</td><td>Between 9 and 12 samplers used</td></tr>
<tr><td>4h</td><td>Between 13 and 16 samplers used</td></tr>
</table>
</td></tr>
<tr><td></td><td>1:0</td><td>
Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td>4</td><td>31:16</td><td>
Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td></td><td>15:5</td><td>
Binding Table Pointer
<table>
<tr><td>Format:</td><td>SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW Binding Table Alignment is alignment is clear.</td></tr>
<tr><td>Format:</td><td>SurfaceStateOffset[18:8]BINDING_TABLE_STATE*256 [] When HW Binding Table Alignment is alignment set to 512KB size</td></tr>
</table>
<table>
<tr><th>Description</th></tr>
<tr><td>Specifies a pointer offset into the binding table. This field is ignored for child threads.

When Binding Table Pool is Disabled, this pointer is relative to the Surface State Base Address.
When Binding Table Pool is Enabled, this pointer is relative to the Binding Table PoolBase Address.

</td></tr>
</table>
</td></tr>
<tr><td></td><td>4:0</td><td>
Binding Table Entry Count
<table><tr><td>Format:</td><td>U5</td></tr></table>
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.
</td></tr>
</table>

INTERFACE_DESCRIPTOR_DATA

Value	Name
[0,31]	

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.

5 — 31:16 — Constant/Indirect URB Entry Read Length

Format:	U16

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 + Cross-Thread Constant Data Read Length).

If **Cross-Thread Constant Data Read Length** for Indirect is greater than 0, then this field must also be greater than 0. The allowed combinations are:

Constant/Indirect URB Entry Read Length **Entry Read Length**	Cross-Thread Constant **Data Read Length**	Notes
=0	=0	No Payload
>0	=0	Per-thread payload only
>0	>0	Both kinds of payload
=0	>0	Only for CURBE payloads

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
[0,1983]		Indicating [0,1983] 256-bit register increments. ROB has 64KB of storage; 2048 entries. However, lowest 64 entries are reserved for VFE/TS to store interface descriptor data. Hence, (URB Entry Read Offset + Read Length) shall not exceed 1984.

6 — 31:24 — Reserved

INTERFACE_DESCRIPTOR_DATA

		Format: MBZ
	23:22	**Rounding Mode** Format: U2
	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. Uses a different encoding to allow encodings for the new 1k and 2k SLM sizes.
	15	**Reserved** Format: MBZ
	14:13	**Reserved** Format: MBZ
	12:10	**Reserved**

Rounding Mode values:

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

Shared Local Memory Size values:

Value	Name	Description
0	Encodes 0K	No SLM used
1	Encodes 1K	
2	Encodes 2K	
3	Encodes 4K	
4	Encodes 8K	
5	Encodes 16K	
6	Encodes 32K	
7	Encodes 64K	

INTERFACE_DESCRIPTOR_DATA		
		Format: MBZ
	9:0	**Number of Threads in GPGPU Thread Group** Format: U10 Specifies the number of threads that are in this thread group. **Value** | **Name** | **Description** [1,112] | | The minimum value is 1, while the maximum value is the number of threads in two subslices for local barriers. See vol1b Configurations for the number of threads per subslice for different products.
7	31:8	**Reserved** Format: MBZ
	7:0	**Cross-Thread Constant Data Read Length** Format: U8 Specifies the amount of constant data in CURBE in 8-DW register increments which will be sent to every thread in the thread group in addition to the per thread ids specified by **Constant URB Entry Read Length**. **Value** | **Name** [0,127] |

INTERRUPT

INTERRUPT	
Source:	BSpec
Access:	RO, R/W, R/WC, R/W
Size (in bits):	128
Default Value:	0x00000000, 0xFFFFFFFF, 0x00000000, 0x00000000

See the Interrupt Definition Tables to find the source event for each interrupt bit. There are multiple instances of this register format.

DWord	Bit	Description
0	31:0	**ISR** Access: RO These are the Interrupt Status Register Bits. This field contains the non-persistent values of the interrupt status bits. The IMR selects which of these interrupt conditions are reported in the persistent IIR Value / Name: 0b — Condition Doesn't exist 1b — Condition Exists **Restriction** Some inputs to this register are short pulses. Do not use this register to sample these conditions.
1	31:0	**IMR** Access: R/W These are the Interrupt Mask Register Bits. This field contains a bit mask which selects which interrupt bits from the ISR are reported in the IIR. Value / Name: FFFFFFFFh — All interrupts masked **[Default]** 0b — Not Masked 1b — Masked **Restriction** For GT interrupts DO NOT use this register to mask interrupt events. Instead program this IMR to all 0s and use the individual GT command streamer MASK bits in the GT register space. This prevents unneeded messaging to DE.
2	31:0	**IIR** Access: R/WC These are the Interrupt Identity Register Bits. This field holds the persistent values of the interrupt bits from the ISR which are unmasked by the IMR. The IER enables an interrupt to be generated when the corresponding bit in the IIR becomes set. A disabled interrupt will still appear in the IIR. Bits set in this register will remain set (persist) until the interrupt condition is cleared by writing a

<table>
<tr><th colspan="3">INTERRUPT</th></tr>
<tr><td></td><td></td><td>'1' to the appropriate bits.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0b</td><td>Condition Not Detected</td></tr>
<tr><td>1b</td><td>Condition Detected</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>For each bit, the IIR can store a second pending interrupt if two or more of the same interrupt conditions occur before the first condition is cleared. Upon clearing the first interrupt, the IIR bit will momentarily go low, then return high to indicate there is second interrupt pending.</td></tr>
</table>
</td></tr>
<tr><td>3</td><td>31:0</td><td>IER
<table>
<tr><td>Access:</td><td>R/W</td></tr>
</table>
These are the Interrupt Enable Register Bits. The field enables an interrupt to be generated when the corresponding bit in the IIR becomes set. A disabled interrupt will still appear in the IIR.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0b</td><td>Disabled</td></tr>
<tr><td>1b</td><td>Enabled</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>The master interrupt enable must be set to 1b for any of these enabled interrupts to propagate to PCI device 2 interrupt processing.</td></tr>
</table>
</td></tr>
</table>

Invalidate After Read Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_IAR - Invalidate After Read Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>Previously, this Enable field was intended to optimize scratch and spill/fill read messages, where the memory was only used by a single thread and did not need to be maintained after the thread completed. If enabled, it caused all lines in the L3 cache accessed by the message to be invalidated after the read occurred, regardless of whether the line contained modified data. It was intended as a performance hint indicating that the data would no longer be used to avoid writing back data to memory.</td></tr>
</table>

JPEG

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.

LOD Message Address Payload Control

MACD_LOD - LOD Message Address Payload Control

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

<table>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:4</td><td>Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
Ignored</td></tr>
<tr><td>3:0</td><td>LOD
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U4</td></tr></table>
Specifies the LOD for this slot.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>[0,14]</td><td></td><td>representing LOD</td></tr></table>
</td></tr>
</table>

Lower Oword Block Data Payload

MDP_OW1L - Lower Oword Block Data Payload		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0-0.3	127:0	**Oword** Format: U128 Specifies the upper Oword data element
0.4-0.7	127:0	**Reserved** Format: MBZ Ignored

LRI Data Entry

LRI_DATA - LRI Data Entry

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

Each LRI command header is followed by LRI_DATA entries. Each of these entries is a pair of Dwords: the MMIO register address and the data to be written.

DWord	Bit	Description
0..1	63:55	**Reserved** Format: MBZ
	54:32	**MMIO** Format: U23 **Programming Notes** Bits [1:0] MBZ
	31:0	**Data** Format: U32

Manageability Engine Interrupt Vector

CSME_INTR_VEC - Manageability Engine Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	15:2	**Reserved**
	1	**CSME Response** Format: U1 CSME sets this bit in the interrupt when responding to initiated transaction for: • Response to wake up request • Payload message sent toto ME_MESG, ME_DATA for a request
	0	**CSME Request** Format: U1 CSME sets this bit in the interrupt when CSME initiates the transaction for: • CSME to wake up request • Payload message sent toto ME_MESG, ME_DATA for CMSE initiated request

MBHRD State Parameters1

MBHRD State Parameters1		
Source:	BSpec	
Size (in bits):	320	
Default Value:	0x00000802, 0x08041400, 0x03060300, 0x000F0000, 0x00000000, 0x00000002, 0x0D131100, 0x0006E4B5, 0x00000077, 0x00000000	

Please note that DW0-9, correspond to DW100 - 109 of **WiGig Parameters**.

DWord	Bit	Description
0	31:14	**Reserved** Format: MBZ
	13:8	**Max Value of Slice QP Increase For MB HRD** This is the max value of QP increase from one slice to the next. Value / Name: 0-40 / 8 / **[Default]**
	7:6	**Reserved** Format: MBZ
	5:0	**Max Value of Slice QP Decrease For MB HRD** This is the max value of QP decrease from one slice to the next. Value / Name: 0-10 / 2 / **[Default]**
1	31:29	**Reserved** Format: MBZ
	28:24	**MinDelay 1** Minimum delay 1 relative to initial delay. This field is used for MB-HRD computation in WiGig Mode. This field sets the minimum allowed delay as (initial delay * MinDelay1) » 4. Value / Name: 0-31 / 8 / **[Default]**
	23:21	**Reserved** Format: MBZ
	20:16	**MinDelay 2** Exists If: //[Mode] == 'WiGig'

MBHRD State Parameters1

DWord	Bit	Description
		Minimum delay 2 relative to initial delay. This field is used for MB-HRD computation in WiGig Mode. This field sets the minimum allowed delay as (initial delay * MinDelay2) » 4. MinDelay1 >= Mindealy2 **Value** | **Name** 0-31 | 4 | **[Default]**
	15:13	**Reserved** Format: | MBZ
	12:8	**MaxDelay** Maximum delay relative to initial delay. This field is used for MB-HRD computation in WiGig Mode. This field sets the maximum allowed delay as (initial delay * MaxDelay) » 4. **Value** | **Name** 0-31 | 20 | **[Default]**
	7:0	**Reserved** Format: | MBZ
2	31:30	**Reserved** Format: | MBZ
	29:24	**Delta Slice QP Increase 1** Delta slice QP increase when delay gets below MinDelay1. This field is used for MB-HRD computation in WiGig Mode. **Value** | **Name** 0-40 | 3 | **[Default]**
	23:22	**Reserved** Format: | MBZ
	21:16	**Delta Slice QP Increase 2** Delta slice QP increase when delay gets below MinDelay2. This field is used for MB-HRD computation in WiGig Mode. Delta Slice QP Increase 2 is >= Delta Slice QP Increase 1. **Value** | **Name** 0-40 | 6 | **[Default]**
	15:14	**Reserved**
	13:8	**Delta Slice QP Decrease**

MBHRD State Parameters1

DWord	Bit	Description
		Delta slice QP decrease when delay gets above MaxDelay. This field is used for MB-HRD computation in WiGig Mode. **Value** | **Name** 0-40 | 3 | **[Default]**
	7:0	**Reserved** Format: | MBZ
3	31:21	**Reserved** Format: | MBZ
	20:16	**Max Initial Delay** This field is used for MB-HRD computation in WiGig Mode. Max allowed initial delay relative to B/2R where B is the cpb size capacity used by the encoder and R is the peak transmission rate. This sets the maximum allowed initial delay as ((Max Initial Delay +1)*(B/2R)) »4. **Value** | **Name** 0-31 | 15 | **[Default]**
	15:0	**Reserved** Format: | MBZ
4	31:0	**Reserved** Format: | MBZ
5	31:8	**Reserved** Format: | MBZ
	7:0	**Guard Band Clocks for MB HRD Panic** Number of MB time delay as a guard band before triggering panic mode (max 256 MBs per slice). **Value** | **Name** 2-255 | 2 | **[Default]** **Programming Notes** This can be computed as follows: Ave_Bits_per_MB = FrameTSizeByte*8 / (FrameMBWidth*FrameMBHeight) IPCM_bits = chroma_format_idc==3 ? 6400 : 3200 (6400 for 444; 3200 for 420) Guardband = Max(0, ceil((FrameMBWidth*64-3/2*IPCM_bits)/Ave_Bits_per_MB))
6	31:29	**Reserved** Format: | MBZ

![intel]

<table>
<tr><th colspan="3">MBHRD State Parameters1</th></tr>
<tr><td></td><td>28:24</td><td>RUT_Precision
The fractional precision of fRemovalUnitTime, maximum value is the length of max number of MBs per frame (36864 for 4096x2304), which is 16.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0-16</td><td></td></tr><tr><td>13</td><td>[Default]</td></tr></table></td></tr>
<tr><td></td><td>23:21</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>20:16</td><td>T Unit Over R_Precision
Fractional precision of FrameTime Over R, the maximum value is the length of the max slice size (256*16*16*3*8), which is 21.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0-21</td><td></td></tr><tr><td>19</td><td>[Default]</td></tr></table></td></tr>
<tr><td></td><td>15:13</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>12:8</td><td>Delta QP Fractional_Precision
The fractional precision of 1/(fRemovalUnitTime*K) = the length of uiDeltaBetweenTxRv. The max value is 25 if assuming one frame time as the largest arrival/removal delay.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0-25</td><td></td></tr><tr><td>17</td><td>[Default]</td></tr></table></td></tr>
<tr><td></td><td>7:0</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>7</td><td>31:25</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>24:0</td><td>f Removal Unit Time
The removal interval of each MB, the fractional precision depends on the number of MBs of each frame.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0-27MB</td><td></td><td>1~27million clocks, each tick represents a period of 27Mhz clock.</td></tr><tr><td>451765</td><td>[Default]</td><td></td></tr></table></td></tr>
<tr><td>8</td><td>31:25</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>24:0</td><td>Delta QP Fractional
The removal interval of each MB, the fractional precision depends on the number of MBs of each frame.
<table><tr><th>Value</th><th>Name</th></tr></table></td></tr>
</table>

MBHRD State Parameters1		
		0-1FFFFFFh
		119 **[Default]**
9	31:0	**Reserved** Format: MBZ

MBHRD State Parameters2

MBHRD State Parameters2	
Source:	BSpec
Exists If:	//WGBOX Mode
Size (in bits):	320
Default Value:	0x00000000, 0x00000000, 0x000172CA, 0x000019D9, 0x000019DA, 0x000107AC, 0x00001262, 0x00001262, 0x0006DDD0, 0x0000330A

Please note that DW0-9, correspond to DW110 - 119 of **WiGig Parameters** and is used for MB-HRD computation in WGBOX Mode Only.

DWord	Bit	Description
0	31:0	**Reserved** Format: MBZ
1	31:0	**Reserved** Format: MBZ
2	31:25	**Reserved** Format: MBZ
	24:0	**Delay Between Transmitter and Receiver (Non Scaling)** The delay between transmitter and receiver (B/R_NoScale). Value / Name: 0-27MB 94922 — **[Default]** Programming Notes: Please note that this field is not used in WiDi mode and MBZ.
3	31:25	**Reserved** Format: MBZ
	24:0	**Delay Between Transmitter and Receiver IPCM** The delay between transmitter and receiver (B/R_IPCM). Value / Name: 0-27MB 6617 — **[Default]** Programming Notes: Please note that this field is not used in WiDi mode and MBZ.
4	31:25	**Reserved** Format: MBZ
	24:0	**Delay Between Transmitter and Receiver CPB Cap**

MBHRD State Parameters2		
		The delay between transmitter and receiver (B/R_CPB). **Value** / **Name** 0-27MB / 6618 / **[Default]** **Programming Notes** Please note that this field is not used in WiDi mode and MBZ.
5	31:26	**Reserved** Format: MBZ
	25:0	**27MHz/R_NoScale-Fractional Precision** 27MHz/R_NoScale, fractional precision defined by TUnitOverR_precision. **Value** / **Name** 0-54MB / 67500 / **[Default]** **Programming Notes** Please note that this field is not used in WiDi mode and MBZ.
6	31:26	**Reserved** Format: MBZ
	25:0	**27MHz/R_IPCM, Fractional Precision** 27MHz/R_CPB_Cap, fractional precision defined by TUnitOverR_precision. **Value** / **Name** 0-54MB / 4706 / **[Default]** **Programming Notes** Please note that this field is not used in WiDi mode and MBZ.
7	31:26	**Reserved** Format: MBZ
	25:0	**27MHz/R_CPB_Cap, Fractional Precision** 27MHz/R_IPCM, fractional precision defined by TUnitOverR_precision. Integer part is at most 27 assuming minimum rate of 1Mbps. **Value** / **Name** 0-54MB / 4706 / **[Default]** **Programming Notes** Please note that this field is not used in WiDi mode and MBZ.

<table>
<tr><th colspan="3">MBHRD State Parameters2</th></tr>
<tr><td>8</td><td>31:25</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>24:0</td><td>One Frame Time in 27MHz Clocks
The number of clocks for one frame, which is 27MHz/framerate.<table><tr><th>Value</th><th>Name</th></tr><tr><td>0-27MB</td><td></td></tr><tr><td>450000</td><td>[Default]</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>Please note that this field is not used in WiDi mode and MBZ.</td></tr></table></td></tr>
<tr><td>9</td><td>31:14</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>13:8</td><td>MaxQP for MB HRD<table><tr><td>Default Value:</td><td>51</td></tr></table>This is the absolute maximum value/upper bound of QP allowed by MB-HRD RC.<table><tr><th>Programming Notes</th></tr><tr><td>It is recommended that MaxQP be set to a larger value (40 or above) to give flexibility to the MB-HRD algorithm to avoid the underflow condition.</td></tr></table></td></tr>
<tr><td></td><td>7:6</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>5:0</td><td>MinQP for MB HRD
This is the absolute minimum value/lower bound of QP allowed by MB-HRD RC.<table><tr><th>Value</th><th>Name</th></tr><tr><td>10-26</td><td></td></tr><tr><td>10</td><td>[Default]</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td><u>To ensure that the MB size doesn't exceed 3200 bits</u>, Cr/Cb QP cannot go below 10. The Value of MinQp for Luma and Chroma Offset is programmed in such a way to ensure this.
E.g. If chroma offset = -5, MinQp should be >= 15. This would ensure that the Final Chroma QP >= (-5+15 = 10).</td></tr></table></td></tr>
</table>

MBHRD State Parameters3

MBHRD State Parameters3	
Source:	BSpec
Exists If:	//WGBOX Mode
Size (in bits):	64
Default Value:	0x0000031E, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

Please note that DW0-7, correspond to DW120 - 127 of **WiDi Parameters** and is used for MB-HRD computation in WGBOX Mode Only.

Programming Notes
This field is MBZ for WiDi Mode.

DWord	Bit	Description
0	31:11	**Reserved** Format: MBZ
	10:8	**DeltaQPWhenFracDecr** Format: U3 QP increase when the fractional value of the target frame bytes increases by 1/8. Value: 3 — Name: **[Default]** Value: 0-7
	7:6	**Reserved** Format: MBZ
	5:0	**DeltaQPWhenUnderFlow** Format: U6 QP increase when AvoidUnderflow condition is true. Value: 30 — Name: **[Default]** Value: 5-40
1..7	31:0	**Reserved** Format: MBZ

MEDIA_SURFACE_STATE

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	31:30	**Rotation**

Value	Name
00b	No Rotation or 0 Degree
01b	90 Degree Rotation
10b	180 Degree Rotation
11b	270 Degree Rotation

Programming Notes
Rotation is only supported only with AVS function messages and not with HDC direct write and 16x8 AVS messages.

DWord	Bit	Description
	29:27	**Reserved**

Format:	MBZ

DWord	Bit	Description
	26:20	**X Offset**

Exists If:	//[Surface Format] is one of Planar Formats
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 4-high pixel) resolution.

Value	Name	Description
[0,127]		In multiples of 4 (low 2 bits missing)

Programming Notes
For linear surfaces and Packed Formats, this field must be zero.

MEDIA_SURFACE_STATE

DWord	Bit	Description
		For **Surface Format** with 8 bits per element, this field must be a multiple of 16. For **Surface Format** with 16 bits per element, this field must be a multiple of 8.
	26:16	**Reserved** Exists If: //[Surface Format] is not one of Planar Formats Format: MBZ
	19:16	**Y Offset** Exists If: //[Surface Format] is one of Planar Formats Format: RowOffset[5:2] 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)
	15:12	**Reserved** Format: MBZ
	11:0	**Reserved** Format: MBZ
1	31:18	**Height** Format: U14-1 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.
	17:4	**Width**

Y Offset values:

Value	Name	Description
[0,15]		In multiples of 4 (low two bits missing)

Programming Notes
For linear surfaces and Packed Formats, this field must be zero.

Height values:

Value	Name	Description	Exists If
[0,16383]		representing heights [1,16384]	[Surface Type] != FM_STRBUF_*
[0, 16383]			[SurfaceType] == FM_STRBUF_*

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.
When the format is structure buffer, this field is valid for reading the Data base Structure buffer (or) Test Vector Structure Buffer (or) Index Table. The Number of entries * Pitch should be less than 2^40.

MEDIA_SURFACE_STATE

Format:	U14-1

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	Exists If
[0,16383]		representing widths [1,16384]	[Surface Type] != FM_STRBUF_*
[0,16383]		Contains bits [13:0] of the number of entries in the buffer - 1	[SurfaceType] == FM_STRBUF_*

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.
• For Y8_UNORM_VA format width should be in multiple of 4, for Y16_UNORM_VA format width should be in multiple of 2, for Y1_UNORM format width should be in multiple of 32 • When Address Control = Mirror, the total width should be in multiple of 4bytes.
Width (field value + 1) must be a multiple of 2 for PLANAR_420_16
For Y16_UNORM format width should be in multiple of 2
When the format is structure buffer, this field is valid for reading the Data base Structure buffer (or) Test Vector Structure Buffer (or) Index Table. The Number of entries * Pitch should be less than 2^40.

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**

Default Value:	0
Format:	U0.2

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

MEDIA_SURFACE_STATE

		direction **Programming Notes** This field is ignored for all formats except for PLANAR_420_8 and PLANAR_420_16 This offset has been increased from 2 bits to 3 bits to support U1.2 format, and the MSB bit is added as Pixel Offset V Direction MSB in DWord 2. Valid values for the combined field range from 0 to 4.
2	31:27	**Surface Format**

Description
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.
Note: Y8_UNORM_VA, Y16_UNORM and Y16_SNORM are used for all functions of sample_8x8 except AVS where rest of the formats are not used. These two formats are packed as 32bits in L1 though the individual pixels are either 8bpp or 16bpp respectively.

Value	Name	Description
0	YCRCB_NORMAL	
1	YCRCB_SWAPUVY	
2	YCRCB_SWAPUV	
3	YCRCB_SWAPY	
4	PLANAR_420_8	
5	Y8_UNORM_VA	Sample_8x8 only except AVS
6	Y16_SNORM	Sample_8x8 only except AVS
7	Y16_UNORM_VA	Sample_8x8 only except AVS
8	R10G10B10A2_UNORM	Sample_8x8 only
9	R8G8B8A8_UNORM	Sample_8x8 AVS only
10	R8B8_UNORM (CrCb)	Sample_8x8 AVS only
11	R8_UNORM (Cr/Cb)	Sample_8x8 AVS only
12	Y8_UNORM	Sample_8x8 AVS only
13	A8Y8U8V8_UNORM	Sample_8x8 AVS only
14	B8G8R8A8_UNORM	Sample_8x8 AVS only
15	R16G16B16A16	Sample_8x8 AVS only
16	Y1_UNORM	Sample_8x8 only for boolean surfaces (1bit/pixel)
17	Y32_UNORM	For Integral Image (32bpp)
18	PLANAR_422_8	Sample_8x8 AVS only
19	FM_STRBUF_Y1	Structure Buffer 1bit/element Sample_8x8 only feature matching

MEDIA_SURFACE_STATE

20	FM_STRBUF_Y8	Structure Buffer 8bit/element Sample_8x8 only feature matching
21	FM_STRBUF_Y16	Structure Buffer 16bit/element Sample_8x8 only feature matching
22	FM_STRBUF_Y32	Used for Index Table only. 32bit per entry.
23	PLANAR_420_16	Sample_8x8 AVS only
24	R16B16_UNORM (CrCb)	Sample_8x8 AVS only
25	R16_UNORM (Cr/Cb)	Sample_8x8 AVS only
26	Y16_UNORM	Sample_8x8 AVS only
Others	Reserved	

Programming Notes

For FM_STRBUF_Y1 format, data must be programmed with byte alignment.

26 **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.

25 **Cr(V)/Cb(U) Pixel Offset U Direction**

Default Value:	0
Format:	U0.1

Description

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

Programming Notes

This field is must be zero for all formats except PLANAR_420_16, PLANAR_420_8, PLANAR_422_8, YCRCB_NORMAL, YCRCB_SWAPUVY, YCRCB_SWAPUV, YCRCB_SWAPY.

24 **Cr(V)/Cb(U) Pixel Offset V Direction MSB**

Default Value:	0
Format:	U1

Description

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

MEDIA_SURFACE_STATE

		Programming Notes This field is must be zero for all formats except?PLANAR_420_16 and PLANAR_420_8 This offset has been increased from 2 bits to 3 bits as U1.2 format and this bit is used in conjunction with the bits in the Cr(V)/Cb(U) Pixel Offset V Direction field in DWord 1, which contain the rest of the bits for offset V-direction. Valid values for the combined field range from 0 to 4.
	23	**Memory Compression Mode** Distinguishes Vertical from Horizontal compression. Value | Name | Description 0 | Horizontal Compression Mode **[Default]** | 1 | Vertical Compression Mode |
	22	**Memory Compression Enable** Format: Enable This surface may contain compressed or compressible pixels. Memory compression will be attempted for writes to this surface. Reads from this surface will check for compressed data. **Programming Notes** The compression control must have 0 value for non-tileY modes. Please refer to vol1a Memory Data Formats chapter -- section Media Memory Compression for more details, including format restrictions.
	21	**Address Control** Value | Name | Description 0 | CLAMP | Clamp 1 | MIRROR | Mirror
	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. If Media Memory Compression is enabled, the following max pitch size restriction must be

MEDIA_SURFACE_STATE

		honored. For larger resolution, Media Memory compression Must be disabled. Tiling Mode Pixel Format Max Frame Width (bytes) Max Frame Width (pixels) Max Pitch (bytes) Legacy 4K 8bpp 16k 16k 16k + 127 16bpp 16k 8k 16k + 127 32bpp 16k 4k 16k + 127 64bpp 16k 2k 16k + 127 128bpp 16k 1k 16k + 127 TileYF 8bpp 8k 8k 8k + 63 16bpp 16k 8k 16k + 127 32bpp 16k 4k 16k + 127 64bpp 16k 2k 16k + 255 128bpp 16k 1k 16k + 255 TileYS 8bpp 16k 16k 16k + 255 16bpp 16k 8k 16k + 511 32bpp 16k 4k 16k + 511 64bpp 16k 2k 16k + 1023 128bpp 16k 1k 16k + 1023
		For FM_STRBUF_Y* surface Formats, Max Pitch programmable is 2048 bytes Must be a power of 2.
		For FM_STRBUF_Y* surface Formats, Pitch must be a multiple of 64 bytes.
	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. **Programming Notes** Must be Zero as this field is not used.
	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.

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

MEDIA_SURFACE_STATE

DWord	Bit	Description
		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 Y-plane origin to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave Chroma is enabled. Resultant X-offset = 'X-offset of the surface (Y-plane)' + 'X offset for U(Cb)' For TileYS and TileYF this offset should be integral multiple of Tile width of Luma plane. **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 Y-plane origin to the start (origin) of the U(Cb) plane or the interleaved UV plane if Interleave Chroma is enabled. Resultant X-offset = 'Y-offset of the surface (Y-plane)' + 'Y offset for U(Cb)' For TileYS and TileYF this offset should be integral multiple of Tile width of Luma plane. **Programming Notes** This field must be aligned by 4 bit[1:0] = 00 This field must be aligned by 4 bit[1:0] = 00 for all format besides PLANAR_420_*
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** For Planar surfaces this field specifies the horizontal offset in pixels from the Y-plane origin to the start (origin) of the V(Cb) plane. Resultant X-offset = 'X-offset of the surface (Y-plane)' + 'X offset for V(Cb)' For TileYS and TileYF this offset should be integral multiple of Tile width of Luma plane.

MEDIA_SURFACE_STATE

DWord	Bit	Description
		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** For Planar surfaces this field specifies the vertical offset in rows from the Y-plane origin to the start (origin) of the V(Cb) plane. Resultant Y-offset = 'Y-offset of the surface (Y-plane)' + 'Y offset for V(Cb)' For TileYS and TileYF this offset should be integral multiple of Tile width of Luma plane. **Programming Notes** This field must indicate a multiple of 4 (bit 0 & 1 = 00).
5	31	**Vertical Line Stride** Format: U1 in lines to skip between logically adjacent lines For Surfaces accessed via the sample_8x8 message:Specifies number of lines (0 or 1) to skip between logically adjacent lines - provides support of interleaved (field) surfaces as textures.For Other Surfaces:Vertical Line Stride must be zero.
	30	**Vertical Line Stride Offset** Format: U1 in lines of initial offset (when Vertical Line Stride == 1) For Surfaces accessed via the sample_8x8 message: Specifies the offset of the initial line from the beginning of the buffer. For Other Surfaces: Vertical Line Stride Offset must be zero. **Programming Notes** This field must be set to 0 if Vertical Line Stride is 0.
	29:24	**Reserved** Format: MBZ
	23:20	**Depth** Format: U4 This field specifies the upper nibbles of the number of entries in the structure buffer. **Value** | **Name** | **Description** | **Exists If** [0-15] | | Contains bits [31:28] of the number of entries in the buffer - 1 | [SurfaceType] == FM_STRBUF_* **Programming Notes** This field is valid for reading the Data base Structure buffer (or) Test Vector Structure Buffer (or)

MEDIA_SURFACE_STATE

DWord	Bit	Description
		Index Table. The Number of entries * Pitch should be less than 2^40.
	19:18	**Tiled Resource Mode** Format: U2 **For Sampling Engine, Render Target, and Typed/Untyped Surfaces:** This field specifies the tiled resource mode. **For other surfaces:** This field is ignored.

Value	Name	Description
0h	TRMODE_NONE	No tiled resource
1h	TRMODE_TILEYF	4KB tiled resources
2h	TRMODE_TILEYS	64KB tiled resources
3h	Reserved	

Programming Notes
If **Tile Mode** is not set to TILEMODE_YMAJOR, this field must be set to TRMODE_NONE.
If this field is not set to TRMODE_NONE, the **Surface Format** must be one with 8, 16, 32, 64, or 128 bits per element, or one of the compressed texture modes (BC*, ETC*, EAC*, ASTC*). Additionally, YCRCB* formats are supported and treated as 16 bits per element, and the PLANAR_420_8 and PLANAR_422_8 formats are supported and treated as 8 bits per element on the Y plane and 16 bits per element on the UV plane (if **Interleave Chroma** is enabled) or 8 bits per element on the U and V planes (if **Interleave Chroma** is disabled.

DWord	Bit	Description
	17:7	**Reserved** Format: MBZ
	6:0	**Surface Memory Object Control State** Default Value: 0h DefaultVaueDesc Format: MEMORY_OBJECT_CONTROL_STATE This 7-bit field is used in various state commands and indirect state objects to define cacheability and other attributes related to memory objects.
6	31:0	**Surface Base Address** Format: GraphicsAddress[31:0] Specifies the low 32 bits of 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

MEDIA_SURFACE_STATE		
		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.Certain message types used to access surfaces have more stringent alignment requirements. Please refer to the specific message documentation for additional restrictions. In Feature matching, for indirect database fetch (index surface) the surface base address should be cacheline aligned
7	31:16	**Reserved** Format: MBZ
	15:0	**Surface Base Address High** Format: GraphicsAddress[47:32] Specifies the high 16 bits of the byte-aligned base address of the surface. Refer to Surface Base Address [31:0] for programming notes applying to this field.

MemoryAddressAttributes

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

This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface. It defines the attributes for VDBOX addresses.

DWord	Bit	Description
0	31:15	**Reserved**

Format:	MBZ

DWord	Bit	Description
	14:13	**Base Address - Tiled Resource Mode**

Format:	U2

For Media Surfaces: This field specifies the tiled resource mode.

Value	Name	Description
00b	TRMODE_NONE	TileY resources
01b	TRMODE_TILEYF	4KB tiled resources
10b	TRMODE_TILEYS	64KB tiled resources
11b	Reserved	

DWord	Bit	Description
	12	**Base Address - Row Store Scratch Buffer Cache Select**

Format:	U1

Description
This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.
When this is programmed to "1" (going to Media Cache), the corresponding base address will be programmed with the starting position in the media cache. The programming table is in "Buffer Size Requirement Page" in HEVC section

Value	Name	Description
0		Buffer going to LLC.
1		Buffer going to Internal Media Storage.

DWord	Bit	Description
	11	**Reserved**

Format:	MBZ

DWord	Bit	Description
	10	**Base Address - 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	Name

MemoryAddressAttributes

		0b — Horizontal Compression Mode **Programming Notes** Must be zero; vertical compression is not supported.
	9	**Base Address - Memory Compression Enable** Format: Enable Memory compression will be attempted for this surface.
	8:7	**Base Address - Arbitration Priority Control** Format: HEVC_ARBITRATION_PRIORITY
	6:1	**Base Address - 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**

Merged Media Block Message Header

MH_MBM - Merged Media Block Message Header

Source:	EuSubFunctionDataPort1
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:0	**X Offset** Format: S31 X offset (in bytes) of the upper left corner of the block into the surface.
1	31:0	**Y Offset** Format: S31 Y offset (in rows) of the upper left corner of the block into the surface.
2	31:0	**Merged Media Block Message Control** Format: MHC_MBM_CONTROL Specifies the Merged message subtype and additional input parameters.
3	31:0	**Mask** Format: U32 The Mask is ignored by the Merged Media Block message: all Dwords are always returned on reads, and always enabled to be written on writes.
4	31:0	**FFTID** Format: MHC_FFTID Fixed Function Thread ID
5..7	95:0	**Reserved** Format: MBZ Ignored

Merged Media Block Message Header Control

MHC_MBM_CONTROL - Merged Media Block Message Header Control

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

DWord	Bit	Description
0	31:30	**Message Mode** Specifies the Media Block Read message is Normal subtype. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>00h</td><td>Normal</td><td>The Block Height and Block Width fields are specified in this Dword. The Mask is ignofed by a media block read message.</td></tr><tr><td>Others</td><td>Reserved</td><td>Reserved.</td></tr></table>
	29	**Reserved** Format: MBZ Ignored
	28:24	**Sub-Register Offset** Format: U5 Provides the sub-register offset in unit of bytes of a Merged Media Block Read message. This field is ignored (reserved) for a media block write message. Range = [0, 28]. Only a multiple of BasePitch, including 0, is valid. **Programming Notes** Sub-Register Offset and Register Pitch Control allow software to assembly multiple media block reads directly into a shared GRF register set. For example, if both are set to zero, the read data are written to GRF registers, aligning to the least significant bits of the first register, and the register pitch is equal to the next power-of-2 that is greater than or equal to the Block Width. If Register Pitch Control is non-zero, multiple media block read messages sharing the same Register Pitch Control but with different Sub-Register Offset can fill in the same set of GRF registers with media block data line interleaved. **Restriction** For the Sampler Cache Data, this field must be zero. BasePitch is defined as the next the power-of-2 that is greater than or equal to the Block Width. Minimum BasePitch is 1 DWord. Sub-Register Offset must be aligned to BasePitch (therefore will be a multiple of DWords as

MHC_MBM_CONTROL - Merged Media Block Message Header Control

		well). When Register Pitch Control = 0, Sub-Register Offset must align to BasePitch*Block Height. ensuring the output fits in a single GRF register. In general (and specifically when Sub-Register Offset is greater than 0), when the resulting data will cross a GRF register boundary, the data must be placed symmetrically between GRF registers.
	23:22	**Reserved** Format: MBZ Ignored
	21:16	**Block Height** Format: U6 Height in rows of block being accessed. Range = [0,63] representing 1 to 64 rows **Restriction** If Block Width (bytes), then Maximum Block Height (rows) is constrained by (# Dwords width) * (# rows) <= 64 Dwords.
	15:10	**Reserved** Format: MBZ Ignored
	9:8	**Register Pitch Control** Format: U2 Controls the register pitch for a Merged Media Block Read message. This field is ignored (reserved) for a media block write message. Register Pitch Control is only allowed to be non-zero when Block Width is a multiple of DWords. Restriction : For the Sampler Cache Data, this field must be zero.

Value	Name	Description
0h	RPC_1 **[Default]**	1 Block
1h	RPC_2	2 Blocks
3h	RPC_4	4 Blocks

Restriction
BasePitch is defined as the next the power-of-2 that is greater than or equal to the Block Width. The effective register pitch (RPC*BasePitch)+SRO must be less than or equal to 32 bytes (to fit in a single GRF register).

	7:6	**Reserved** Format: MBZ Ignored
	5:0	**Block Width**

MHC_MBM_CONTROL - Merged Media Block Message Header Control

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U6</td></tr></table>Width in bytes of the block being accessed. Range = [0,31] representing 1 to 32 Bytes.

Message Descriptor - Render Target Write

Message Descriptor - Render Target Write

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

DWord	Bit	Description
0	31	**Reserved** Format: MBZ
	30	**Data Format** Format: U1 Value / Name / Description: 0 / Single Precision / 32b 1 / Half Precision / 16b **Programming Notes** This field is applicable for Render Target Write Messages ONLY.
	29:14	**Reserved** Format: MBZ
	13	**Per-Sample PS outputs enable** This bit must not be set when Render Target is not bound to pixel-shader OR when Render Target is not multisampled. This bit must be set when PS runs at sample-frequency i.e. pixel shader dispatch mode is PER_SAMPLE. By setting this bit, PS sends Render Target Write Message that outputs color, depth(optional) and stencil(optional) phases on per sample basis for each slot. When Render Target is multisampled and this bit is reset, Render Target outputs color, depth(optional) and stencil(optional) at pixel frequency. It should be noted that the latter case is applicable for only per-pixel PS invocation.
	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

Message Descriptor - Render Target Write

		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
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_TM	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_REPDATA message must not be used in SIMD8 pixel-shaders.

	7:0	**Reserved**

Format:	MBZ

Message Descriptor - Sampling Engine

Message Descriptor - Sampling Engine

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

DWord	Bit	Description
0	31	**EOT**
	30	**Return Format** Format: U1 Value / Name / Description: 0 / 32-bit / Return data is 32b 1 / 16-bit / Return data is 16b **Programming Notes** This field must be set to 32-bit for messages with SIMD Mode of SIMD4x2 or SIMD32/64. This field must be set to 32 for resinfo, LOD and sampleinfo messages.
	29	**SIMD Mode[2]** Format: U1 This field is the upper bit of the 3-bit SIMD Mode field.
	28:25	**Message Length** Format: U4 This field specifies the number of 256-bit GRF registers starting from (src) to be sent out on the request message payload. Value / Name: [1,15] / **Programming Notes** A value of 0 is considered erroneous.
	24:20	**Response Length** Format: U5 This field indicates the number of 256-bit registers expected in the message response. Value / Name: [0,16] / **Programming Notes** A value 0 indicates that the request message does not expect any response. The largest response supported is 16 GRF registers.

Message Descriptor - Sampling Engine

19 — **Header Present**

Format:	Enable

Description
Specifies whether the message includes a header phase. If the header is not present (this field is zero), all of the fields normally contained in the header are assumed to be 0.
If the header is not present, in some cases the **Write Channel Mask** fields are set according to the Response Length. For more details, please refer to the **Payload Parameter Definition** section, under *Shared Functions*..

18:17 — **SIMD Mode[1:0]**

Format:	U2

Description
Specifies the SIMD mode of the message being sent.
A third bit SIMD Mode[2] is added to this field (bit 29 of message descriptor). Encodings now range from 0-7. SIMD Mode[2:0] SIMD 000 Reserved 001 SIMD8 010 SIMD16 011 SIMD32/64 100 Reserved 101 SIMD8H 110 SIMD16H 111 Reserved

16:12 — **Message Type**

Format:	U5

Specifies the type of message being sent. For more details, please refer to **Message Format** section for the definition of these 5 bits..

11:8 — **Sampler Index**

Format:	U4

Specifies the index into the sampler state table. Ignored for ld, resinfo, sampleinfo, and cache_flush type messages.

Value	Name
[0,15]	

Programming Notes
• For the deinterlace message, this field must be a multiple of 2 (even). • For the sample_8x8 message, this field must be a multiple of 4.

7:0 — **Binding Table Index**

Format:	U8

Message Descriptor - Sampling Engine		
		Specifies the index into the **binding table**. Ignored for cache_flush type messages. Values of 255 and 253 indicate stateless. 254 indicates SLM. 252 indicates bindless. **Value** | **Name** [0,255] |

MFD_MPEG2_BSD_OBJECT Inline Data Description

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	31:24	**Slice Horizontal Position** Format: U8 in Macroblocks This field indicates the horizontal position of the first macroblock in the slice.
	23:16	**Slice Vertical Position** Format: U8 in Macroblocks This field indicates the vertical position of the first macroblock in the slice.
	15:8	**Macroblock Count** Format: U8 in Macroblocks This field indicates the number of macroblocks in the slice, including skipped macroblocks.
	7	**Slice Concealment Override Bit** This bit forces hardware to handle the current slice in Conceal or Deocde Mode. If this bit is set to one, VIN will force the current slice to do concealment or to decode from bitstream regardless if the slice boundary has errors or not. Value 1h: VIN will use driver-provided "Slice Concealment Type" regardless of valid slice boundary Value 0h: Driver must program "Slice Concealment Type" to '0'. VIN will set "Slice Concealment Type" depending if the slice boundary has error or not
	6	**Slice Concealment Type Bit** This bit can be forced by driver ("Slice Concealment Override Bit") or set by VINunit depending on slice boundary errors. Value 1h: VMD will conceal all MBs of the slice regardless of bitstream. (If driver does not force the value of this bit, VIN will set this bit depending on slice boundary error. If the next slice position of the current slice is out-of-bound or the same or earlier than the current slice start position, VIN will set this bit for the next slice) Value 0h: VMD will decode MBs from the bitstream until the bitstream is run-out. Then VMD will conceal the remaining MBs. **Programming Notes** VIN can turn this bit from 0 to 1 internally if "Slice Concealment Disable Bit" is "0" and VIN detects slice boundary errors.

<table>
<tr><th colspan="3">MFD_MPEG2_BSD_OBJECT Inline Data Description</th></tr>
<tr><td></td><td>5</td><td>Last Pic Slice
This bit is added to support error concealment at the end of a picture.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1h</td><td></td><td>The current Slice is the last Slice of the entire picture</td></tr>
<tr><td>0h</td><td></td><td>The current Slice is not the last Slice of current picture</td></tr>
</table></td></tr>
<tr><td></td><td>4</td><td>Reserved</td></tr>
<tr><td></td><td>3</td><td>Is Last MB
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1h</td><td></td><td>The current MB is the last MB in the current Slice</td></tr>
<tr><td>0h</td><td></td><td>The current MB is not the last MB in the current Slice</td></tr>
</table></td></tr>
<tr><td></td><td>2:0</td><td>First Macroblock Bit Offset
<table><tr><td>Format:</td><td>U3</td></tr></table>
This field provides the bit offset of the first macroblock in the first byte of the input bitstream.</td></tr>
<tr><td>1</td><td>31:29</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>28:24</td><td>Quantizer Scale Code
<table><tr><td>Format:</td><td>U5</td></tr></table>
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.</td></tr>
<tr><td></td><td>23:17</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>16:8</td><td>Next Slice Vertical Position
<table><tr><td>Format:</td><td>U9 in macroblocks</td></tr></table>
This field indicates the vertical position (in macroblock units) of the first macroblock in the next slice.
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This field is primarily used for error concealment. In the case that current slice is the last slice, this field should set to the height of the picture (field picture will be in height of field) (since y-direction is zero-based numbering).</td></tr>
</table></td></tr>
<tr><td></td><td>7:0</td><td>Next Slice Horizontal Position
<table><tr><td>Format:</td><td>U8 in macroblocks</td></tr></table>
This field indicates the horizontal position (in macroblock units) of the first macroblock in the next slice.
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This field is primarily used for error concealment. In the case that current slice is the last slice, this field should set 0.</td></tr>
</table></td></tr>
</table>

MFX_REFERENCE_PICTURE_BASE_ADDR

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

DWord	Bit	Description
0..1	63:48	**Reserved** Format: MBZ
	47:32	**Reference Picture Address [n] High** Format: GraphicsAddress[47:32] This field is for the upper range of Reference Picture Addresses
	31:6	**Reference Picture Address [n]** Format: GraphicsAddress[31:6] Specifies the 64 byte aligned reference frame buffer addresses for the motion compensation operation in AVC/ /MPEG2. AVC can specify up to 16 YUV frame-based surfaces for both forward and backward references, i.e. L0+L1 total = 16 max. Any entry can be assigned to L0 or L1 or both lists.But VC1 and MPEG2, worst case, can use up to 2 YUV frame-based surfaces for both forward and backward references: • P-MB : RefAddr[0] - temporal closest previous field of a reference frame (can be the current frame) • RefAddr[1]- next temporal closest previous field of a reference frame (must be different from the current frame) It is a variant (without the LongTermRefPic specification) of the RefFrameList[16] defined in AVC DXVA Spec. RefAddr[0-15] is indexed by frame_storeID »1. It is not a packed list, i.e. invalid entries can scatter among the list. All invalid addresses must be set to a valid address RefAddr[0] by the driver. The same applies to VC1 and MPEG2. **Programming Notes** AVC: Always specifies all 16 addresses even some of them are not needed as indicated by the max num of active reference pictures. This is done for preventing data corruption (error, fault condition, etc.) by having all the references being set to a legal location.
	5:0	**Reserved** Format: MBZ

Motion Decision Setting Parameters0

Motion Decision Setting Parameters0

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

Please note that DW0-1, correspond to DW64-69 of **WiGig Parameters**.

DWord	Bit	Description
0	31:30	**Reserved** Format: MBZ
	29	**Intra16x16 prediction Enable** Value 1: Intra16x16 enabled Value 0: Intra16x16 disabled **Programming Notes** **Restriction:** This Field is always enabled.
	28	**Intra8x8 prediction Enable** Value 1: Intra8x8 enabled Value 0: Intra8x8 disabled
	27	**Intra4x4 prediction Enable** Value 1: Intra4x4 enabled Value 0: Intra4x4 disabled
	26:21	**Reserved** Format: MBZ
	20	**Constrained Intra Prediction Flag** Exists If: //WiDi and WiGig Modes It is set to the value of the syntax element in the current active PPS. Value 0 — Intra and Inter Neighboring MB — Allows both intra and inter neighboring MB to be used in the intra-prediction encoding of the current MB. Value 1 — Intra Neighboring MB — Allows only to use neighboring Intra MBs in the intra-prediction encoding of the current MB. If the neighbor is an inter MB, it is considered as not available.
	19:0	**Reserved** Format: MBZ

Motion Decision Setting Parameters0		
1	31:24	**I-Intra 16x16** Intra 16x16 prediction mode bias for I-frame
	23:16	**I-Intra 8x8** Intra 8x8 prediction mode bias for I-frame
	15:8	**I-Intra 4x4** Intra 4x4 prediction mode bias for I-frame
	7:0	**Reserved** Format: MBZ
2	31:24	**P-Intra 16x16** Intra 16x16 prediction mode bias for P-frame
	23:16	**P-Intra 8x8** Intra 8x8 prediction mode bias for P-frame
	15:8	**P-Intra 4x4** Intra 4x4 prediction mode bias for P-frame
	7:0	**Reserved** Format: MBZ
3	31:0	**Reserved** Format: MBZ
4	31:0	**Reserved** Format: MBZ
5	31:24	**Block BasedSkip Threshold -QP50-51**
	23:16	**Block BasedSkip Threshold -QP48-49**
	15:8	**Block BasedSkip Threshold -QP46-47**
	7:0	**Block BasedSkip Threshold -QP44-45**

Motion Decision Setting Parameters1

Motion Decision Setting Parameters1		
Source:	BSpec	
Size (in bits):	320	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
Please note that DW0-9, correspond to DW70-79 of **WiGig Parameters**.		
DWord	**Bit**	**Description**
0	31:24	**Block BasedSkip Threshold -QP42-43**
	23:16	**Block BasedSkip Threshold -QP40-41**
	15:8	**Block BasedSkip Threshold -QP38-39**
	7:0	**Block BasedSkip Threshold -QP36-37**
1	31:24	**Block BasedSkip Threshold -QP34-35**
	23:16	**Block BasedSkip Threshold -QP32-33**
	15:8	**Block BasedSkip Threshold -QP30-31**
	7:0	**Block BasedSkip Threshold -QP28-29**
2	31:24	**Block BasedSkip Threshold -QP26-27**
	23:16	**Block BasedSkip Threshold -QP24-25**
	15:8	**Block BasedSkip Threshold -QP22-23**
	7:0	**Block BasedSkip Threshold -QP20-21**
3	31:24	**Block BasedSkip Threshold -QP18-19**
	23:16	**Block BasedSkip Threshold -QP16-17**
	15:8	**Block BasedSkip Threshold -QP14-15**
	7:0	**Block BasedSkip Threshold -QP12-13**
4	31:24	**Block BasedSkip Threshold -QP10-11** Format: U4.4 This field is used as Block BaseSkip threshold - for conditional replenishment, after checking the ZMV location, a MB will be coded as skip if all subblocks (4x4) distortions are less than or equal to this threshold.
	23:0	**Reserved** Format: MBZ
5	31:0	**Reserved**
6	15:12	**Reserved**
7..9	95:0	**Reserved** Format: MBZ

MPEG2

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.

MSAA Sample Number Message Address Control

MACD_MSAA_SN - MSAA Sample Number Message Address Control		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:4	**Reserved** Format: MBZ Ignored
	3:0	**Sample Number** Format: U4 Specifies the sample number for the slot. If the sample number is larger than the **Number of Multisamples** in the Surface State, then the access is out of bounds.

MsgDescpt31

<table>
<tr><th colspan="3">MsgDescpt31</th></tr>
<tr><td colspan="3">Source: Eulsa
Size (in bits): 29
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>28:25</td><td>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.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>1-15</td><td>Number of MRF Registers</td></tr></table></td></tr>
<tr><td>24:20</td><td>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.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0-16</td><td>Number of Registers</td></tr></table></td></tr>
<tr><td>19</td><td>Header Present
<table><tr><td>Format:</td><td>Enable</td></tr></table>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.</td></tr>
<tr><td>18:0</td><td>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.</td></tr>
</table>

No Event Data Payload

MDP_NO_EVENT - No Event Data Payload		
Source:	EuSubFunctionGateway	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..7	255:0	**Reserved** Format: MBZ

Normal Media Block Message Header

MH_MB - Normal Media Block Message Header

Source:	EuSubFunctionDataPort1
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:0	**X Offset** Format: S31 X offset (in bytes) of the upper left corner of the block into the surface. **Programming Notes** Must be DWord aligned (Bits 1:0 MBZ) for the write form of the message.
1	31:0	**Y Offset** Format: S31 Y offset (in rows) of the upper left corner of the block into the surface.
2	31:0	**Normal Media Block Message Control** Format: MHC_MB_CONTROL Specifies the Normal message subtype and additional input parameters.
3	31:0	**Mask** Format: U32 The Mask is ignored by the Normal Media Block message: all Dwords are always returned on reads, and always enabled to be written on writes.
4	31:0	**FFTID** Format: MHC_FFTID Fixed Function Thread ID
5..7	95:0	**Reserved** Format: MBZ Ignored

Normal Media Block Message Header Control

MHC_MB_CONTROL - Normal Media Block Message Header Control

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

<table>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:30</td><td>

Message Mode

Specifies the interpretation of M0.3 (Pixel or Byte Mask). For the Sampler Cache Data Port, this field is ignored, behaving as if always set to NORMAL.

<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>00h</td><td>Normal</td><td>The Block Height and Block Width fields are specified in this Dword. The Mask is ignored by a media block read message and behaves as if it is set to all ones for a media block write message.</td></tr>
<tr><td>Others</td><td>Reserved</td><td>Reserved.</td></tr>
</table>

<table>
<tr><th>Programming Notes</th></tr>
<tr><td>The Media Block Read message is Normal subtype when both Sub-Register Offset and Register Pitch Control are zero. The Media Block Read message is Merged subtype when either Sub-Register Offset or Register Pitch Control are non-zero.</td></tr>
</table>

</td></tr>
<tr><td></td><td>29</td><td>

Reserved

<table>
<tr><td>Format:</td><td>MBZ</td></tr>
</table>

Ignored

</td></tr>
<tr><td></td><td>28:24</td><td>

Sub-Register Offset

<table>
<tr><td>Default Value:</td><td>0</td></tr>
<tr><td>Format:</td><td>U5</td></tr>
</table>

The sub-register offset must be 0 for Normal Media Block Read message subtype. This field is ignored (reserved) for a media block write message.

</td></tr>
<tr><td></td><td>23:22</td><td>

Reserved

<table>
<tr><td>Format:</td><td>MBZ</td></tr>
</table>

Ignored

</td></tr>
<tr><td></td><td>21:16</td><td>

Block Height

</td></tr>
</table>

MHC_MB_CONTROL - Normal Media Block Message Header Control

		Format: U6 Height in rows of block being accessed. Range = [0,63] representing 1 to 64 rows **Restriction** If Block Width (bytes), then Maximum Block Height (rows) is constrained by (# Dwords width) * (# rows) <= 64 Dwords.
	15:10	**Reserved** Format: MBZ Ignored
	9:8	**Register Pitch Control** Default Value: 0 Format: U2 The register pitch must be 0 for a Normal Media Block Read message. This field is ignored (reserved) for a media block write message.
	7:6	**Reserved** Format: MBZ Ignored
	5:0	**Block Width** Format: U6 Width in bytes of the block being accessed. For normal Media Block Writes, Range = [0,63] representing 1 to 64 Bytes. For normal Media Block Reads and for masked and merged Media Block messages, Range = [0,31] representing 1 to 32 Bytes. **Programming Notes** Must be DWord aligned for the write form of the message.

oMask Message Data Payload Register

MDPR_OMASK - oMask Message Data Payload Register		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:16	**oMask1** Format: U16 oMask for Pixels [15:0] of Slot 1. Not used for Slot Group HI.
	15:0	**oMask0** Format: U16 oMask for Pixels [15:0] of Slot 0. Not used for Slot Group HI.
1	31:16	**oMask3** Format: U16 oMask for Pixels [15:0] of Slot 3. Not used for Slot Group HI.
	15:0	**oMask2** Format: U16 oMask for Pixels [15:0] of Slot 2. Not used for Slot Group HI.
2	31:16	**oMask5** Format: U16 oMask for Pixels [15:0] of Slot 5. Not used for Slot Group HI.
	15:0	**oMask4** Format: U16 oMask for Pixels [15:0] of Slot 4. Not used for Slot Group HI.
3	31:16	**oMask7** Format: U16 oMask for Pixels [15:0] of Slot 7. Not used for Slot Group HI.
	15:0	**oMask6**

MDPR_OMASK - oMask Message Data Payload Register

DWord	Bit	Description
		Format: U16 oMask for Pixels [15:0] of Slot 6. Not used for Slot Group HI.
4	31:16	**oMask9** Format: U16 oMask for Pixels [15:0] of Slot 9. Used only if Slot Group HI or SIMD16.
	15:0	**oMask8** Format: U16 oMask for Pixels [15:0] of Slot 8. Used only if Slot Group HI or SIMD16.
5	31:16	**oMask11** Format: U16 oMask for Pixels [15:0] of Slot 11. Used only if Slot Group HI or SIMD16.
	15:0	**oMask10** Format: U16 oMask for Pixels [15:0] of Slot 10. Used only if Slot Group HI or SIMD16.
6	31:16	**oMask13** Format: U16 oMask for Pixels [15:0] of Slot 13. Used only if Slot Group HI or SIMD16.
	15:0	**oMask12** Format: U16 oMask for Pixels [15:0] of Slot 12. Used only if Slot Group HI or SIMD16.
7	31:16	**oMask15** Format: U16 oMask for Pixels [15:0] of Slot 15. Used only if Slot Group HI or SIMD16.
	15:0	**oMask14**

		Format:	U16
		oMask for Pixels [15:0] of Slot 14. Used only if Slot Group HI or SIMD16.	

OM Replicated SIMD16 Render Target Data Payload

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]

OM S0A SIMD8 Render Target Data Payload

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

OM S0A SIMD16 Render Target Data Payload

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]**

MDP_RTW_MA16 - OM S0A SIMD16 Render Target Data Payload

		Format: **MDP_DW_SIMD8** Slots [15:8] Red
5.0-5.7	255:0	**Green[7:0]** Format: **MDP_DW_SIMD8** 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

OM SIMD8 Dual Source Render Target Data Payload

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**

MDP_RTW_M8DS - OM SIMD8 Dual Source Render Target Data Payload

		Format: MDP_DW_SIMD8 Slots[7:0] or [15:8] of Src1 Red
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

OM SIMD8 Render Target Data Payload

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

OM SIMD16 Render Target Data Payload

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**

MDP_RTW_M16 - OM SIMD16 Render Target Data Payload

		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 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

OS OM S0A SIMD8 Render Target Data Payload

MDP_RTW_SMA8 - OS 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**

MDP_RTW_SMA8 - OS OM S0A SIMD8 Render Target Data Payload

		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	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] Stencil

OS OM SIMD8 Dual Source Render Target Data Payload

MDP_RTW_SM8DS - OS 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

MDP_RTW_SM8DS - OS OM SIMD8 Dual Source Render Target Data Payload		
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** 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	**Stencil** Format: MDPR_STENCIL Slots [7:0] or [15:8] of Stencil

OS OM SIMD8 Render Target Data Payload

MDP_RTW_SM8 - OS 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	**Stencil**

MDP_RTW_SM8 - OS OM SIMD8 Render Target Data Payload

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**MDPR_STENCIL**</td></tr></table>Slots [7:0] Stencil

OS S0A SIMD8 Render Target Data Payload

MDP_RTW_SA8 - OS 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	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

OS SIMD8 Dual Source Render Target Data Payload

MDP_RTW_S8DS - OS 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**

MDP_RTW_S8DS - OS SIMD8 Dual Source Render Target Data Payload

		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** 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	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] or [15:8] of Stencil

OS SIMD8 Render Target Data Payload

MDP_RTW_S8 - OS 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	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] Stencil

OS SZ OM S0A SIMD8 Render Target Data Payload

MDP_RTW_SZMA8 - OS SZ OM S0A SIMD8 Render Target Data Payload

Source:	BSpec
Size (in bits):	2048
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**

MDP_RTW_SZMA8 - OS SZ OM S0A SIMD8 Render Target Data Payload

		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
7.0-7.7	255:0	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] Stencil

OS SZ OM SIMD8 Dual Source Render Target Data Payload

MDP_RTW_SZM8DS - OS SZ OM SIMD8 Dual Source 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 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

MDP_RTW_SZM8DS - OS SZ OM SIMD8 Dual Source Render Target Data Payload

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** 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
10.0-10.7	255:0	**Stencil** Format: **MDPR_STENCIL** Slots [7:0] or [15:8] of Stencil

OS SZ OM SIMD8 Render Target Data Payload

MDP_RTW_SZM8 - OS SZ OM SIMD8 Render Target Data Payload

Source:	BSpec
Size (in bits):	1792
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_SZM8 - OS SZ OM SIMD8 Render Target Data Payload

5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
6.0-6.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

OS SZ S0A SIMD8 Render Target Data Payload

MDP_RTW_SZA8 - OS SZ 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	**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_SZA8 - OS SZ S0A SIMD8 Render Target Data Payload		
5.0-5.7	255:0	**Source Depth** Format: MDP_DW_SIMD8 Slots [7:0] Source Depth
6.0-6.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] Stencil

OS SZ SIMD8 Dual Source Render Target Data Payload

MDP_RTW_SZ8DS - OS SZ 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	**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

MDP_RTW_SZ8DS - OS SZ SIMD8 Dual Source Render Target Data Payload

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 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
9.0-9.7	255:0	**Stencil** Format: MDPR_STENCIL Slots [7:0] or [15:8] of Stencil

OS SZ SIMD8 Render Target Data Payload

MDP_RTW_SZ8 - OS SZ SIMD8 Render Target Data Payload		
Source:	BSpec	
Size (in bits):	1536	
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
5.0-5.7	255:0	**Stencil**

MDP_RTW_SZ8 - OS SZ SIMD8 Render Target Data Payload

<table>
<tr><td></td><td></td><td>
<table>
<tr><td></td><td></td></tr>
<tr><td>Format:</td><td>MDPR_STENCIL</td></tr>
</table>
Slots [7:0] Stencil
</td></tr>
</table>

Oword 2 Block Data Payload

<table>
<tr><th colspan="3">MDP_OW2 - Oword 2 Block Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.3</td><td>127:0</td><td>Oword0
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U128</td></tr></table>Specifies the Oword data for block element 0</td></tr>
<tr><td>0.4-0.7</td><td>127:0</td><td>Oword1
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U128</td></tr></table>Specifies the Oword data for block element 1</td></tr>
</table>

Oword 4 Block Data Payload

MDP_OW4 - Oword 4 Block 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	**Data[1:0]** Format: MDCR_OW Specifies the Oword data for block elements [1:0]
1.0-1.7	255:0	**Data[3:2]** Format: MDCR_OW Specifies the Oword data for block elements [3:2]

Oword 8 Block Data Payload

MDP_OW8 - Oword 8 Block Data Payload

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

DWord	Bit	Description
0.0-0.7	255:0	**Data[1:0]** Format: **MDCR_OW** Specifies the Oword data for block elements [1:0]
1.0-1.7	255:0	**Data[3:2]** Format: **MDCR_OW** Specifies the Oword data for block elements [3:2]
2.0-2.7	255:0	**Data[5:4]** Format: **MDCR_OW** Specifies the Oword data for block elements [5:4]
3.0-3.7	255:0	**Data[7:6]** Format: **MDCR_OW** Specifies the Oword data for block elements [7:6]

Oword A64 SIMD8 Atomic Operation CMPWR16B Message Data Payload

MDP_A64_AOP8_OW2 - Oword A64 SIMD8 Atomic Operation CMPWR16B Message Data Payload

Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0.0-0.7	255:0	**Slot[1:0] Src0** Format: MDCR_OW Specifies the Slot [1:0] Source 0 data
1.0-1.7	255:0	**Slot[3:2] Src0** Format: MDCR_OW Specifies the Slot [3:2] Source 0 data
2.0-2.7	255:0	**Slot[5:4] Src0** Format: MDCR_OW Specifies the Slot [5:4] Source 0 data
3.0-3.7	255:0	**Slot[7:6] Src0** Format: MDCR_OW Specifies the Slot [7:6] Source 0 data
4.0-4.7	255:0	**Slot[1:0] Src1**

MDP_A64_AOP8_OW2 - Oword A64 SIMD8 Atomic Operation CMPWR16B Message Data Payload

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**MDCR_OW**</td></tr></table>Specifies the Slot [1:0] Source 1 data
5.0-5.7	255:0	**Slot[3:2] Src1**<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**MDCR_OW**</td></tr></table>Specifies the Slot [3:2] Source 1 data
6.0-6.7	255:0	**Slot[5:4] Src1**<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**MDCR_OW**</td></tr></table>Specifies the Slot [5:4] Source 1 data
7.0-7.7	255:0	**Slot[7:6] Src1**<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>**MDCR_OW**</td></tr></table>Specifies the Slot [7:6] Source 1 data

Oword Data Blocks Message Descriptor Control Field

MDC_DB_OW - Oword Data Blocks Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	3	
Default Value:	0x00000000	

DWord	Bit	Description
0	2:0	**Data Blocks** Specifies the number of Oword blocks to be read or written

Value	Name	Description
00h	OW1L	1 Oword, read into or written from the low 128 bits of the destination register
01h	OW1U	1 Oword, read into or written from the high 128 bits of the destination register
02h	OW2	2 Owords
03h	OW4	4 Owords
04h	OW8	8 Owords
Others	Reserved	Ignored

Oword Data Payload Register

MDCR_OW - Oword Data Payload Register		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0.0-0.3	127:0	**Oword0** Format: U128 Specifies the slot 0 data in this payload register
0.4-0.7	127:0	**Oword1** Format: U128 Specifies the slot 1 data in this payload register

Oword Dual Data Blocks Message Descriptor Control Field

MDC_DB_OWD - Oword Dual Data Blocks Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	

DWord	Bit	Description
0	1:0	**OW Dual Data Blocks** Specifies the number of Oword Blocks to be read or written <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>00h</td><td>OWD1</td><td>1 Hword register, 2 Owords</td></tr><tr><td>02h</td><td>OWD4</td><td>4 Hword registers, 8 Owords</td></tr><tr><td>Others</td><td>Reserved</td><td>Ignored</td></tr></table>

PALETTE_ENTRY

PALETTE_ENTRY		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:24	**Alpha** Format: U8 Alpha channel value for this entry in the texture color palette.
	23:16	**Red** Format: U8 Red channel value for this entry in the texture color palette.
	15:8	**Green** Format: U8 Green channel value for this entry in the texture color palette.
	7:0	**Blue** Format: U8 Blue channel value for this entry in the texture color palette.

Performance Counter Report Format 101b

Performance Counter Report Format 101b		
Source:	BSpec	
Size (in bits):	2048	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0	31:0	**RPT_ID**
1	31:0	**TIME_STAMP**
2	31:0	**CTX_ID**
3	31:0	**GPU_TICKS**
4	31:0	**A-Cntr 0 (low dword)**
5	31:0	**A-Cntr 1 (low dword)**
6	31:0	**A-Cntr 2 (low dword)**
7	31:0	**A-Cntr 3 (low dword)**
8	31:0	**A-Cntr 4 (low dword)**
9	31:0	**A-Cntr 5 (low dword)**
10	31:0	**A-Cntr 6 (low dword)**
11	31:0	**A-Cntr 7 (low dword)**
12	31:0	**A-Cntr 8 (low dword)**
13	31:0	**A-Cntr 9 (low dword)**
14	31:0	**A-Cntr 10 (low dword)**
15	31:0	**A-Cntr 11 (low dword)**
16	31:0	**A-Cntr 12 (low dword)**
17	31:0	**A-Cntr 13 (low dword)**
18	31:0	**A-Cntr 14 (low dword)**
19	31:0	**A-Cntr 15 (low dword)**
20	31:0	**A-Cntr 16 (low dword)**
21	31:0	**A-Cntr 17 (low dword)**

Performance Counter Report Format 101b

22	31:0	**A-Cntr 18 (low dword)**
23	31:0	**A-Cntr 19 (low dword)**
24	31:0	**A-Cntr 20 (low dword)**
25	31:0	**A-Cntr 21 (low dword)**
26	31:0	**A-Cntr 22 (low dword)**
27	31:0	**A-Cntr 23 (low dword)**
28	31:0	**A-Cntr 24 (low dword)**
29	31:0	**A-Cntr 25 (low dword)**
30	31:0	**A-Cntr 26 (low dword)**
31	31:0	**A-Cntr 27 (low dword)**
32	31:0	**A-Cntr 28 (low dword)**
33	31:0	**A-Cntr 29 (low dword)**
34	31:0	**A-Cntr 30 (low dword)**
35	31:0	**A-Cntr 31 (low dword)**
36	31:0	**A-Cntr 32 (low dword)**
37	31:0	**A-Cntr 33 (low dword)**
38	31:0	**A-Cntr 34 (low dword)**
39	31:0	**A-Cntr 35 (low dword)**
40	31:24	**High byte of A3**
	23:16	**High byte of A2**
	15:8	**High byte of A1**
	7:0	**High byte of A0**
41	31:24	**High byte of A7**
	23:16	**High byte of A6**
	15:8	**High byte of A5**
	7:0	**High byte of A4**
42	31:24	**High byte of A11**
	23:16	**High byte of A10**
	15:8	**High byte of A9**
	7:0	**High byte of A8**
43	31:24	**High byte of A15**
	23:16	**High byte of A14**
	15:8	**High byte of A13**
	7:0	**High byte of A12**
44	31:24	**High byte of A19**
	23:16	**High byte of A18**

Performance Counter Report Format 101b		
	15:8	**High byte of A17**
	7:0	**High byte of A16**
45	31:24	**High byte of A23**
	23:16	**High byte of A22**
	15:8	**High byte of A21**
	7:0	**High byte of A20**
46	31:24	**High byte of A27**
	23:16	**High byte of A26**
	15:8	**High byte of A25**
	7:0	**High byte of A24**
47	31:24	**High byte of A31**
	23:16	**High byte of A30**
	15:8	**High byte of A29**
	7:0	**High byte of A28**
48	31:0	**B-Cntr 0**
49	31:0	**B-Cntr 1**
50	31:0	**B-Cntr 2**
51	31:0	**B-Cntr 3**
52	31:0	**B-Cntr 4**
53	31:0	**B-Cntr 5**
54	31:0	**B-Cntr 6**
55	31:0	**B-Cntr 7**
56	31:0	**C-Cntr 0**
57	31:0	**C-Cntr 1**
58	31:0	**C-Cntr 2**
59	31:0	**C-Cntr 3**
60	31:0	**C-Cntr 4**
61	31:0	**C-Cntr 5**
62	31:0	**C-Cntr 6**
63	31:0	**C-Cntr 7**

Per Thread Scratch Space Message Header Control

MHC_PTSS - Per Thread Scratch Space Message Header Control

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

DWord	Bit	Description
0	31:4	**Reserved** Format: MBZ Ignored
	3:0	**Per Thread Scratch Space** Format: U4 Specifies the amount of scratch space allowed to be used by this thread for messages in which the Binding Table Index is Stateless model, otherwise this field is ignored.The data port will use this to bounds check scratch space messages. Value range = [0,11] represents [1KB, 2MB] in powers of two. **Programming Notes** Writes out of bounds will be ignored. Reads out of bounds will return 0.

PIXEL_HASH_TABLE_1BIT_32ENTRY

PIXEL_HASH_TABLE_1BIT_32ENTRY

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

Description
2-way pixel hashing table. Table is 32-entries:8X,4Y in [Y][X] format. Each entry is a single bit that indicates which sub-slice hardware block the indicated xy pixel block is mapped.
pixelhash_id maps to dual-subslice. A value of 0 indicates the larger DSS, or first enbled DSS if both enabled DSS are balanced (have same number of enabled subslices)

DWord	Bit	Description
0	31:24	**Pixel Hashing Table Entries y[3]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=3 and x=7..0
	23:16	**Pixel Hashing Table Entries y[2]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=2 and x=7..0
	15:8	**Pixel Hashing Table Entries y[1]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=1 and x=7..0
	7:0	**Pixel Hashing Table Entries y[0]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=0 and x=7..0

PIXEL_HASH_TABLE_1BIT_64ENTRY

PIXEL_HASH_TABLE_1BIT_64ENTRY	
Source:	BSpec
Size (in bits):	64
Default Value:	0x00000000, 0x00000000

Description
2-way pixel hashing table. Table is 64-entries:8X,8Y in [Y][X] format. Each entry is a single bit that indicates which sub-slice hardware block the indicated xy pixel block is mapped.
pixelhash_id maps to dual-subslice. A value of 0 indicates the larger DSS, or first enbled DSS if both enabled DSS are balanced (have same number of enabled subslices)

DWord	Bit	Description
0	31:24	**Pixel Hashing Table Entries y[3]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=3 and x=7..0
	23:16	**Pixel Hashing Table Entries y[2]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=2 and x=7..0
	15:8	**Pixel Hashing Table Entries y[1]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=1 and x=7..0
	7:0	**Pixel Hashing Table Entries y[0]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=0 and x=7..0
1	31:24	**Pixel Hashing Table Entries y[7]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=7 and x=7..0
	23:16	**Pixel Hashing Table Entries y[6]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=6 and x=7..0
	15:8	**Pixel Hashing Table Entries y[5]x[7:0]** Format: U8 Indicates the pixelhash_id for the pixel block that has y=5 and x=7..0
	7:0	**Pixel Hashing Table Entries y[4]x[7:0]**

PIXEL_HASH_TABLE_1BIT_64ENTRY		
		Format: U8 Indicates the pixelhash_id for the pixel block that has y=4 and x=7..0

PIXEL_HASH_TABLE_1BIT_128ENTRY

PIXEL_HASH_TABLE_1BIT_128ENTRY	
Source:	BSpec
Size (in bits):	128
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000

Description
2-way pixel hashing table. Table is 128-entries:16X,8Y in [Y][X] format. Each entry is a single bit that indicates which sub-slice hardware block the indicated xy pixel block is mapped.
pixelhash_id maps to dual-subslice. A value of 0 indicates the larger DSS, or first enbled DSS if both enabled DSS are balanced (have same number of enabled subslices)

DWord	Bit	Description
0	31:16	**Pixel Hashing Table Entries y[1]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=1 and x=15..0
	15:0	**Pixel Hashing Table Entries y[0]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=0 and x=15..0
1	31:16	**Pixel Hashing Table Entries y[3]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=3 and x=15..0
	15:0	**Pixel Hashing Table Entries y[2]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=2 and x=15..0
2	31:16	**Pixel Hashing Table Entries y[5]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=5 and x=15..0
	15:0	**Pixel Hashing Table Entries y[4]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=4 and x=15..0
3	31:16	**Pixel Hashing Table Entries y[7]x[15:0]** Format: U16 Indicates the pixelhash_id for the pixel block that has y=7 and x=15..0
	15:0	**Pixel Hashing Table Entries y[6]x[15:0]**

PIXEL_HASH_TABLE_1BIT_128ENTRY		
		Format: U16 Indicates the pixelhash_id for the pixel block that has y=6 and x=15..0

PIXEL_HASH_TABLE_2BIT_64ENTRY

PIXEL_HASH_TABLE_2BIT_64ENTRY	
Source:	BSpec
Size (in bits):	128
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000

Description
3-wayor 4-way pixel hashing table. Table is 64-entries:8X,8Y in [Y][X] format. Each entry is two bits that indicates which sub-slice hardware block the indicated xy pixel block is mapped.
pixelhash_id maps to subslice. A value of 0 indicates the first enabled subslice. A value of 1 indicates the second enabled subslice.

DWord	Bit	Description
0	31:30	**Pixel Hashing Table Entry y[1]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=1
	29:28	**Pixel Hashing Table Entry y[1]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=1
	27:26	**Pixel Hashing Table Entry y[1]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=1
	25:24	**Pixel Hashing Table Entry y[1]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=1
	23:22	**Pixel Hashing Table Entry y[1]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=1
	21:20	**Pixel Hashing Table Entry y[1]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=1
	19:18	**Pixel Hashing Table Entry y[1]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=1
	17:16	**Pixel Hashing Table Entry y[1]x[0]**

PIXEL_HASH_TABLE_2BIT_64ENTRY

		Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=1
	15:14	**Pixel Hashing Table Entry y[0]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=0
	13:12	**Pixel Hashing Table Entry y[0]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=0
	11:10	**Pixel Hashing Table Entry y[0]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=0
	9:8	**Pixel Hashing Table Entry y[0]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=0
	7:6	**Pixel Hashing Table Entry y[0]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=0
	5:4	**Pixel Hashing Table Entry y[0]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=0
	3:2	**Pixel Hashing Table Entry y[0]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=0
	1:0	**Pixel Hashing Table Entry y[0]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=0
1	31:30	**Pixel Hashing Table Entry y[3]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=3
	29:28	**Pixel Hashing Table Entry y[3]x[6]**

PIXEL_HASH_TABLE_2BIT_64ENTRY		
		Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=3
	27:26	**Pixel Hashing Table Entry y[3]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=3
	25:24	**Pixel Hashing Table Entry y[3]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=3
	23:22	**Pixel Hashing Table Entry y[3]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=3
	21:20	**Pixel Hashing Table Entry y[3]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=3
	19:18	**Pixel Hashing Table Entry y[3]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=3
	17:16	**Pixel Hashing Table Entry y[3]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=3
	15:14	**Pixel Hashing Table Entry y[2]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=2
	13:12	**Pixel Hashing Table Entry y[2]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=2
	11:10	**Pixel Hashing Table Entry y[2]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=2
	9:8	**Pixel Hashing Table Entry y[2]x[4]**

PIXEL_HASH_TABLE_2BIT_64ENTRY		
		Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=2
	7:6	**Pixel Hashing Table Entry y[2]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=2
	5:4	**Pixel Hashing Table Entry y[2]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=2
	3:2	**Pixel Hashing Table Entry y[2]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=2
	1:0	**Pixel Hashing Table Entry y[2]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=2
2	31:30	**Pixel Hashing Table Entry y[5]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=5
	29:28	**Pixel Hashing Table Entry y[5]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=5
	27:26	**Pixel Hashing Table Entry y[5]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=5
	25:24	**Pixel Hashing Table Entry y[5]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=5
	23:22	**Pixel Hashing Table Entry y[5]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=5
	21:20	**Pixel Hashing Table Entry y[5]x[2]**

PIXEL_HASH_TABLE_2BIT_64ENTRY

		Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=5
	19:18	**Pixel Hashing Table Entry y[5]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=5
	17:16	**Pixel Hashing Table Entry y[5]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=5
	15:14	**Pixel Hashing Table Entry y[4]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=4
	13:12	**Pixel Hashing Table Entry y[4]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=4
	11:10	**Pixel Hashing Table Entry y[4]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=4
	9:8	**Pixel Hashing Table Entry y[4]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=4
	7:6	**Pixel Hashing Table Entry y[4]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=4
	5:4	**Pixel Hashing Table Entry y[4]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=4
	3:2	**Pixel Hashing Table Entry y[4]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=4
	1:0	**Pixel Hashing Table Entry y[4]x[0]**

PIXEL_HASH_TABLE_2BIT_64ENTRY

		Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=4
3	31:30	**Pixel Hashing Table Entry y[7]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=7
	29:28	**Pixel Hashing Table Entry y[7]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=7
	27:26	**Pixel Hashing Table Entry y[7]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=7
	25:24	**Pixel Hashing Table Entry y[7]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=7
	23:22	**Pixel Hashing Table Entry y[7]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=7
	21:20	**Pixel Hashing Table Entry y[7]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=7
	19:18	**Pixel Hashing Table Entry y[7]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=7
	17:16	**Pixel Hashing Table Entry y[7]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=7
	15:14	**Pixel Hashing Table Entry y[6]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=6
	13:12	**Pixel Hashing Table Entry y[6]x[6]**

PIXEL_HASH_TABLE_2BIT_64ENTRY		
		Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=6
	11:10	**Pixel Hashing Table Entry y[6]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=6
	9:8	**Pixel Hashing Table Entry y[6]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=6
	7:6	**Pixel Hashing Table Entry y[6]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=6
	5:4	**Pixel Hashing Table Entry y[6]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=6
	3:2	**Pixel Hashing Table Entry y[6]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=6
	1:0	**Pixel Hashing Table Entry y[6]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=6

PIXEL_HASH_TABLE_2BIT_128ENTRY

PIXEL_HASH_TABLE_2BIT_128ENTRY	
Source:	BSpec
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

Description
3-wayor 4-way pixel hashing table. Table is 128-entries:16X,8Y in [Y][X] format. Each entry is two bits that indicates which sub-slice hardware block the indicated xy pixel block is mapped.
pixelhash_id maps to subslice. A value of 0 indicates the first enabled subslice. A value of 1 indicates the second enabled subslice.

DWord	Bit	Description
0	31:30	**Pixel Hashing Table Entry y[0]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=0
	29:28	**Pixel Hashing Table Entry y[0]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=0
	27:26	**Pixel Hashing Table Entry y[0]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=0
	25:24	**Pixel Hashing Table Entry y[0]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=0
	23:22	**Pixel Hashing Table Entry y[0]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=0
	21:20	**Pixel Hashing Table Entry y[0]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=0
	19:18	**Pixel Hashing Table Entry y[0]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=0

PIXEL_HASH_TABLE_2BIT_128ENTRY

	17:16	**Pixel Hashing Table Entry y[0]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=0
	15:14	**Pixel Hashing Table Entry y[0]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=0
	13:12	**Pixel Hashing Table Entry y[0]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=0
	11:10	**Pixel Hashing Table Entry y[0]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=0
	9:8	**Pixel Hashing Table Entry y[0]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=0
	7:6	**Pixel Hashing Table Entry y[0]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=0
	5:4	**Pixel Hashing Table Entry y[0]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=0
	3:2	**Pixel Hashing Table Entry y[0]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=0
	1:0	**Pixel Hashing Table Entry y[0]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=0
1	31:30	**Pixel Hashing Table Entry y[1]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=1

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	29:28	**Pixel Hashing Table Entry y[1]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=1
	27:26	**Pixel Hashing Table Entry y[1]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=1
	25:24	**Pixel Hashing Table Entry y[1]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=1
	23:22	**Pixel Hashing Table Entry y[1]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=1
	21:20	**Pixel Hashing Table Entry y[1]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=1
	19:18	**Pixel Hashing Table Entry y[1]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=1
	17:16	**Pixel Hashing Table Entry y[1]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=1
	15:14	**Pixel Hashing Table Entry y[1]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=1
	13:12	**Pixel Hashing Table Entry y[1]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=1
	11:10	**Pixel Hashing Table Entry y[1]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=1

PIXEL_HASH_TABLE_2BIT_128ENTRY

	9:8	**Pixel Hashing Table Entry y[1]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=1
	7:6	**Pixel Hashing Table Entry y[1]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=1
	5:4	**Pixel Hashing Table Entry y[1]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=1
	3:2	**Pixel Hashing Table Entry y[1]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=1
	1:0	**Pixel Hashing Table Entry y[1]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=1
2	31:30	**Pixel Hashing Table Entry y[2]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=2
	29:28	**Pixel Hashing Table Entry y[2]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=2
	27:26	**Pixel Hashing Table Entry y[2]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=2
	25:24	**Pixel Hashing Table Entry y[2]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=2
	23:22	**Pixel Hashing Table Entry y[2]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=2

PIXEL_HASH_TABLE_2BIT_128ENTRY

	21:20	**Pixel Hashing Table Entry y[2]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=2
	19:18	**Pixel Hashing Table Entry y[2]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=2
	17:16	**Pixel Hashing Table Entry y[2]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=2
	15:14	**Pixel Hashing Table Entry y[2]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=2
	13:12	**Pixel Hashing Table Entry y[2]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=2
	11:10	**Pixel Hashing Table Entry y[2]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=2
	9:8	**Pixel Hashing Table Entry y[2]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=2
	7:6	**Pixel Hashing Table Entry y[2]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=2
	5:4	**Pixel Hashing Table Entry y[2]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=2
	3:2	**Pixel Hashing Table Entry y[2]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=2

PIXEL_HASH_TABLE_2BIT_128ENTRY

	1:0	**Pixel Hashing Table Entry y[2]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=2
3	31:30	**Pixel Hashing Table Entry y[3]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=3
	29:28	**Pixel Hashing Table Entry y[3]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=3
	27:26	**Pixel Hashing Table Entry y[3]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=3
	25:24	**Pixel Hashing Table Entry y[3]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=3
	23:22	**Pixel Hashing Table Entry y[3]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=3
	21:20	**Pixel Hashing Table Entry y[3]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=3
	19:18	**Pixel Hashing Table Entry y[3]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=3
	17:16	**Pixel Hashing Table Entry y[3]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=3
	15:14	**Pixel Hashing Table Entry y[3]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=3

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	13:12	**Pixel Hashing Table Entry y[3]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=3
	11:10	**Pixel Hashing Table Entry y[3]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=3
	9:8	**Pixel Hashing Table Entry y[3]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=3
	7:6	**Pixel Hashing Table Entry y[3]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=3
	5:4	**Pixel Hashing Table Entry y[3]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=3
	3:2	**Pixel Hashing Table Entry y[3]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=3
	1:0	**Pixel Hashing Table Entry y[3]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=3
4	31:30	**Pixel Hashing Table Entry y[4]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=4
	29:28	**Pixel Hashing Table Entry y[4]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=4
	27:26	**Pixel Hashing Table Entry y[4]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=4

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	25:24	**Pixel Hashing Table Entry y[4]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=4
	23:22	**Pixel Hashing Table Entry y[4]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=4
	21:20	**Pixel Hashing Table Entry y[4]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=4
	19:18	**Pixel Hashing Table Entry y[4]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=4
	17:16	**Pixel Hashing Table Entry y[4]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=4
	15:14	**Pixel Hashing Table Entry y[4]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=4
	13:12	**Pixel Hashing Table Entry y[4]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=4
	11:10	**Pixel Hashing Table Entry y[4]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=4
	9:8	**Pixel Hashing Table Entry y[4]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=4
	7:6	**Pixel Hashing Table Entry y[4]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=4

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	5:4	**Pixel Hashing Table Entry y[4]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=4
	3:2	**Pixel Hashing Table Entry y[4]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=4
	1:0	**Pixel Hashing Table Entry y[4]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=4
5	31:30	**Pixel Hashing Table Entry y[5]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=5
	29:28	**Pixel Hashing Table Entry y[5]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=5
	27:26	**Pixel Hashing Table Entry y[5]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=5
	25:24	**Pixel Hashing Table Entry y[5]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=5
	23:22	**Pixel Hashing Table Entry y[5]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=5
	21:20	**Pixel Hashing Table Entry y[5]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=5
	19:18	**Pixel Hashing Table Entry y[5]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=5

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	17:16	**Pixel Hashing Table Entry y[5]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=5
	15:14	**Pixel Hashing Table Entry y[5]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=5
	13:12	**Pixel Hashing Table Entry y[5]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=5
	11:10	**Pixel Hashing Table Entry y[5]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=5
	9:8	**Pixel Hashing Table Entry y[5]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=5
	7:6	**Pixel Hashing Table Entry y[5]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=5
	5:4	**Pixel Hashing Table Entry y[5]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=5
	3:2	**Pixel Hashing Table Entry y[5]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=5
	1:0	**Pixel Hashing Table Entry y[5]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=5
6	31:30	**Pixel Hashing Table Entry y[6]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=6

PIXEL_HASH_TABLE_2BIT_128ENTRY

	29:28	**Pixel Hashing Table Entry y[6]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=6
	27:26	**Pixel Hashing Table Entry y[6]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=6
	25:24	**Pixel Hashing Table Entry y[6]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=6
	23:22	**Pixel Hashing Table Entry y[6]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=6
	21:20	**Pixel Hashing Table Entry y[6]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=6
	19:18	**Pixel Hashing Table Entry y[6]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=6
	17:16	**Pixel Hashing Table Entry y[6]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=6
	15:14	**Pixel Hashing Table Entry y[6]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=6
	13:12	**Pixel Hashing Table Entry y[6]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=6
	11:10	**Pixel Hashing Table Entry y[6]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=6

PIXEL_HASH_TABLE_2BIT_128ENTRY

	9:8	**Pixel Hashing Table Entry y[6]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=6
	7:6	**Pixel Hashing Table Entry y[6]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=6
	5:4	**Pixel Hashing Table Entry y[6]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=6
	3:2	**Pixel Hashing Table Entry y[6]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=6
	1:0	**Pixel Hashing Table Entry y[6]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=6
7	31:30	**Pixel Hashing Table Entry y[7]x[15]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=15 and y=7
	29:28	**Pixel Hashing Table Entry y[7]x[14]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=14 and y=7
	27:26	**Pixel Hashing Table Entry y[7]x[13]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=13 and y=7
	25:24	**Pixel Hashing Table Entry y[7]x[12]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=12 and y=7
	23:22	**Pixel Hashing Table Entry y[7]x[11]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=11 and y=7

PIXEL_HASH_TABLE_2BIT_128ENTRY		
	21:20	**Pixel Hashing Table Entry y[7]x[10]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=10 and y=7
	19:18	**Pixel Hashing Table Entry y[7]x[9]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=9 and y=7
	17:16	**Pixel Hashing Table Entry y[7]x[8]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=8 and y=7
	15:14	**Pixel Hashing Table Entry y[7]x[7]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=7 and y=7
	13:12	**Pixel Hashing Table Entry y[7]x[6]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=6 and y=7
	11:10	**Pixel Hashing Table Entry y[7]x[5]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=5 and y=7
	9:8	**Pixel Hashing Table Entry y[7]x[4]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=4 and y=7
	7:6	**Pixel Hashing Table Entry y[7]x[3]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=3 and y=7
	5:4	**Pixel Hashing Table Entry y[7]x[2]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=2 and y=7
	3:2	**Pixel Hashing Table Entry y[7]x[1]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=1 and y=7

PIXEL_HASH_TABLE_2BIT_128ENTRY

	1:0	**Pixel Hashing Table Entry y[7]x[0]** Format: U2 Indicates the pixelhash_id for the pixel block that has x=0 and y=7

Pixel Sample Mask Render Target Message Header Control

MHC_RT_PSM - Pixel Sample Mask Render Target Message Header Control

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

<table>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:16</td><td>Dispatched Pixel/Sample Enables
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16</td></tr></table>
One bit per pixel (or sample within pixel) indicating which pixels/samples were originally enabled when the thread was dispatched. The Dispatched Pixel/Sample Enables must be unmodified from the ones sent when the pixel shader thread was initiated. If the Dispatched Pixel/Sample Enables are modified, behavior is undefined.
<table><tr><th>Programming Notes</th></tr><tr><td>When operating in PER_SAMPLE mode these bits correspond to samples, not pixels. Each subspan slot (4 bits) corresponds to a specific sample location for the subspan. Note that in NUMSAMPLES_1 mode, a pixel and sample are synonomous. When operating in PER_PIXEL mode, this field is ignored, and instead the SampleEnableMask (obtained via bypass) are used to clear the Depth Scoreboard.</td></tr></table>
</td></tr>
<tr><td>15:0</td><td>Pixel/Sample Enables
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U16</td></tr></table>
Specifies which pixels/samples are still lit based on kill instruction activity in the pixel shader. This mask is AND'd with the Dispatched Pixel/Sample Enables mask, and that is used to control actual accesses to the color buffer. Pixels/samples will be dropped on masked writes, and the GRF is not modified for masked reads.
<table><tr><th>Programming Notes</th></tr><tr><td>When operating in PER_SAMPLE mode these bits correspond to samples, not pixels, as the PS is run per-sample. Each subspan slot (4 bits) corresponds to a specific sample location for the subspan. When operating in PER_PIXEL mode, these bits still correspond to pixels, as the PS is run per-pixel. Each pixel's mask bit is replicated according to Number of Multisamples and combined with other masks to control writes to the multisample locations.</td></tr></table>
</td></tr>
</table>

Power Clock State Format

Power Clock State Format	
Source:	RenderCS
Size (in bits):	32
Default Value:	0x00000088

Known Uses

- R_PWR_CLK_STATE - Render Power Clock State Register
- PM_PWR_CLK_STATE - PM Power Clock State Request (Intended, in GT/GTI space, not yet in use)
- PM_PWR_CLK_STATE (Intended, in GT/GTI space, not yet in use)

DWord	Bit	Description
0	31	**Reserved** Format: MBZ
	30:20	**Reserved** Access: RO Format: MBZ
	19	**Reserved** Access: RO Format: MBZ
	18	**Enable Slice Count Request** Access: R/W Enable Slice Count Request.
	17:12	**Slice Count Request** Access: R/W Slice Count Request. This is limited to the number of slices allowed in a given SKU. Note: In software programs this register as if GT consists of 2 slices with 4 subslices in each slice. Hardware maps this to the LP 1 slice/8-subslice physical layout.

Bit 18 values:

Value	Name	Description
0h	Disable	Use async PMunit slice count request.
1h	Enable	Use SliceCount from this register.

Bits 17:12 values:

Value	Name	Description
000001b		1 slice.

Power Clock State Format

	000010b		2 slices.
	000011b		3 slices.
	000100b		4 slices.
	000101b		5 slices. Hardware will revert to 4 slices
	000110b		6 slices.
	000111b		7 slices.
	001000b		8 slices.

11 SSCountEn

Description
Enable Subslice Count Request. 0 = Use Async subslice count 1 = Use SScount in this register

10:8 SScount

Description
Number of subslices to power. This value only applies when slice 0 is the only one powered (otherwise all available subslices are used per slice) The valid values are further limited by the actual subslice count of the part

Value	Name	Description
000001b		1 sub slice.
000010b		2 sub slices.
000011b		3 sub slices.
100b		4 sub slices

7:4 EUmax

Access:	R/W

Maximum number of EUs to power (per subslice if multiple subslices enabled). To specify an exact number of subslices, set EUmax equal to EUmin.

Value	Name	Description
0010b		2 EUs
0100b		4 EUs
0110b		6 EUs
1000b	**[Default]**	8 EUs

Power Clock State Format

		Programming Notes EUmin and EUmax need to be even and odd numbers are illegal; hardware will clip odd EU counts to an even value.
	3:0	**EUmin** Access: R/W Minimum number of EUs to power (per subslice if multiple subslices enabled). To specify an exact number of subslices, set EUmax equal to EUmin.

Value	Name	Description
0010b		2 EUs
0100b		4 EUs
0110b		6 EUs
1000b	**[Default]**	8 EUs

Programming Notes
EUmin and EUmax need to be even and odd numbers are illegal; hardware will clip odd EU counts to an even value.

PPHWSP_LAYOUT

PPHWSP_LAYOUT - PPHWSP_LAYOUT	
Source:	BSpec
Size (in bits):	32672
Default Value:	0x00000000, 0x00000000,

PPHWSP_LAYOUT - PPHWSP_LAYOUT
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,

PPHWSP_LAYOUT - PPHWSP_LAYOUT
0x00000000, 0x00000000,

PPHWSP_LAYOUT - PPHWSP_LAYOUT

0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000

DWord	Bit	Description
0..3	127:0	**Reserved**
4	31:0	**Ring Head Pointer Storage** **Description** 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).

<table>
<tr><th colspan="3">PPHWSP_LAYOUT - PPHWSP_LAYOUT</th></tr>
<tr><td>5..15</td><td>351:0</td><td>Reserved</td></tr>
<tr><td>16</td><td>0</td><td>Cumulative Context Run Time
This has the cumulative run time of the context on HW. HW reports CTX_TIMESTAMP to this location on a context switch.
This value is written after the context save is complete. The value that is saved in the context image does not include the time between the saving of the cumulative value to context to the time we complete the save. If required for the value to always increment and not take the context save into consideration, driver must look at the value in the context image.</td></tr>
<tr><td rowspan="2">17</td><td>31:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>Element Switch
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>Indicates the context is not submitted as the first element in the execlist.</td></tr>
<tr><td>1</td><td></td><td>Indicates the corresponding context has been submitted as first element of the execlist. Preempt Request Received Timestamp is the time when the pending execlist has been submitted to HW. Note that across multiple submissions a given context could be first or second element of an execlist. This bit will get set if the context has been submitted as the first element in the execlist.</td></tr>
</table></td></tr>
<tr><td>18..19</td><td>63:0</td><td>Preempt Request Received Timestamp
TIMESTAMP register sampled on preemption request is reported.</td></tr>
<tr><td>20..21</td><td>63:0</td><td>Context Restore Complete Timestamp
TIMESTAMP register sampled on context restore complete is reported.</td></tr>
<tr><td>22..23</td><td>63:0</td><td>Context Save Finished Timestamp
TIMESTAMP register sampled on context save completion is reported.</td></tr>
<tr><td>24..27</td><td>127:0</td><td>MI_SEMAPHORE_WAIT
MI_SEMAPHORE_WAIT command on which the context got switched out due to semaphore wait. This field is only valid and must be looked at when the context switch reason in context status buffer is stated as “Wait on Semaphore”.</td></tr>
<tr><td>28..31</td><td>127:0</td><td>Reserved</td></tr>
<tr><td>32..33
This field describes the most recent context switch status of the corresponding context.</td><td>63:0</td><td>Context Switch Status Qword</td></tr>
<tr><td>34..1020</td><td>31583:0</td><td>Reserved</td></tr>
</table>

Predicate Barrier Message Data Payload

MDP_PREDICATE_BARRIER - Predicate Barrier Message Data Payload

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

This response message is sent back only if the Gateway Barrier Message specifies that this is a predicated barrier. This response is written to the GRF writeback location, and the response length specified in the send message to the EU must be 1.

DWord	Bit	Description
0	31:16	**Reserved** Format: MBZ
	15:0	**Predicated Barrier Mask Sum** Format: U16 This field is a sum of the predicate mask bits sent by each thread. This field (and the DW containing it) is not written if the barrier is not marked as a predicated barrier. The kernel should compare this field to 0 for the predicated OR function and compare it to the workgroup size for the predicated AND function.
1..7	223:0	**Reserved** Format: MBZ

Qword Data Payload Register

MDCR_QW - Qword Data Payload Register		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0.0-0.1	63:0	**Qword0** Format: U64 Specifies the slot 0 data in this payload register
0.2-0.3	63:0	**Qword1** Format: U64 Specifies the slot 1 data in this payload register
0.4-0.5	63:0	**Qword2** Format: U64 Specifies the slot 2 data in this payload register
0.6-0.7	63:0	**Qword3** Format: U64 Specifies the slot 3 data in this payload register

Qword SIMD8 Atomic Operation CMPWR8B Message Data Payload

MDP_AOP8_QW2 - Qword SIMD8 Atomic Operation CMPWR8B Message Data Payload		
Source:	BSpec	
Size (in bits):	1024	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0-0.7	255:0	**Slot[7:0] Src0[31:0]** Format: MDCR_DW Specifies the lower 32-bits of Slot [7:0] Source 0 data
1.0-1.7	255:0	**Slot[7:0] Src0[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Slot [7:0] Source 0 data
2.0-2.7	255:0	**Slot[7:0] Src1[31:0]** Format: MDCR_DW Specifies the lower 32-bits of Slot [7:0] Source 1 data
3.0-3.7	255:0	**Slot[7:0] Src1[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Slot [7:0] Source 1 data

Qword SIMD8 Atomic Operation CMPWR Message Data Payload

MDP_A64_AOP8_QW2 - Qword SIMD8 Atomic Operation CMPWR Message Data Payload		
Source:	BSpec	
Size (in bits):	1024	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0-0.7	255:0	**Slot[3:0] Src0** Format: MDCR_QW Specifies the Slot [3:0] Source 0 data
1.0-1.7	255:0	**Slot[7:4] Src0** Format: MDCR_QW Specifies the Slot [7:4] Source 0 data
2.0-2.7	255:0	**Slot[3:0] Src1** Format: MDCR_QW Specifies the Slot [3:0] Source 1 data
3.0-3.7	255:0	**Slot[7:4] Src1** Format: MDCR_QW Specifies the Slot [7:4] Source 1 data

Qword SIMD8 Atomic Operation Return Data Message Data Payload

<table>
<tr><th colspan="3">MDP_AOP8_QW1 - Qword SIMD8 Atomic Operation Return Data Message Data Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 512
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Slot[7:0] Qword[31:0]
Format: MDCR_DW
Specifies the lower 32-bits of Slot [7:0] Return data</td></tr>
<tr><td>1.0-1.7</td><td>255:0</td><td>Slot[7:0] Qword[63:32]
Format: MDCR_DW
Specifies the upper 32-bits of Slot [7:0] Return data</td></tr>
</table>

Qword SIMD8 Data Payload

MDP_QW_SIMD8 - Qword SIMD8 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	**Data[3:0]** Format: MDCR_QW Specifies the Slot [3:0] data
1.0-1.7	255:0	**Data[7:4]** Format: MDCR_QW Specifies the Slot [7:4] data

Qword SIMD16 Atomic Operation CMPWR8B Message Data Payload

MDP_AOP16_QW2 - Qword SIMD16 Atomic Operation CMPWR8B Message Data Payload		
Source:	BSpec	
Size (in bits):	2048	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0-0.7	255:0	**Slot[7:0] Src0[31:0]** Format: MDCR_DW Specifies the lower 32-bits of Source 0 data for Slot [7:0]
1.0-1.7	255:0	**Slot[15:8] Src0[31:0]** Format: MDCR_DW Specifies the lower 32-bits Source 0 data for Slot [15:8]
2.0-2.7	255:0	**Slot[7:0] Src0[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Source 0 data for Slot [7:0]
3.0-3.7	255:0	**Slot[15:8] Src0[63:32]** Format: MDCR_DW Specifies the upper 32-bits Source 0 data for Slot [15:8]
4.0-4.7	255:0	**Slot[7:0] Src1[31:0]**

MDP_AOP16_QW2 - Qword SIMD16 Atomic Operation CMPWR8B Message Data Payload

		Format: MDCR_DW Specifies the lower 32-bits of Source 1 data for Slot [7:0]
5.0-5.7	255:0	**Slot[15:8] Src1[31:0]** Format: MDCR_DW Specifies the lower 32-bits Source 1 data for Slot [15:8]
6.0-6.7	255:0	**Slot[7:0] Src1[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Source 1 data for Slot [7:0]
7.0-7.7	255:0	**Slot[15:8] Src1[63:32]** Format: MDCR_DW Specifies the upper 32-bits Source 1 data for Slot [15:8]

Qword SIMD16 Atomic Operation Return Data Message Data Payload

MDP_AOP16_QW1 - Qword SIMD16 Atomic Operation Return Data Message Data Payload

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

DWord	Bit	Description
0.0-0.7	255:0	**Slot[7:0] Qword[31:0]** Format: MDCR_DW Specifies the lower 32-bits of Return data for Slot [7:0]
1.0-1.7	255:0	**Slot[15:8] Qword[31:0]** Format: MDCR_DW Specifies the lower 32-bits of Return data for Slot [15:8]
2.0-2.7	255:0	**Slot[7:0] Qword[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Return data for Slot [7:0]
3.0-3.7	255:0	**Slot[15:8] Qword[63:32]** Format: MDCR_DW Specifies the upper 32-bits of Return data for Slot [15:8]

Qword SIMD16 Data Payload

MDP_QW_SIMD16 - Qword SIMD16 Data Payload		
Source:	BSpec	
Size (in bits):	1024	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0.0-0.7	255:0	**Data[3:0]** Format: MDCR_QW Specifies the Slot [3:0] data
1.0-1.7	255:0	**Data[7:4]** Format: MDCR_QW Specifies the Slot [7:4] data
2.0-2.7	255:0	**qw11_qw8** Format: MDCR_QW Specifies the Slot [11:8] data
3.0-3.7	255:0	**qw15_qw12** Format: MDCR_QW Specifies the Slot [15:12] data

Read-Only Data Port Message Types

MT_DP_RO - Read-Only Data Port Message Types		
Source:	EuSubFunctionReadOnlyDataPort	
Size (in bits):	5	
Default Value:	0x00000000	
Lists all the Message Types in a Read-Only Data Port Message Descriptor [18:14]. Read operations from the Constant Cache and Sampler Cache are encoded in the Read-Only Data Port. Many of the operations are also implemented in Data Port 0, and those operations use the same Message Header.		
DWord	**Bit**	**Description**
0	4:0	**Message Type** Specifies type of message

Value	Name	Description
00h	MT_CC_OWB **[Default]**	Oword Block Read Constant Cache message
01h	MT_CC_OWUB	Unaligned Oword Block Read Constant Cache message
03h	MT_CC_DWS	Dword Scattered Read Constant Cache message
04h	MT_SC_OWUB	Unaligned Oword Block Read Sampler Cache message
05h	MT_SC_MB	Media Block Read Sampler Cache message
06h	MT_RSI	Read Surface Info message
Others	Reserved	Ignored

Read Surface Info 32-Bit Address Payload

MAP32B_RSI - Read Surface Info 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	31:0	**U** Format: U32 Specifies the U channel address offset.
0.1	31:0	**V** Format: U32 Specifies the V channel address offset.
0.2	31:0	**R** Format: U32 Specifies the R channel address offset.
0.3	31:0	**LOD** Format: MACD_LOD Specifies the LOD.
0.4-0.7	127:0	**Reserved** Format: MBZ Ignored

Read Surface Info Data Payload

MDP_RSI - Read Surface Info 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
0.0-0.5	191:0	**Reserved** Format: MBZ Ignored
0.6-0.7	63:0	**Instruction Base Address** Format: GraphicsAddress[63:0] Instruction Base Address from STATE_BASE_ADDRESS, extended to 64-bit format. **Programming Notes** The 48-bit address is returned in a 64-bit address in canonical form.
1.0	31:0	**Width** Format: U32 Surface Width generally computed from RENDER_SURFACE_STATE Width (stored as width minus 1). The value is 0 for NULL surface, and in all other cases (Width+1) » LOD. Surface Width from RENDER_SURFACE_STATE (U14), zero extended to 32 bits.
1.1	31:0	**Height** Format: U32 Surface Height, generally computed from RENDER_SURFACE_STATE Height (stored as height minus 1). The value for a 1D array is RENDER_SUFACE_STATE's (Depth + 1). The value for 1D non-array, BUFFER, and NULL surface is 0. In all other case, the value is (Height + 1) » LOD.
1.2	31:0	**Depth** Format: U32 Surface Depth, generally computed from RENDER_SURFACE_STATE Depth (which is stored depth minus 1). If 2D Array or Cube Array surface, value is the (Depth+1). If 3D surface, value is (Depth+1) » LOD. In all other case, the value is 0.

MDP_RSI - Read Surface Info Data Payload

1.3	31:0	**MIP Count** Format: U32 MIP Count from RENDER_SURFACE_STATE, range [0, 14], zero extended to 32 bits.
1.4	31:0	**Surface Type** Format: U32 Surface Type from RENDER_SURFACE_STATE, zero extended to 32 bits
1.5	31:0	**Surface Format** Format: U32 Surface Format from RENDER_SURFACE_STATE (U9), zero extended to 32 bits.
1.6-1.7	63:0	**Reserved** Format: MBZ Ignored

Surface Type values:

Value	Name	Description
0h	SURFTYPE_1D	1-dimensional map or array of maps
1h	SURFTYPE_2D	2-dimensional map or array of maps
2h	SURFTYPE_3D	3-dimensional map (volumetric) of maps
3h	SURFTYPE_CUBE	Cube map or array of cube maps
4h	SURFTYPE_BUFFER	Element in a buffer
5h	SURFTYPE_STRBUF	Structured buffer surface
7h	SURTYPE_NULL	Null surface
Others	Reserved	Reserved

RENDER_SURFACE_STATE

RENDER_SURFACE_STATE	
Source:	BSpec
Exists If:	//[MessageType] != 'Sample_8x8'
Size (in bits):	512
Default Value:	0x00000000, 0x80000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 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** 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 specifically 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:

- 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.

DWord	Bit	Description
	28	**Surface Array**

RENDER_SURFACE_STATE		
		Format: Enable This field, if enabled, indicates that the surface is an array. **Programming Notes** If this field is *enabled*, the **Surface Type** must be SURFTYPE_1D, SURFTYPE_2D, or 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	**ASTC_Enable** Format: Enable This field, if enabled, indicates that the surface is one of ASTC compression formats. **Programming Notes** If this field is *enabled*, the definition of **Surface Format** encoding will follow a new convention defined by ASTC. If this field is *disabled*, the definition of **Surface Format** will follow the legacy convention defined in non-ASTC style.
	26:18	**Surface Format** Format: SURFACE_FORMAT **Description** This field specifies the format of the surface or element within this surface. This field is ignored for all data port messages other than the render target message and streamed vertex buffer write message. Some forms of the media block messages use the surface format. If **ASTC_Enable** is set to 0, the supported formats and their encoding is listed in the table (x) in Section (y); Otherwise the supported formats and their encoding is listed in the table (x+1) in Section (y). **Programming Notes** If **ASTC_Enable** is set to 0: 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 compressed texture format (BC*, DXT*, FXT*, ETC*, EAC*) • Any YCRCB* format If **ASTC_Enable** is set to 1: • ASTC_Profile: Bit [26] **value**: 0: LDR-Profile, only support 14 2D footprint in compression block, and 10 LDR color endpoint modes (CEM). **value**: 1: Full-Profile, support all ASTC footprint in 2D and 3D, and all 16 CEM including both HDR and LDR modes. • ASTC_BlockDimention: Bit [25]

value: 0: 2D
value: 1: 3D

- ASTC_DecodedFormat: Bit [24]

  value: 0: UNORM8_sRGB;
  value: 1: FLOAT16

- ASTC_2DBlockWidth [23:21]

Value	0h	1h	2h	3h	4h	5h	6h	7h
Width	4	5	6	res	8	res	10	12

- ASTC_2DBlockHeight [20:18]

Value	0h	1h	2h	3h	4h	5h	6h	7h
Height	4	5	6	res	8	res	10	12

- ASTC_3DBlockWidth [23:22]

Value	0h	1h	2h	3h
Width	3	4	5	6

- ASTC_3DBlockHeight [21:20]

Value	0h	1h	2h	3h
Height	3	4	5	6

- ASTC_3DBlockDepth [19:18]

Value	0h	1h	2h	3h
Depth	3	4	5	6

Programming Notes: ASTC_2DBlockHeight and ASTC_2DBlockWidth fields are defined if ASTC_BlockDimention is 0 (2D); While ASTC_3DBlockDepth, ASTC_3DBlockHeight and ASTC_3DBlockWidth are defined if ASTC_BlockDimention is 1 (3D).

This field cannot ASTC format if the **Surface Type** is SURFTYPE_BUFFER or SURFTYPE_STRBUF
This field cannot be ASTC format if the **Surface Type** is SURFTYPE_1D.

This field cannot be a YUV (YCRCB*) or compressed (BC*, DXT*, FXT*, ETC*, EAC*) format if the **Surface Type** is SURFTYPE_BUFFER or SURFTYPE_STRBUF
This field cannot be a planar YUV (PLANAR_*) or compressed (BC*, DXT*, FXT*, ETC*, EAC*) format if the **Surface Type** is SURFTYPE_1D.

17:16 **Surface Vertical Alignment**

Description
For Sampling Engine and Render Target Surfaces: This field specifies the vertical alignment requirement in elements for the surface. Refer to the "Memory Data Formats" chapter for details on how this field changes the layout of the surface in memory. An *element* is defined as a pixel in uncompressed surface formats, and as a compression block in compressed surface formats. For MSFMT_DEPTH_STENCIL type multisampled surfaces, an element is a sample.
This field is used for 2D, CUBE, and 3D surface alignment when Tiled Resource Mode is TRMODE_NONE (Tiled Resource Mode is disabled). This field is ignored for 1D surfaces and

also when Tiled Resource Mode is not TRMODE_NONE (e.g. Tiled Resource Mode is enabled). See the appropriate Alignment table in the "Surface Layout and Tiling" section under Common Surface Formats for the table of alignment values for Tiled Resources.

For other surfaces: This field is ignored.

Value	Name	Description
0h	Reserved	Reserved
1h	VALIGN 4	Vertical alignment factor j = 4
2h	VALIGN 8	Vertical alignment factor j = 8
3h	VALIGN 16	Vertical alignment factor j = 16

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 increases memory usage.
This field is intended to be set to VALIGN_8 only if the surface was rendered as a stencil buffer, since stencil buffer surfaces support only alignment of 8. If set to VALIGN_8, Surface Format must be R8_UINT.
For uncompressed surfaces, the units of "j" are rows of pixels on the physical surface. For compressed texture formats, the units of "j" are in compression blocks, thus each increment in "j" is equal to h pixels, where h is the height of the compression block in pixels.

15:14 **Surface Horizontal Alignment**

Description
For Sampling Engine and Render Target Surfaces: This field specifies the horizontal alignment requirement for the surface.
This field is used for alignment when LOD >= Mip Tail Start LOD This field is ignored when Tiled Resource Mode is not TRMODE_NONE (i.e. Tiled Resources are enabled). See the "Surface Layout and Tiling" section under Common Surface Formats for the table of alignment values for Tile Resources.
For other surfaces: This field is ignored.

Value	Name	Description
0h	Reserved	Reserved
1h	HALIGN 4	Horizontal alignment factor j = 4
2h	HALIGN 8	Horizontal alignment factor j = 8
3h	HALIGN 16	Horizontal alignment factor j = 16

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. In this case it must be set to HALIGN_8 since these surfaces support only alignment of 8. For Z32 formats it must be set to HALIGN_4. Use of HALIGN_8 for other surfaces is supported, but increases memory usage. For uncompressed surfaces, the units of "i" are pixels on the physical surface. For compressed texture formats, the units of "i" are in compression blocks, thus each increment in "i" is equal to w pixels, where w is the width of the compression block in pixels.
		When Auxiliary Surface Mode is set to AUX_CCS_D or AUX_CCS_E, HALIGN 16 must be used.
		For surface format = 32 bpp, num_multisamples = 1 , MIpcount > 0 and surface walk = TiledY, HALIGN must be programmed to 8
	13:12	**Tile Mode** 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.

Value	Name	Description
0h	LINEAR	Linear mode (no tiling)
1h	WMAJOR	W major tiling
2h	XMAJOR	X major tiling
3h	YMAJOR	Y major tiling

Programming Notes
For linear mip-mapped surafces, all MIP levels must have the same pixel/texel format i.e. re-description of the sub-resource is not allowed.
• 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. • Use of WMAJOR is valid only for sampling engine, Data Cache Data Port and render target surfaces and **Surface Format** must be R8_UINT. Vertical Line Stride must be zero. In addition to W tiling, this mode implies that the surface is stored as a stencil buffer. Refer to *Memory Data Formats* section for details on stencil buffer surface layout. • 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. • If **Surface Type** is SURFTYPE_BUFFER, this field must be TILEMODE_LINEAR • If **Number of Multisamples** is not MULTISAMPLECOUNT_1, this field must be YMAJOR.
If **Surface Type** is SURFTYPE_STRBUF, this field must be TILEMODE_LINEAR.
If **Surface Type** is SURFTYPE_1D this field must be TILEMODE_LINEAR, unless **Sampler Legacy 1D Map Layout Disable** is set to 0, in which case TILEMODE_YMAJOR and TILEMODE_WMAJOR are also allowed. **Tiled Resource Mode** must be set to TRMODE_NONE for these cases.
TILEMODE_XMAJOR is only allowed if Surface Type is SURFTYPE_2D.

RENDER_SURFACE_STATE		
		If **Surface Format** is ASTC*, this field must be TILEMODE_YMAJOR.
	11	**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 Cache Data Port:** Specifies number of lines (0 or 1) to skip between logically adjacent lines - provides support of 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*, FXT1, ETC*, EAC*). This bit must not be set if the surface format is compressed type ASTC*. This bit must not be set if the **Auxiliary Surface Mode** is not AUX_NONE.
	10	**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 Cache 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.
	9	**Sampler L2 Out of Order Mode Disable** Format: Disable If disabled this will forced formats which would have bypassed the L2 and been filled into the L1 out of order to be cached in the L2 and send in order to the L1. In general that is any format which is expanded 1:2 in L1 or not expanded at all. This would include all lossless compressed cases For all other formats this will have no affect. **Programming Notes** This bit must be set for the following surface types: BC2_UNORM BC3_UNORM BC5_UNORM BC5_SNORM BC7_UNORM
	8	**Render Cache Read Write Mode** **For Surfaces accessed via the Data Port to Render Cache:** This field specifies the way Render Cache treats a write request. If unset, 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: MBZ

Value	Name	Description
0h	Write-Only Cache	Allocating write-only cache for a write miss
1h	Read-Write Cache	Allocating read-write cache for a write miss

RENDER_SURFACE_STATE

		Programming Notes This field is provided for performance optimization for Render Cache read/write accesses (from EU's point of view).
	7:6	**Media Boundary Pixel Mode** **For 2D Non-Array Surfaces accessed via the Data Port Media Block Read Message or Data Port Transpose 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 or Data Port Transpose 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:** Reserved : 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 **Cube Face Enable - Negative X**

Exists If:	[Surface Type] == 'SURFTYPE_CUBE'
Format:	Enable

For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine: This field enable the individual face 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.

Programming Notes
When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).

5:0 **Reserved**

Exists If:	[Surface Type] != 'SURFTYPE_CUBE'
Format:	MBZ

4 **Cube Face Enable - Positive X**

Exists If:	[Surface Type] == 'SURFTYPE_CUBE'
Format:	Enable

RENDER_SURFACE_STATE

		For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine: This field enable the individual face 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. **Programming Notes** When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).
	3	**Cube Face Enable - Negative Y** Exists If: [Surface Type] == 'SURFTYPE_CUBE' Format: Enable **For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine:** This field enable the individual face 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. **Programming Notes** When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).
	2	**Cube Face Enable - Positive Y** Exists If: [Surface Type] == 'SURFTYPE_CUBE' Format: Enable **For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine:** This field enable the individual face 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. **Programming Notes** When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).
	1	**Cube Face Enable - Negative Z** Exists If: [Surface Type] == 'SURFTYPE_CUBE' Format: Enable **For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine:** This field enable the individual face 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. **Programming Notes** When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).

RENDER_SURFACE_STATE

DWord	Bit	Description
	0	**Cube Face Enable - Positive Z**

Exists If:	[Surface Type] == 'SURFTYPE_CUBE'
Format:	Enable

For SURFTYPE_CUBE Surfaces accessed via the Sampling Engine: This field enable the individual face 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.

Programming Notes
When TEXCOORDMODE_CLAMP is used when accessing a cube map, this field must be programmed to 1b (face enabled).

DWord	Bit	Description
1	31	**Enable Unorm Path in Color Pipe**

Format:	Enable

Enables Unorm Path (fixed Point Conversion of floating point for fill and blend in DAPRSS) in color Pipe.

Value	Name	Description
1	ENABLE **[Default]**	Enables Unorm Path in Color Pipe.
0	DISABLE	Disables Unorm path in Color Pipe.

DWord	Bit	Description
	30:24	**Memory Object Control State**

Format:	MEMORY_OBJECT_CONTROL_STATE

Specifies the memory object control state for this surface and the associated Auxiliary surface (if any).

DWord	Bit	Description
	23:19	**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.

Programming Notes
This field also exists in SAMPLER_STATE. If both fields are zero, the Base Mip Level is zero. If one is nonzero, Base Mip Level is the nonzero field. It is illegal to have both Base Mip Level fields nonzero.

DWord	Bit	Description
	18	**Corner Texel Mode**

Format:	Enable

This field, when ENABLED, indicates when a surface is using corner texel-mode for sampling.

RENDER_SURFACE_STATE

		Corner Texel Mode is ignored for Planar YUV/YCrCb surface formats. Corner Texel Mode is ignored for sample_8X8 and sample_unorm message types. Corner Texel Mode is not supported with Non-Normalized coordinates. Does not support legacy sampler features set0 See legacy sampler page for more details **Value** | **Name** | **Description** 0h | Disable **[Default]** | When programmed to 0h, Corner Texel Mode is disabled. This means texel coordinate references use standard texel reference mode, with respect to the center of the texel. 1h | Enable | When programmed to 1h, Corner Texel Mode is enabled. Texel coordinate references are with respect to the upper left corner of a texel. **Programming Notes** Corner texel mode cannot be enabled for 1D surfaces unless 3DSTATE_DEPTH_BUFFER::Surface Type == SURFTYPE_NULL
	17	**Reserved** Format: | MBZ
	16	**Reserved** Format: | MBZ
	15	**Sample Tap Discard Disable** This bit forces sample tap discard filter mode to be disabled for this surface state. This bit must be set for surfaces which are no Alpha Channel such as R8G8B8_UNORM. **Value** | **Name** | **Description** 0h | ENABLE **[Default]** | When programmed to 0h, Sample Tap Discard filter mode is allowed and is not disabled by this bit. This bit is ignored if Sample Tap Discard is not enabled in the Sampler State. 1h | DISABLE | When programmed to 1h, Sample Tap Discard filter mode will be disabled even if enabled through Sampler State
	14:0	**Surface QPitch** Format: | U15[16:2] **Description** The interpretation of this field is dependent on Surface Type as follows: • SURFTYPE_1D: distance in *pixels* between array slices • SURFTYPE_2D/CUBE: distance in *rows* between array slices. For Quilted Textures this field specifies the distance in rows between *quilt* slices. For compressed texture formats, one row contains a complete compression block vertically. • SURFTYPE_3D: distance in *rows* between R-slices [**Note:** these *rows* are only in the vertical dimension without considering the depth dimension]. For compressed texture formats, one row contains a complete compression block vertically. • Other surface types: field is ignored

RENDER_SURFACE_STATE

Value	Name	Description
[1h,7FFFh]		Range [4h,1FFFCh] in multiples of 4 (low 2 bits missing)

Programming Notes
For Surface Type 1D: This field must be set to an integer multiple of the **Surface Horizontal Alignment** **For Surface Type 2D, CUBE:** This field must be set to an integer multiple of the **Surface Vertical Alignment** **For Surface Type 3D:** *Tile Mode != Linear:* This field must be set to an integer multiple of the tile height (2^Cv) *Tile Mode == Linear:* This field must be set to an integer multiple of the Surface Vertical Alignment **Note:** for compressed textures (BC*, FXT1, ETC*, EAC*), this field is in units of rows of compression blocks.
Note: for the compressed texture ASTC Surface Format, this field is in units of rows of compression blocks.
Software must ensure that this field is set to a value sufficiently large such that the array slices in the surface do not overlap. Refer to the Memory Data Formats section for information on how surfaces are stored in memory.

2 | 31:30 **Reserved**

Format:	MBZ

29:16 **Height**

Format:	U14-1

This field specifies the height of the surface, minus 1. 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	Exists If
[0,0]		must be zero	[Surface Type] == 'SURFTYPE_1D'
[0,16383]		height of surface - 1 (y/v dimension)	[SurfaceType] == 'SURFTYPE_2D'
[0,2047]		height of surface -1 (y/v dimension)	[SrufaceType]== 'SURFTYPE_3D'
[0,16383]		height of surface - 1 (y/v dimension)	[SurfaceType] == 'SURFTYPE_CUBE'
[0,16383]		contains bits [20:7] of the number of entries in the buffer - 1	([SurfaceType] == 'SURFTYPE_BUFFER') || ([SurfaceType] == 'SURFTYPE_STRBUF')

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

RENDER_SURFACE_STATE

		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 frame
		The **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 Surface Format is PLANAR*, see **Planar Memory Organization** section for restrictions on the value of this field.
	15:14	**Reserved** Format: MBZ
	13:0	**Width** Format: U14-1

Description
This field specifies the width of the surface, minus 1. 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.
For surfaces accessed with the Transpose Read Message, this field is in units of DWords.

Value	Name	Description	Exists If
[0,16383]		width of surface - 1 (x/u dimension)	[SurfaceType] == 'SURFTYPE_1D'
[0,16383]		width of surface - 1 (x/u dimension)	[SurfaceType] == 'SURFTYPE_2D'
[0,2047]		width of surface -1 (x/u dimension)	[SrufaceType] == 'SURFTYPE_3D'
[0,16383]		width of surface - 1 (x/u dimension)	[SurfaceType] == 'SURFTYPE_CUBE'
[0,127]		contains bits [6:0] of the number of entries in the buffer - 1	([SurfaceType] == 'SURFTYPE_BUFFER') || ([SurfaceType] == 'SURFTYPE_STRBUF')

Programming Notes
• For surface types other than SURFTYPE_BUFFER or STRBUF The Width specified by this

RENDER_SURFACE_STATE

		field must be less than or equal to the surface pitch (specified in bytes via the Surface Pitch field). • For cube maps, Width must be set equal to the Height. • 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 **Surface Format** is PLANAR*, this field must be a multiple of 2
		If **Number of Multisamples** is MULTISAMPLECOUNT_16, then Width must be 8K texels or less, or the surface must not use the a multisample control surface (MCS).
3	31:21	**Depth**

Format:	U11-1

This field specifies the total number of levels, minus 1, for a volume texture or the number of array elements, minus 1, 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	Exists If
[0,2047]		number of array elements - 1	[SurfaceType] == 'SURFTYPE_1D'
[0,2047]		number of array elements - 1	[SurfaceType] == 'SURFTYPE_2D'
[0,2047]		depth of surface - 1 (z/r dimension)	[SurfaceType] == 'SURFTYPE_3D'
[0,340]		number of array elements - 1 [see programming notes for range]	[SurfaceType] == 'SURFTYPE_CUBE'
[0,2047]		contains bits [31:21] of the number of entries in the buffer - 1	([SurfaceType] == SURFTYPE_BUFFER) OR ([SurfaceType] == 'SURFTYPE_STRBUF')

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 and Typed Data Port 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_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].

RENDER_SURFACE_STATE		
	20	**Tile Address Mapping Mode** Format: U1 This field is used to select between Tile Address Mapping mode and for TileYs and TileYf.

Value	Name	Description
0h		Tile Address Mapping Mode Thou shalt program the bit to 0h. Thou shalt NOT program the bit to 1h. Thou shalt not program the bit to 2h. 3h is *right out*. The number of the programming shall be 0h and 0h shall be the number of the programming. Fractional numbers, being evil in our site (and impossible) must also not be used. Great disappointment and functional woes shall be seen if this bit is not programmed to 0h.
1h		Tile Address Mapping Mode (for Standard Tiling). Thou shalt not program the bit to 1h. Thou shalt program the bit to 0h. Thou shalt not program the bit to 2h. 3h is *right out*. The number of the programming shall be 0h and 0h shall be the number of the programming. Fractional numbers, being evil in our site (and impossible) must also not be used. Great disappointment and functional woes shall be seen if this bit is not programmed to 0h.

Programming Notes
Tile Address Mapping Mode must be set when surface type is SURFTYPE_3D.

	19	**Standard Tiling Mode Extensions**

Description
It changes in the MIP Tail Packing. When enabled (programmed to 1h), MIP Tail packing for Volumetric and 1D are changed as defined in the Surface Layout and Tiling section.
This bit controls enabling of some Standard Tiling extensions:

Value	Name	Description
0h	Disable **[Default]**	When programmed to 0h, the extensions to support Standard Tiling are disabled. Behavior reverts to Miptail packing.
1h	Enable	When programmed to 1h, the changes to support Standard Tiling Extensions are enabled. See the Surface Layout and Tiling section for details.

	18	**Reserved** Format: MBZ
	17:0	**Surface Pitch** Format: U18-1 Pitch in #Bytes Surface Pitch Range: • For surfaces of type SURFTYPE_BUFFER: [0,2047] -> [1B, 2048B] • For surfaces of type SURFTYPE_STRBUF: [0,2047] -> [1B, 2048B]

RENDER_SURFACE_STATE

- For other linear surfaces: [0, 262143] -> [1B, 256KB]
- For X-tiled surface: [511, 262143] -> [512B, 256KB] = [1 tile, 512 tiles]
- For Y-tiled surfaces: [127, 262143]->[128B, 256KB] = [1 tile, 2048 tiles]
- For W-tiled surfaces: [127, 262143]->[128B, 256KB] = [1 tile, 2048 tiles]
- For TileYF and TileYS surfaces, the range is dependent on the Cu parameter (refer to *Memory Data Formats* section for the definition of the Cu parameter depending on the case). The range in bytes is [2^{Cu}-1,262143] -> [(2^{Cu})B,256KB] = [1 tile, 256KB/(2^{Cu}) tiles]

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.

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 untyped data port messages, which are only supported with **Surface Type** SURFTYPE_BUFFER, the pitch is ignored and assumed to be 1 byte.
- For linear surfaces with **Surface Type** of SURFTYPE_STRBUF, the pitch must be a multiple of 4 bytes.
- For linear surfaces with **Surface Type** of SURFTYPE_BUFFER and **Surface Format** RAW, the pitch must be 1 byte.
- 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 (and thus has **Tile Mode** set to TILEMODE_WMAJOR), 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).

- The width of a tile depends on the surface format if Tiled Resource Enable is enabled. Refer to the Tiled Resource Enable field to determine which sub-mode applies to the surface format in use, and determine the Cu parameter from the Surface Layout section. The tile width is equal to 2^Cu bytes.
- For surfaces of type SURFTYPE_1D, this field is ignored.

The following table indicates the maximum byte width, frame width, and pitch size allowed when memory compression is on.

Tiling Mode	Pixel Format	Max Frame Width (bytes)	Max Frame Width (pixels)	Max Pitch (bytes)
Legacy 4K	8bpp	16k	16k	16k + 127
	16bpp	16k	8k	16k + 127
	32bpp	16k	4k	16k + 127

RENDER_SURFACE_STATE

	64bpp	16k	2k	16k + 127
	128bpp	16k	1k	16k + 127
TileYF	8bpp	8k	8k	8k + 63
	16bpp	16k	8k	16k + 127
	32bpp	16k	4k	16k + 127
	64bpp	16k	2k	16k + 255
	128bpp	16k	1k	16k + 255
TileYS	8bpp	16k	16k	16k + 255
	16bpp	16k	8k	16k + 511
	32bpp	16k	4k	16k + 511
	64bpp	16k	2k	16k + 1023
	128bpp	16k	1k	16k + 1023

4 | 31 | **Reserved**

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

31:0 | **Reserved**

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

30:29 | **Render Target And Sample Unorm Rotation**

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

Description
For Render Target Surfaces: This field specifies the rotation of this render target surface when being written to memory.
For sample_unorm Messages: This field specifies the rotation of the data returned by sampler for sample_unorm message.
For Other Surfaces: This field is ignored.

Value	Name	Description
0h	0DEG	No rotation (0 degrees)
1h	90DEG	Rotate by 90 degrees
2h	180DEG	Rotate by 180 degrees [for sample_unorm message]
3h	270DEG	Rotate by 270 degrees

Programming Notes
Programming Notes for Render Target Surfaces only
• Rotation is not supported for render targets of any type other than simple, non-mip-

RENDER_SURFACE_STATE

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: R8G8B8A8_UNORM_SRGB, B8G8R8[A|X]8_UNORM, B8G8R8[A|X]8_UNORM_SRGB, B10G10R10[A|X]2_UNORM, R10G10B10A2_UNORM, SRGB, R16G16B16A16_FLOAT, R16G16B16X16_FLOAT

28:18 **Minimum Array Element**

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

17:7 **Render Target View Extent**

Exists If:	[Surface Type] != 'SURFTYPE_STRBUF'
Format:	U11-1

Range [0,2047] to indicate extent of [1,2048]

For Render Target and Typed Dataport 3D Surfaces:
This field indicates the extent of the accessible 'R' coordinates minus 1 on the LOD currently being rendered to.

For Render Target and Typed Dataport 1D and 2D Surfaces:
This field must be set to the same value as the Depth field.

For Other Surfaces:
This field is ignored.

6 **Multisampled Surface Storage Format**

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

This field indicates the storage format of the multisampled surface.

Value	Name	Description
0h	MSS	Multisampled surface was/is rendered as a render target
1h	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_MSS
- IF this field is MSFMT_DEPTH_STENCIL, the only sampling engine messages allowed are "ld2dms", "resinfo", and "sampleinfo".
- This field is ignored if **Number of Multisamples** is MULTISAMPLECOUNT_1

5:3 **Number of Multisamples**

RENDER_SURFACE_STATE

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

This field indicates the number of multisamples on the surface.

Value	Name
0h	MULTISAMPLECOUNT_1
1h	MULTISAMPLECOUNT_2
2h	MULTISAMPLECOUNT_4
3h	MULTISAMPLECOUNT_8
4h	MULTISAMPLECOUNT_16
5h-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.

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
[0,7]	

5 31:25 **X Offset**

Format:	U7[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 4-high pixel) resolution.

Format:

```
PixelOffset[8:2]
```

Value	Name	Description
[0,127]		Range [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

RENDER_SURFACE_STATE

		is assumed to be 32 bits in width. • For surfaces accessed with the **Data Port Transpose Read 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** not SURFTYPE_2D, this field must be zero. • If **MIP Count** is not zero, this field must be zero. • If **Number of Multisamples** is not MULTISAMPLECOUNT_1, this field must be zero. • If **Surface Array** is enabled, this field must be zero. • If **Auxiliary Surface Mode** is not AUX_NONE, this field must be zero. • If **Surface Vertical Alignment** is VALIGN_8, this field must be a multiple of 8. • For **Surface Format** with 8 bits per element, this field must be a multiple of 16. • For **Surface Format** with 16 bits per element, this field must be a multiple of 8. • If **Tiled Resource Mode** is not TRMODE_NONE, this field must be zero.
	24	**Reserved** Format: MBZ
	23:21	**Y Offset** Format: U3[4:2] 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.) Format: `RowOffset[4:2]`

Value	Name	Description
[0,7]		Range [0,28] in multiples of 4 (low two bits 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.
- If **Surface Type** not SURFTYPE_2D, this field must be zero.
- If **MIP Count** is not zero, this field must be zero.

RENDER_SURFACE_STATE

		• If **Number of Multisamples** is not MULTISAMPLECOUNT_1, this field must be zero. • If **Surface Array** is enabled, this field must be zero. • If **Auxiliary Surface Mode** is not AUX_NONE, this field must be zero. • If **Tiled Resource Mode** is not TRMODE_NONE, this field must be zero. This field must be zero if Surface Format is Planar and the U and V planes are half-pitch (e.g. YV12 format).
	20	**EWA Disable For Cube**

Format:	Disable

Specifies if EWA mode for LOD quality improvement needs to be disabled for cube maps.

Value	Name	Description
0h	Enable **[Default]**	EWA is enabled for cube maps
1h	Disable	EWA is disabled for cube maps

Programming Notes
This field indicates if EWA mode for LOD quality improvement needs to be disabled for cube maps. By default EWA would be on for cube maps hence this field must be 0. If there is any spec violation seen with EWA on cube maps then this field must be set to 1 to disable EWA for cubes.

19:18 **Tiled Resource Mode**

For Sampling Engine, Render Target, and Typed/Untyped Surfaces: This field specifies the tiled resource mode. **For other surfaces:** This field is ignored.

Value	Name	Description	Exists If
0h	NONE	No tiled resource	
1h	4KB	4KB tiled resources	[SurfaceType] == 'SURFTYPE_1D'
2h	64KB	64KB tiled resources	[SurfaceType] == 'SURFTYPE_1D'
1h	TILEYF	4KB tiled resources	[SurfaceType] != 'SURFTYPE_1D'
2h	TILEYS	64KB tiled resources	[SurfaceType] != 'SURFTYPE_1D'
3h	Reserved		

Programming Notes
If **Tile Mode** is not set to TILEMODE_YMAJOR, this field must be set to TRMODE_NONE, unless the Surface Type is SURFTYPE_1D.
If this field is not set to TRMODE_NONE, the **Surface Format** must be one with 8, 16, 32, 64, or 128 bits per element, or one of the compressed texture modes (BC*, ETC*, EAC*, ASTC*). Additionally, YCRCB* formats are supported and treated as 16 bits per element, and the PLANAR_420_8 format is support and treated as 8 bits per element on the Y plane and 16 bits per element on the UV plane (if **Separate UV Plane Enable** is disabled) or 8 bits per element on the U and V planes (if **Separate UV Plane Enable** is enabled).

RENDER_SURFACE_STATE

If this field is set to TRMODE_NONE, the surface cannot contain any null pages unless **Surface Type** is BUFFER or STRBUF. A BUFFER or STRBUF surface with null pages must have **Surface Base Address** and **Surface Pitch** set to an integer multiple of the element size, and **Surface Format** must be one with 8, 16, 32, 64, or 128 bits per element.

If **Surface Format** is PLANAR, the surface cannot contain any null pages.

17:16 Reserved

Format:	MBZ

15 Reserved

Format:	MBZ

14 Coherency Type

Specifies the type of coherency maintained for this surface.

Value	Name	Description
0h	GPU coherent	Surface memory is kept coherent with GPU threads using GPU read/write ordering rules. Surface memory is backed by system memory but is not kept coherent with CPU (LLC).
1h	IA coherent	Surface memory is kept coherent with CPU (LLC).

Programming Notes
This field may optionally be 1 (IA coherent) for messages sent to SFID_DP_DC0 or SFID_DP_DC1 or SFID_DP_DC2. This field is typically set to 0 (GPU coherent) if the context is operating in a non-SVM legacy mode (for example, Ring Buffer or a Execlist using 32-bit Virtual Address Legacy Context PPGTT32).

13:12 Reserved

Format:	MBZ

11:8 Mip Tail Start LOD

Format:	U4 in LOD Units

For Sampling Engine, Render Target, and Typed Surfaces: This field indicates which LOD is the first one in the MIP tail if **Tiled Resource Mode** is not TRMODE_NONE. The MIP tail has a different layout than the rest of the surface. Refer to the *Memory Data Formats* section for more details. **For other surfaces:** This field is ignored.

Programming Notes
This field is ignored if **Tiled Resource Mode** is TRMODE_NONE.
If **Tiled Resource Mode** is not TRMODE_NONE, this field must be set to ensure that mips within the mip tail do not overlap given the storage algorithms given in the Memory Data Formats section.
If **Tiled Resource Mode** is not TRMODE_NONE, to disable the Mip Tail this field must be set to a mip that larger than those present in the surface (i.e. 15). This is recommended for non-mip-mapped surfaces.

RENDER_SURFACE_STATE

The following table indicates the *maximum* size of the mip that is set to be the Mip Tail Start LOD for various cases:

Surface Type	Tiling Mode	#MS	Bits Per Element				
			8	16	32	64	128
1D	64KB	1	16384	8192	4096	2048	1024
	4KB	1	1024	512	256	128	64
2D/ CUBE	TIleYS	1	128x256	128x128	64x128	64x64	32x64
		2	128x128	128x64	64x64	64x32	32x32
		4	64x128	64x64	32x64	32x32	16x32
		8	64x64	64x32	32x32	32x16	16x16
		16	32x64	32x32	16x32	16x16	8x16
	TileYF	any	32x64	32x32	16x32	16x16	8x16
3D	TIleYS	1	32x32x32	16x32x32	16x32x16	16x16x16	8x16x16
	TIleYF	1	16x8x16	8x8x16	8x8x8	8x4x8	4x4x8

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.

3:0 **MIP Count / LOD**

Format:	**Sampling Engine and Typed Surfaces:** U4 in (LOD units - 1) **Render Target Surfaces:** U4 in LOD units
Range	**Sampling Engine and Typed Surfaces:** [0,14] representing [1,15] MIP levels **Render Target Surfaces:** [0,14] representing LOD **Other Surfaces:** [0]

For Sampling Engine and Typed 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

RENDER_SURFACE_STATE

		target surfaces. **For Other Surfaces:** This field is reserved : MBZ **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. For sampling engine surfaces with YCRCB* or PLANAR* surface format, **MIP Count** must be zero.
6	31	**Reserved** Exists If: ([Surface Format] != 'PLANAR') Format: MBZ
	31	**Separate UV Plane Enable** Exists If: ([Surface Format] == 'PLANAR') Format: Enable If enabled, this field indicates that the U and V are present as separate planes. If disabled, the UV data is interleaved on a single plane. **Programming Notes** See the section "Planar Memory Organization" for a description of how the size and location of the chroma planes (U and V) are calculated.
	30	**Half Pitch for Chroma** Exists If: ([Surface Format] == 'PLANAR') This bit enables support for half-pitch chroma planes for Planar YUV surfaces. It is ignored for Non-Planar surfaces. For planar surfaces it allows the chroma planes to be one-half the width of a the Y (Luma) plane. For example, should be set to 0h for NV12 surfaces. Must be set to 1h for YV12 surfaces.
	30:16	**Auxiliary Surface QPitch** Exists If: ([Surface Format] != 'PLANAR') Format: U15[16:2] This field specifies the distance in rows between array slices on the auxiliary surface.

Half Pitch for Chroma values:

Value	Name	Description
0h	Disable **[Default]**	Setting this bit to 0h (default) causes Chroma planes to be treated as full width (same as Y plane).
1h	Enable	Setting this bit to 1h causes Chroma planes (U and V) to be treated as half the width of the Luma (Y) plane.

Auxiliary Surface QPitch values:

Value	Name	Description
[1h,7FFFh]		Range [4h,1FFFCh] in multiples of 4 (low 2 bits missing)

RENDER_SURFACE_STATE

		Programming Notes This field must be set to an integer multiple of the **Surface Vertical Alignment** Software must ensure that this field is set to a value sufficiently large such that the array slices in the auxiliary surface do not overlap. Refer to the Memory Data Formats section for information on how surfaces are stored in memory. For non-multisampled render target's CCS auxiliary surface, QPitch must be computed with Horizontal Alignment = 128 and Surface Vertical Alignment = 256. These alignments are only for CCS buffer and not for associated render target.
	29:16	**X Offset for U or UV Plane** Exists If: ([Surface Format] == 'PLANAR') Format: U14 This field specifies the horizontal offset in pixels from the **Surface Base Address** to the start (origin) of the U plane or interleaved UV plane, depending on the setting of **Separate UV Plane Enable**. **Programming Notes** This field must be a multiple of 4 (bits 1:0 MBZ). If **Tiled Resource Mode** is enabled, this field must be a multiple of the tile width in pixels. **Auxiliary Surface Mode** is forced to AUX_NONE.
	15	**YUV Interpolation Enable** Format: Enable This bit controls whether a Non-Planar YUV4:2:2 and Planar YUV4:2:0 surface use interpolated or replicated U and V channels for input to the Sampler filter. Programming to 1h causes interpolation of U and V channels. In this case the chrominance for odd pixels is computed by an interpolation between adjacent even pixels. Programming to 0h causes the chrominance to be copied from the pixel to the left. **Value** | **Name** | **Description** 0h | Disable **[Default]** | Programming to 0h causes the sampler to replicate U and V channels. This will lead to lower quality in certain cases where the YUV surface is being filtered (e.g. linear). 1h | Enable | Programming to 1h causes the sampler to interpolate the U and V channels between the horizontally neighboring pixels. This will improve image quality if the surface is being filtered.
	14	**Reserved** Exists If: ([Surface Format] == 'PLANAR') Format: MBZ
	14:12	**Reserved** Exists If: ([Surface Format] != 'PLANAR') Format: MBZ
	13:0	**Y Offset for U or UV Plane**

RENDER_SURFACE_STATE

		Exists If: ([Surface Format] == 'PLANAR') Format: U14 This field specifies the vertical offset in rows from the **Surface Base Address** to the start (origin) of the U plane or interleaved UV plane, depending on the setting of **Separate UV Plane Enable**. **Programming Notes** For surfaces where **Surface Format** = PLANAR* and **Separate UV Plane** is Enabled, the Y Offset must be programmed in multiples of **half-rows**. For example, for a surface where Y is physically followed by U and then V in memory, the Y Offset to U plane would be (2*Y-Height). For all other PLANAR YUV formats this is programmed in multiples of full rows. For all format besides PLANAR_420_* This field must be a multiple of 4 (bits 1:0 MBZ). For formats PLANAR_420_* with separate chroma planes (e.g. YV12) this field must be multiple of 4 if U plane is the first chroma plane after the Y (luma) plane. It can be a multiple of 2 if it is the second chroma plane in memory. For formats PLANAR_420_* with interleaved chroma planes (e.g. NV12) this field can be multiple of 2. If **Tiled Resource Mode** is enabled, this field must be a multiple of the tile height in rows. **Auxiliary Surface Mode** is forced to AUX_NONE.
	11:3	**Auxiliary Surface Pitch** Exists If: ([Surface Format] != 'PLANAR') Format: U9-1 This field specifies the Auxiliary surface pitch in (#Tiles - 1). Value: [0, 511] — Name: — Description: -> [1 tile, 512 tiles]
	2:0	**Auxiliary Surface Mode** Exists If: ([Surface Format] != 'PLANAR') Format: U3 Specifies what type of surface the Auxiliary surface is. The Auxiliary surface has its own base address and pitch, but otherwise shares or overrides other fields set for the primary surface, detailed in the programming notes below.

Value	Name	Description
0h	AUX_NONE	No Auxiliary surface is used
1h	AUX_CCS_D	The Auxiliary surface is a CCS (Color Control Surface) with compression disabled or an MCS with compression enabled, depending on **Number of Multisamples**. MCS (Multisample Control Surface) is a special type of CCS.
2h	AUX_APPEND	The Auxiliary surface is an append buffer
3h	AUX_HIZ	The Auxiliary surface is a hierarchical depth buffer [] AUX_HIZ is not a supported value for surfaces being sampled by the 3D sample. Programming to 3h will be ignored by the 3D sampler and interpreted as AUX_NONE.
4h	Reserved	

RENDER_SURFACE_STATE

5h	AUX_CCS_E	The Auxiliary surface is a CCS with compression enabled or an MCS with compression enabled, depending on **Number of Multisamples**.
6h-7h	Reserved	

Programming Notes
The CCS and hierarchical depth Auxiliary surface shares **Height, Width, Depth, Surface Type, Surface Array, Surface Min LOD, MIP Count / LOD, Surface Object Control State, Resource Min LOD**, and **Minimum Array Element** with the primary surface. The hierarchical depth Auxiliary surface uses **Surface Horizontal Alignment** of 16, **Surface Vertical Alignment** of 8, regardless of the primary surface's values for these fields. **X & Y Offset** are set to zero for the purpose of accessing the Auxiliary surface. If this field is set to AUX_HIZ, **Surface Format** must be one of the following: R32_FLOAT, R24_UNORM_X8_TYPELESS, or R16_UNORM, and the format must match the format used when the surface was used as a depth buffer (with R channel corresponding to D channel).
CCS and hierarchical depth Auxiliary surfaces are TileY with **Tiled Resource Mode** of TRMODE_NONE regardless of the tile mode of the primary surface, and **Mip Tail Start LOD** is ignored for these surfaces.
The CCS Auxiliary surface for non-multisampled render targets has Horizontal Alignment = 128 and Vertical alignment = 64.
The CCS Auxiliary surface for **Number of Multisamples** > 1 uses **Surface Horizontal Alignment** of 16 and **Surface Vertical Alignment** of 4 regardless of the primary surface's values for these fields.
If this field is set to AUX_HIZ, **Number of Multisamples** must be MULTISAMPLECOUNT_1, and Surface Type cannot be SURFTYPE_3D.
If **Number of Multisamples** is MULTISAMPLECOUNT_1, AUX_CCS_E setting is only allowed if **Surface Format** is supported for Render Target Compression. This setting enables render target compression.
If **Number of Multisamples** is MULTISAMPLECOUNT_1, AUX_CCS_D setting is only allowed if **Surface Format** supported for Fast Clear. In addition, if the surface is bound to the sampling engine, **Surface Format** must be supported for Render Target Compression for surfaces bound to the sampling engine. For render target surfaces, this setting disables render target compression. For sampling engine surfaces, this mode behaves the same as AUX_CCS_E.
If **Number of Multisamples** is *not* MULTISAMPLECOUNT_1, both AUX_CCS_E and AUX_CCS_D settings indicate that the auxiliary surface is a multisample control surface (MCS), and multisample compression is enabled.
If **Number of Multisamples** is MULTISAMPLECOUNT_1, and if **Tiled Resource Mode** is NOT TRMODE_NONE, then, if CCS tile is NULL, Render Target Tiles represented by that CCS tile are assumed to be NULL by HW.

DWord	Bit	Description
7	31	**Memory Compression Mode** Distinguishes Vertical from Horizontal compression.

Value	Name	Description
0	Horizontal **[Default]**	

RENDER_SURFACE_STATE

		1 | Vertical
	30	**Memory Compression Enable** Format: Enable This surface may contain compressed or compressible pixels. Memory compression will be attempted for writes to this surface. Reads from this surface will check for compressed data. **Programming Notes** The compression control must have 0 value for non-tileY modes. The Memory Compression Enable can be non-zero only for the surface state that has media messages. That is for 3d case the compression control bits will be 0 in normal surface state but can be non-zero in normal surface state for media messages. E.g. sample_unorm. The only sampler messages supported with memory compression enabled are *sample_8x8*, *sample_unorm*, and SIMD16 *sample*. Please refer to vol1a Memory Data Formats chapter >section Media Memory Compression for more details, including format restrictions.
	29	**Reserved** Format: MBZ
	28	**Reserved**
	27:25	**Shader Channel Select Red** Format: **Shader Channel Select** Enumerated Type Specifies which surface channel is read or written in the Red shader channel. **Programming Notes** The Shader channel selects also define which shader channels are written to which surface channel. If the Shader channel select is SCS_ZERO or SCS_ONE then it is not written to the surface. If the shader channel select is SCS_RED it is written to the surface red channel and so on. If more than one shader channel select is set to the same surface channel only the first shader channel in RGBA order will be written. Each shader channel select must be set to the same surface channel (R = SCS_RED, G = SCS_GREEN, B = SCS_BLUE, A = SCS_ALPHA) if the surface is accessed via the sampler's sample_unorm* or sample_8x8 messages. The Shader Channel Select fields do not affect the following sampling engine message types: resinfo, sampleinfo, LOD, and ld_mcs. These messages behave as if each Shader Channel Select is set to the same color surface channel. For the sampling engine *gather4** messages, the Gather4 Source Channel Select field in the message header defines which channel's Shader Channel Select is used to select the surface channel to be sampled. Other Shader Channel Select fields are ignored. For the sampling engine *sample*_c* and *gather4*_c* messages, the compare operation always occurs on the red channel from the surface regardless of the setting of the Shader Channel Select fields. For Render Target, Red, Green and Blue Shader Channel Selects MUST be such that only valid components can be swapped i.e. only change the order of components in the pixel. Any other values for these Shader Channel Select fields are not valid for Render Targets. This also means

RENDER_SURFACE_STATE

		that there MUST not be multiple shader channels mapped to the same RT channel. When multiple Channel selects have the same value and shader channel is disabled, disable channel writes 0s to memory. This behavior does not match with Data Port message via HDC. The output channel is undefined if the source is to a channel is not present for the current surface format. For example, If the surface format is R16_float and the shader channel select green specifies green as the source the output is undefined. It should instead select 0 which is the default for a missing color channel..
	24:22	**Shader Channel Select Green** Format: **Shader Channel Select** Enumerated Type See **Shader Channel Select Red** for details.
	21:19	**Shader Channel Select Blue** Format: **Shader Channel Select** Enumerated Type See **Shader Channel Select Red** for details.
	18:16	**Shader Channel Select Alpha** Format: **Shader Channel Select** Enumerated Type See **Shader Channel Select Red** for details. **Programming Notes** For Render Target, this field MUST be programmed to value = SCS_ALPHA.
	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. **Value** / **Name**: [0,14] / **Programming Notes** This field must be zero if the **ChromaKey Enable** is enabled in the associated sampler.
8..9	63:0	**Surface Base Address** Format: GraphicsAddress[63:0]SurfaceBase 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

RENDER_SURFACE_STATE		
		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. The base address must be aligned to element size. • Linear depth buffer surface base addresses must be 64-byte aligned. Note that while render targets (color) can be SURFTYPE_BUFFER, depth buffers cannot. • Mipmapped surfaces are stored in a "monolithic" (fixed) format, and only require a single address for the base MIP. All other MIPs are positioned relative to the base MIP. • The Base Address for linear (non-tiled) 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 (non-tiled) surfaces have no alignment requirements (byte alignment is sufficient). • 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. Tiles are inherently page-aligned (4K or 64K). • Certain message types used to access surfaces have more stringent alignment requirements. Please refer to the specific data-port message documentation for additional restrictions.
		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 tile aligned (64KB aligned for TileYS, 4KB aligned for all other tile modes). For 1D surfaces, the base address must be 64KB aligned if Tiled Resource Mode is TRMODE_64KB, and 4KB aligned if Tiled Resource Mode is TRMODE_4KB. Compressed (BC*, ASTC, etc.) surface data is usually copied by re-describing each MIP/slice as a separate surface, using a size-equivalent RGBA format. But a MIP/slice within a packed **MIP Tail** doesn't have the tile-aligned **Surface Base Address** required for the re-description. This case must be specially handled by re-describing the packed **MIP Tail** as a single-MIP surface with the width/pitch/height/depth of a single tile, and then use drawing geometry to "reach out" to the desired tail slot (*x*, *y*, *z*) offset.
10..11	63:62	**Reserved** Exists If: ([Surface Format] == 'PLANAR') Format: MBZ
	63:12	**Auxiliary Surface Base Address**

RENDER_SURFACE_STATE

		Exists If: ([Surface Format] != 'PLANAR') AND [Memory Compression Enable] == 0 Format: GraphicsAddress[63:12] Specifies the 4kbyte-aligned base address of the Auxiliary surface associated with the primary surface specified in other SURFACE_STATE fields.
	61:48	**X Offset for V Plane** Exists If: ([Surface Format] == 'PLANAR') Format: U14 This field specifies the horizontal offset in pixels from the **Surface Base Address** to the start (origin) of the V plane. **Programming Notes** This field must be a multiple of 4 (bits 1:0 MBZ). If **Tiled Resource Mode** is enabled, this field must be a multiple of the tile width in pixels. This field is ignored if **Separate UV Plane Enable** is disabled.
	47:46	**Reserved** Exists If: ([Surface Format] == 'PLANAR') Format: MBZ
	45:32	**Y Offset for V Plane** Exists If: ([Surface Format] == 'PLANAR') Format: U14 This field specifies the vertical offset in rows from the **Surface Base Address** to the start (origin) of the V plane. **Programming Notes** For surfaces where **Surface Format** = PLANAR* and **Separate UV Plane** is Enabled, the Y Offset must be programmed in multiples of **half-rows**. For example, for a surface where Y is physically followed by U and then V in memory, the Y Offset to V plane would be (2*Y-Height+ U-Height). For all other PLANAR YUV formats this is programmed in multiples of full rows (e.g Y-Height + U-Height). For all format besides PLANAR_420_* This field must be a multiple of 4 (bits 1:0 MBZ). For formats PLANAR_420_* this field must be multiple of 4 if U plane is the first chroma plane after the Y (luma) plane. It can be a multiple of 2 if it is the second chroma plane. For formats PLANAR_420_* when this field is not a multiple of 4 the Out-of-Bounds Suppression check must be disabled to avoid false out of bound detection. If **Tiled Resource Mode** is enabled, this field must be a multiple of the tile height in rows. This field is ignored if **Separate UV Plane Enable** is disabled.
	31:21	**Auxiliary Table Index for Media Compressed Surface** Exists If: [Memory Compression Enable] ==1 This field is valid only if Media Memory Compression is on for the surface(Memory Compression Enable == 1). In that case, the Auxiliary Surface Base address is never expected to be used and hence can be overloaded. This represents the 11 bit index into the table in memory which maps the surface to the auxiliary base address.

RENDER_SURFACE_STATE

	Bits	Description
	11	**Reserved** Format: MBZ
	10	**Clear Value Address Enable** Format: Enable This field enables HW Managed Clear Value Layout for the Surface State. If this bit is enabled, Clear Value Address is present instead of explicit clear values.

Value	Name	Description
0h	Disable	Clear values are present in the surface state explicitly.
1h	Enable	Clear value Address is present instead of explicit clear values.

Programming Notes
If this bit is cleared, then no clear value is being used for the surface. In this case, 3D Sampler will not fetch any clear value from memory and it is assumed that the AUX_CCS auxiliary surface will never indicate the clear state for this surface.
This field must be enabled to program the discrad bit. If this field is not enabled, HW does not discard the color surfaces during the Tile Pass.

9:5 Quilt Height

Format:	U5

This field specifies the height of a quilted texture in units of quilt slices. Refer to the section on Quilted Textures for more details.

Value	Name	Description
[0,31]		representing height of quilt - 1 (y/v dimension)

Programming Notes

Programming Notes

- Only power-of-2 **Quilt Height** and **Quilt Width** values are allowed: (1,2,4,8,16,32) mapping to (0,1,3,7,15,31) values in the fields.
- A surface is defined as a quilted texture if either **Quilt Height** or **Quilt Width** is nonzero (actual field value, not the incremented value).
- A quilted texture
  - is only supported by the sampling engine (other shared functions will ignore the **Quilt Width** and **Quilt Height** field, behaving as if they are set to zero).
  - must have a **Surface Type** of SURFTYPE_2D.
  - must have **Number of Multisamples** set to NUMSAMPLES_1.
  - must have **Vertical Line Stride** set to 0.
  - must have **Auxiliary Surface Mode** set to AUX_NONE.
  - **Depth** indicates the array dimension of the quilted texture if **Surface Array** is enabled. The valid range of **Depth** is [0, 2048 / (QuiltWidth * QuiltHeight) - 1], i.e. the total number of underlying array slices including quilt slices cannot exceed 2048.

RENDER_SURFACE_STATE

		• cannot be accessed with any ld* message type or using a sampler with the **Non-Normalized Coordinate Enable** field enabled.
	4:0	**Quilt Width** Format: U5 This field specifies the width of a quilted texture in units of quilt slices. Refer to the section on Quilted Textures for more details. **Value** | **Name** | **Description** [0,31] | | representing width of quilt - 1 (x/u dimension)
12	31:6	**Clear Color Address** Exists If: [Clear Value Address Enable] == 'Enable'] Format: GraphicsAddress[31:6]SurfaceState **Description** Specifies the lower bits of Graphics Address where clear value is stored in.This address is also used to store the per surface discard bit in PTBR mode The memory layout of the clear color pointed to by this address is a value stored in the lower-order bytes of a 64-byte cache-line. The data will be formatted as 32-bit IEEE Floating-point per channel,32-bit UINT per channel,32-bit SINT per channel, or SRGB depending on the surface type (e.g. R32G32B32A32_UINT surfaces assume use 32-bit UINT for clear color). These supported formats are identical the definition for **Red Clear Color** field defined in the RENDER_SURFACE_STATE. For D24X8 depth surfaces (R24_UNORM_X8_TYPELESS), the format of the data at this location shall be UNORM24_X8 rather than a 32-bit format.
	31:6	**Clear Depth Address Low** Exists If: ([Auxiliary Surface Mode] == 'AUX_HIZ') AND ([Clear Value Address Enable] == 'Enable') Format: GraphicsAddress[31:6]SurfaceState **For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_HIZ:** Specifies the lower bits of Graphics Address where the depth clear value is stored. The memory format is IEEE 32 bit float. The numeric range is required to match the numeric range limitations of 3DSTATE_CLEAR_PARAMS:Depth Clear Value.
	31:6	**Reserved** Exists If: ([Auxiliary Surface Mode] != 'AUX_CCS_D') AND ([Auxiliary Surface Mode] != 'AUX_CCS_E') AND ([Auxiliary Surface Mode] != 'AUX_HIZ') Format: MBZ
	31:0	**Red Clear Color** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E' OR [Auxiliary Surface Mode] == 'AUX_HIZ') AND [Clear Value Address Enable] == 'Disable')

RENDER_SURFACE_STATE

		For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_CCS: Specifies the clear value for the red channel. **For Depth Buffer Surfaces with Auxiliary Surface Mode set to AUX_HIZ and Clear Value Address Enable set to 'Disable':** Specifies the depth clear value. **For Other Surfaces:** This field is ignored. **Programming Notes** Legacy clear color is deprecated. This field shall not be used to store color or depth clear values.
	5	**Clear Color Conversion Enable** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Enable') Enables Pixel backend hw to convert clear values into native format and write back to clear address, so that display and sampler can use the converted value for resolving fast cleared RTs

Value	Name	Description
1h	Enable **[Default]**	Enables Pixel backend hw to convert clear values into native format and write back to clear address, so that display and sampler can use the converted value for resolving fast cleared RTs
0h	Disable	Disable hw conversion and write back of clear value

	5:0	**Reserved** Exists If: ([Auxiliary Surface Mode] == 'AUX_HIZ') AND ([Clear Value Address Enable] == 'Enable') Format: MBZ
	5:0	**Reserved** Exists If: ([Auxiliary Surface Mode] != 'AUX_CCS_D') AND ([Auxiliary Surface Mode] != 'AUX_CCS_E') AND ([Auxiliary Surface Mode] != 'AUX_HIZ') Format: MBZ
	4:0	**Reserved** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Enable') Format: MBZ
13	31:16	**Reserved** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E' OR [Auxiliary Surface Mode] == 'AUX_HIZ') AND [Clear Value Address Enable] == 'Enable') Format: MBZ
	31:0	**Reserved** Exists If: [Auxiliary Surface Mode] == 'AUX_HIZ' Format: MBZ
	31:0	**Green Clear Color**

RENDER_SURFACE_STATE		
		Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Disable') **For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_CCS:** Specifies the clear value for the green channel. **For Other Surfaces:** This field is ignored. **Programming Notes** Legacy clear color is deprecated. This field shall not be used to store color clear values.
	31:0	**Reserved** Exists If: ([Auxiliary Surface Mode] != 'AUX_CCS_D') AND ([Auxiliary Surface Mode] != 'AUX_CCS_E') AND ([Auxiliary Surface Mode] != 'AUX_HIZ') Format: MBZ
	15:0	**Clear Color Address High** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Enable') Format: GraphicsAddress[47:32]SurfaceState **For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_CCS:** Specifies the higher bits of Graphics Address where clear value is stored from RGBA (R in the LSB and A in the MSB - in that order) **For Other Surfaces:** This field is ignored.
	15:0	**Clear Depth Address High** Exists If: ([Auxiliary Surface Mode] == 'AUX_HIZ') AND ([Clear Value Address Enable] == 'Enable') Format: GraphicsAddress[47:32]SurfaceState
14	31:0	**Reserved** Exists If: [Auxiliary Surface Mode] == 'AUX_HIZ' Format: MBZ
	31:0	**Blue Clear Color** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Disable') **For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_CCS:** Specifies the clear value for the green channel. **For Other Surfaces:** This field is ignored. **Programming Notes** Legacy clear color is deprecated. This field shall not be used to store color clear values.
	31:0	**Reserved** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Enable') Format: MBZ
15	31:0	**Reserved** Exists If: [Auxiliary Surface Mode] == 'AUX_HIZ'

RENDER_SURFACE_STATE

		Format: MBZ
	31:0	**Alpha Clear Color** Exists If: (([Auxiliary Surface Mode] == 'AUX_CCS_D' OR [Auxiliary Surface Mode] == 'AUX_CCS_E') AND [Clear Value Address Enable] == 'Disable') **For Sampling Engine Surfaces and Render Targets with Auxiliary Surface Mode set to AUX_CCS:** Specifies the clear value for the green channel. **For Other Surfaces:** This field is ignored. **Programming Notes** Legacy clear color is deprecated. This field shall not be used to store color clear values.

Render Data Port Message Types

MT_DP_RT - Render Data Port Message Types		
Source:	EuSubFunctionRenderDataPort	
Size (in bits):	5	
Default Value:	0x0000000C	
Lists all the Message Types in a Render Data Port Message Descriptor [18:14].		

DWord	Bit	Description
0	4	**Reserved** Format: MBZ Ignored
	3:0	**Message Type** Specifies type of message

Value	Name	Description
0Ch	MT_RTW **[Default]**	Render Target Write message
0Dh	MT_RTR	Render Target Read message
Others	Reserved	Ignored

Render Engine Interrupt Vector

RENDER_INTR_VEC - Render Engine Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	15	**Catastrophic Error** This interrupt signals that a unrecoverable errorduring the engine processing. When Memory interface signals this error, the Command Streamer will stop parsing any more instructions. Scheduler is expected to reset the engine to evict the context
	14	**EU Restart Interrupt**
	13	**Spare 13**
	12	**Spare 12**
	11	**CS Wait On Semaphore**
	10	**Spare 10**
	9	**CS TR Invalid Tile Detection**
	8	**CS Context Switch Interrupt**
	7	**Legacy Context Per Process Page Fault Interrupt** This Fault interrupt is only delivered to the Host SW. Fault interrupt is generated by GA fabric, not by the CS This interrupt is for handling Legacy Page Fault. When Fault Repair Mode is enabled, Interrupt mask register value is not looked at to generate interrupt due to page fault. Please refer to vol1c "page fault support" section for more details.
	6	**CS Watchdog Counter Expired**
	5	**Spare 5**
	4	**CS PIPE_CONTROL Notify**
	3	**CS Error Interrupt**
	2	**Spare 2**
	1	**Reserved**
	0	**CS MI User Interrupt**

Render Target Index Message Header Control

MHC_RT_RTI - Render Target Index Message Header Control

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

DWord	Bit	Description
0	31:3	**Reserved** Format: Ignore Ignored
	2:0	**Render Target Index** Format: U3 Specifies the render target index that will be used to select blend state from BLEND_STATE.

Render Target Message Header

MH_RT - Render Target Message Header		
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
0.0-0.0	31:0	**Render Target Controls 0** Format: **MHC_RT_C0** Specifies controls for Render Target Write and Read messages.
0.1-0.1	31:0	**Color Calculator State Pointer** Format: **MHC_RT_CCSP** For Render Target Write message, specifies the HWORD-aligned GeneralStateOffset for Color State. Ignored by Render Target Read message.
0.2-0.2	31:0	**Render Target Index** Format: **MHC_RT_RTI** For Render Target Write message, specifies the render target index used to select blend state from BLEND_STATE. Ignored by Render Target Read message.
0.3-0.4	63:0	**Reserved** Format: Ignore Ignored
0.5-0.5	31:0	**Color Code** Format: **MHC_RT_CC** Hardware uses to track synchronizing events and free resources on thread completion.
0.6-0.7	63:0	**Reserved** Format: Ignore Ignored

MH_RT - Render Target Message Header

1.0-1.0	31:0	**Reserved** Format: Ignore Ignored
1.1-1.1	31:0	**Reserved** Format: Ignore Ignored
1.2-1.2	31:0	**Subspan 0** Format: MHC_RT_SUBSPAN Upper left corner of subspan 0
1.3-1.3	31:0	**Subspan 1** Format: MHC_RT_SUBSPAN Upper left corner of subspan 1
1.4-1.4	31:0	**Subspan 2** Format: MHC_RT_SUBSPAN Upper left corner of subspan 2
1.5-1.5	31:0	**Subspan 3** Format: MHC_RT_SUBSPAN Upper left corner of subspan 3
1.6-1.6	31:0	**Reserved** Format: Ignore Ignored
1.7-1.7	31:0	**Pixel Sample Enables** Format: MHC_RT_PSM

MH_RT - Render Target Message Header		
		Pixel Sample Enables

Render Target Message Header Control

MHC_RT_C0 - Render Target Message Header Control

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

DWord	Bit	Description
0	31	**Reserved** Format: Ignore Ignored
	30:27	**Viewport Index** Format: U4 For Render Target Write message, specifies the index of the viewport currently being used. Range = [0,15] Ignored by Render Target Read message.
	26:16	**Render Target Array Index** Format: U11 Specifies the array index to be used for the following surface types: SURFTYPE_1D: specifies the array index. Range = [0,511] SURFTYPE_2D: specifies the array index. Range = [0,511] SURFTYPE_3D: specifies the Z or R coordinate. Range = [0,2047] SURFTYPE_BUFFER: must be zero. SURFTYPE_CUBE: specifies the face identifier. Mapping (0,+x) (1,-x) (2,+y) (3,-y) (4,+z) (5,-z). **Programming Notes** The Render Target Array Index used by hardware for access to the Render Target is overridden with the Minimum Array Element defined in SURFACE_STATE if it is out of the range between Minimum Array Element and Depth. For cube surfaces, a depth value of 5 is used for this determination. SW must use the EXT_FUNC_CTRL on side-band to avoid sending header, when either this bit is set or Render Target Index needs to be programmed. The typical use case of Multi-Render Target Write messages requires setting these bit fields and avoiding to send header improves HW performance.
	15	**Front/Back Facing Polygon** Format: U1 Determines whether the polygon is front or back facing. Used by the render cache to determine which stencil test state to use. Value | Name | Description 0h | Front facing | All 1h | Back facing | All
	14	**Stencil Present to Render Target**

MHC_RT_C0 - Render Target Message Header Control

		Format: Enable For Render Target Write message, indicates that computed stencil is included in the message. Must be zero for Render Target Read message.
	13	**Source Depth Present to Render Target** Format: Enable For Render Target Write Message, indicates that source depth data is included in the message. Must be zero for Render Target Read message.
	12	**oMask to Render Target** Format: Enable For Render Target Write message, indicates that oMask data is present in the message and is to be used to mask off samples. Must be zero for Render Target Read message.
	11	**Source0 Alpha Present to Render Target** Format: Enable For Render Target Write message, indicates that Source0 Alpha (aka o0.a) data is included in RTWrite message. If present, these alpha values are used as inputs to AlphaTest and AlphaToCoverage functions. This is required to meet the API rules when writing to multiple render targets (MRTs). Must be zero for Render Target Read message. **Programming Notes** This bit should not be set when write to RT0, though sending and using redundant alpha will provide the correct results (at lower performance). This bit is not supported on Dual-Source Blend message types, as source0 alpha is already included in those messages. This bit is not supported on replicated data message types. SW must use the EXT_FUNC_CTRL on side-band to avoid sending header, when either this bit is set or Render Target Index needs to be programmed. The typical use case of Multi-Render Target Write messages requires setting these bit fields and avoiding to send header improves HW performance.
	10	**Reserved** Format: Ignore Ignored
	9:6	**Starting Sample Pair Index or Sample Index** Format: U4 When pixel shader is dispatched in per-sample mode or per-pixel mode with Per-Sample PS Enable bit cleared, this field indicates the index of the first sample pair of the dispatch. Range =

MHC_RT_C0 - Render Target Message Header Control		
		[0,7]. When pixel shader is dispatched in per-pixel mode with Per-Sample PS Enable bit set, this field indicates the index of a sample referenced by per-sample RT read or RT write messages. Range = [0, 15].
	5:0	**Reserved** Format: Ignore Ignored

Replicated Pixel Render Target Data Payload Register

MDPR_RGBA - Replicated Pixel Render Target Data Payload Register

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

DWord	Bit	Description
0	31:0	**Red** Format: U32 Specifies the value of all slots' red channel.
1	31:0	**Green** Format: U32 Specifies the value of all slots' green channel.
2	31:0	**Blue** Format: U32 Specifies the value of all slots' blue channel.
3	31:0	**Alpha** Format: U32 Specifies the value of all slots' alpha channel.
4..7	127:0	**Reserved** Format: Ignore Ignored

Replicated SIMD16 Render Target Data Payload

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]

Reversed SIMD Mode 2 Message Descriptor Control Field

MDC_SM2R - Reversed SIMD Mode 2 Message Descriptor Control Field

Source:	BSpec
Size (in bits):	1
Default Value:	0x00000000

DWord	Bit	Description
0	0	**SIMD Mode** Specifies the SIMD mode of the message (number of slots processed)

Value	Name	Description
00h	SIMD16	SIMD16
01h	SIMD8	SIMD8

RoundingPrecisionTable_3_Bits

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

S0A SIMD8 Render Target Data Payload

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

S0A SIMD16 Render Target Data Payload

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]**

MDP_RTW_A16 - S0A SIMD16 Render Target Data Payload		
		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]** 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_BORDER_COLOR_STATE

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

Description

The interpretation of the border color depends on the Texture Border Color Mode field in SAMPLER_STATE as follows:

- 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.
- DX10/OGL mode:the format of the border color depends on the format of the surface being sampled. If the map format is UINT, then the border color format is R32G32B32A32_UINT. If the map format is SINT, then the border color format is R32G32B32A32_SINT. Otherwise, the border color format is R32G32B32A32_FLOAT.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.

If the Texture Border Color Mode field in SAMPLER_STATE is set to DX9 and theMMIO register bit " Enable Missing Alpha Format Fix" is set in register E194h, then the interpretation of the border color format depends of the format of the surface being sampled:

1. If the map format is UINT, border color is R8G8B8A8_UINT
2. If the map format is SINT, border color is R8G8B8A8_SINT
3. Otherwise, border color is R8G8B8A8_UNORM

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.

- The border color itself is accessed through the texture cache hierarchy rather than the state cache

SAMPLER_BORDER_COLOR_STATE

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.

If the Texture Border Color Mode field in SAMPLER_STATE is set to DX9 and the MMIO register bit " Enable Missing Alpha Format Fix" is set in register E194h and a surface with format SINT is being sampled, then each channel of the border color must be 00h or 01h (0 or 1 in SINT encoding).

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
	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
	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 Texture Border Color Mode = DX9
	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**

SAMPLER_BORDER_COLOR_STATE

		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_INDIRECT_STATE_BORDER_COLOR

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

Description

This structure is a one version of the SAMPLER_INDIRECT_STATE structure, suitable for many needs.
An instance of this structure is pointed to by the **Indirect State Pointer** field in SAMPLER_STATE.
The interpretation of the border color depends on the **Texture Border Color Mode** field in SAMPLER_STATE as follows:

- In **8BIT** 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 **OGL** mode, the format of the border color is R32G32B32A32_FLOAT, R32G32B32A32_SINT, or R32G32B32A32_UINT, depending on 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.

Programming Notes

- 8BIT 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 As U8** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: U8
	31:0	**Border Color Red As S31**

SAMPLER_INDIRECT_STATE_BORDER_COLOR

		Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == 'OGL' AND (Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: S31 Format: U32 Format: IEEE Float
	23:16	**Border Color Blue As U8** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: U8
	15:8	**Border Color Green As U8** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: U8
	7:0	**Border Color Red As U8** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: U8
1	31:0	**Reserved** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: MBZ
	31:0	**Border Color Green As S31** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == 'OGL' AND (Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: S31 Format: U32 Format: IEEE Float
2	31:0	**Reserved** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: MBZ
	31:0	**Border Color Blue As S31** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == 'OGL' AND (Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: S31

<table>
<tr><th colspan="3">SAMPLER_INDIRECT_STATE_BORDER_COLOR</th></tr>
<tr><td></td><td></td><td><table><tr><td>Format:</td><td>U32</td></tr><tr><td>Format:</td><td>IEEE Float</td></tr></table></td></tr>
<tr><td rowspan="2">3</td><td>31:0</td><td>Reserved<table><tr><td></td><td></td></tr><tr><td>Exists If:</td><td>//Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT'</td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>31:0</td><td>Border Color Alpha As S31<table><tr><td></td><td></td></tr><tr><td>Exists If:</td><td>//Structure[SAMPLER_STATE][Texture Border Color Mode] == 'OGL' AND (Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true'</td></tr><tr><td>Format:</td><td>S31</td></tr><tr><td>Format:</td><td>U32</td></tr><tr><td>Format:</td><td>IEEE Float</td></tr></table></td></tr>
</table>

SAMPLER_INDIRECT_STATE

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

Description

Note: There are three variations of this structure, defined separately because their payloads have different lengths. Currently only SAMPLER_INDIRECT_STATE_BORDER_COLOR is fully defined.
This structure is pointed to by **Indirect State Pointer** (SAMPLER_STATE).
The interpretation of the border color depends on the **Texture Border Color Mode** field in SAMPLER_STATE as follows:

- In **8BIT** 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 **OGL** mode, the format of the border color is R32G32B32A32_FLOAT, R32G32B32A32_SINT, or R32G32B32A32_UINT, depending on 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

- 8BIT 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.

- 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

SAMPLER_INDIRECT_STATE		
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.		
DWord	**Bit**	**Description**
0	31:24	**Border Color Alpha** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: UNORM8 Texture Border Color Mode = 8BIT
	31:0	**Border Color Red** Exists If: //Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: SINT32 (2's complement) for all SINT surface formats Format: UINT32 for all UINT surface formats Format: IEEE_FP for all other surface formats
	23:16	**Border Color Blue** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: UNORM8 Texture Border Color Mode = 8BIT
	15:8	**Border Color Green** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: UNORM8 Texture Border Color Mode = 8BIT
	7:0	**Border Color Red** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: UNORM8 Texture Border Color Mode = 8BIT
1	31:0	**Reserved** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: MBZ
	31:0	**Border Color Green**

SAMPLER_INDIRECT_STATE		
		Exists If: //Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: IEEE_FP Format: S31 Format: U32
2	31:0	**Reserved** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: MBZ
	31:0	**Border Color Blue** Exists If: //Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: IEEE_FP Format: S31 Format: U32
3	31:0	**Reserved** Exists If: //Structure[SAMPLER_STATE][Texture Border Color Mode] == '8BIT' Format: MBZ
	31:0	**Border Color Alpha** Exists If: //Structure[RENDER_SURFACE_STATE][Surface Format]Property[IsSigned]=='true' Format: IEEE_FP Format: S31 Format: U32
4..15	383:0	**Reserved**

SAMPLER_STATE_8x8_1D_CONVOLVE

SAMPLER_STATE_8x8_1D_CONVOLVE

Source:	BSpec
Exists If:	//(Function==00001b && 1D Vertical Convolve) || (Function==1001b && 1D Horizontal Convolve)
Size (in bits):	224
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0	31:24	**Reserved** Format: MBZ
	23:20	**MSB WIDTH**
	19:16	**MSB HEIGHT**
	15:13	**Reserved** Format: MBZ
	12	**Size of the Coefficient**

MSB WIDTH (bits 23:20)

Value	Name	Description
1		Extends the Filter Size Width upto 31.
0		No Change to the Filter Size

Programming Notes
For 1D Vertical Convolve this should always be 0.

MSB HEIGHT (bits 19:16)

It contains the MSB HEIGHT of the kernel and is used to extend the kernel width range to 31. Used along with bits[3:0] which represents the LSB for the kernel Height.

Value	Name	Description
1h		Extends the filter size height upto 31.

Programming Notes
For 1D Horizontal Convolve this should always be 0.

Size of the Coefficient (bit 12)

Value	Name	Description
0	8bit	The lower 8 bits of the accumulator is forced to zero or ignored during the accumulation operation.
1	16bit	The lower 8 bits are also included for the operation. The final result of the accumulator is shifted before clamping the result as specified by the Scale down value.: Result[15:0] = Clamp(Accum[40:12] » scale_down)

<table>
<tr><th colspan="3">SAMPLER_STATE_8x8_1D_CONVOLVE</th></tr>
<tr><td></td><td>11:8</td><td>Scale down value
<table><tr><td>Exists If:</td><td>//Convolve Only</td></tr></table>
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>[0,10]</td><td></td><td>The final result is shifted by this value before clamp is done.</td></tr></table></td></tr>
<tr><td></td><td>7:4</td><td>WIDTH
It contains the WIDTH of the kernel.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>[2-15]</td><td></td></tr></table>
<table><tr><th>Programming Notes</th></tr><tr><td>For 1D Vertical Convolve this should always be 0.</td></tr></table></td></tr>
<tr><td></td><td>3:0</td><td>HEIGHT
It contains the HEIGHT of the kernel.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>[2-15]</td><td></td></tr></table>
<table><tr><th>Programming Notes</th></tr><tr><td>For 1D Horizontal Convolve this should always be 0.</td></tr></table></td></tr>
<tr><td>1..15</td><td>31:0</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>16</td><td>31:16</td><td>Filter Coefficient[1]
<table><tr><td>Exists If:</td><td>//Filtering Operation</td></tr><tr><td>Format:</td><td>S3.4(8bit)/S3.12(16bit) in 2's Complement</td></tr></table>
Range: [-8.0, +8.0)
<table><tr><th>Programming Notes</th></tr><tr><td>If not used in the filtering operation, must be zero.</td></tr></table></td></tr>
<tr><td></td><td>15:0</td><td>Filter Coefficient[0]
<table><tr><td>Exists If:</td><td>//Filtering Operation</td></tr><tr><td>Format:</td><td>S3.4(8bit)/S3.12(16bit) in 2's Complement</td></tr></table>
Range: [-8.0, +8.0)
<table><tr><th>Programming Notes</th></tr><tr><td>If not used in the filtering operation, must be zero.</td></tr></table></td></tr>
<tr><td>17</td><td>31:16</td><td>Filter Coefficient[3]
<table><tr><td>Exists If:</td><td>//Filtering Operation</td></tr><tr><td>Format:</td><td>S3.4(8bit)/S3.12(16bit) in 2's Complement</td></tr></table>
Range: [-8.0, +8.0)
<table><tr><th>Programming Notes</th></tr></table></td></tr>
</table>

<table>
<tr><th colspan="3">SAMPLER_STATE_8x8_1D_CONVOLVE</th></tr>
<tr><td></td><td></td><td>If not used in the filtering operation, must be zero.</td></tr>
<tr><td></td><td>15:0</td><td>Filter Coefficient[2]
Exists If: //Filtering Operation
Format: S3.4(8bit)/S3.12(16bit) in 2's Complement
Range: [-8.0, +8.0)
Programming Notes
If not used in the filtering operation, must be zero.</td></tr>
<tr><td>18..19</td><td>31:0</td><td>Filter Coefficient[7:4]
This table has the same layout as shown above.</td></tr>
<tr><td>20..23</td><td>31:0</td><td>Filter Coefficient[15:8]
This table has the same layout as shown above.</td></tr>
<tr><td>24..31</td><td>31:0</td><td>Filter Coefficient[31:16]
This table has the same layout as shown above.</td></tr>
</table>

intel

SAMPLER_STATE_8x8_AVS_COEFFICIENTS

SAMPLER_STATE_8x8_AVS_COEFFICIENTS

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

Description
ExistsIf = AVS && (Function_mode = 0)

DWord	Bit	Description
0	31:24	**Table 0Y Filter Coefficient[n,1]** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**Table 0X Filter Coefficient[n,1]** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**Table 0Y Filter Coefficient[n,0]** Format: S1.6 2's Complement **Range:** [-2, +2) **Programming Notes** If the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM, this field MBZ.
	7:0	**Table 0X Filter Coefficient[n,0]** Format: S1.6 2's Complement **Range:** [-2, +2) **Programming Notes** If the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM, this field MBZ.
1	31:24	**Table 0Y Filter Coefficient[n,3]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**Table 0X Filter Coefficient[n,3]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	15:8	**Table 0Y Filter Coefficient[n,2]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)

SAMPLER_STATE_8x8_AVS_COEFFICIENTS

	7:0	**Table 0X Filter Coefficient[n,2]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
2	31:24	**Table 0Y Filter Coefficient[n,5]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**Table 0X Filter Coefficient[n,5]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	15:8	**Table 0Y Filter Coefficient[n,4]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0) **Programming Notes** If the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM, this field MBZ.
	7:0	**Table 0X Filter Coefficient[n,4]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0) **Programming Notes** If the format is R10G10B10A2_UNORM or R8G8B8A8_UNORM, this field MBZ.
3	31:24	**Table 0Y Filter Coefficient[n,7]** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**Table 0X Filter Coefficient[n,7]** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**Table 0Y Filter Coefficient[n,6]** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**Table 0X Filter Coefficient[n,6]** Format: S1.6 2's Complement **Range:** [-2, +2)

SAMPLER_STATE_8x8_AVS_COEFFICIENTS		
4	31:24	**Table 1X Filter Coefficient[n,3]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**Table 1X Filter Coefficient[n,2]** Format: S1.6 2's Complement **Description** **Range:** [-2.0, +2.0)
	15:0	**Reserved** Format: MBZ
5	31:16	**Reserved** Format: MBZ
	15:8	**Table 1X Filter Coefficient[n,5]** Format: S1.6 2's Complement **Description** **Range:** [-2.0, +2.0)
	7:0	**Table 1X Filter Coefficient[n,4]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
6	31:24	**Table 1Y Filter Coefficient[n,3]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**Table 1Y Filter Coefficient[n,2]** Format: S1.6 2's Complement **Description** **Range:** [-2.0, +2.0)
	15:0	**Reserved** Format: MBZ
7	31:16	**Reserved** Format: MBZ
	15:8	**Table 1Y Filter Coefficient[n,5]** Format: S1.6 2's Complement

SAMPLER_STATE_8x8_AVS_COEFFICIENTS		
		Description **Range:** [-2.0, +2.0)
	7:0	**Table 1Y Filter Coefficient[n,4]** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)

SAMPLER_STATE_8x8_AVS

SAMPLER_STATE_8x8_AVS	
Source:	BSpec
Size (in bits):	8960
Default Value:	0x0294806C, 0x00000000, 0x39CFD1FF, 0x839F0000, 0x9A6E4000, 0x02601180, 0xFFFE2F2E, 0x00000000, 0xD82E0000, 0x8285ECEC, 0x00008282, 0x00000000, 0x02117000, 0xA38FEC96, 0x00008CC8, 0x00000000,

SAMPLER_STATE_8x8_AVS

	0x00000000, 0x00000000

Description
ExistsIf = AVS && (Function_mode = 0)

DWord	Bit	Description
0	31:28	**Reserved** Format: MBZ
	27:23	**R3c Coefficient** Default Value: 5 Format: U0.5
	22:18	**R3x Coefficient** Default Value: 5 Format: U0.5
	17:12	**Strong Edge Threshold** Default Value: 8 Format: U6 If EM > **Strong Edge Threshold**, the basic VSA detects a strong edge.
	11:6	**Weak Edge Threshold** Default Value: 1 Format: U6 If **Strong Edge Threshold** > EM > **Weak Edge Threshold**, the basic VSA detects a weak edge.
	5:0	**Gain Factor** Default Value: 44 Format: U6 User control sharpening strength

intel

SAMPLER_STATE_8x8_AVS		
1	31:0	**Reserved** Format: MBZ
2	31:27	**R5c Coefficient** Default Value: 7 Format: U0.5
	26:22	**R5cx Coefficient** Default Value: 7 Format: U0.5
	21:17	**R5x Coefficient** Default Value: 7 Format: U0.5
	16:14	**Strong Edge Weight** Default Value: 7 Format: U3 Sharpening strength when a strong edge is found in basic VSA.
	13:11	**Regular Weight** Default Value: 2 Format: U3 Sharpening strength when a weak edge is found in basic VSA.
	10:8	**Non Edge Weight** Default Value: 1 Format: U3 Sharpening strength when no edge is found in basic VSA.
	7:0	**Global Noise Estimation** Default Value: 255

SAMPLER_STATE_8x8_AVS

		Format: U8 Global noise estimation of previous frame.
3	31	**Reserved**
	30	**Reserved**
	29:28	**Enable 8-tap filter**
	27:22	**Hue_Max**
	21:16	**Sat_Max**

Enable 8-tap filter (29:28)

Description
Adaptive Filtering (Mode = 11) ExistsIf: R10G10B10A2_UNORM R8G8B8A8_UNORM (AYUV also) R8B8G8A8_UNORM B8G8R8A8_UNORM R16G16B16A16
Enable 8-tap Filtering on UV channel (Mode = 10) ExistsIf: R16B16_UNORM, R16_UNORM
Enable 8-tap Filtering on UV channel (Mode = 10) ExistsIf: R10G10B10A2_UNORM R8G8B8A8_UNORM (AYUV also) R8B8_UNORM (CrCb) R8_UNORM R8B8G8A8_UNORM B8G8R8A8_UNORM R16G16B16A16 Y8_UNORM

Value	Name	Description
00b		4-tap filter is only done on all channels.
01b		Enable 8-tap Adaptive filter on G-channel. 4-tap filter on other channels.
10b		8-tap filter is done on all channels (UV-ch uses the Y-coefficients)
11b		Enable 8-tap Adaptive filter all channels (UV-ch uses the Y-coefficients).

Programming Notes
For 00 and 10, are applicable for RGB surfaces only or surface without Y-ch. In case it is a YUV surface it will default to adaptive mode automatically which is 01 and 11 respectively. Alpha channel is always bi-linear filter irrespective of the above modes.
Mode 01 and 00 are legacy support and are supported on all surface formats.
When Mode is 10 and Surface format is Y8_UNORM, Bypass X/Y Adaptive Filtering must be 1, and Default Sharp Level must be 255

Hue_Max (27:22)

Default Value:	14
Format:	U6

Rectangle half width.

Sat_Max (21:16)

Default Value:	31
Format:	U6

SAMPLER_STATE_8x8_AVS

		Rectangle half length
	15:8	**Cos(alpha)** Format: S0.7 2's Complement Deafult Value: 79/128
	7:0	**Sin(alpha)** Format: S0.7 2's Complement Deafult Value: 101/128
4	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	**VY_STD_Enable** Format: Enable Enables STD in the VY subspace.
	14:12	**Diamond Margin** Default Value: 4 Format: U3
	11	**Shuffle_OutputWriteback for sample_8x8**

Value	Name	Description
0		Writeback same as Original Sample_8x8
1		Writeback of Sample_8x8 Is Modified to Suite Sample_Unorm

SAMPLER_STATE_8x8_AVS

DWord	Bit	Description
	10:0	**S3U** Format: S2.8 2's Complement Deafult Value: 0/256
5	31	**SkinDetailFactor** Format: S0 **Value** | **Name** | **Description** 1 | | sign(SkinDetailFactor) is equal to +1, and the content of the detected skin tone area is not detail revealed. 0 | | sign(SkinDetailFactor) is equal to -1, and the content of the detected skin tone area is detail revealed.
	30:24	**Diamond_du** Default Value: 2 Format: S6 2's Complement Rhombus center shift in the sat-direction, relative to the rectangle center.
	23:21	**HS_margin** Default Value: 3 Format: U3 Defines rectangle margin
	20:13	**Diamond_alpha** Format: U2.6 Deafault Value: 100/64 1 / tan(β)
	12:7	**Diamond_Th** Default Value: 35 Format: U6 Half length of the rhombus axis in the sat-direction.
	6:0	**Diamond_dv**

SAMPLER_STATE_8x8_AVS

DWord	Bit	Description
		Default Value: 0 Format: S6 2's Complement Rhombus center shift in the hue-direction, relative to the rectangle center.
6	31:24	**Y_point_4** Default Value: 255 Format: U8 Fourth point of the Y piecewise linear membership function.
	23:16	**Y_point_3** Default Value: 254 Format: U8 Third point of the Y piecewise linear membership function.
	15:8	**Y_point_2** Default Value: 47 Format: U8 Second point of the Y piecewise linear membership function.
	7:0	**Y_point_1** Default Value: 46 Format: U8 First point of the Y piecewise linear membership function.
7	31:16	**Reserved** Format: MBZ
	15:0	**INV_Margin_VYL** Format: U0.16 1/Margin_VYL = 3300/65536
8	31:24	**P1L**

SAMPLER_STATE_8x8_AVS

DWord	Bit	Description
		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	**INV_Margin_VYU** 1/Margin_VYU = 1600/65536
9	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** Default Value: 236 Format: U8 Y Point 2 of the lower part of the detection PWLF.

SAMPLER_STATE_8x8_AVS		
10	31:27	**Y_Slope_2** Format: U2.3 Deafault Value: 31/8 Slope between points Y3 and Y4.
	26:16	**S0L** Format: S2.8 2's Complement Deafault Value: -5/256 Slope 0 of the lower part of the detection PWLF.
	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
11	31:22	**Reserved** Format: MBZ
	21:11	**S2L** Format: S2.8 2's Complement Default Value: 0/256 Slope 2 of the lower part of the detection PWLF.
	10:0	**S1L** Format: S2.8 2's Complement Default Value: 0/256 Slope 1 of the lower part of the detection PWLF.
12	31:27	**Y_Slope1**

SAMPLER_STATE_8x8_AVS

DWord	Bit	Description
		Format: U2.3 Default Value: 31/8 Slope between points Y1 and Y2.
	26:19	**P1U** Default Value: 66 Format: U8 Y Point 1 of the upper part of the detection PWLF.
	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 Default Value: 0/256 Slope 3 of the lower part of the detection PWLF.
13	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

SAMPLER_STATE_8x8_AVS

		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.
14	31:27	**Reserved** Format: MBZ
	26:16	**S0U** Format: S2.8 2's Complement Default Value: 256/256 Slope 0 of the upper part of the detection PWLF.
	15:8	**B3U** Default Value: 140 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.
15	31:22	**Reserved** Format: MBZ
	21:11	**S2U** Format: S2.8 2's Complement Deafult Value: -179/256

SAMPLER_STATE_8x8_AVS

DWord	Bit	Description
		Slope 2 of the upper part of the detection PWLF.
	10:0	**S1U** Format: S2.8 2's Complement Deafult Value: 113/256 Slope 1 of the upper part of the detection PWLF.
16..151	4351:0	**Filter Coefficient[0..16]** Format: SAMPLER_STATE_8x8_AVS_COEFFICIENTS[17]
152	31:24	**Default Sharpness Level** Format: U8 When adaptive scaling is off, determines the balance between sharp and smooth scalers. Value / Name / Description: 0 / **[Default]** / Contribute 1 from the smooth scalar 255 / / 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 4 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.
153	31:23	**Reserved** Format: MBZ
	22	**Bypass X Adaptive Filtering**

<table>
<tr><th colspan="3">SAMPLER_STATE_8x8_AVS</th></tr>
<tr><td></td><td></td><td>
<table><tr><td>Format:</td><td>Disable</td></tr></table>
When disabled, the X direction will use Default Sharpness Level to blend between the smooth and sharp filters rather than the calculated value.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td>Disble</td><td>Disable X Adaptive Filtering</td></tr>
<tr><td>0</td><td>Enable</td><td>Enable X Adaptive Filtering</td></tr>
</table>
</td></tr>
<tr><td></td><td>21</td><td>
Bypass Y Adaptive Filtering
<table><tr><td>Format:</td><td>Disable</td></tr></table>
When disabled, the Y direction will use Default Sharpness Level to blend between the smooth and sharp filters rather than the calculated value.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td>Disble</td><td>Disable Y Adaptive Filtering</td></tr>
<tr><td>0</td><td>Enable</td><td>Enable Y Adaptive Filtering</td></tr>
</table>
</td></tr>
<tr><td></td><td>20:2</td><td>
Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td></td><td>1</td><td>
Adaptive Filter for all channels
<table><tr><td>Format:</td><td>Enable</td></tr></table>
Only to be enabled if 8-tap Adaptive filter mode is on, eElse it should be disabled.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td>Enable</td><td>Enable Adaptive Filter on UV/RB Channels</td></tr>
<tr><td>0</td><td>Disble</td><td>Disable Adaptive Filter on UV/RB Channels</td></tr>
</table>
</td></tr>
<tr><td></td><td>0</td><td>
RGB Adaptive
<table><tr><td>Format:</td><td>Enable</td></tr></table>
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.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>1</td><td>Enable</td><td>Enable the RGB Adaptive filter using the equation (Y=(R+2G+B)»2)</td></tr>
<tr><td>0</td><td>Disble</td><td>Disable the RGB Adaptive equation and use G-Ch directly for adaptive filter</td></tr>
</table>
</td></tr>
<tr><td>154..159</td><td>191:0</td><td>
Reserved
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td>160..279</td><td>3839:0</td><td>
Filter Coefficient[17..31]
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>SAMPLER_STATE_8x8_AVS_COEFFICIENTS[15]</td></tr></table>
</td></tr>
</table>

SAMPLER_STATE_8x8_CONVOLVE_COEFFICIENTS

SAMPLER_STATE_8x8_CONVOLVE_COEFFICIENTS		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31:16	**Filter Coefficient[0,1]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,0]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
1	31:16	**Filter Coefficient[0,3]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,2]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
2	31:16	**Filter Coefficient[0,5]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,4]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
3	31:16	**Filter Coefficient[0,7]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,6]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)

SAMPLER_STATE_8x8_CONVOLVE_COEFFICIENTS		
4	31:16	**Filter Coefficient[0,9]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,8]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
5	31:16	**Filter Coefficient[0,11]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,10]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
6	31:16	**Filter Coefficient[0,13]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,12]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
7	31:16	**Filter Coefficient[0,15]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)
	15:0	**Filter Coefficient[0,14]** Format: S3.4(8bit)/S3.12(16bit) in 2's Complement **Range:** [-8.0, +8.0)

SAMPLER_STATE_8x8_CONVOLVE

SAMPLER_STATE_8x8_CONVOLVE	
Source:	BSpec
Size (in bits):	16384
Default Value:	0x00000000, 0x00000000,

SAMPLER_STATE_8x8_CONVOLVE
0x00000000, 0x00000000
Description

SAMPLER_STATE_8x8_CONVOLVE

Function: 0001b ExistsIf: [Convolve] && [mode==0] && [(Kernel Size) =< (15x15)]
Function: 1010b ExistsIf: "[1Pixel Convolution] && [(Kernel Size) =< (15x15)]
Function: 0001b ExistsIf: [Convolve] && [mode==1] && [(Kernel Size) > (15x15)]

DWord	Bit	Description
0	31:21	**Reserved** Format: MBZ
	20	**MSB WIDTH** Exists If: //[Convolve] Only It contains the MSB Width of the kernel and is used to extend the kernel width range to 31. Used along with bits[3:0] which represents the LSB for the kernel Height. Value / Name / Description: 1 / EXTENDED / Extends the Filter Size Width upto 31. 0 / NO_CHANGE / No Change to the Filter Size
	19:17	**Reserved** Format: MBZ
	16	**MSB HEIGHT** It contains the MSB HEIGHT of the kernel and is used to extend the kernel width range to 31. Used along with bits[3:0] which represents the LSB for the kernel Height. Value / Name / Description: 1 / EXTENDED / Extends the filter size height upto 31. 0 / NO_CHANGE / No Change to the Filter Size
	15:13	**Reserved** Format: MBZ
	12	**Size of the Coefficient** Value / Name / Description: 0 / 8bit / The lower 8 bits of the accumulator is forced to zero or ignored during the accumulation operation. 1 / 16bit / The lower 8 bits are also included for the operation. The final result of the accumulator is shifted before clamping the result as specified by the Scale down value.: Result[15:0] = Clamp(Accum[40:12] » scale_down)
	11:8	**Scale down value** Exists If: //[Convolve] Only Value / Name / Description: [0,10] / / The final result is shifted by this value before clamp is done.
	7:4	**WIDTH**

SAMPLER_STATE_8x8_CONVOLVE		
		Exists If: //[Convolve] Only It contains the WIDTH of the kernel. **Value** / **Name** [2-15]
	3:0	**HEIGHT** Exists If: //[Convolve] Only It contains the HEIGHT of the kernel. **Value** / **Name** [2-15]
1..15	479:0	**Reserved** Format: MBZ
16..143	4095:0	**Filter Coefficient[15:0,15:0]** Exists If: //[Filtering] Operation Format: SAMPLER_STATE_8x8_CONVOLVE_COEFFICIENTS[16] Columns [15:0] of the coefficient containing 16 coefficients for [15:0] rows. **Programming Notes** Please note that this field is MBZ if not used in the Filtering Mode.
144..511	11775:0	**Filter Coefficient[15:0,15:0]to[30:0,31:0]** Format: SAMPLER_STATE_8x8_CONVOLVE_COEFFICIENTS[46] Expands Filter Coefficient[15:0,15:0] to account for columns [15:0] of the coefficient containing 16 coefficients for [31:16] rows and columns [30:16] of the coefficient containing 31 coefficients for [31:0] rows. **Programming Notes** Filter Coefficient beyond [15:0,15:0] are present only when Kernel size is greater than 15x15 otherwise it is not present.

SAMPLER_STATE_8x8_ERODE_DILATE_MINMAXFILTER

SAMPLER_STATE_8x8_ERODE_DILATE_MINMAXFILTER

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

Description
The table is valid for the following funstions: 0100 - Erode && (Function_mode==0) 0101 - Dilate && (Function_mode==0) 0011 - MinMaxFilter && (Function_mode==0)

Programming Notes
Max kernel size is 15x15. For sizes less than 15x15 the coefficients not used should be zeroed out.

DWord	Bit	Description
0	31:16	**16bit Mask for Row0 [15:0]**
	15:8	**Reserved** Format: MBZ
	7:4	**Width Of The Kernel** Value: 2-15 / Name: (blank)
	3:0	**Height Of The Kernel** Value: 2-15 / Name: (blank)
1	31:16	**16bit Mask for Row2 [15:0]**
	15:0	**16bit Mask for Row1 [15:0]**
2	31:16	**16bit Mask for Row4 [15:0]**
	15:0	**16bit Mask for Row3 [15:0]**
3	31:16	**16bit Mask for Row6 [15:0]**
	15:0	**16bit Mask for Row5 [15:0]**
4	31:16	**16bit Mask for Row8 [15:0]**
	15:0	**16bit Mask for Row7 [15:0]**
5	31:16	**16bit Mask for Row10 [15:0]**
	15:0	**16bit Mask for Row9 [15:0]**
6	31:16	**16bit Mask for Row12 [15:0]**
	15:0	**16bit Mask for Row11 [15:0]**
7	31:16	**16bit Mask for Row14 [15:0]**
	15:0	**16bit Mask for Row13 [15:0]**

SAMPLER_STATE

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.

DWord	Bit	Description
	30	**CPS LOD Compensation Enable**

Format:	Enable

This field, if enabled, causes derivatives used to compute LOD to be adjusted by scale factors for coarse pixel shading. The adjustment only occurs if the following are all true:

- This field is enabled
- **CPS Message LOD Compensation Enable** in the message header is enabled

The scale.x and scale.y factors are computed in hardware and delivered to the sampler at thread dispatch time.
The following adjustments generate new derivatives as follows:

$\frac{du}{dx} = \frac{du}{dx} *scale.x$	$\frac{dv}{dx} = \frac{dv}{dx} *scale.x$	$\frac{dr}{dx} = \frac{dr}{dx} *scale.x$
$\frac{du}{dy} = \frac{du}{dy} *scale.y$	$\frac{dv}{dy} = \frac{dv}{dy} *scale.y$	$\frac{dr}{dy} = \frac{dr}{dy} *scale.y$

DWord	Bit	Description
	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	OGL	New mode for interpreting the border color
1h	8BIT	Earlier mode for interpreting the border color

Programming Notes

SAMPLER_STATE

		SET0_LEGACY: Undefined behavior if DX9 border is used with any feature added. See Legacy sampler feature page for details.
		This field must not be set to DX9 if there are null tiles in use
		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 except when setting BORDER COLOR RED/GREEN/BLUE and ALPHA to 0
		This field must be set to DX10/OGL mode if REDUCTION_MINIMUM or REDUCTION_MAXIMUM or message type is sample_min or sample_max.
	28:27	**LOD PreClamp Mode**

This field determines whether the computed LOD is clamped to [max,min] mip level before the mag-vs-min determination is performed.

PRECLAMP_OGL: LOD pre-clamped to **Min LOD** and **Max LOD**

OpenGL API currently clamps LOD to the **Min LOD** and **Max LOD** (from Sampler State) prior to performing min/mag determination, and therefore it is expected that an OpenGL driver would need to set this field to PRECLAMP_OGL.

Value	Name	Description
0h	NONE	LOD PreClamp disabled
1h	Reserved	
2h	OGL	LOD PreClamp enabled (OGL mode)

26:22 Coarse LOD Quality Mode

Format:	U5

This field configures the coarse LOD image quality mode for the sample_d, sample_l, and sample_b messages in the sampling engine. In general, performance will increase and power consumption will decrease with each step of reduced quality (performance gain for sample_l and sample_b will be minimal).

Value	Name	Description
0h	Disabled	Full quality is enabled, matching prior products
01h-1Fh		Quality degrades with each larger value, performance improves with each larger value

Programming Notes
Although allowed, it is not recommended to program this field to a value greater than 17h to avoid masking the exponent which may generate incorrect LOD values.

SAMPLER_STATE

	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.
Mip Mode Filter must be set to MIPFILTER_NONE or MIPFILTER_NEAREST if Surface Format for the associated surface is UINT or SINT. However, all settings of this field are allowed with UINT/SINT if a minimum or maximum operation is being performed.
Mip Mode Filter must be set to MIPFILTER_NONE for Planar YUV surfaces.

	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
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	Reserved	
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_ANISOTROPIC may cause artifacts at cube edges if enabled for cube maps with the

SAMPLER_STATE

		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	Reserved	
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-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	**LOD algorithm** Format: U1 Enumerated Type Controls which algorithm is used for LOD calculation. Generally, the EWA approximation algorithm results in higher image quality than the legacy algorithm.

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

Programming Notes
The EWA Algorithm should only be enabled for Anisotropic Filtering modes. It must not be enabled for non-anisotropic filtering as the increased accuracy of the LOD calculation will is not

SAMPLER_STATE

		required and will increase the power and reduce overall efficiency.
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.
	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 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	**ChromaKey Enable** Format: Enable This field enables the chroma key function. **Programming Notes** upported only on a specific subset of surface formats. See section titled: "Surface Formats" in this section for supported formats. This field must be disabled if min or mag filter is MAPFILTER_ANISOTROPIC. This field must be disabled if used with a surface of type SURFTYPE_3D.
	6:5	**ChromaKey Index** Format: U2

SAMPLER_STATE

		Range: [0, 3] 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.
	4	**ChromaKey Mode**
	3:1	**Shadow Function**

ChromaKey Mode (bit 4)

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_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	Description
0h	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.
1h	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.

Shadow Function (bits 3:1)

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

![intel]

SAMPLER_STATE

DWord	Bit	Description
		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
2	31:24	**Reserved** Format: MBZ
	23:6	**Indirect State Pointer** Format: DynamicStateOffset[23:6]SAMPLER_INDIRECT_STATE_BORDER_COLOR [] **Description** This field specifies the pointer to SAMPLER_INDIRECT_STATE, which contains the border color This pointer is relative to the Dynamic State Base Address. This pointer is relative to the Dynamic State Base Address for Non-Bindless sampler state, and is relative to the Sample State Base Address for Bindless sampler state
	5	**Reserved** Format: MBZ
	4	**Reserved** Format: MBZ
	3	**Return Filter Weight for Border Texels** This bit, when set, returns the filter_weight in the Alpha channel of all non-border texels. Red, Green, and Blue channels will contain the sample result with border texels excluded. For cases where the surface format contains an Alpha channel, the result returned will be ovewritten to return the filter weight. For cases where the surface format does not contain Alpha, the result will still be returned in the

		Alpha Channel.

Value	Name	Description
0h	Disable **[Default]**	When programmed to 0h, normal data will be returned on RGBA channels, including contribution from border color texels.
1h	Enable	When programmed to 1h, RGB channels return filter data contributed from non-border color texels, and A channel returns filter weight of contributing texels.

Programming Notes
If this bit is set then the border color and the Border Color Mode field (in SAMPLER_STATE) are ignored. Certain message types such as sample_c, sample_min/max and gather4_* have restrictions on the use of this mode. See the Messages section of the 3D sampler for more information.

2 **Return Filter Weight for Null Texels**

This bit, when set, causes samples to return filter_weight of all non-NULL texels in the Alpha channel; Red, Green, and Blue channels are contain the filter result with NULL texels excluded; A non-NULL texel is a texel which does not reference a Null Tile.
For cases where Tiled_Resource_Mode is TR_NONE, the result will always be 1.0 since no texels would be NULL.
For cases where the surface format contains an Alpha channel, the result returned will be overridden to return the filter weight.
For cases where the surface format does not contain Alpha, the result will still be returned in the Alpha Channel.

Value	Name	Description
0h	Disable **[Default]**	When programmed to 0h, filter weight will not be returned, and normal data will be returned on the Alpha channel.
1h	Enable	When programmed to 1h, filter weight will be returned on the Alpha channel rather than the normal data expected on the Alpha channel.

Programming Notes
Certain message types such as sample_c, sample_min/max and gather4_* have restrictions on the use of this mode. See the Messages section of the 3D sampler for more information.

1 **SRGB DECODE**

This bit controls whether the 3D sampler will decode an sRGB formatted surface into RGB prior to any filtering operation.
When set, it does not convert to linear RGB (via a reverse gamma conversion). This bit is ignored for ASTC formats, which are always converted to linear RGB prior to filtering.

Value	Name	Description
0h	DECODE_EXT **[Default]**	When set to 0h, the 3D sampler will convert texels from an sRGB surface to linear RGB prior to filtering and/or returning the value.
1h	SKIP_DECODE_EXT	When set to 1h, the 3D sampler will not convert texels to linear RGB before filtering and returning results.

SAMPLER_STATE		
	0	**LOD Clamp Magnification Mode** Format: U1 Enumerated Type This field allows the flexibility to control how LOD clamping is handled when in magnification mode. **Value** | **Name** | **Description** 0h | MIPNONE | When in magnification mode, Sampler will clamp LOD as if the **Mip Mode Filter** is MIPFILTER_NONE. This is how OpenGL defines magnification, and therefore it is expected that those drivers would not set this bit. 1h | MIPFILTER | When in magnification mode, Sampler will clamp LOD based on the value of **Mip Mode Filter**.
3	31:26	**Reserved** Format: MBZ
	25	**Reserved** Default Value: 0
	24	**Allow low quality LOD calculation** Setting this bit will allow sampler to use the low quality LODcalculationmode for power savings. Note that this will not force low quality and sampler will only do it if the follow conditions are also true. If they are not true it will use the same algorithm as before as selected by the EWA bit Message type sample/sample_l Min/Mag/Mip_filter = nearest or linear. Map type = 2D //No arrays Indirect offsets must be zero Coordinates must be normalized No clamp border or half border Sampler must not be disabled //Sampler state bit No chromakey No posh
	23:22	**Reduction Type** Format: U2 Enumerated Type This field defines the type of reduction that will be performed on the texels in the footprint defined by the **Min/Mag/Mip Filter Mode** fields. This field is ignored if **Reduction Type Enable** is disabled. **Value** | **Name** | **Description** 0h | STD_FILTER | standard filter 1h | COMPARISON | comparison followed by standard filter 2h | MINIMUM | minimum of footprint 3h | MAXIMUM | maximum of footprint **Programming Notes** The following message types ignore this field: *sample_min, sample_max, sample_unorm*, resinfo, sampleinfo, LOD, ld*, sample_8x8*.

SAMPLER_STATE

		If the current min/mag filter mode is MAPFILTER_MONO, this field is ignored. The *sample_c, sample_l_c, sample_d_c, sample_b_c, gather4_c*, and *gather4_po_c* message types, when used with STD_FILTER, MINIMUM, or MAXIMUM settings of this field, perform the operation of the message of the same name without the "_c". The ref parameter is ignored by hardware. For message types not listed above, when used with COMPARISON setting of this field, perfom the operation of the message of the same name with "_c" included. The ref parameter used by the operation (since it is not delivered in the message) is set to zero. Restrictions applying to the message whose behavior is being performed must be followed. For example, a sample message used with COMPARISON reduction filter must follow all of the restrictions of *sample_c*. An exception to this is the MINIMUM and MAXIMUM reduction types allow SURFTYPE_1D, 2D, 3D, and CUBE, including with **Surface Array** enabled, even though the sample_min/max messages only allow 2D. Restrictions applying to the message delivered need not be followed. For example, a *sample_c* message used with STD_FILTER reduction filter needs to follow only the restrictions of sample, not the restrictions of *sample_c*.
	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
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 **not** force rounding enable.
	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)

SAMPLER_STATE		
		in either mag or min filter mode. **Programming Notes** Hardware will **not** force rounding enable.
	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 **not** force rounding enable.
	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 **not** force rounding enable.
	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. **Programming Notes** Hardware will **not** force rounding enable.
	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 **not** force rounding enable.
	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.
1	TRIQUAL_HIGH/MAG_CLAMP_MIPFILTER	High Quality.
2	MED	Medium Quality.

SAMPLER_STATE

		3 | LOW | Low Quality.
	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.
	9	**Reduction Type Enable** Format: Enable This field enables the **Reduction Type** field to modify the behavior of messages based on its setting. If this field is disabled, all messages behave as defined and the **Reduction Type** field is ignored.
	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, 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.

SAMPLER_STATE		
	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 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 **Programming Notes** TCZ Address Control Mode Cannot use MIRROR_101 mode. MIRROR_101 mode only works for 2D surfaces.

SCALER_COEFFICIENT_FORMAT

<table>
<tr><th colspan="3">SCALER_COEFFICIENT_FORMAT</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 16
Default Value: 0x00000000</td></tr>
<tr><td colspan="3">Scaler coefficients are stored in sign-exponent-mantissa format. Two coefficients are stored in each dword, the table below show the data packing in each dword.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="5">0</td><td>15</td><td>Sign<table><tr><th>Value</th><th>Name</th></tr><tr><td>0b</td><td>Positive</td></tr><tr><td>1b</td><td>Negative</td></tr></table></td></tr>
<tr><td>14</td><td>Reserved</td></tr>
<tr><td>13:12</td><td>Exponent
Represented as 2^(-n)<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>00b</td><td>2</td><td>2 or mantissa is b.bbbbbbbb</td></tr><tr><td>01b</td><td>1</td><td>1 or mantissa is 0.bbbbbbb..</td></tr><tr><td>10b</td><td>0.5</td><td>0.5 or mantissa is 0.0bbbbbbbb..</td></tr><tr><td>11b</td><td>0.25</td><td>0.25 or mantissa is 0.00bbbbbbb..</td></tr><tr><td>Others</td><td>Reserved</td><td>Reserved</td></tr></table></td></tr>
<tr><td>11:3</td><td>Mantissa
All the tap coefficients use all 9 bits of mantissa.</td></tr>
<tr><td>2:0</td><td>Reserved</td></tr>
</table>

SCISSOR_RECT

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.

Restriction
Restriction : When executed in the POCS command stream, this command programs the scissor state for the SFR stage of the POCS pipeline

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**

SCISSOR_RECT

		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	Name
0-16383	

Scratch Hword Block Message Header

MH_A32_HWB - Scratch Hword Block Message Header		
Source:	EuSubFunctionDataPort0	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..2	95:0	**Reserved** Format: MBZ Ignored
3	31:0	**Per Thread Scratch Space** Format: MHC_PTSS Specifies amount of scratch space used by this thread, for Stateless bounds checking.
4	31:0	**Reserved** Format: MBZ Ignored
5	31:0	**Buffer Base Address** Format: MHC_A32_BBA Specifies the surface address offset page [31:10] for A32 stateless messages.
6..7	63:0	**Reserved** Format: MBZ Ignored

SF_CLIP_VIEWPORT

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	

Restriction
Restriction : When executed in the POCS command stream, this command programs the viewport state for the CLR and SFR stage of the POCS pipeline.

DWord	Bit	Description
0	31:0	**Viewport Matrix Element m00** Format: IEEE_FLOAT32
1	31:0	**Viewport Matrix Element m11** Format: IEEE_FLOAT32
2	31:0	**Viewport Matrix Element m22** Format: IEEE_FLOAT32
3	31:0	**Viewport Matrix Element m30** Format: IEEE_FLOAT32
4	31:0	**Viewport Matrix Element m31** Format: IEEE_FLOAT32
5	31:0	**Viewport Matrix Element m32** Format: IEEE_FLOAT32
6	31:0	**Reserved** Format: MBZ
7	31:0	**Reserved** Format: MBZ
8	31:0	**X Min Clip Guardband** Format: IEEE_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** Format: IEEE_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**

SF_CLIP_VIEWPORT		
		Format: IEEE_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.
11	31:0	**Y Max Clip Guardband** Format: IEEE_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	31:0	**X Min ViewPort** Format: IEEE_Float This 32-bit float represents the Viewport.XMin. This is the X min of the viewport extents as programmed by API, and this value should be programmed in Screen Space coordinate and not as normalized coordinate.
13	31:0	**X Max ViewPort** Format: IEEE_FLOAT32 This 32-bit float represents the Viewport.XMax. This is the X max of the viewport extents as programmed by API, and this value should be programmed in Screen Space coordinate and not as normalized coordinate.
14	31:0	**Y Min ViewPort** Format: IEEE_FLOAT32 This 32-bit float represents the Viewport.YMin. This is the Y min of the viewport extents as programmed by API, and this value should be programmed in Screen Space coordinate and not as normalized coordinate.
15	31:0	**Y Max ViewPort** Format: IEEE_FLOAT32 This 32-bit float represents the Viewport.Ymax. This is the Y max of the viewport extents as programmed by API, and this value should be programmed in Screen Space coordinate and not as normalized coordinate.

SF_OUTPUT_ATTRIBUTE_DETAIL

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

DWord	Bit	Description
0	15	**Component Override W** Format: Enable If set, the W component of this output Attribute is overridden by the W component of the constant vector specified by ConstantSource.
	14	**Component Override Z** Format: Enable If set, the Z component of this output Attribute is overridden by the Z component of the constant vector specified by ConstantSource.
	13	**Component Override Y** Format: Enable If set, the Y component of output Attribute is overridden by the Y component of the constant vector specified by ConstantSource.
	12	**Component Override X** Format: Enable If set, the X component of output Attribute is overridden by the X component of the constant vector specified by ConstantSource.
	11	**Swizzle Control Mode** Format: U1 Enumerated Type When Attribute Swizzle Enable is ENABLED, this bit controls whether attributes 0-15 or 16-31 are subject to the following swizzle controls: • Component Override X/Y/Z/W • Constant Source • Swizzle Select • Source Attribute • WrapShortest Enables Note that the Number of SF Output Attributes field specifies how many attributes are output. Note: This field does not impact any functions which provide separate states for all 32 attributes (e.g., Point sprite, Constant interpolation).

SF_OUTPUT_ATTRIBUTE_DETAIL

		Note: This field is only valid for the first indexed attribute (Attribute[0]). For all other indices, it is Reserved and MBZ.
	10:9	**Constant Source** Format: U2 enumerated type This state selects a constant vector which can be used to override individual components of this Attribute

Value	Name	Description
0h	CONST_0000	Constant.xyzw = 0.0,0.0,0.0,0.0
1h	CONST_0001_FLOAT	Constant.xyzw = 0.0,0.0,0.0,1.0
2h	CONST_1111_FLOAT	Constant.xyzw = 1.0,1.0,1.0,1.0
3h	PRIM_ID	Constant.xyzw = PrimID (replicated)

	8	**Reserved** Format: MBZ
	7:6	**Swizzle Select** Format: U2 enumerated type This state, along with Source Attribute, specifies the source for this output Attribute.

Value	Name	Description
0h	INPUTATTR	This attribute is sourced from AttrInputReg[SourceAttribute]
1h	INPUTATTR_FACING	If the object is front-facing, this attribute is sourced from AttrInputReg[SourceAttribute]. If the object is back-facing, this attribute is sourced from AttrInputReg[SourceAttribute+1].
2h	INPUTATTR_W	This attribute is sourced from AttrInputReg[SourceAttribute]. The W component is copied to the X component.
3h	INPUTATTR_FACING_W	If the object is front-facing, this attribute is sourced from AttrInputReg[SourceAttribute]. If the object is back-facing, this attribute is sourced from AttrInputReg[SourceAttribute+1]. The W component is copied to the X component.

	5	**Reserved** Format: MBZ
	4:0	**Source Attribute** Format: U5 This field selects the source attribute for this Attribute. Source attribute 0 corresponds to the first 128 bits of data indicated by Vertex URB Entry Read Offset

SFC_8x8_AVS_COEFFICIENTS

SFC_8x8_AVS_COEFFICIENTS	
Source:	BSpec
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

Description
ExistsIf = AVS && (Function_mode = 0)

DWord	Bit	Description
0	31:24	**ZeroYFilterCoefficient1** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**ZeroXFilterCoefficient1** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**ZeroYFilterCoefficient0** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**ZeroXFilterCoefficient0** Format: S1.6 2's Complement **Range:** [-2, +2)
1	31:24	**ZeroYFilterCoefficient3** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**ZeroXFilterCoefficient3** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**ZeroYFilterCoefficient2** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**ZeroXFilterCoefficient2** Format: S1.6 2's Complement

SFC_8x8_AVS_COEFFICIENTS		
		Range: [-2, +2)
2	31:24	**ZeroYFilterCoefficient5** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**ZeroXFilterCoefficient5** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**ZeroYFilterCoefficient4** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**ZeroXFilterCoefficient4** Format: S1.6 2's Complement **Range:** [-2, +2)
3	31:24	**ZeroYFilterCoefficient7** Format: S1.6 2's Complement **Range:** [-2, +2)
	23:16	**ZeroXFilterCoefficient7** Format: S1.6 2's Complement **Range:** [-2, +2)
	15:8	**ZeroYFilterCoefficient6** Format: S1.6 2's Complement **Range:** [-2, +2)
	7:0	**ZeroXFilterCoefficient6** Format: S1.6 2's Complement **Range:** [-2, +2)
4	31:24	**OneXFilterCoefficient3** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**OneXFilterCoefficient2** Format: S1.6 2's Complement

SFC_8x8_AVS_COEFFICIENTS		
		Range: [-1.0, +1.0)
	15:0	**Reserved** Format: MBZ
5	31:16	**Reserved** Format: MBZ
	15:8	**OneXFilterCoefficient5** Format: S1.6 2's Complement **Range:** [-1.0, +1.0)
	7:0	**OneXFilterCoefficient4** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
6	31:24	**OneYFilterCoefficient3** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)
	23:16	**OneYFilterCoefficient2** Format: S1.6 2's Complement **Range:** [-1.0, +1.0)
	15:0	**Reserved** Format: MBZ
7	31:16	**Reserved** Format: MBZ
	15:8	**OneYFilterCoefficient5** Format: S1.6 2's Complement **Range:** [-1.0, +1.0)
	7:0	**OneYFilterCoefficient4** Format: S1.6 2's Complement **Range:** [-2.0, +2.0)

SFC_AVS_CHROMA_COEFF_TABLE_BODY

SFC_AVS_CHROMA_COEFF_TABLE_BODY

Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:56	**Table 1Y Filter Coefficient[[n],5]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for Y-direction. **Programming Notes** Filter tap index3 in U/V 4-tap filtering
	55:48	**Table 1X Filter Coefficient[[n],5]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for X-direction. **Programming Notes** Filter tap index3 in U/V 4-tap filtering
	47:40	**Table 1Y Filter Coefficient[[n],4]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for Y-direction. **Programming Notes** Filter tap index 2 in U/V 4-tap filtering

SFC_AVS_CHROMA_COEFF_TABLE_BODY

	39:32	**Table 1X Filter Coefficient[[n],4]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for X-direction. **Programming Notes** Filter tap index 2 in U/V 4-tap filtering
	31:24	**Table 1Y Filter Coefficient[[n],3]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for Y-direction. **Programming Notes** Filter tap index1 in U/V 4-tap filtering
	23:16	**Table 1X Filter Coefficient[[n],3]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for X-direction. **Programming Notes** Filter tap index1 in U/V 4-tap filtering
	15:8	**Table 1Y Filter Coefficient[[n],2]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for Y-direction. **Programming Notes** Filter tap index0 in U/V 4-tap filtering
	7:0	**Table 1X Filter Coefficient[[n],2]** Format: S1.6 2's Complement **Range:** [-2, +2) Chroma table for X-direction. **Programming Notes**

SFC_AVS_CHROMA_COEFF_TABLE_BODY		
		Filter tap index0 in U/V 4-tap filtering
2..63	1983:0	**Filter Coefficients** Format: Chroma_Filter_Coefficients_Array[31]

SFC_AVS_LUMA_COEFF_TABLE_BODY

SFC_AVS_LUMA_COEFF_TABLE_BODY

Source:	BSpec
Size (in bits):	4096
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0..3	127:120	**Table 0Y Filter Coefficient[[n],7]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	119:112	**Table 0X Filter Coefficient[[n],7]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction.

SFC_AVS_LUMA_COEFF_TABLE_BODY		
		Programming Notes For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	111:104	**Table 0Y Filter Coefficient[[n],6]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	103:96	**Table 0X Filter Coefficient[[n],6]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	95:88	**Table 0Y Filter Coefficient[[n],5]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	87:80	**Table 0X Filter Coefficient[[n],5]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	79:72	**Table 0Y Filter Coefficient[[n],4]** Format: S1.6 2's Complement **Range:** [-2, +2)

SFC_AVS_LUMA_COEFF_TABLE_BODY

		Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	71:64	**Table 0X Filter Coefficient[[n],4]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	63:56	**Table 0Y Filter Coefficient[[n],3]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	55:48	**Table 0X Filter Coefficient[[n],3]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	47:40	**Table 0Y Filter Coefficient[[n],2]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	39:32	**Table 0X Filter Coefficient[[n],2]** Format: S1.6 2's Complement

SFC_AVS_LUMA_COEFF_TABLE_BODY

		Range: [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	31:24	**Table 0Y Filter Coefficient[[n],1]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	23:16	**Table 0X Filter Coefficient[[n],1]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	15:8	**Table 0Y Filter Coefficient[[n],0]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for Y-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
	7:0	**Table 0X Filter Coefficient[[n],0]** Format: S1.6 2's Complement **Range:** [-2, +2) Luma table for X-direction. **Programming Notes** For 5x5 filter mode, table 0, 6 and 7 must be programmed to zero.
4..127	3967:0	**Filter Coefficients**

SFC_AVS_LUMA_COEFF_TABLE_BODY		
		Format: Luma_Filter_Coefficients_Array[31]

SFC_AVS_STATE_BODY

SFC_AVS_STATE_BODY		
Source:	BSpec	
Size (in bits):	96	
Default Value:	0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:24	**Sharpness Level** Format: U8 When adaptive scaling is off, determines the balance between sharp and smooth scalers. **Value** | **Name** | **Description** 0 | | Contribute 1 from the smooth scalar 255 | | Contribute 1 from the sharp scalar
	23:7	**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.
1	31:24	**Reserved** Format: MBZ
	23:16	**Max Derivative 4 Pixels** Format: U8 Used in adaptive filtering to specify the lower boundary of the smooth 4 pixel area.
	15:8	**Reserved** Format: MBZ
	7:0	**MAX Derivative Point 8** Format: U8 Used in adaptive filtering to specify the lower boundary of the smooth 8 pixel area.
2	31:12	**Reserved**

![intel logo]

SFC_AVS_STATE_BODY

		Format: MBZ
	11:8	**Input Horizontal Siting Value - Specifies the horizontal siting of the input**
	7:4	**Reserved**
	3:0	**Input Vertical Siting - Specifies the vertical siting of the input**

Input Horizontal Siting Value (11:8)

Value	Name
0000b	0(fraction in integer)
0001b	1/8
0010b	2/8
0011b	3/8
0100b	4/8
0101b	5/8
0110b	6/8
0111b	7/8
1000b	8/8

Programming Notes
For 444 format, horizontal chroma siting should be programmed to zero.

Input Vertical Siting (3:0)

Value	Name
000b	0
0001b	1/8
0010b	2/8
0011b	3/8
0100b	4/8
0101b	5/8
0110b	6/8
0111b	7/8
1000b	8/8

Programming Notes
For 444 and 422 format, vertical chroma siting should be programmed to zero.

SFC_FRAME_START_BODY

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

DWord	Bit	Description
0	31:0	**Reserved** Format: MBZ

SFC_HDR_STATE

SFC_HDR_STATE

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

DWord	Bit	Description
0..1	63:48	**Reserved** Format: MBZ
	47:12	**Address** Format: GraphicsAddress[47:12] Specifies the graphics base address used to fetch SFC_EOTF_OETF_STATE surfacetable into SFC.
	11:0	**Reserved** Format: MBZ
2	31:15	**Reserved** Format: MBZ
	14:13	**Surface Tiled Mode** Format: U2 **For Media Surfaces:** This field specifies the tiled resource mode. Value / Name / Description: 0 / TRMODE_NONE / No tiled resources 1 / TRMODE_TILEYF / 4KB tiled resources 2 / TRMODE_TILEYS / 64KB tiled resources 3 / Reserved /
	12	**Reserved** Format: MBZ
	11	**Scratch Buffer Cache Select** Default Value: 0 Disable Format: U1 **Programming Notes** This must be set to 0
	10	**Compression Type** Default Value: 0 Disable Format: boolean This field is applicable only when Memory compression is enabled.As memory compression is not supported on this surface, it must be 0.

SFC_HDR_STATE		
	9	**Memory Compression Enable** Default Value: 0 Disable Format: Enable **Programming Notes** Memory compression is not supported for this surface Must be 0.
	8:7	**Arbitration Priority Control** Format: HEVC_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**

SFC_IEF_STATE_BODY

SFC_IEF_STATE_BODY		
Source:	BSpec	
Size (in bits):	736	
Default Value:	0x0294806C, 0x39CFD1FF, 0x039F0000, 0x9A6E4000, 0x00601180, 0xFFFE2F2E, 0x00000000, 0xD82E0000, 0x8285ECEC, 0x00008282, 0x00000000, 0x02117000, 0xA38FEC96, 0x00008CC8, 0x00000000, 0x00002000, 0x00000000, 0x00000400, 0x00000000, 0x00000400, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:28	**Reserved** Format: MBZ
	27:23	**R3c Coefficient** Default Value: 5 Format: U0.5 IEF smoothing coefficient, *see IEF map.*
	22:18	**R3x Coefficient** Default Value: 5 Format: U0.5 IEF smoothing coefficient, *see IEF map.*
	17:12	**Strong Edge Threshold** Default Value: 8 Format: U6 If EM > Strong Edge Threshold → the basic VSA detects a strong edge.
	11:6	**Weak Edge Threshold** Default Value: 1 Format: U6 If Strong Edge Threshold > EM > Weak Edge Threshold → the basic VSA detects a weak edge.
	5:0	**Gain Factor** Default Value: 44 Format: U6 User control sharpening strength.
1	31:27	**R5c Coefficient** Default Value: 7

SFC_IEF_STATE_BODY

DWord	Bit	Description
		Format: U0.5 IEF smoothing coefficient, *see IEF map*.
	26:22	**R5cx Coefficient** Default Value: 7 Format: U0.5 IEF smoothing coefficient, *see IEF map*.
	21:17	**R5x Coefficient** Default Value: 7 Format: U0.5 IEF smoothing coefficient, *see IEF map*.
	16:14	**Strong Edge Weight** Default Value: 7 Format: U3 Sharpening strength when a <u>STRONG</u> edge is found in basic VSA.
	13:11	**Regular Weight** Default Value: 2 Format: U3 Sharpening strength when a <u>WEAK</u> edge is found in basic VSA.
	10:8	**Non Edge Weight** Default Value: 1 Format: U3 . Sharpening strength when <u>NO EDGE</u> is found in basic VSA.
	7:0	**Global Noise Estimation** Default Value: 255 Format: U8 Global noise estimation of previous frame.
2	31:28	**Reserved** Format: MBZ
	27:22	**Hue_Max** Default Value: 14 Format: U6 Rectangle half width.

SFC_IEF_STATE_BODY		
	21:16	**Sat_Max** Default Value: 31 Format: U6 Rectangle half length.
	15:8	**STD Cos(alpha)** Format: S0.7 2's Complement **Default Value** = 79/128
	7:0	**STD Sin(alpha)** Format: S0.7 2's Complement **Default Value** = 101/128
3	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	**VY_STD_Enable** Format: Enable Enables STD in the VY subspace.
	14:12	**Diamond Margin** Default Value: 4 Format: U3
	11	**Reserved** Format: MBZ
	10:0	**S3U** Format: S2.8 -2's Complement Slope 3 of the upper part of the detection PWLF. **Deafult:** 0/256
4	31	**Skin Detail Factor**

SFC_IEF_STATE_BODY

		Format: U1 Enumerated Type This flag bit is in operation only when one of the following conditions exists: • when the control bit **SkinToneTunedIEF_Enable** is on. • When **SkinDetailFactor** is equal to 0, sign(**SkinDetailFactor**) is equal to +1, and the content of the detected skin tone area is detail revealed.When **SkinDetailFactor** is equal to 1, sign(**SkinDetailFactor**) is equal to -1, and the content of the detected skin tone area is not detail revealed. **Value** / **Name**: 0 — Detail Revealed **[Default]** 1 — Not Detail Revealed
	30:24	**Diamond_du** Default Value: 0 Format: S6 -2's Complement Rhombus center shift in the sat-direction, relative to the rectangle center.
	23:21	**HS_margin** Default Value: 3 Format: U3 Defines rectangle margin.
	20:13	**Diamond_alpha** Format: U2.6 1 / tan(β) **Deafult:** 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 Rhombus center shift in the hue-direction, relative to the rectangle center.
5	31:24	**Y_point_4** Default Value: 255 Format: U8

SFC_IEF_STATE_BODY		
		Fourth point of the Y piecewise linear membership function.
	23:16	**Y_point_3** Default Value: 254 Format: U8 Third point of the Y piecewise linear membership function.
	15:8	**Y_point_2** Default Value: 47 Format: U8 Second point of the Y piecewise linear membership function.
	7:0	**Y_point_1** Default Value: 46 Format: U8 First point of the Y piecewise linear membership function.
6	31:16	**Reserved** Format: MBZ
	15:0	**INV_Margin_VYL** Format: U0.16 1 / Margin_VYL **Default:** 3300/65536
7	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	**INV_Margin_VYU** Format: U0.16 1 / Margin_VYL **Default:** 1600/65536

SFC_IEF_STATE_BODY		
8	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** Default Value: 236 Format: U8 Y Point 2 of the lower part of the detection PWLF.
9	31:27	**Y_Slope_2** Format: U2.3 Slope between points Y3 and Y4. **Default:** 31/8
	26:16	**S0L** Format: S2.8 -2's Complement Slope 0 of the lower part of the detection PWLF. **Default:** -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.

SFC_IEF_STATE_BODY		
10	31:22	**Reserved** Format: MBZ
	21:11	**S2L** Format: S2.8 -2's Complement **Default:** 0/256 Slope 2 of the lower part of the detection PWLF.
	10:0	**S1L** Format: S2.8 -2's Complement **Default:** 0/256 Slope 1 of the lower part of the detection PWLF.
11	31:27	**Y_Slope1** Format: U2.3 Slope between points Y1 and Y2. **Default:** 31/8
	26:19	**P1U** Default Value: 66 Format: U8 Y Point 1 of the upper part of the detection PWLF.
	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. **Default:** 0/256
12	31:24	**B1U** Default Value: 163 Format: U8 V Bias 1 of the upper part of the detection PWLF.

SFC_IEF_STATE_BODY		
	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.
13	31:27	**Reserved** Format: MBZ
	26:16	**S0U** Format: S2.8 -2's Complement Slope 0 of the upper part of the detection PWLF. **Default:** 256/256
	15:8	**B3U** Default Value: 140 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.
14	31:22	**Reserved** Format: MBZ
	21:11	**S2U** Format: S2.8 -2's Complement **Default:** -179/256 Slope 2 of the upper part of the detection PWLF.

SFC_IEF_STATE_BODY		
	10:0	**S1U** Format: S2.8 -2's Complement **Default:** 113/256 Slope 1 of the upper part of the detection PWLF.
15	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	**Reserved** Format: MBZ
	1	**YUV Channel Swap**
	0	**Transform Enable**
16	31:26	**Reserved** Format: MBZ
	25:13	**C3** Default Value: 0 Format: S2.10 -2's Complement Transform coefficient
	12:0	**C2** Default Value: 0 Format: S2.10 -2's Complement Transform coefficient
17	31:26	**Reserved** Format: MBZ
	25:13	**C5** Default Value: 0 Format: S2.10 -2's Complement

SFC_IEF_STATE_BODY		
		Transform coefficient
	12:0	**C4** Default Value: 1024 Format: S2.10 -2's Complement Transform coefficient
18	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
19	31:13	**Reserved** Format: MBZ
	12:0	**C8** Default Value: 1024 Format: S2.10 -2's Complement Transform coefficient
20	31:22	**Reserved** Format: MBZ
	21:11	**Offset out 1** Default Value: 0 Format: S2.8 -2's Complement Offset out for Y/R.
	10:0	**Offset in 1** Default Value: 0 Format: S2.8 -2's Complement Offset in for Y/R.
21	31:22	**Reserved**

SFC_IEF_STATE_BODY

		Format: MBZ
	21:11	**Offset out 2** Default Value: 0 Format: S2.8 -2's Complement Offset out for U/G.
	10:0	**Offset in 2** Default Value: 0 Format: S2.8 -2's Complement Offset in for U/G.
22	31:22	**Reserved** Format: MBZ
	21:11	**Offset out 3** Default Value: 0 Format: S2.8 -2's Complement Offset out for V/B.
	10:0	**Offset in 3** Default Value: 0 Format: S2.8 -2's Complement Offset in for V/B.

SFC_LOCK_BODY

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

DWord	Bit	Description
0	31:2	**Reserved** Format: MBZ
	1	**Pre-Scaled Output Surface Output Enable** **VD - Reconstructed Pixel Output Enable** For VD Mode, this field specifies the enabling of writing out the display reconstructed pixel to memory. It could be pre or post- ILDB filter pixel output based on the pre- and post- filter setting in the AVC state command.
	0	**VE-SFC Pipe Select**

Pre- Deblock Flag	Post- Deblock Flag	VD Pixels Output to Memory	VD Pixels Output to SFC
0	0	Invalid for SFC Mode	Invalid for SFC Mode
0	1	Filtered Pixels (allow ON/OFF)	Filter Pixels Sent to SFC for Scaling
1	0	Non-filter (bypass) pixels (allow ON/OFF)	Non-Filter Pixels Sent to SFC for Scaling
1	1	Non-filter (bypass) pixels (allow ON/OFF) Filtered pixels (always OFF)	Filter Pixels Sent to SFC for Scaling.

SFC_STATE_BODY

SFC_STATE_BODY		
Source:	BSpec	
Size (in bits):	1472	
Default Value:	0x00000000, 0x00001000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0	31:12	**Reserved** Format: MBZ
	11	**Reserved** Format: MBZ
	10:8	**VD/VE Input Ordering Mode** Format: U3 • VD mode: (SFC pipe mode set as "0") • VE mode: (pipe mode set as "1 and 4") For values for each mode, please refer to the table below: • HCP mode : SFC Pipe Mode set as "2" For values for each mode, please refer to the table below:

Value	Name	Description	Exists If
0		16x16 block z-scan order - no shift	//VD Mode
1		16x16 block z-scan order - 4 pixels shift upward	//VD Mode
2		8x8 block jpeg z-scan order	//VD Mode
3		16x16 block jpeg z-scan order	//VD Mode
4		16x16 block VP8 row-scan order - no shift	//VD Mode
5-7		Reserved	//VD Mode
0		16x16 block HEVC Decoderrow-scan order -4 pixel shift upward	//HCP Mode
1		32x32block HEVC Decoderrow-scan order -4 pixel shift upward	//HCP Mode
2		64x64 block HEVC Decoder row-scan order -4 pixel shift upward	//HCP Mode

SFC_STATE_BODY

3		64x64 block VP9 Decoderrow-scan order - 8 pixel shift upward	//HCP Mode
[4-7]		Reserved	//HCP Mode
0		8x4 block column order, 64 pixel column	//VE Mode
1		4x4 block column order, 64 pixel column	//VE Mode
[2-7]		Reserved	//VE Mode

Programming Notes

This field shall be programmed according to video modes used in VDBOX. NOTE: SFC supports progressive input and output only (Interlaced/MBAFF is not supported).

Video Mode	Surface Format	SFC Input Chroma Sub-Sampling	VD/VE Input Ordering Mode
VC1 w/o LF and w/o OS Note: VC1 LF applies for either ILDB	420 (NV12)	1	0
VC1 w/ LF or w/ OS or w/ both Note: VC1 LF applies for either ILDB		INVALID with SFC	INVALID with SFC
AVC w/o LF	Monochrome	0	0
AVC w/o LF	420 (NV12)	1	0
AVC with LF	Monochrome	0	1
AVC/VP8 with LF	420 (NV12)	1	1
VP8 w/o LF	420 (NV12)	1	4
JPEG (YUV Interleaved)	Monochrome	0	2
JPEG (YUV Interleaved)	420	1	3
JPEG (YUV Interleaved)	422H_2Y	2	2
JPEG (YUV Interleaved)	422H_4Y	2	3
JPEG (YUV Interleaved)	444	4	2
JPEG (YUV Interleaved)	411	5	2

This field shall be programmed according to Image enhancement modes used in VEBOX.

VEBOX MODE	VEBOX Single Pipe Enable Bit	SFC Input Surface Format	SFC Input Chroma Sub Sampling	VD/VE Input Ordering Mode
1. DN/HP with RGB input 2. Camera pipe (DM) enabled 3. IECP with FECSC, CCM, FGC filters enabled	1	Monochrome	0	1
	1	420 (NV12)	1	1
	1	422H	2	1
	1	444	4	1
All other modes:	0	Monochrome	0	0

SFC_STATE_BODY

(Legacy DN/DI/IECP features)	0	420 (NV12)	1	0
	0	422H	2	0
	0	444	4	0

This field shall be programmed according to video mode used in HCP. Note: SFC supports progressive input and output only (interlace/mbaff is not supported).

Video Mode	Surface Format	SFC Input Chroma Sub Sampling	VD/VE Input ordering mode
HEVC 16x16 LCU	420/422/444	1 / 2 / 4	0
HEVC 32x32 LCU	420/422/444	1 / 2 / 4	1
HEVC 64x64 LCU	420/422/444	1/ 2 / 4	2
VP9 64x64 LCU	420/444	1 / 4	3 / 4

7:4 **SFC Input Chroma Sub-Sampling**

Value	Name	Description
0	4:0:0	SFC to insert UV channels
1	4:2:0	
2	4:2:2 Horizonatal	VD: 2:1:1
3	Reserved	
4	4:4:4 Progressive/Interleaved	

Programming Notes

This field shall be programmed according to video modes used in VDBOX. NOTE: SFC supports progressive input and output only (Interlaced/MBAFF is not supported).

Video Mode	Surface Format	SFC Input Chroma Sub-Sampling	VD/VE Input Ordering Mode
VC1 w/o LF and w/o OS Note: VC1 LF applies for either ILDB	420 (NV12)	1	0
VC1 w/ LF or w/ OS or w/ both Note: VC1 LF applies for either ILDB		INVALID with SFC	INVALID with SFC
AVC w/o LF	Monochrome	0	0
AVC w/o LF	420 (NV12)	1	0
AVC with LF	Monochrome	0	1
AVC/VP8 with LF	420 (NV12)	1	1
VP8 w/o LF	420 (NV12)	1	4
JPEG (YUV Interleaved)	Monochrome	0	2
JPEG (YUV Interleaved)	420	1	3

SFC_STATE_BODY

JPEG (YUV Interleaved)	422H_2Y	2	2
JPEG (YUV Interleaved)	422H_4Y	2	3
JPEG (YUV Interleaved)	444	4	2

This field shall be programmed according to Image enhancement modes used in VEBOX.

VEBOX MODE	Surface Format	SFC Input Chroma Sub Sampling	VD/VE Input Ordering Mode
Legacy DN/DI/IECP features	Monochrome	0	0
Legacy DN/DI/IECP features	420 (NV12)	1	0
Legacy DN/DI/IECP features	422H	2	0
Legacy DN/DI/IECP features	444	4	0
Capture/Camera pipe enabled(Demosaic)	Monochrome	0	1
Capture/Camera pipe enabled(Demosaic)	420 (NV12)	1	1
Capture/Camera pipe enabled(Demosaic)	422H	2	1
Capture/Camera pipe enabled(Demosaic)	444	4	1

3:0 **SFC Pipe Mode**

Value	Name	Description
0		VD-to-SFC AVS
1		VE-to-SFC AVS + IEF + Rotation
2		HCP-to-SFC AVS
3		Reserved
4		VE-to-SFC Integral Image
5		
[6,15]		Reserved

Programming Notes
Note: for SFC Pipe mode set to VE-to-SFC AVS mode. IECP pipeline mode MUST be enabled. However, each sub-IECP feature can be turned on/off independently.

1 31:30 **Reserved**

Format:	MBZ

29:16 **Input Frame Resolution Height**

Format:	U14-1

SFC_STATE_BODY

		Minus 1 in unit of pixel [13:0]. It is set to the value of the output resolution or number of pixels streaming into SFC from VD/HCP or VEBOX. Since the Max value support in 16K pixels,the max value allowed in 16K minus 1. • VDBOX frame height is multiple of 16 for Video source and JPEG formats other than 400, 444 and 422H_2Y. • VDBOX frame height is multiple of 8 for JPEG formats 400, 444 and 422H_2Y. • VEBOX frame height is multiple of 4. • HEVC frame height is multiple of 8 • VP9 frame height is multiple of 8. Min Resolution is 128 pixels. Max Resolution is up to 16K pixel eg. for 1920x1080 content, FrameHeightInMBsMinus1 is equal to 1087 (1080 rounded up 16 pixel boundary, minus 1. i.e. effectively specified as 1088 instead). Restriction : For Integral Image Mode, this field is Reserved and MBZ.
	15:14	**Reserved** Format: MBZ
	13:0	**Input Frame Resolution Width** Format: U14-1 Minus 1 in unit of pixel [13:0]. It is set to the value of the output resolution or number of pixels streaming into SFC from VD/HCP or VEBox. Since the max value support is 16k pixels, the max value allowed is 16K minus 1. • VDBOX frame width is multiple of 16 for Video source and JPEG formats other than 400, 444 and 422H_2Y. • VDBOX frame width is multiple of 8 for JPEG formats 400, 444 and 422H_2Y. • VEBOX frame width is multiple of 16. • HEVC frame width is multiple of 8. • VP9 frame width is multiple of 8. *Min Resolution* is 128 pixels. *Max Resolution* is up to 16K pixels. e.g. for 1920x1080 content, FrameHeightInMBsMinus1 is equal to1087 (1080 rounded up 16 pixel boundary, minus 1. i.e. effectively specified as 1088 instead). **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
2	31:23	**Reserved** Format: MBZ
	22:18	**Reserved**

SFC_STATE_BODY

		<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
	17	**Reserved** <table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MBZ</td></tr></table>
	16	**Input Color Space - 0- YUV/1 - RGB** THis specifies the color space of the input format. RGB is valid only with the VE-SFC mode. <table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>YUV Color Space</td></tr><tr><td>1</td><td>RGB Color Space</td></tr></table>
	15:12	**Output Chroma Downsampling co-siting position Horizontal Direction** <table><tr><td>Format:</td><td>U4</td></tr></table>This field specifies the fractional position of the bilinear filter for chroma downsampling. In the X-axis. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0000b</td><td>0/8 (Left full pixel)</td><td>0 (fraction_in_integer)</td></tr><tr><td>0001b</td><td>1/8</td><td>1 (fraction_in_integer)</td></tr><tr><td>0010b</td><td>1/4 (2/8)</td><td>2 (fraction_in_integer)</td></tr><tr><td>0011b</td><td>3/8</td><td>3 (fraction_in_integer)</td></tr><tr><td>0100b</td><td>1/2 (4/8)</td><td>4 (fraction_in_integer)</td></tr><tr><td>0101b</td><td>5/8</td><td>5 (fraction_in_integer)</td></tr><tr><td>0110b</td><td>3/4 (6/8)</td><td>6 (fraction_in_integer)</td></tr><tr><td>0111b</td><td>7/8</td><td>7 (fraction_in_integer)</td></tr><tr><td>1000b</td><td>8/8</td><td></td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>For 444 format, horizontal chroma-siting should be programmed to zero.</td></tr></table>
	11:8	**Output Chroma Downsampling co-siting position Vertical Direction** <table><tr><td>Format:</td><td>U4</td></tr></table>This field specifies the fractional position of the bilinear filter for chroma downsampling. In the Y-axis. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0000b</td><td>0/8 (Left full pixel)</td><td>0 (fraction_in_integer)</td></tr><tr><td>0001b</td><td>1/8</td><td>1 (fraction_in_integer)</td></tr><tr><td>0010b</td><td>1/4 (2/8)</td><td>2 (fraction_in_integer)</td></tr><tr><td>0011b</td><td>3/8</td><td>3 (fraction_in_integer)</td></tr><tr><td>0100b</td><td>1/2 (4/8)</td><td>4 (fraction_in_integer)</td></tr></table>

SFC_STATE_BODY

			0101b | 5/8 | 5 (fraction_in_integer) 0110b | 3/4 (6/8) | 6 (fraction_in_integer) 0111b | 7/8 | 7 (fraction_in_integer) 1000b | 8/8 |

0101b	5/8	5 (fraction_in_integer)
0110b	3/4 (6/8)	6 (fraction_in_integer)
0111b	7/8	7 (fraction_in_integer)
1000b	8/8	

Programming Notes
For 444 and 422 format, vertical chroma co-siting value should be programmed to zero.

7:6 **Reserved**

Format:	MBZ

5 **RGBA_Channel_Swap Enable**

Default Value:	0
Format:	Enable

This bit should only be used with RGB output formats and CSC conversion is turned on. When this bit is set, the R and B channels are swapped into the output RGB channels as shown in the following table:

Name	Bits	MSB Color Order	Swapped
RGBA8	8:8:8:8	A:B:G:R	A:R:G:B
RGBA10	2:10:10:10	A:R:G:B	A:B:G:R
RGB 5:6:5	5:6:5	R:G:B	B:G:R

4 **Reserved**

Format:	MBZ

3:0 **Output Surface Format type**

SFC output surface format type.
Reserved

Value	Name	Description	Exists If
0		AYUV 4:4:4 (8:8:8:8 MSB-A:Y:U:V)	//Tile-Y/ Tile-X/Linear
1		RGBA8 4:4:4:4 (8:8:8:8 MSB-A:B:G:R)	//Tile-Y/ Tile-X/Linear
2		RGBA10 10:10:10:2 (2:10:10:10 MSB-A:R:G:B)	//Tile-Y/ Tile-X/Linear
3		RGB 5:6:5 (5:6:5 MSB-R:G:B)	//Tile-Y/ Tile-X/Linear
4		Planar NV12 4:2:0 8-bit	//Tile-Y
5		Packed YUYV 4:2:2 8-bit	//Tile-Y/ Tile-X/Linear
6		Packed UYVY 4:2:2 8-bit	//Tile-Y/ Tile-X/Linear
7		Packed integral Image 32-bit	//Linear
8		Packed integral Image 64-bit	//Linear

SFC_STATE_BODY

DWord	Bit	Description

		9 | | P016 format | //Tile-Y
		10 | | Y210 / Y216 Format
		BitDepth = 0 => Y210
		BitDepth = 1 => Y216 | //Tile-Y / Tile-X / Linear
		11 | | Y410 / Y416 Format
		BitDepth = 0 => Y410
		BitDepth = 1 => Y416 | //Tile-Y / Tile-X / Linear
		
		Restriction
		For Integral Image Mode, output surface format type must be set to 32/64-bit Integral Image Plane. Driver/SW must ensure the max accumulated integral image value does not exceed the programmable output precision. HW will simply generate wrong value once it overflow in wrap around case.

3	31:23	**Reserved** Format: MBZ
	22	**Tile Type** Format: bool 0 : Real HCP Tile Mode 1 : Virtual HCP Tile Mode **Programming Notes** This field is only used when SFC Pipe Mode is HCP-to-SFC. In Real HCP Tile Mode, video streams defines the tile boundary. In Virtual HCP Tile Mode, driverstreams defines the tile boundary.
	21:20	**BitDepth** This field is valid only for output formats P016/Y216/Y416. This field is used to specify how many of the LSB bits have valid data. Value | Name | Description 0 | 10BitFormat | Higher 10 bits are valid and lower 6 bits are 0
	19	**CSC Enable** This field is set when YUV to RGB or RGB to YUV conversion is required or the RGB/YUV range conversion is required. CSC conversion matrix need to be programmed accordingly. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	18	**Color Fill Enable** **Programming Notes** This field could be enabled only if the scaled resolution is smaller than the output/display resolution. If enabled, HW will fill the gap with programmable pixel values. Else, nothing will be filled in the gap region. **Usage:** Color fill must be enabled for the first time/pass when a new surface is allocated/ used. Optional for subsequence frames since the gap region is filled with default pixels by prior

SFC_STATE_BODY

		passes. In scalability mode ie. (SFC Engine Mode != 00), gray fill should be set only for left mosttile and for other tiles it should be disabled.
	17:16	**Rotation Mode**

Format:	U2

Value	Name
00b	0 (degrees)
01b	90 Clockwise
10b	180 Clockwise
11b	270 Clockwise

Programming Notes
SFC rotation (90, 180 and 270) should be set only on VEBox input mode and SFC output set to TileY. Restriction: • For Integral Image Mode, this field is Reserved and MBZ. • For VDBox Mode, this field is Reserved and MBZ. • For linear or TileX SFC output, this field is Reserved and MBZ.

15:13 **Reserved**

Format:	MBZ

12 **Chroma Upsampling Enable**

This field enables the high-quality UV channel upsampler prior to IEF filter process. This field should be disabled when the source pixels and output pixels are kept with the same chroma sub-sample type and IEF is disabled.

Restriction
For Integral Image Mode, this field is Reserved and MBZ.

11 **Reserved**

Format:	MBZ

10 **RGB Adaptive**

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.
0: Disable the RGB Adaptive equation and use G-Ch directly for adaptive filter
1: Enable the RGB Adaptive filter using the equation (Y=(R+2G+B)»2)

9 **Bypass X Adaptive Filtering**

Value	Name	Description
0	Enable X Adaptive Filtering	

SFC_STATE_BODY

	1 | Disable X Adaptive Filtering | The X direction will use **Default Sharpness Level** to blend between the smooth and sharp filters rather than the calculated value.

8 **Bypass Y Adaptive Filtering**

Value	Name	Description
0	Enable Y Adaptive Filtering	
1	Disable Y Adaptive Filtering	The Y direction will use **Default Sharpness Level** to blend between the smooth and sharp filters rather than the calculated value.

7 **AVS Scaling Enable**

Value	Name	Description
1	Enable	
0	Disable	The scaling factor is ignored and a scaling ratio of 1:1 is assumed.

6 **Adaptive Filter for all Channels**

Value	Name	Description
1	Enable Adaptive Filter on UV/RB Channels	8-tap Adaptive Filter Mode is on
0	Disable Adaptive Filter on UV/RB Channels	

Programming Notes
The field can be enabled if 8-tap Adaptive filter mode is on. Else it should be disabled.

5:4 **AVS Filter Mode**

Value	Name
0	5x5 Poly-phase filter + Bilinear (adaptive)
1	8x8 poly-phase filter + Bilinear (adaptive)
2	Bilinear filter only
3	Reserved

Programming Notes
In VD-to-SFC mode, value of 1 is not allowed.

3 **Enable 8 tap for Chroma channels filtering**

This bit enables 8 tap filtering for Chroma Channels.

Programming Notes
8tap enable should only be enabled when SFC Input Chroma Sub-Sampling = 4 (ie. 444 input format to SFC).

2 **IEF4Smooth_Enable**

Value	Name	Description
0	**[Default]**	IEF is operating as a content adaptive detail filter based on 5x5 region.
1		IEF is operating as a content adaptive smooth filter based on 3x3 region

SFC_STATE_BODY

		Restriction For Integral Image Mode, this field is Reserved and MBZ.
	1	**Skin Tone Tuned IEF_Enable** Exists If: //IEF Enable = 1 **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	0	**IEF Enable** Value | Name | Description 1 | Enable | IEF Filter is Enabled 0 | Disable | IEF Filter is Disabled **Restriction** For Integral Image Mode and VD Mode, this field is Reserved and MBZ.
4	31:30	**Reserved** Format: MBZ
	29:16	**Source Region Height** Format: U14-1 **Source/Crop Region Height Minus 1 of the Input Frame in Unit of Pixel [13:0].** This field specifies the source/crop region of the input frame used for scaling of the graphic view. It defines the out-of-frame boundary used prior to AVS/IEF interpolation operation. The max value should be programmed to be equal or small than the input FrameHeightinMBminus 1 field. e.g. for 1920x1080 content, FrameHeightInMBsMinus1 is equal to1087 (1088 lines); however, the crop region height should be set to 1079(1080 lines). The last 8 lines are assumed to be not usable and should not be used as source pixels for Scaling or IEF operations. Otherwise, the bad pixels will breach and cause artifacts into the scaled output frame. Restriction : For Integral Image Mode, this field is Reserved and MBZ. Restriction : For AVS mode, the restriction is tied to chroma input format type: 420 - multiple of 2. 422/444/400 - no restrictions, except for AVS bypass case (ie. 1:1 scaling) where restriction is tied to chroma output format.**Min Resolution** is 128 pixels. **Max Resolution** is 16K pixels. **Restriction** In VD-to-SFC and HCP-to-SFC modes, this field must be programmed to same value as Input Frame Resolution Height.
	15:14	**Reserved**

SFC_STATE_BODY

		Format: MBZ
	13:0	**Source Region Width** Format: U14-1 **Source/Crop Region Width Minus 1 of the Input Frame in Unit of Pixel [13:0].** This field specifies the source/crop region of the input frame used for scaling of the graphic view. It defines the out-of-frame boundary used prior to AVS/IEF interpolation operation. The max value should be programmed to be equal or small than the input FrameWidthinMBminus 1 field. e.g. for 1920x1080 content, FrameWidthInMBsMinus1 is equal to1919 (1920 pixel wide); however, the crop region width should be set to less than 1909(1910 pixel wide). The last 10 pixels of the frame are assumed to be not usable and should not be used as source pixels for Scaling or IEF operations. Otherwise, the bad pixels will breach and cause artifacts into the scaled output frame. Restriction : For Integral Image Mode, this field is Reserved and MBZ. Restriction : For AVS mode, the restriction is tied to chroma input format type: 420 - multiple of 2. 422 - multiple of 2. 444/400 - no restrictions, except for AVS bypass case (ie. 1:1 scaling) where restriction is tied to chroma output format.**Min Resolution** is 128 pixels. **Max Resolution** is 16K pixels. **Restriction** In VD-to-SFC and HCP-to-SFC modes, this field must be programmed to same value as Input Frame Resolution Width.
5	31:30	**Reserved** Format: MBZ
	29:16	**Source Region Vertical Offset** Format: U14 **Description** **Vertical Offset Of The SRC Region Relative To The Starting Position Of The Input Frame In Unit Of Pixel [13:0]** This field specifies the vertical offset of the starting position of the scaled region relatives to the starting position (pixel 0,0) of the output frame. It defines the out-of-frame boundary used prior to AVS/IEF interpolation operation. This value should be set to zero if the starting corner of the crop region is same as the input frame region. The sum of this value and the src/crop region size heightminus1 must be programmed to be equal or small than the input FrameHeightinMBminus 1 field. Restriction : For Integral Image Mode, this field is Reserved and MBZ.

SFC_STATE_BODY

<table>
<tr><td></td><td></td><td>Restriction : For AVS mode, the restriction is tied to chroma input format type: 420 - multiple of 2. 422/444/400 - no restrictions.

Restriction
In VD-to-SFC and HCP-to-SFC modes, this field is Reserved and MBZ..</td></tr>
<tr><td></td><td>15:14</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>13:0</td><td>Source Region Horizontal Offset
Format: U14

Description
Horizontal Offset Of The SRC Region Relative To The Starting Position Of The Input Frame In Unit Of Pixel [13:0]
This field specifies the horizontal offset of the starting position of the scaled region relatives to the starting position (pixel 0,0) of the output frame. It defines the out-of-frame boundary used prior to AVS/IEF interpolation operation. This value should be set to zero if the starting corner of the crop region is same as the input frame region. The sum of this value and the src/crop region size widthminus1 must be programmed to be equal or small than the input FrameWidthinMBminus 1 field.
Restriction : For Integral Image Mode, this field is Reserved and MBZ.
Restriction : For AVS mode, the restriction is tied to chroma input format type: 420 - multiple of 2. 422 - multiple of 2. 444/400 - no restrictions.

Restriction
In VD-to-SFC and HCP-to-SFC modes, this field is Reserved and MBZ..</td></tr>
<tr><td>6</td><td>31:30</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>29:16</td><td>Output Frame Height
Format: U14-1

It is set to the value of the final output resolution of the graphic view. Since the max value support is 16k pixels, the max value allowed is 16K minus 1.
Restriction : For Integral Image Mode, this field is Reserved and MBZ.
Restriction : For AVS mode, the restriction is tied to chroma output format type: 420 - multiple of 2. 422/444/400 - no restrictions. Min Resolution is 128 pixels. Max Resolution is 16K pixels.</td></tr>
</table>

SFC_STATE_BODY

DWord	Bit	Description
	15:14	**Reserved** Format: MBZ
	13:0	**Output Frame Width** Format: U14-1 It is set to the value of the final output resolution of the graphic view. Since the max value support is 16k pixels, the max value allowed is 16K minus 1. Restriction : For Integral Image Mode, this field is Reserved and MBZ. Restriction : For AVS mode, the restriction is tied to chroma output format type: 420 - multiple of 2. 422 - multiple of 2. 444/400 - no restrictions. **Min Resolution** is 128 pixels. **Max Resolution** is 16K pixels.
7	31:30	**Reserved** Format: MBZ
	29:16	**Scaled Region Size Height** Format: U14-1 It is set to the height of the scaled region over the output frame of the graphic view. Restriction : For AVS mode, if rotation_mode = 0/180, the restriction is tied to chroma output format type: 420 - multiple of 2. 422/444/400 - no restrictions. For AVS mode, if rotation_mode = 90/270, the restriction is tied to chroma output format type: 420/422 - multiple of 2. 444/400 - no restrictions. **Min Resolution** is 128 pixels. **Max Resolution** is 16K pixels. **Programming Notes** The Max Value =< [The Output Frame Height Minus1].
	15:14	**Reserved** Format: MBZ
	13:0	**Scaled Region Size Width** Format: U14-1 It is set to the Width of the scaled region over the output frame of the graphic view. Restriction : For AVS mode, the restriction is tied to chroma output format type: 420 - multiple

SFC_STATE_BODY

DWord	Bit	Description
		of 2. 422 - multiple of 2. 444/400 - no restrictions. **Min Resolution** is 128 pixels. **Max Resolution** is 16K pixels **Programming Notes** The Max Value =< [The Output Frame Width Minus1].
8	31	**Reserved** Format: MBZ
	30:16	**Scaled Region Vertical Offset** Format: S14 **Vertical Offset (in pixels) Of The Scaled Region Relatives to The Starting Position Of The Output Frame In Unit Of Pixel [13:0]** This field specifies the vertical offset of the starting position of the scaled region relatives to the starting position (pixel 0,0) of the output frame. The gap between the scaled and output frame shall be filled by hardware with a set of programmed YUV/RGB values (Grey Bar). This value should be set to zero if the starting corner of the scaled region is same as the output frame region. The sum of this value and the scaled region size Heightminus1 must be programmed to be equal or small than the output FrameHeightinMBminus 1 field plus 16. **Programming Notes** This field must be set to zero if SFC Output surface format type is P010/P016. **Restriction** For Integral Mode, this field is reserved and MBZ For AVS mode, the restriction is tied to chroma output format type: 420 - multiple of 2. 422/444/400 - no restrictions. This field must be set to zero if SFC Output surface format type is NV12.
	15	**Reserved** Format: MBZ
	14:0	**Scaled Region Horizontal Offset** Format: S14 **Description** **Horizontal Offset (in pixels) Of The Scaled Region Relatives to The Starting Position Of The Output Frame In Unit Of Pixel [13:0]** This field specifies the horizontal offset of the starting position of the scaled region relatives to the starting position (pixel 0,0) of the output frame. The gap between the scaled and output frame shall be filled by hardware with a set of programmed YUV/RGB values (Grey Bar). This

SFC_STATE_BODY		
		value should be set to zero if the starting corner of the scaled region is same as the output frame region. The sum of this value and the scaled region size Widthminus1 must be programmed to be equal or small than the output FrameWidthinMBminus 1 field plus 16. Restriction : For Integral Image Mode, this field is Reserved and MBZ. Restriction : For AVS mode, the restriction is tied to chroma output format type: 420 - multiple of 2. 422 - multiple of 2. 444/400 - no restrictions. Restriction : This field must be set to zero if SFC output surface format type is NV12. This field must be set to zero if SFC Output surface format type is P010/P016.
9	31:26	**Reserved** Format: MBZ
	25:16	**Gray Bar Pixel - Y/R** Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This is the default value used to fill in the area between the scaled region and the output frame size (aka Gray Bar) in Y or R channel on the AYUV or RGBA domain respectively. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	15:10	**Reserved** Format: MBZ
	9:0	**Gray Bar Pixel - U/G** Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This is the default value used to fill in the area between the scaled region and the output frame size (aka Gray Bar) in U or G channel on the AYUV or RGBA domain respectively. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
10	31:26	**Reserved** Format: MBZ
	25:16	**Gray Bar Pixel - V/B** Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This is the default value used to fill in the area between the scaled region and the output frame size (aka Gray Bar) in V or B channel on the AYUV or RGBA domain respectively.

SFC_STATE_BODY

DWord	Bit	Description
		Restriction For Integral Image Mode, this field is Reserved and MBZ.
	15:10	**Reserved** Format: MBZ
	9:0	**Gray Bar Pixel - A** Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This is the default value used to fill in the area between the scaled region and the output frame size (aka Gray Bar) in A channel on the AYUV or RGBA domain respectively. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
11	31:26	**Reserved** Format: MBZ
	25:16	**UV Default value for V channel (For Mono Input Support)** Exists If: //Input NOT originated by VEBOX. Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This field specifies the UV default value fill in to the UV output channels when input from VDBOX is set to Monochrome. **Restriction** Not used when input is originated by VEBOX (Including Integral Image Mode).
	15:10	**Reserved** Format: MBZ
	9:0	**UV Default value for U channel (For Mono Input Support)** Exists If: //Input NOT originated by VEBOX. Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This field specifies the UV default value fill in to the UV output channels when input from VDBOX is set to Monochrome. **Restriction** Not used when input is originated by VEBOX (Including Integral Image Mode).
12	31:10	**Reserved**

SFC_STATE_BODY		
		Format: MBZ
	9:0	**Alpha Default Value** Format: 10-bit UNORM Type **Range:**[0.0, +1.0] This field specifies the Alpha default value fill into the alpha output channel when output format type is set to RGBA8/10. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
13	31:28	**Reserved** Format: MBZ
	27:5	**Scaling Factor Height** Format: U4.19 This field specifies the scaling ratio of the vertical sizes between the crop/source region and the scaled region. The destination pixel coordinate, y-axis, is multiplied with this scaling factor to mapping back to the source input pixel coordinate. The field specifies the ratio of crop height resolution/ scaled height resolution. This implies $1/sf_u$ in the equation.
	4:0	**Reserved** Format: MBZ
14	31:28	**Reserved** Format: MBZ
	27:5	**Scale Factor Width** Format: U4.19 This field specifies the scaling ratio of the horizontal sizes between the crop/source region and the scaled region. The destination pixel coordinate, x-axis, is multiplied with this scaling factor to mapping back to the source input pixel coordinate. The field specifies the ratio of crop width resolution/ scaled width resolution. This implies $1/sf_u$ in the equations above.
	4:0	**Reserved** Format: MBZ
15	31:0	**Reserved** Format: MBZ
16	31:12	**Output Frame Surface Base Address** Specifies the 4K byte aligned frame buffer address for outputting the scaled up/down image. Data is stored in Tile-Y format.

SFC_STATE_BODY

DWord	Bit	Description
		For Integral Image mode, the accumulated integral image values will be packed linear in this surface. **Programming Notes** This field is ignored if I-frame only mode is set to 0 (Disable).
	11:0	**Reserved** Format: MBZ
17	31:16	**Reserved** Format: MBZ
	15:0	**Output Frame Surface Base Address High** This field is for the upper range [47:32] of Output Frame Surface Base Address. For Integral Image mode, the accumulated integral image values will be packed linear in this surface.
18	31:15	**Reserved** Format: MBZ
	14:13	**Output Surface Tiled Mode** Format: U2 **For Media Surfaces:** This field specifies the tiled resource mode. Value / Name / Description: 0h / TRMODE_NONE / No tiled resource 1h / TRMODE_TILEYF / 3h / Reserved /
	12	**Output Frame Surface Base Address - Row Store Scratch Buffer Cache Select** Format: MBZ Value / Name / Description: 0 / Disable **[Default]** / This field must be programmed to 0 **Programming Notes** This must be set to 0
	11	**Reserved** Format: MBZ
	10	**Compression Type** Format: U1 This field is applicable only when Memory compression is enabled. Value / Name: 0 / Media Compression Enabled **[Default]**

intel

SFC_STATE_BODY

DWord	Bit	Description
		1 | Render Compression Enabled
	9	**Output Frame Surface Base Address - Memory Compression Enable** Format: | Enable Memory compression will be attempted for this surface.
	8:7	**Output Frame Surface Base Address - Arbitration Priority Control** Format: | HEVC_ARBITRATION_PRIORITY
	6:1	**Output Frame Surface Base Address - 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**
19	31:12	**AVS Line Buffer Surface Base Address** Specifies the 4K byte aligned frame buffer address for scratch space used for row/column store. This surface is used only if the internal buffer inside the SFC HW is not large enough to contain all row/column memory accesses. The AVS line buffer needs to be a valid address even for 1:1 scaling if SFC is used. **Programming Notes** This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00(Scalability workloads).
	11:0	**Reserved** Format: | MBZ
20	31:16	**Reserved** Format: | MBZ
	15:0	**AVS Line Buffer Surface Base Address High** This field is for the upper range [47:32] of AVS Line Buffer Surface Base Address. AVS Line buffer address needs to be valid even for 1:1 scaling if SFC is used. **Programming Notes** This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00.
21	31:15	**Reserved** Format: | MBZ
	14:13	**AVS Line Buffer Tiled Mode** Format: | U2 **For Media Surfaces:** This field specifies the tiled resource mode.

SFC_STATE_BODY

		Value / **Name** / **Description** table below

Value	Name	Description
0h	TRMODE_NONE	No tiled resource
1h	TRMODE_TILEYF	4KB tiled resources
2h	TRMODE_TILEYS	64KB tiled resources
3h	Reserved	

12 — **AVS Line Buffer Base Address - Row Store Scratch Buffer Cache Select**

Format:	U1

This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.

Value	Name	Description
0	LLC	Buffer going to LLC

Programming Notes
This suface does not support to put in Row Store Scratch Buffer. Must be set to 0

11 — **Reserved**

Format:	MBZ

10 — **AVS Line Buffer Base Address - Memory Compression Mode**

Default Value:	0 Horizontal Compression Mode
Format:	U1

Distinguishes vertical from horizontal compression. Please refer to vol1a? **Memory Data Formats chapter - section** ?media Memory Compression for more details.

Programming Notes
Memory compression is not supported. ?This bit is not used. Default to 0

9 — **AVS Line Buffer Base Address - Memory Compression Enable**

Default Value:	0 Disable
Format:	Enable

This bit control memory compression for this surface

Programming Notes
This bit must be set to 0 (Memory compression is not supported in this surface)

8:7 — **AVS Line Buffer Base Address - Arbitration Priority Control**

Format:	HEVC_ARBITRATION_PRIORITY

6:1 — **AVS Line Buffer Base Address - 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.

SFC_STATE_BODY

DWord	Bit	Description
		The field is defined to populate 64 different surface controls to be used concurrently. Related control registers can be updated during runtime.
	0	**Reserved**
22	31:12	**IEF Line Buffer Surface Base Address** Specifies the 4K byte aligned frame buffer address for the scratch space used for row/column store. This surface is used only if the internal buffer inside SFC HW is not large enough to contain all row/column memory accesses. **Programming Notes** This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	11:0	**Reserved** Format: MBZ
23	31:16	**Reserved** Format: MBZ
	15:0	**IEF Line Buffer Surface Base Address High** This field is for the upper range [47:32] of IEF Line Buffer Surface Base Address. **Programming Notes** This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
24	31:15	**Reserved** Format: MBZ
	14:13	**IEF Line Buffer Tiled Mode** Format: U2 **For Media Surfaces:** This field specifies the tiled resource mode.
	12	**IEF Line Buffer Base Address - Row Store Scratch Buffer Cache Select** Format: U1 This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.

IEF Line Buffer Tiled Mode values:

Value	Name	Description
0h	TRMODE_NONE	No tiled resource
1h	TRMODE_TILEYF	4KB tiled resources
2h	TRMODE_TILEYS	64KB tiled resources
3h	Reserved	

SFC_STATE_BODY

		Value / **Name** / **Description**: 0 / LLC / Buffer going to LLC **Programming Notes** This surface does not support Rowstore Scratch Buffer Cache. ?Must be programmed to 0
	11	**Reserved** Format: MBZ
	10	**IEF Line Buffer Base Address - Memory Compression Mode** Default Value: 0 Format: U1 Distinguishes vertical from horizontal compression. **Programming Notes** Must be zero; memory compression is not supported for this surface. Default to0
	9	**IEF Line Buffer Base Address - Memory Compression Enable** Default Value: 0 Disable Format: Enable **Programming Notes** Memory compression is not supported for this surface Must be 0.
	8:7	**IEF Line Buffer Base Address - Arbitration Priority Control** Format: HEVC_ARBITRATION_PRIORITY
	6:1	**IEF Line Buffer Base Address - 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**
25	31:12	**SFD Line Buffer Surface Base Address** Specifies the 4K byte aligned frame buffer address for the scratch space used for row/column store. This surface is used only if the internal buffer inside SFC HW is not large enough to contain all row/column memory accesses. **Programming Notes** This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00. **Restriction**

<table>
<tr><th colspan="3">SFC_STATE_BODY</th></tr>
<tr><td></td><td></td><td>For Integral Image Mode, this field is Reserved and MBZ.</td></tr>
<tr><td></td><td>11:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>26</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>15:0</td><td>SFD Line Buffer Surface Base Address High
This field is for the upper range [47:32] of SFD Line Buffer Surface Base Address.
Programming Notes
This field needs to be programmed separately and exclusively for each pipe when SFC engine mode is not programmed 2'b00.
Restriction
For Integral Image Mode, this field is Reserved and MBZ.</td></tr>
<tr><td>27</td><td>31:15</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>14:13</td><td>SFD Line Buffer Tiled Mode
Format: U2
For Media Surfaces: This field specifies the tiled resource mode.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>TRMODE_NONE</td><td>No tiled resource</td></tr>
<tr><td>1h</td><td>TRMODE_TILEYF</td><td>4KB tiled resources</td></tr>
<tr><td>2h</td><td>TRMODE_TILEYS</td><td>64KB tiled resources</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table></td></tr>
<tr><td></td><td>12</td><td>SFD Line Buffer Base Address - Row Store Scratch Buffer Cache Select
Format: U1
This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>LLC</td><td>Buffer going to LLC</td></tr>
<tr><td>1</td><td>Media Storage [Default]</td><td>Data will first cache in Media Storage</td></tr>
</table>
Programming Notes
This surface does not support Rowstore Scratch Buffer Cache. Must be programmed to 0</td></tr>
<tr><td></td><td>11</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>10</td><td>SFD Line Buffer Base Address - Memory Compression Mode
Default Value: 0</td></tr>
</table>

SFC_STATE_BODY

		Format: U1 Distinguishes vertical from horizontal compression. Please refer to vol1a **Memory Data Formats chapter - section** media Memory Compression for more details. **Programming Notes** Must be zero; memory compression is not supported for this surface. Default to0
	9	**SFD Line Buffer Base Address - Memory Compression Enable** Default Value: 0 Disable Format: Enable **Programming Notes** Memory compression is not supported for this surface Must be 0.
	8:7	**SFD Line Buffer Base Address - Arbitration Priority Control** Format: HEVC_ARBITRATION_PRIORITY
	6:1	**SFD Line Buffer Base Address - 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**
28	31:28	**Output Surface Format**
	27	**Output Surface Interleave Chroma Enable**
	26:20	**Reserved** Format: MBZ
	19:3	**Output Surface Pitch** Format: U17-1 Pitch in (Bytes - 1) This field specifies the surface pitch.

Value	Name	Description
[0,2047]	SURFTYPE_BUFFER Surfaces	[1B, 2048B]
[0,131071]	Other Linear Surfaces	[64B, 512KB] = [1 CL, 8K CLs]
[511,131071]	X-tiled Surface	[512B, 256KB] = [1tile, 512 tiles]
[127,131071]	Y-tiled surfaces	[128B,256KB] = [1 tile, 2048 tiles]

Programming Notes

SFC_STATE_BODY

- For tiled surfaces, the pitch must be a multiple of the tile width
- For Linear surfaces, the pitch must be a multiple of CL (64B) width
- 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.

If Media Memory Compression is enabled, the following max pitch size restriction must be honored. For larger resolution, Media Memory compression Must be disabled.

Tiling Mode	Pixel Format	Max Frame Width (bytes)	Max Frame Width (pixels)	Max Pitch (bytes)
Legacy 4K	8bpp	16k	16k	16k + 127
	16bpp	16k	8k	16k + 127
	32bpp	16k	4k	16k + 127
	64bpp	16k	2k	16k + 127
	128bpp	16k	1k	16k + 127
TileYF	8bpp	8k	8k	8k + 63
	16bpp	16k	8k	16k + 127
	32bpp	16k	4k	16k + 127
	64bpp	16k	2k	16k + 255
	128bpp	16k	1k	16k + 255
TileYS	8bpp	16k	16k	16k + 255
	16bpp	16k	8k	16k + 511
	32bpp	16k	4k	16k + 511
	64bpp	16k	2k	16k + 1023
	128bpp	16k	1k	16k + 1023

2

Output Surface Half Pitch For Chroma

Exists If:	//PLANAR Surface Formats Only
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.

1

Output Surface Tiled

Format:	Boolean

This field specifies whether the surface is tiled.

Value	Name	Description
1	True	Tiled
0	FALSE	Linear

Programming Notes

SFC_STATE_BODY

		• 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	**Output Surface Tile Walk** Format: SFC_Tile_Walk 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. **Value** / **Name** 0 – TILEWALK_XMAJOR 1 – 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. This field is ignored when the surface is linear.
29	31:30	**Reserved** Format: MBZ
	29:16	**Output Surface X Offset For U** Exists If: //PLANAR Surface Formats Only Format: U14 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. **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must be zero.
	15:14	**Reserved** Format: MBZ
	13:0	**Output Surface Y Offset For U** Exists If: //PLANAR Surface Formats Only Format: U14 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. **Programming Notes**

SFC_STATE_BODY		
		For PLANAR_420 and PLANAR_422 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs.
30	31:30	**Reserved** Format: MBZ
	29:16	**Output Surface X Offset For V** Exists If: //PLANAR Surface Formats with Interleaved Chroma Disable Format: U14 Pixel Offset 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.
	15:14	**Reserved** Format: MBZ
	13:0	**Output Surface Y Offset For V** Exists If: //PLANAR Surface Formats with Interleaved Chroma Disable Format: U14 Pixel Offset This field specifies the vertical offset in rows 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.
31	31:1	**Reserved**
32	31:0	**Reserved**
33	31:30	**Reserved** Format: MBZ
	29:16	**SourceEndX** Format: U14 Indicates the X-direction end location in the original input frame to SFC. For 420/422 this field should be in multiple of 2. **Programming Notes** This field is only programmed when SFC Pipe Mode is HCP-to-SFC This should be in sync with tile widthsize programmed inHCP_TILE_CODING command
	15:14	**Reserved**

SFC_STATE_BODY

DWord	Bit	Description
		Format: MBZ
	13:0	**SourceStartX** Format: U14 Indicates the X-direction start location in the original input frame to SFC. For 420/422 this field should be in multiple of 2. **Programming Notes** This field is only programmed when SFC Pipe Mode is HCP-to-SFC This should be in sync with tile width size programmed in HCP_TILE_CODING command
34	31:30	**Reserved** Format: MBZ
	29:16	**DestinationEndX** Format: U14 Indicates the X-direction end location in the output frame of SFC. **Programming Notes** This field is valid only in Scalability Mode. Please refer to SFC Programming Model to program this field.
	15:14	**Reserved** Format: MBZ
	13:0	**DestinationStartX** Format: U14 Indicate the X-direction start location in the output frame of SFC. **Programming Notes** This field is valid only in Scalability Mode. Please refer to SFC Programming Model to program this field.
35	31:29	**Reserved** Format: MBZ
	28:5	**Xphaseshift** Format: s4.19 Xphaseshift would be programmed to do output centering in x-direction. **Programming Notes** This field allows user to program the horizontal address/coordinate of the center of scaling. For the valid programming where the scaling center is within the original image, the numerical/floatingvalue for the Xphaseshift would be c_x*(1/sf_hor - 1). The sf_hor in the above equation is the numerical/floating value of the horizontal scaling factor while c_x corresponds to the normalized horizontal coordinate of the scaling center (i.e., 0 <= c_x <= 1). For example, if (c_x, c_y) = (0, 0), the scaling center would be the legacy top-left mode while (c_x, c_y) = (0.5, 0.5) would be the center mode which corresponds to the default of many other display solutions.

SFC_STATE_BODY		
	4:0	**Reserved** Format: MBZ
36	31:29	**Reserved** Format: MBZ
	28:5	**Yphaseshift** Format: s4.19 Yphaseshift would be programmed to do output centering in y-direction. **Programming Notes** This field allows user to program the verticaladdress/coordinate of the center of scaling. For the valid programming where the scaling center is within the original image, the numerical/floatingvalue for the Yphaseshift would be c_y*(1/sf_ver- 1). The sf_verin the above equation is the numerical/floating value of the verticalscaling factor while c_ycorresponds to the normalized verticalcoordinate of the scaling center (i.e., 0 <= c_y<= 1). For example, if (c_x, c_y) = (0, 0), the scaling center would be the legacy top-left mode while (c_x, c_y) = (0.5, 0.5) would be the center mode which corresponds to the default of many other display solutions.
	4:0	**Reserved** Format: MBZ
37	31:12	**AVS Line Tile Buffer Surface Base Address** Specifies the 4K byte aligned frame buffer address for the scratch space used for row/column store. This surface is used only if the internal buffer inside SFC HW is not large enough to contain all row/column memory accesses. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	11:0	**Reserved** Format: MBZ
38	31:16	**Reserved** Format: MBZ
	15:0	**AVS Line Tile Buffer Surface Base Address High** This field is for the upper range [47:32] of AVS Line Tile Buffer Surface Base Address. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
39	31:15	**Reserved** Format: MBZ
	14:13	**AVS Line Tile Buffer Tiled Mode** Format: U2 **For Media Surfaces:** This field specifies the tiled resource mode. Value | Name | Description

SFC_STATE_BODY

0h	TRMODE_NONE	No tiled resource
1h	TRMODE_TILEYF	4KB tiled resources
2h	TRMODE_TILEYS	64KB tiled resources
3h	Reserved	

12 **AVS Line Tile Buffer Base Address - Row Store Scratch Buffer Cache Select**

Format:	U1

This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.

Value	Name	Description
0	LLC **[Default]**	Buffer going to LLC

Programming Notes
This surface does not support Rowstore Scratch Buffer Cache. Must be programmed to 0

11 **Reserved**

Format:	MBZ

10 **AVS Line Tile Buffer Base Address - Memory Compression Mode**

Default Value:	0
Format:	U1

Distinguishes vertical from horizontal compression. Please refer to vol1a **Memory Data Formats chapter - section** media Memory Compression for more details.

Programming Notes
Must be zero; memory compression is not supported for this surface. Default to0

9 **AVS Line Tile Buffer Base Address - Memory Compression Enable**

Default Value:	0 Disable
Format:	Enable

Programming Notes
Memory compression is not supported for this surface Must be 0.

8:7 **AVS Line Tile Buffer Base Address - Arbitration Priority Control**

Format:	HEVC_ARBITRATION_PRIORITY

6:1 **AVS Line Tile Buffer Base Address - 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.

<table>
<tr><th colspan="3">SFC_STATE_BODY</th></tr>
<tr><td></td><td>0</td><td>Reserved</td></tr>
<tr><td rowspan="2">40</td><td>31:12</td><td>IEF Line Tile Buffer Surface Base Address
Specifies the 4K byte aligned frame buffer address for the scratch space used for row/column store. This surface is used only if the internal buffer inside SFC HW is not large enough to contain all row/column memory accesses.
<table><tr><th>Restriction</th></tr><tr><td>For Integral Image Mode, this field is Reserved and MBZ.</td></tr></table></td></tr>
<tr><td>11:0</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td rowspan="2">41</td><td>31:16</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>15:0</td><td>IEF Line Tile Buffer Surface Base Address High
This field is for the upper range [47:32] of IEF Line Tile Buffer Surface Base Address.
<table><tr><th>Restriction</th></tr><tr><td>For Integral Image Mode, this field is Reserved and MBZ.</td></tr></table></td></tr>
<tr><td rowspan="5">42</td><td>31:15</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>14:13</td><td>IEF Line Tile Buffer Tiled Mode
<table><tr><td>Format:</td><td>U2</td></tr></table>
For Media Surfaces: This field specifies the tiled resource mode.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>TRMODE_NONE</td><td>No tiled resource</td></tr>
<tr><td>1h</td><td>TRMODE_TILEYF</td><td>4KB tiled resources</td></tr>
<tr><td>2h</td><td>TRMODE_TILEYS</td><td>64KB tiled resources</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr></table></td></tr>
<tr><td>12</td><td>IEF Line Tile Buffer Base Address - Row Store Scratch Buffer Cache Select
<table><tr><td>Format:</td><td>U1</td></tr></table>
This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>LLC [Default]</td><td>Buffer going to LLC</td></tr></table>
<table><tr><th>Programming Notes</th></tr><tr><td>This surface does not support Rowstore Scratch Buffer Cache. Must be programmed to 0</td></tr></table></td></tr>
<tr><td>11</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>10</td><td>IEF Line Tile Buffer Base Address - Memory Compression Mode
<table><tr><td>Default Value:</td><td>0</td></tr></table></td></tr>
</table>

SFC_STATE_BODY

		Format: U1 Distinguishes vertical from horizontal compression. Please refer to vol1a **Memory Data Formats chapter - section** media Memory Compression for more details. **Programming Notes** Must be zero; memory compression is not supported for this surface. Default to0
	9	**IEF Line Tile Buffer Base Address - Memory Compression Enable** Default Value: 0 Disable Format: Enable **Programming Notes** Memory compression is not supported for this surface Must be 0.
	8:7	**IEF Line Tile Buffer Base Address - Arbitration Priority Control** Format: HEVC_ARBITRATION_PRIORITY
	6:1	**IEF Line Tile Buffer Base Address - 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**
43	31:12	**SFD Line Tile Buffer Surface Base Address** Specifies the 4K byte aligned frame buffer address for the scratch space used for row/column store. This surface is used only if the internal buffer inside SFC HW is not large enough to contain all row/column memory accesses. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
	11:0	**Reserved** Format: MBZ
44	31:16	**Reserved** Format: MBZ
	15:0	**SFD Line Tile Buffer Surface Base Address High** This field is for the upper range [47:32] of SFD Line Tile Buffer Surface Base Address. **Restriction** For Integral Image Mode, this field is Reserved and MBZ.
45	31:15	**Reserved** Format: MBZ

SFC_STATE_BODY

<table>
<tr><td></td><td>14:13</td><td>
SFD Line Tile Buffer Tiled Mode
<table><tr><td>Format:</td><td>U2</td></tr></table>
For Media Surfaces: This field specifies the tiled resource mode.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>TRMODE_NONE</td><td>No tiled resource</td></tr>
<tr><td>1h</td><td>TRMODE_TILEYF</td><td>4KB tiled resources</td></tr>
<tr><td>2h</td><td>TRMODE_TILEYS</td><td>64KB tiled resources</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>12</td><td>
SFD Line Tile Buffer Base Address - Row Store Scratch Buffer Cache Select
<table><tr><td>Format:</td><td>U1</td></tr></table>
This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td>LLC [Default]</td><td>Buffer going to LLC</td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This surface does not support Rowstore Scratch Buffer Cache. Must be programmed to 0</td></tr>
</table>
</td></tr>
<tr><td></td><td>11</td><td>
Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td></td><td>10</td><td>
SFD Line Tile Buffer Base Address - Memory Compression Mode
<table><tr><td>Default Value:</td><td>0</td></tr><tr><td>Format:</td><td>U1</td></tr></table>
Distinguishes vertical from horizontal compression. Please refer to vol1a Memory Data Formats chapter - section media Memory Compression for more details.
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>Must be zero; memory compression is not supported for this surface. Default to0</td></tr>
</table>
</td></tr>
<tr><td></td><td>9</td><td>
SFD Line Tile Buffer Base Address - Memory Compression Enable
<table><tr><td>Default Value:</td><td>0 Disable</td></tr><tr><td>Format:</td><td>Enable</td></tr></table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>Memory compression is not supported for this surface
Must be 0.</td></tr>
</table>
</td></tr>
<tr><td></td><td>8:7</td><td>
SFD Line Tile Buffer Base Address - Arbitration Priority Control
<table><tr><td>Format:</td><td>HEVC_ARBITRATION_PRIORITY</td></tr></table>
</td></tr>
<tr><td></td><td>6:1</td><td>
SFD Line Tile Buffer Base Address - Index to Memory Object Control State (MOCS) Tables
<table><tr><td>Format:</td><td>U6</td></tr></table>
</td></tr>
</table>

SFC_STATE_BODY		
		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**

SIMD1 Untyped BUFFER Surface 64-Bit Address Payload

<table>
<tr><th colspan="3">MAP64B_USU_SIMD1 - SIMD1 Untyped BUFFER Surface 64-Bit Address Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 64
Default Value: 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.1</td><td>63:0</td><td>U0
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U64</td></tr></table>Specifies the U channel for slot [0]</td></tr>
</table>

SIMD8 Dual Source Render Target Data Payload

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

MDP_RTW_8DS - SIMD8 Dual Source Render Target Data Payload

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 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

SIMD8 LOD Message Address Payload Control

MACR_LOD_SIMD8 - SIMD8 LOD Message Address Payload Control		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0.0	31:0	**Slot0 LOD** Format: MACD_LOD Specifies the LOD for slot 0
0.1	31:0	**Slot1 LOD** Format: MACD_LOD Specifies the LOD for slot 1
0.2	31:0	**Slot2 LOD** Format: MACD_LOD Specifies the LOD for slot 2
0.3	31:0	**Slot3 LOD** Format: MACD_LOD Specifies the LOD for slot 3
0.4	31:0	**Slot4 LOD** Format: MACD_LOD Specifies the LOD for slot 4
0.5	31:0	**Slot5 LOD** Format: MACD_LOD Specifies the LOD for slot 5

MACR_LOD_SIMD8 - SIMD8 LOD Message Address Payload Control		
0.6	31:0	**Slot6 LOD** Format: MACD_LOD Specifies the LOD for slot 6
0.7	31:0	**Slot7 LOD** Format: MACD_LOD Specifies the LOD for slot 7

SIMD8 MSAA Typed Surface 32-Bit Address Payload

MAP32B_MSAA_TS_SIMD8 - SIMD8 MSAA Typed Surface 32-Bit Address Payload		
Source:	BSpec	
Size (in bits):	1280	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0.0	31:0	**Slot0 Sample Number** Format: MACD_MSAA_SN Specifies the sample number for slot 0
0.1	31:0	**Slot1 Sample Number** Format: MACD_MSAA_SN Specifies the sample number for slot 1
0.2	31:0	**Slot2 Sample Number** Format: MACD_MSAA_SN Specifies the sample number for slot 2
0.3	31:0	**Slot3 Sample Number** Format: MACD_MSAA_SN Specifies the sample number for slot 3
0.4	31:0	**Slot4 Sample Number** Format: MACD_MSAA_SN Specifies the sample number for slot 4
0.5	31:0	**Slot5 Sample Number**

MAP32B_MSAA_TS_SIMD8 - SIMD8 MSAA Typed Surface 32-Bit Address Payload

		Format: **MACD_MSAA_SN** Specifies the sample number for slot 5
0.6	31:0	**Slot6 Sample Number** Format: **MACD_MSAA_SN** Specifies the sample number for slot 6
0.7	31:0	**Slot7 Sample Number** Format: **MACD_MSAA_SN** Specifies the sample number for slot 7
1.0-1.7	255:0	**U** Format: **MACR_32b** Specifies the U channel for slots [7:0]
2.0-2.7	255:0	**V** Format: **MACR_32b** Specifies the V channel for slots [7:0]
3.0-3.7	255:0	**R** Format: **MACR_32b** Specifies the R channel for slots [7:0]
4.0-4.7	255:0	**LOD** Format: **MACR_LOD_SIMD8** Specifies the LOD for slots [7:0]

SIMD8 Render Target Data Payload

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

SIMD8 Typed Surface 32-Bit Address Payload

MAP32B_TS_SIMD8 - SIMD8 Typed Surface 32-Bit Address Payload		
Source:	BSpec	
Size (in bits):	1024	
Default Value:	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]
2.0-2.7	255:0	**R** Format: MACR_32b Specifies the R channel for slots [7:0]
3.0-3.7	255:0	**LOD** Format: MACR_LOD_SIMD8 Specifies the LOD for slots [7:0]

SIMD8 Untyped BUFFER Surface 32-Bit Address Payload

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]

SIMD8 Untyped BUFFER Surface 64-Bit Address Payload

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]

SIMD8 Untyped STRBUF Surface 32-Bit Address Payload

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]

SIMD8 URB Channel Mask Message Address Payload

MAPU_CMASK_SIMD8 - SIMD8 URB Channel Mask Message 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	**Per Slot Channel Mask** Format: MACD_URB_CMASK[8] Each slot's mask field is combined with the execution mask to determine which Dwords are written to the URB.

SIMD8 URB Offset Message Address Payload

MAPU_SIMD8 - SIMD8 URB Offset Message 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	**Slot Offset** Format: U32[8] Each slot's offset field is added to the **Global Offset**(specified in the message descriptor) and the slot's URB Handle (specified in the message header)to generate the URB address for this access. This offset and the Global Offset are specified as Oword units (128 bits). Value: [0-2047] — Name:

Value	Name
[0-2047]	

SIMD16 Render Target Data Payload

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_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

MDP_RTW_16 - SIMD16 Render Target Data Payload		
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
7.0-7.7	255:0	**Alpha[15:7]** Format: MDP_DW_SIMD8 Slots [15:7] Alpha

SIMD16 Untyped BUFFER Surface 32-Bit Address Payload

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
0.0-1.7	511:0	**U** Format: U32[16] Specifies the U channel for slots [15:0]

SIMD16 Untyped BUFFER Surface 64-Bit Address Payload

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]

SIMD16 Untyped STRBUF Surface 32-Bit Address Payload

MAP32B_USUV_SIMD16 - SIMD16 Untyped STRBUF Surface 32-Bit Address Payload		
Source:	BSpec	
Size (in bits):	1024	
Default Value:	0x00000000, 0x00000000	

DWord	Bit	Description
0.0-1.7	511:0	**U** Format: U32[16] Specifies the U channel for slots [15:0]
2.0-3.7	511:0	**V** Format: U32[16] Specifies the V channel for slots [15:0]

SIMD 32-Bit Address Payload Control

MACR_32B - SIMD 32-Bit Address Payload Control		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0	31:0	**Offset0** Format: U32 Specifies the address offset for slot 0 in this payload register.
0.1	31:0	**Offset1** Format: U32 Specifies the address offset for slot 1 in this payload register.
0.2	31:0	**Offset2** Format: U32 Specifies the address offset for slot 2 in this payload register.
0.3	31:0	**Offset3** Format: U32 Specifies the address offset for slot 3 in this payload register.
0.4	31:0	**Offset4** Format: U32 Specifies the address offset for slot 4 in this payload register.
0.5	31:0	**Offset5** Format: U32 Specifies the address offset for slot 5 in this payload register.
0.6	31:0	**Offset6**

MACR_32B - SIMD 32-Bit Address Payload Control

<table>
<tr><td></td><td></td><td><table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U32</td></tr></table>Specifies the address offset for slot 6 in this payload register.</td></tr>
<tr><td>0.7</td><td>31:0</td><td>Offset7<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U32</td></tr></table>Specifies the address offset for slot 7 in this payload register.</td></tr>
</table>

SIMD 64-Bit Address Payload Control

<table>
<tr><th colspan="3">MACR_64B - SIMD 64-Bit Address Payload Control</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.1</td><td>63:0</td><td>Offset0
Format: U64
Specifies the address offset for slot 0 in this payload register.</td></tr>
<tr><td>0.2-0.3</td><td>63:0</td><td>Offset1
Format: U64
Specifies the address offset for slot 1 in this payload register.</td></tr>
<tr><td>0.4-0.5</td><td>63:0</td><td>Offset2
Format: U64
Specifies the address offset for slot 2 in this payload register.</td></tr>
<tr><td>0.6-0.7</td><td>63:0</td><td>Offset3
Format: U64
Specifies the address offset for slot 3 in this payload register.</td></tr>
</table>

SIMD8 32-Bit Address Payload

<table>
<tr><th colspan="3">MAP32B_SIMD8 - SIMD8 32-Bit Address Payload</th></tr>
<tr><td>Source:</td><td colspan="2">BSpec</td></tr>
<tr><td>Size (in bits):</td><td colspan="2">256</td></tr>
<tr><td>Default Value:</td><td colspan="2">0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-0.7</td><td>255:0</td><td>Offset[7:0]
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>MACR_32b</td></tr></table>Specifies the address offset for Slots [7:0].</td></tr>
</table>

SIMD8 64-Bit Address Payload

MAP64B_SIMD8 - SIMD8 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	**Offset[3:0]** Format: **MACR_64b** Specifies the address offset for slots [3:0].
1.0-1.7	255:0	**Offset[7:4]** Format: **MACR_64b** Specifies the address offset for slots [7:4].

SIMD16 32-Bit Address Payload

<table>
<tr><th colspan="3">MAP32B_SIMD16 - SIMD16 32-Bit Address Payload</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 512
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0.0-1.7</td><td>511:0</td><td>Offset
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U32[16]</td></tr></table>Specifies the address offset for slots [15:0].</td></tr>
</table>

SIMD16 64-Bit Address Payload

MAP64B_SIMD16 - SIMD16 64-Bit Address Payload

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

DWord	Bit	Description
0.0-0.7	255:0	**Offset[3:0]** Format: MACR_64b Specifies the address offsets for slots [3:0].
1.0-1.7	255:0	**Offset[7:4]** Format: MACR_64b Specifies the address offsets for slots [7:4].
2.0-2.7	255:0	**Offset[11:8]** Format: MACR_64b Specifies the address offsets for slots [11:8].
3.0-3.7	255:0	**Offset[15:12]** Format: MACR_64b Specifies the address offsets for slots [15:12].

SIMD Mode 2 Message Descriptor Control Field

MDC_SM2 - SIMD Mode 2 Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**SIMD Mode** Specifies the SIMD mode of the message (number of slots processed)

Value	Name	Description
00h	SIMD8	SIMD8
01h	SIMD16	SIMD16

SIMD Mode 3 Message Descriptor Control Field

MDC_SM3 - SIMD Mode 3 Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	1:0	**SIMD Mode** Specifies the SIMD mode of the message (number of slots processed)

Value	Name	Description
00h	Reserved	Ignored
01h	SIMD16	SIMD16
02h	SIMD8	SIMD8
03h	Reserved	Ignored

SLICE_HASH_TABLE

SLICE_HASH_TABLE - SLICE_HASH_TABLE

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

Description
The slice hash table state is stored as an array tables (2 slices-8 slices), each of which contains the 32 DWords described here. 16x16 lookup table for slice indexed by lower bits of pixel block address. Each entry in the table indicates the **physical**slice_id to map that XY. If a slice is disabled, then it must not be present in the table. Entries in the table that point to disabled slice will be mapped to lowest enabled slice_id.

DWord	Bit	Description
0..31	1023:0	**Slice Hashing Table Entries** Format: U4[16][16]

SLM Block Message Header

MH_SLM_GO - SLM Block Message Header		
Source:	EuSubFunctionDataPort0	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..1	63:0	**Reserved** Format: MBZ Ignored
2	31:0	**Global Offset** Format: U32 Specifies the global element index into the buffer, in units of Hwords, Owords, Dwords, or Bytes (depending on the message). **Programming Notes** The Global Offset for Aligned Block operations is specified as a Dword-aligned byte offset (offset bits [1:0] = 0), or Oword-aligned byte offset (offset bits [3:0]=0), or Hword-aligned byte offset (offset bits [4:0]=0).
3..7	159:0	**Reserved** Format: MBZ Ignored

Slot Group 2 Message Descriptor Control Field

MDC_SG2 - Slot Group 2 Message Descriptor Control Field		
Source: BSpec Size (in bits): 1 Default Value: 0x00000000		
DWord	**Bit**	**Description**
0	0	**SIMD Mode** Controls which 8 bits of Pixel/Sample Mask in the message header are ANDed with the execution mask to determine which slots are accessed. This field is ignored if the header is not present.

Value	Name	Description
00h	SG8L	Use low 8 slots
01h	SG8U	Use high 8 slots

Slot Group 3 Message Descriptor Control Field

MDC_SG3 - Slot Group 3 Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	

DWord	Bit	Description
0	1:0	**SIMD Mode** Controls which 8 bits of Pixel/Sample Mask in the message header are ANDed with the execution mask to determine which slots are accessed. This field is ignored if the header is not present.

Value	Name	Description
00h	Reserved	Ignored
01h	SG8L	Use low 8 slots
02h	SG8U	Use high 8 slots
03h	Reserved	Ignored

Slot Group Select Render Cache Message Descriptor Control Field

MDC_RT_SGS - Slot Group Select Render Cache Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**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
00h	SLOTGRP_LO	Choose bypassed data for slots 15:0
01h	SLOTGRP_HI	Choose bypassed data for slots 31:16

SO_DECL

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: Enable 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-

SO_DECL

		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	Name
[0,32]	
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:	U4 Format: Enable[4] 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	SO_DECL_COMPMASK_NONE **[Default]**
xxx1b	SO_DECL_COMPMASK_X
xx1xb	SO_DECL_COMPMASK_Y
x1xxb	SO_DECL_COMPMASK_Z
1xxxb	SO_DECL_COMPMASK_W

SO_DECL_ENTRY

SO_DECL_ENTRY		
Source:	RenderCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0..1	63:48	**Stream 3 Decl** Format: **SO_DECL** This field contains Stream 3 SO_DECL [n]
	47:32	**Stream 2 Decl** Format: **SO_DECL** This field contains Stream 2 SO_DECL [n]
	31:16	**Stream 1 Decl** Format: **SO_DECL** This field contains Stream 1 SO_DECL [n]
	15:0	**Stream 0 Decl** Format: **SO_DECL** This field contains Stream 0 SO_DECL [n]

Split_coding_unit_flags

Split_coding_unit_flags		
Source:	VideoCS	
Size (in bits):	21	
Default Value:	0x00000000	
Contains the split level flags, level 0 through 2.		
DWord	**Bit**	**Description**
0	20	**Split_flag_level0** Format: U1
	19:16	**Split_flag_level1** Format: U4 [19:16] is in raster order. Bit16 is for partition0 in raster order.
	15:12	**Split_flag_level2 level1part3** Format: U4 Split flags for bit19 partition. [15:12] is in raster order. Bit12 is for partition0 in raster order.
	11:8	**Split_flag_level2 level1part2** Format: U4 Split flags for bit18 partition. [11:8] is in raster order. Bit8 is for partition0 in raster order.
	7:4	**Split_flag_level2 level1part1** Format: U4 Split flags for bit17 partition. [7:4] is in raster order. Bit4 is for partition0 in raster order.
	3:0	**Split_flag_level2 level1part0** Format: U4 Split flags for bit16 partition. [3:0] is in raster order. Bit0 is for partition0 in raster order.

SplitBaseAddress4KByteAligned

SplitBaseAddress4KByteAligned		
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Specifies a 64-bit (48-bit canonical) 4K-byte aligned memory base address. GraphicsAddress is a 64-bit value [63:0], but only a portion of it is used by hardware. The upper reserved bits are ignored and MBZ.		
DWord	**Bit**	**Description**
0..1	63:12	**Base Address** Format: GraphicsAddress63-12
	11:0	**Reserved** Format: MBZ

SplitBaseAddress64ByteAligned

SplitBaseAddress64ByteAligned		
Source:	BSpec	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Specifies a 64-bit (48-bit canonical) 64-byte aligned memory base address.		
DWord	**Bit**	**Description**
0..1	63:6	**Base Address** Format: GraphicsAddress63-6
	5:0	**Reserved** Format: MBZ

SrcRegNum

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

Description
Register Number The register number for the operand. For a GRF register, is the part of a register address that aligns to a 256-bit (32-byte) boundary. For an ARF register, this field is encoded such that MSBs identify the architecture register type and LSBs provide the register number. An ARF register can only be dst or src0. Any src1 or src2 operands cannot be ARF registers. RegNum and SubRegNum together provide the byte-aligned address for the origin of a register region. RegNum provides bits 12:5 of that address. For one-source and two-source instructions, SubregNum provides bits 4:0. For three-source instructions, the address must be DWord-aligned; SubRegNum provides bits 4:2 of the address and bits 1:0 are zero. This field is present for the direct addressing mode and not present for indirect addressing. This field applies to both source and destination operands.

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 Execution Environment chapter for details.

SrcSubRegNum

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

Description
Subregister Number The subregister number for the operand. For a GRF register, is the byte address within a 256-bit (32-byte) register. For an ARF register, determines the sub-register number according to the specified encoding for the given architecture register. RegNum and SubRegNum together provide the byte-aligned address for the origin of a GRF register region. RegNum provides bits 12:5 of that address. For one-source and two-source instructions, SubregNum provides bits 4:0. For three-source instructions, the address must be DWord-aligned; SubRegNum provides bits 4:2 of the address and bits 1:0 are zero.

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 Execution Environment chapter for details.

Stateless Binding Table Index Message Descriptor Control Field

MDC_STATELESS - Stateless Binding Table Index Message Descriptor Control Field

Source:	BSpec
Size (in bits):	8
Default Value:	0x000000FF

DWord	Bit	Description
0	7:0	**Binding Table Index** Specifies the message is Stateless

Value	Name	Description
0FFh	A32_A64 **[Default]**	Specifies a A32 or A64 Stateless access that is locally coherent (coherent within a thread group)
0FDh	A32_A64_NC	Specifies a A32 or A64 Stateless access that is non-coherent (coherent within a thread).
Others	Reserved	Ignored

Restriction
When using A32_A64_NC, SW must ensure that 2 threads do not both access the same cache line (64B)

Stateless Block Message Header

MH_A32_GO - Stateless Block Message Header

Source:	EuSubFunctionDataPort0
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..1	63:0	**Reserved** Format: MBZ Ignored
2	31:0	**Global Offset** Format: U32 Specifies the global element index into the buffer, in units of Owords, Dwords, or Bytes (depending on the message). **Programming Notes** If the address offset calculated with the Buffer Base Address and Global Offset is greater than the PTSS size or the GeneralStateBufferSize, then the access is Out-of-Bounds.
3	31:0	**Per Thread Scratch Space** Format: MHC_PTSS Specifies amount of scratch space used by this thread, for Stateless bounds checking.
4	31:0	**Reserved** Format: MBZ Ignored
5	31:0	**Buffer Base Address** Format: MHC_A32_BBA **Description** Specifies the surface address offset page [31:10] for A32 stateless messages. Restriction : When using stateless A32 Data Port messages, General State Base Address[47:12] + Buffer Base Address[31:10] must be less than 2^48. It is illegal for this to be greater or equal than 2^48. **Programming Notes** This field is internally forced to 0 in hardware for CSR cycles.
6..7	63:0	**Reserved** Format: MBZ Ignored

Stateless Surface Message Header

MH1_A32 - Stateless Surface Message Header		
Source:	EuSubFunctionDataPort1	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0..4	159:0	**Reserved** Format: MBZ Ignored
5	31:0	**Buffer Base Address** Format: MHC_A32_BBA Specifies the surface address offset page [31:10] for A32 stateless messages.
6..7	63:0	**Reserved** Format: MBZ Ignored

Stateless Surface Pixel Mask Message Header

MH1_A32_PSM - Stateless Surface Pixel Mask Message Header

Source:	EuSubFunctionDataPort1
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..4	159:0	**Reserved** Format: MBZ Ignored
5	31:0	**Buffer Base Address** Format: MHC_A32_BBA Specifies the surface address offset page [31:10] for A32 stateless messages.
6	31:0	**Reserved** Format: MBZ Ignored
7	31:0	**Reserved** Format: MBO If the optional header is delivered, this field must be all ones.

Static Frame Control Parameters0

Static Frame Control Parameters0		
Source:	BSpec	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000	
Please note that DW0-3, correspond to DW96-99 of **WiGig Parameters**.		
DWord	**Bit**	**Description**
0	31:0	**Reserved** Format: MBZ
1	31:0	**Reserved** Format: MBZ
2	31:0	**Reserved** Format: MBZ
3	31:18	**Reserved** Format: MBZ
	17	**Skip Frame Enable** When display asserts skip frame flag and target QP is reached, WDBOX converts all MB of a frame to SKIP. No new reference picture is generated.
	16:14	**Reserved** Format: MBZ
	13:8	**QP Skip Threshold** Each MB is CR coded (provided CR is enabled feature) or Intra coded (CR feature is disabled) when both previous and current frame QPs are less than or equal to this field (threshold) and the current frame is static frame.
	7:0	**Reserved** Format: MBZ

Stencil Message Data Payload Register

MDPR_STENCIL - Stencil Message Data Payload Register

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

DWord	Bit	Description
0	31:24	**Stencil3** Format: U8 Stencil for Slot 3.
	23:16	**Stencil2** Format: U8 Stencil for Slot 2.
	15:8	**Stencil1** Format: U8 Stencil for Slot 1.
	7:0	**Stencil0** Format: U8 Stencil for Slot 0.
1	31:24	**Stencil7** Format: U8 Stencil for Slot 7.
	23:16	**Stencil6** Format: U8 Stencil for Slot 6.
	15:8	**Stencil5** Format: U8 Stencil for Slot 5.
	7:0	**Stencil4** Format: U8 Stencil for Slot 4.
2..7	191:0	**Reserved** Format: Ignore Ignored

Subset Atomic Integer Trinary Operation Message Descriptor Control Field

MDC_AOP3S - Subset Atomic Integer Trinary Operation Message Descriptor Control Field

Source:	BSpec
Size (in bits):	4
Default Value:	0x0000000E

DWord	Bit	Description
0	3:0	**Atomic Integer Operation Type** Specifies the atomic integer trinary operation to be performed

Value	Name	Description
0Eh	AOP_CMPWR **[Default]**	new_dst = (src0 == old_dst) ? src1 : old_dst
Others	Reserved	Ignored

Programming Notes
When Return Data Control is set, old_dst is returned.

Subset Reversed SIMD Mode 2 Message Descriptor Control Field

<table>
<tr><th colspan="3">MDC_SM2RS - Subset Reversed SIMD Mode 2 Message Descriptor Control Field</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 1
Default Value: 0x00000001</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>SIMD Mode
Specifies the SIMD mode of the message (number of slots processed)<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0h</td><td>Reserved</td><td>Not used</td></tr><tr><td>01h</td><td>SIMD8 [Default]</td><td>SIMD8</td></tr></table></td></tr>
</table>

Subset SIMD Mode 2 Message Descriptor Control Field

MDC_SM2S - Subset SIMD Mode 2 Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	1	
Default Value:	0x00000000	

DWord	Bit	Description
0	0	**SIMD Mode** Specifies the SIMD mode of the message (number of slots processed)

Value	Name	Description
00h	SIMD8	SIMD8
01h	Reserved	Ignored

Subset SIMD Mode 3 Message Descriptor Control Field

MDC_SM3S - Subset SIMD Mode 3 Message Descriptor Control Field		
Source:	BSpec	
Size (in bits):	2	
Default Value:	0x00000000	

DWord	Bit	Description
0	1:0	**SIMD Mode** Specifies the SIMD mode of the message (number of slots processed)

Value	Name	Description
00h	Reserved	Ignored
01h	Reserved	Ignored
02h	SIMD8	SIMD8
03h	Reserved	Ignored

Subspan Render Target Message Header Control

<table>
<tr><th colspan="3">MHC_RT_SUBSPAN - Subspan Render Target Message Header Control</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:16</td><td>Y
Format: U16
Y coordinate for upper-left pixel of this subspan</td></tr>
<tr><td>15:0</td><td>X
Format: U16
X coordinate for upper-left pixel of this subspan</td></tr>
</table>

Surface Binding Table Index Message Descriptor Control Field

MDC_BTS - Surface Binding Table Index Message Descriptor Control Field

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

DWord	Bit	Description
0	7:0	**Binding Table Index** Specifies the Binding Table index for the message, which must be a Surface State Model.

Value	Name	Description
00h-0EFh	BTS	Index of Binding Table State Surfaces
0F0h-0FAh	Reserved	Reserved
0FCh	SSO_BINDLESS	Specifies a Surface State Offset into the Bindless Surface State heap, supplied by the extended message descriptor.
0FBh	Reserved	
Others	Reserved	Ignored

Programming Notes
For Render Target Views, the Binding Table index need to be confined to the 00h to 0Fh range if Binding Table is not in the 256B alignment (18:8) mode. In the 256B alignment mode, the Binding Table Index need to be confined to the 00h to 3Fh range if slice common register 3. state cache perf fix disabled is set to 1, the entire range of BTI is supported.

Surface or Stateless Binding Table Index Message Descriptor Control Field

MDC_BTS_A32 - Surface or Stateless Binding Table Index Message Descriptor Control Field

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

DWord	Bit	Description
0	7:0	**Binding Table Index** Specifies the surface for the message, either Surface State Model or Stateless.

Value	Name	Description
00h-0EFh	BTS	Index of Binding Table State Surfaces
0F0h-0FAh	Reserved	Reserved
0FCh	SSO_BINDLESS	Specifies a Surface State Offset into the Bindless Surface State heap, supplied by the extended message descriptor.
0FBh	Reserved	
0FFh	A32_A64	Specifies a A32 or A64 Stateless access that is locally coherent (coherent within a thread group)
0FDh	A32_A64_NC	Specifies a A32 or A64 Stateless access that is non-coherent (coherent within a thread).
Others	Reserved	Ignored

Restriction
When using A32_A64_NC, SW must ensure that 2 threads do not both access the same cache line (64B)

Surface Pixel Mask Message Header

MH1_BTS_PSM - Surface Pixel Mask Message Header		
Source:	EuSubFunctionDataPort1	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0..6	223:0	**Reserved** Format: MBZ Ignored
7	31:0	**Reserved** Format: MBO Restriction : This field must be all ones when this header is required.

SW Generated BINDING_TABLE_STATE

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

Description
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 64-byte boundary.
Binding table indexes beyond 256 will automatically be mapped to entry 0 by the HW, w/ the exception of any messages which support the special indexes 240 through 255, inclusive.

DWord	Bit	Description
0	31:6	**Surface State Pointer** Format: SurfaceStateOffset[31:6] This 64-byte aligned address points to a surface state block. This pointer is relative to the **Surface State Base Address**
	5	**Reserved** Format: MBZ
	4:0	**Reserved** Format: MBZ

SZ OM S0A SIMD8 Render Target Data Payload

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

SZ OM S0A SIMD16 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
9.0-10.7	511:0	**Alpha**

MDP_RTW_ZMA16 - SZ OM S0A SIMD16 Render Target Data Payload

		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

SZ OM SIMD8 Dual Source Render Target Data Payload

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**

MDP_RTW_ZM8DS - SZ OM SIMD8 Dual Source Render Target Data Payload

		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** 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

SZ OM SIMD8 Render Target Data Payload

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

SZ OM SIMD16 Render Target Data Payload

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]**

MDP_RTW_ZM16 - SZ OM SIMD16 Render Target Data Payload

		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 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

SZ S0A SIMD8 Render Target Data Payload

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

SZ S0A SIMD16 Render Target Data Payload

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**

MDP_RTW_ZA16 - SZ S0A SIMD16 Render Target Data Payload

		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]** 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

SZ SIMD8 Dual Source Render Target Data Payload

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**

MDP_RTW_Z8DS - SZ SIMD8 Dual Source Render Target Data Payload

		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** 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

SZ SIMD8 Render Target Data Payload

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

SZ SIMD16 Render Target Data Payload

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]**

MDP_RTW_Z16 - SZ SIMD16 Render Target Data Payload		
		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
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 EOT Message Descriptor

TS_EOT - Thread EOT Message Descriptor		
Source:	RenderCS	
Size (in bits):	32	
Default Value:	0x02000000	

End of Thread message is sent to SFID_TS (07h) to end GPGPU and Media threads. The EU send instruction must also set the EOT control (bit 5) of the extended message descriptor.
This message is sent with single register message payload, which is a copy of the R0 thread payload sent with the thread dispatch.

DWord	Bit	Description
0	31:29	**Reserved** Format: MBZ
	28:25	**Message Length** Default Value: 1h One GRF Format: U4
	24:20	**Response Length** Default Value: 0h Zero GRF Format: U5
	19	**Header Present** Format: MBZ
	18:1	**Reserved** Format: MBZ
	0	**Message Type** Default Value: 0h End Thread Format: Opcode End of Thread message opcode

TILE_RECT

TILE_RECT		
Source:	RenderCS, PositionCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:16	**Tile Rectangle Y Min** Format: U16 Specifies Y Min coordinate of (inclusive) Tile Region used for tile rendering test. Value: [0,16383] — Name:
	15:0	**Tile Rectangle X Min** Format: U16 Specifies X Min coordinate of (inclusive) Tile Region used for tile rendering test. Value: [0,16383] — Name:
1	31:16	**Tile Rectangle Y Max** Format: U16 Specifies Y Max coordinate of (inclusive) Tile Region used for tile rendering test. Value: [0,16383] — Name:
	15:0	**Tile Rectangle X Max** Format: U16 Specifies X Max coordinate of (inclusive) Tile Region used for tile rendering test. Value: [0,16383] — Name:

TileW SIMD8 Data Control Dword

MDCD_TILEW - TileW SIMD8 Data Control Dword

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

DWord	Bit	Description
0	31:8	**Reserved** Format: MBZ Ignored
	7:0	**Red** Format: U8 Specifies the value of the red channel to be read or written.

TileW SIMD8 Data Payload

MDP_TILEW_SIMD8 - TileW SIMD8 Data Payload

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

DWord	Bit	Description
0.0	31:0	**Red Slot0** Format: MDCD_TileW Specifies the Slot 0 red channel data
0.1	31:0	**Red Slot1** Format: MDCD_TileW Specifies the Slot 1 red channel data
0.2	31:0	**Red Slot2** Format: MDCD_TileW Specifies the Slot 2 red channel data
0.3	31:0	**Red Slot3** Format: MDCD_TileW Specifies the Slot 3 red channel data
0.4	31:0	**Red Slot4** Format: MDCD_TileW Specifies the Slot 4 red channel data
0.5	31:0	**Red Slot5** Format: MDCD_TileW Specifies the Slot 5 red channel data
0.6	31:0	**Red Slot6** Format: MDCD_TileW Specifies the Slot 6 red channel data
0.7	31:0	**Red Slot7** Format: MDCD_TileW Specifies the Slot 7 red channel data

Timeout Data Payload

<table>
<tr><th colspan="3">MDP_TIMEOUT - Timeout Data Payload</th></tr>
<tr><td colspan="3">Source: EuSubFunctionGateway
Size (in bits): 256
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:10</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>9:0</td><td>Timeout Value
Format: U10
The amount of time GW should wait before sending a writeback message. This value is in terms of 1024 clocks. Thus, with a 1Ghz clock it would be approximately in terms of uS. 0 and 1 are illegal values since the actual timeout time can be short by up to 1 increment of the timeout value.</td></tr>
<tr><td>1..7</td><td>223:0</td><td>Reserved
Format: MBZ</td></tr>
</table>

Transpose Message Header

MH_T - Transpose Message Header

Source:	EuSubFunctionDataPort1
Size (in bits):	256
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0	31:0	**X Offset** Format: S31 X offset (in bytes) of the upper left corner of the block into the surface. **Programming Notes** This field must be a multiple of the Block Width in bytes. Must be DWORD aligned.
1	31:0	**Y Offset** Format: S31 Y offset (in rows) of the upper left corner of the block into the surface. **Programming Notes** This field must be a multiple of the Block Height.
2	31:0	**Block Dimensions** Format: MHC_BDIM The height and width of the block to transpose.
3..7	159:0	**Reserved** Format: MBZ Ignored

TS_CONSTANTS_REMOVED

TS_CONSTANTS_REMOVED	
Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0..7	255:0	**Push Constants for Texel Shader slot 0.**
8..15	255:0	**Push Constants for Texel Shader slot 1.**
16..23	255:0	**Push Constants for Texel Shader slot 2.**
24..31	255:0	**Push Constants for Texel Shader slot 3.**
32..39	255:0	**Push Constants for Texel Shader slot 4.**
40..47	255:0	**Push Constants for Texel Shader slot 5.**
48..55	255:0	**Push Constants for Texel Shader slot 6.**
56..63	255:0	**Push Constants for Texel Shader slot 7.**

TS_CONSTANTS_REMOVED

TS_CONSTANTS_REMOVED	
Source:	BSpec
Size (in bits):	2048
Default Value:	0x00000000, 0x00000000

DWord	Bit	Description
0..7	255:0	**Push Constants for Texel Shader slot 0.**
8..15	255:0	**Push Constants for Texel Shader slot 1.**
16..23	255:0	**Push Constants for Texel Shader slot 2.**
24..31	255:0	**Push Constants for Texel Shader slot 3.**
32..39	255:0	**Push Constants for Texel Shader slot 4.**
40..47	255:0	**Push Constants for Texel Shader slot 5.**
48..55	255:0	**Push Constants for Texel Shader slot 6.**
56..63	255:0	**Push Constants for Texel Shader slot 7.**

Untyped Write Channel Mask Message Descriptor Control Field

MDC_UW_CMASK - Untyped Write Channel Mask Message Descriptor Control Field

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

DWord	Bit	Description
0	3:0	**Mask** For untyped surface write messages, indicates which channels are included in the message payload and written to the surface.

Value	Name	Description
00h	RGBA **[Default]**	Red, Green, Blue, and Alpha are included
08h	RGB	Red, Green, and Blue are included
0Ch	RG	Red and Green are included
0Eh	R	Red is included
Others	Reserved	Ignored

Upper Oword Block Data Payload

MDP_OW1U - Upper Oword Block Data Payload		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0.0-0.3	127:0	**Reserved** Format: MBZ Ignored
0.4-0.7	127:0	**Oword** Format: U128 Specifies the upper Oword data element

URB Channel Mask Payload Control

MACD_URB_CMASK - URB Channel Mask Payload Control		
Source:	BSpec	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:24	**Reserved** Format: MBZ
	23:16	**Channel Mask** Format: Enable[8] For each channel present in the message data payload, the corresponding channel mask bit is ANDed with the slot's execution mask to determine the final channel enable. When final channel enable is 1 it indicates that Dword data will be written to the surface.
	15:0	**Reserved** Format: MBZ

URB Handle Message Header

MH_URB_HANDLE - URB Handle Message Header		
Source:	BSpec	
Size (in bits):	256	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

DWord	Bit	Description
0.0 This is the URB handle where slot 0 results are written or read.	31:0	**Handle 0** Format: **MHC_URB_HANDLE**
0.1 This is the URB handle where slot 1 results are written or read.	31:0	**Handle 1** Format: **MHC_URB_HANDLE**
0.2 This is the URB handle where slot 2 results are written or read.	31:0	**Handle 2** Format: **MHC_URB_HANDLE**
0.3 This is the URB handle where slot 3 results are written or read.	31:0	**Handle 3** Format: **MHC_URB_HANDLE**
0.4 This is the URB handle where slot 4 results are written or read.	31:0	**Handle 4** Format: **MHC_URB_HANDLE**
0.5 This is the URB handle where slot 5 results are written or read.	31:0	**Handle 5** Format: **MHC_URB_HANDLE**
0.6 This is the URB handle where slot 6 results are written or read.	31:0	**Handle 6** Format: **MHC_URB_HANDLE**
0.7 This is the URB handle where slot 7 results are written or read.	31:0	**Handle 7** Format: **MHC_URB_HANDLE**

URB Handle Message Header Control

MHC_URB_HANDLE - URB Handle Message Header Control		
Source:	BSpec	
Size (in bits):	0	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:25	Reserved
	24:0	Reserved

VC1

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

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	**Command Privilege Violation Error** This bit is set if a command classified as privileged is parsed in a non-privileged batch buffer. The command will be converted to a NOOP and parsing will continue.
	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: 1 — Name: (none) — Description: Instruction Error detected **Programming Notes** This error indications cannot be cleared except by reset (i.e., it is a fatal error).

VD_CONTROL_STATE_BODY

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

DWord	Bit	Description
0	31:1	**Reserved** Format: MBZ
	0	**Pipeline Initialization** This bit, when set, clears internal states for HCP Pipe if Media Instruction Opcode is set for HCP Pipe..
1	31:3	**Reserved** Format: MBZ
	2	**Memory Implicit Flush** This is used to initiate an implicit flush to memory to make sure all the memory request goes to memory. This should be programmed at the end of each frame after frame completion and before MI_FLUSH.
	1	**Scalable Mode Pipe Unlock** This is used for decoder/encoder pipe to unlock all the pipes for scalable mode. It should be programmed at the end of frame.
	0	**Scalable Mode Pipe Lock** This is used for decoder/encoder pipe to lock all the pipes for scalable mode. It should be programmed at the start of frame.

VDENC_64B_Aligned_Lower_Address

VDENC_64B_Aligned_Lower_Address		
Source:	VideoCS	
Size (in bits):	32	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	31:6	**Address** Format: GraphicsAddress[31:6] [31:6] This field is for the 26 bits of the lower address.
	5:0	**Reserved** Format: MBZ

VDENC_64B_Aligned_Upper_Address

VDENC_64B_Aligned_Upper_Address		
Source:	VideoCS	
Size (in bits):	32	
Default Value:	0x00000000	

DWord	Bit	Description
0	31:16	**Reserved** Format: MBZ
	15:0	**Address Upper DWord** Bits [47:32] of the Address. GraphicsAddress is 64-bit value [63:0], but only a portion of it is used by hardware. The uppermost bits [63:48] are ignored and MBZ.

VDENC_Block_8x8_4

VDENC_Block_8x8_4		
Source: VideoCS Size (in bits): 16 Default Value: 0x00000000		
DWord	**Bit**	**Description**
0	15:12	**Block8x8[3]** Format: U4
	11:8	**Block8x8[2]** Format: U4
	7:4	**Block8x8[1]** Format: U4
	3:0	**Block8x8[0]** Format: U4

VDENC_Colocated_MV_Picture

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

DWord	Bit	Description
0	31:0	**Lower Address** Format: VDENC_64B_Aligned_Lower_Address 64 byte aligned buffer. This field is used to write the DMV data by VDEnc. VDEnc only supports spatial direct prediction and not temporal direct. Hence the HW precomputes the ColZeroFlag per 8x8 block and writes 8-bits per macroblock. HW accumulates a CL worth of data before writing it out. This is a linear buffer, can be considered to be a frame level row-store. There is no read/write happening to the surface for any given frame. HW only writes to this surface for P-Frames. HW only reads from this surface for B-Frames. Size = 8-bits/ MB linear buffer.
1	31:0	**Upper Address** Format: VDENC_64B_Aligned_Upper_Address
2	31:0	**Picture Fields** Format: VDENC_Surface_Control_Bits

VDENC_Down_Scaled_Reference_Picture

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

DWord	Bit	Description
0	31:0	**Lower Address** Format: **VDENC_64B_Aligned_Lower_Address** Specifies the 64 byte aligned DownScaled reference frame buffer address. VDEnc supports upto 3 down-scaled reference pictures for HME search. (2 fwd and 1 bwd).
1	31:0	**Upper Address** Format: **VDENC_64B_Aligned_Upper_Address**
2	31:0	**Picture Fields** Format: **VDENC_Surface_Control_Bits**

VDENC_Original_Uncompressed_Picture

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

DWord	Bit	Description
0	31:0	**Lower Address** Format: **VDENC_64B_Aligned_Lower_Address** Specifies the 64 byte aligned frame buffer address for fetching YUV pixel data from the original uncompressed input picture for encoding. This field is only valid in encoding mode.
1	31:0	**Upper Address** Format: **VDENC_64B_Aligned_Upper_Address**
2	31:0	**Picture Fields** Format: **VDENC_Surface_Control_Bits**

VDENC_Reference_Picture

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

DWord	Bit	Description
0	31:0	**Lower Address** Format: **VDENC_64B_Aligned_Lower_Address** Specifies the 64 byte aligned reference frame buffer addresses corresponding to fwd reference index = 0 in the bitstream. VDEnc supports upto 4 reference pictures for IME search. (3 fwd and 1 bwd).
1	31:0	**Upper Address** Format: **VDENC_64B_Aligned_Upper_Address**
2	31:0	**Picture Fields** Format: **VDENC_Surface_Control_Bits**

VDENC_Reference_Surface_State_Fields

VDENC_Reference_Surface_State_Fields

Source:	VideoCS
Size (in bits):	128
Default Value:	0x00000000, 0x00000003, 0x00000000, 0x00000000

DWord	Bit	Description
0	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. Value: [0,16383] — Name: — Description: Representing heights [1,16384] **Programming Notes** This should be a multiple of 8 for HEVC and VP9.
	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: [0,16383] — Name: — Description: 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, 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 currently ignored in the VDEnc.

VDENC_Reference_Surface_State_Fields

DWord	Bits	Description
1	31:28	**Surface Format**

Format:	U4

Specifies the format of the surface. All supported formats are assumed to by Tile-Y.

Value	Name	Description
0	YUY2Variant	YUY2Variant is the modified YUY2 format YUYV/YUY2 (8:8:8:8 MSB V0 Y1 U0 Y0), 8 bit planar 422. The chroma is UV interleaved and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.
1	Reserved	
2	AYUVVariant	AYUV4444Variant is the modified AYUV4444 format, 8 bit planar 444 format. The U channel is below the luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma. The V channel is below the U and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.
3	Reserved	
4	PLANAR_420_8	(NV12, IMC1,2,3,4, YV12)
[5,7]	Reserved	
8	P010Variant	>8 bit planar 420 with MSB together and LSB at an offset in x direction.
9	Reserved	
10	Y416Variant	Y416Variant is the modified Y410/Y412/Y416 format, >8 bit planar 444 with MSB bytes packed together and LSB bytes at an offset in the X-direction where the x-offset is 32-bit aligned.</p> The U channel is below the luma, has identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma. </p> The V channel is below the U, has identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.</p>
11	Reserved	
12	Y216Variant	Y216Variant is the modified Y210/Y216 format, >8 bit planar 422 with MSB bytes packed together and LSB bytes at an offset in the X-direction where the x-offset is 32-bit aligned. The chroma is UV interleaved with identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.
[13,15]	Reserved	

DWord	Bits	Description
	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.</p>
For 444 formats, they are stored as two separate planes one below the other. But on the 422 and 420 formats, they are interleaved.

Value	Name	Description

VDENC_Reference_Surface_State_Fields

		0	Disable
		1	Enable

26:22 Reserved

Format:	MBZ

21:20 Reserved21_20

19:3 Surface Pitch

Format:	U17

-1 pitch in Bytes
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. This field is igored by VDEnc (unless we support YV12).

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

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

VDENC_Reference_Surface_State_Fields		
		of this bit.
	0	**Tile Walk** Format: U1 (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 always treats this as 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.
2	31	**Reserved** Format: MBZ
	30:16	**X Offset for U(Cb)** Default Value: 0 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 for all formats.
	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 all reference formats.
3	31:29	**Reserved** Format: MBZ
	28:16	**X Offset for V(Cr)** Format: U13 Offset in Pixels.</p>

VDENC_Reference_Surface_State_Fields

		</p> This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the V(Cr) plane. This field is only used for PLANAR surface formats with Interleave Chroma disabled.</p> This field MBZ for all reference VDEnc formats.</p>
	15:0	**Y Offset for V(Cr)** Format: U16 Row Offset in Pixels This field specifies the vertical offset in rows from the Surface Base Address to the start (origin) of the V(Cr) plane. It is only used for 444 planar formats for reference frames (AYUV4444V and Y416).</p> This field MBZ for PLANAR_420_8, P010V, YUY2V and Y216V formats.</p>

VDENC_Row_Store_Scratch_Buffer_Picture

VDENC_Row_Store_Scratch_Buffer_Picture

Source: VideoCS

Size (in bits): 96

Default Value: 0x00000000, 0x00000000, 0x00000000

Structure_VDENC_Row_Store_Scratch_Buffer_Picture
Y

DWord	Bit	Description
0	31:0	**Lower Address** Format: VDENC_64B_Aligned_Lower_Address This field provides the base address of the scratch buffer (read/write) used by VDEnc to store MB information of the previous row for processing of each macroblock in the current row. The Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of the current macroblock to address the Row Store. **AVC:** Size = 256 bits/MB. 4K wide picture needs 128 CLs.
1	31:0	**Upper Address** Format: VDENC_64B_Aligned_Upper_Address
2	31:0	**Buffer Picture Fields** Format: VDENC_Surface_Control_Bits

VDENC_Streamin_Data_Picture

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

DWord	Bit	Description
0	31:0	**Lower Address** Format: **VDENC_64B_Aligned_Lower_Address** Specifies the address for per-MB indirect data in memory when the StreamInEnable is set in the VDENC_PIPE_MODE_SELECT command. Currently this surface is intended to have 1CL per MB and is a linear surface. This has parameters such as extra predictors with refidx (intended to be used for IME search), MB level quantization parameters and Region of Interest bits. The individual parameters have enables in the VDENC_IMG_STATE to indicate which of the parameters are valid in the streamin surface. Size = 1CL/MB linear surface.
1	31:0	**Upper Address** Format: **VDENC_64B_Aligned_Upper_Address**
2	31:0	**Picture Fields** Format: **VDENC_Surface_Control_Bits**

VDENC_Sub_Mb_Pred_Mode

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

DWord	Bit	Description
0	7:6	**SubMbPredMode[3]** Format: U2
	5:4	**SubMbPredMode[2]** Format: U2
	3:2	**SubMbPredMode[1]** Format: U2
	1:0	**SubMbPredMode[0]** Format: U2

VDENC_Surface_Control_Bits

<table>
<tr><th colspan="3">VDENC_Surface_Control_Bits</th></tr>
<tr><td colspan="3">Source: VideoCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:15</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>14:13</td><td>Tiled Resource Mode
<table><tr><td></td><td></td></tr><tr><td>Format:</td><td>U2</td></tr></table>
For Media Surfaces: This field specifies the tiled resource mode.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>TRMODE_NONE</td><td>No tiled resource.</td></tr>
<tr><td>1h</td><td>TRMODE_TILEYF</td><td>4KB tiled resources</td></tr>
<tr><td>2h</td><td>TRMODE_TILEYS</td><td>64KB tiled resources</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table>
<table>
<tr><th>Programming Notes</th></tr>
<tr><td>This field should be programmed the same for all these VDEnc surfaces listed below. DS FWD REF0, DS FWD REF1, DS BWD REF0. FWD REF0, FWD REF1, FWD REF2, BWD REF0.</td></tr>
</table></td></tr>
<tr><td></td><td>12</td><td>Cache Select
<table><tr><td>Exists If:</td><td>(Property[Structure_VDENC_Row_Store_Scratch_Buffer_Picture] == 'true')</td></tr><tr><td>Format:</td><td>U1</td></tr></table>
<table>
<tr><th>Description</th></tr>
<tr><td>This field controls if the Row Store is going to store inside Media Cache (rowstore cache) or to LLC.</td></tr>
<tr><td>When this is programmed to "1" (going to Media Cache), the corresponding base address will be programmed with the starting position in the media cache. The programming table is in "Buffer Size Requirement Page" in HEVC section</td></tr>
</table>
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>Buffer going to LLC.</td></tr>
<tr><td>1</td><td></td><td>Buffer going to Internal Media Storage.</td></tr>
</table></td></tr>
<tr><td></td><td>12:11</td><td>Reserved
<table><tr><td>Exists If:</td><td>(Property[Structure_VDENC_Row_Store_Scratch_Buffer_Picture] == 'false')</td></tr><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
</table>

VDENC_Surface_Control_Bits

Bits	Field
11	**Reserved**

Exists If:	(Property[Structure_VDENC_Row_Store_Scratch_Buffer_Picture] == 'true')
Format:	MBZ

Bits	Field
10	**Reserved**
9	**Memory Compression Enable**

Format:	Enable

Memory compression will be attempted for this surface.

Value	Name
0h	Disable
1h	Enable

Programming Notes
This field should be programmed the same for all these VDEnc surfaces listed below. **DS FWD REF0, DS FWD REF1, DS BWD REF0. FWD REF0, FWD REF1, FWD REF2, BWD REF0.**

Bits	Field
8:7	**Arbitration Priority Control**

Format:	U2

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

Value	Name	Description
00b	Highest priority	
01b	Second highest priority	
10b	Third highest priority	
11b	Lowest priority	

Bits	Field
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.

Bits	Field
0	**Reserved**

VDENC_Surface_State_Fields

VDENC_Surface_State_Fields		
Source:	VideoCS	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000003, 0x00000000, 0x00000000	

DWord	Bit	Description
0	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: [0,16383] — Name: — Description: Representing heights [1,16384] **Programming Notes** AVC specific Note: • When surface tiling is TileY, the Frame Height needs to be programmed as 16-pixel aligned. • When surface tiling is Linear (supported only for Source surface), the Frame Height can be programmed 2-pixel aligned.
	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: [0,16383] — Name: — Description: 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, • 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	**Color space selection** Format: U1 0 Use BT.601 Space conversion

VDENC_Surface_State_Fields

		1 Use BT.709 Space conversion
	2	**Reserved**
	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.
1	31:27	**Format**

value	Name	Description	Programming Notes
0h	YUY2 format		
1h	RGB_8 format		
2h	AYUV4444 format		
3h	P010Variant		P010Variant is a modified P010 format, >8 bit planar 420 with MSB together and LSB at an offset in x direction where the x-offset should be 32-bit aligned.
4h	PLANAR_420_8		
5h-	YCRCB_SwapY format		
6h	YCRCB_SwapUV format		
7h	YCRCB_SwapUVY format		
8h	Y216 format	This format is used for source only. Any 422 mode with more than 8 bits per sample component uses this format.	
9h	RGB_10 format		
Ah	Y410 format		
Bh	NV21 Planar_420_8		

VDENC_Surface_State_Fields

	Format		
Ch	Y416 format		
Dh	P010		
Eh	P016		This is added for VP9 8./10/12 bit decode
Fh	Y8 format		
10h	Y16 format		
11h	Y216Variant	Y216Variant is the modifed Y210/Y216 format, 8 bit planar 422 with MSB bytes packed together and LSB bytes at an offset in the X-direction where the x-offset is 32-bit aligned. The chroma is UV interleaved with identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.	
12h	Y416Variant	Y416Variant is the modifed Y410/Y412/Y416 format,8 bit planar 444 with MSB bytes packed together and LSB bytes at an offset in the X-direction where the x-offset is 32-bit aligned. The U channel is below the luma, has identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma The V channel is below the U, has identical MSB and LSB split as luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.	
13h	YUY2Variant	YUY2Variant is the modifed YUY2 format, 8 bit planar 422. The chroma is UV interleaved and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.	
14h	AYUV4444Variant	AYUV4444Variant is the modifed AYUV4444 format, 8 bit planar 444 format. The U channel is below the luma and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma. The V channel is below the and is at an offset in the Y-direction (similar to NV12) but is the same height as the luma.	

VDENC_Surface_State_Fields

15h-1Fh	Reserved		

26:23 Reserved

Format:	MBZ

22:20 Chroma Downsample Filter Control

Format:	U3

Value	Left Side Tap	Center Tap	Right Side Tap
0	0	64	0
1	16	32	16
2	15	34	15
3	14	36	14
4	13	38	13
5	12	40	12
6	11	42	11
7	0	32	32

Programming Notes
For Tile Y 444 -> 420, Filter settings on 0 and 7 are valid. All other combinations are invalid. This is true for 10 bit and 8 bit.

19:3 Surface Pitch

Format:	U17

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

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 = [127,131071] to [128B,128KB] = [1 tile, 1028 tiles].
For TileYF and TileYS surfaces, the range is dependent on the Cu parameter (refer to Memory Data Formats section for the definition of the Cu parameter depending on the case). The range in bytes is [2^{Cu}-1, 131071] -> [(2^{Cu})B, 128KB] = [1 tile, 128KB/($2^{Cu)\ tiles}$] The field specifies the surface pitch in (#Bytes - 1)

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

VDENC_Surface_State_Fields

surface formats. This field is igored by VDEnc (unless we support YV12).

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

Bit 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

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.

Bit 0

Tile Walk

Format:	U1

(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 always treats this as 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.

DWord 2, Bit 31

Reserved

Format:	MBZ

Bit 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).

VDENC_Surface_State_Fields

DWord	Bit	Description
		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. **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must be multiple of 8 pixels - i.e. multiple MBs.
3	31:29	**Reserved** Format: MBZ
	28:16	**X Offset for V(Cr)** Format: U13 Offset in Pixels This field must be zero for NV12 and IMC 1 and 3. This field specifies the horizontal offset in pixels from the Surface Base Address to the start (origin) of the V(Cr) plane. This field is only used for PLANAR surface formats with Interleave Chroma disabled. **Programming Notes** For PLANAR_420 and PLANAR_422 surface formats, this field must indicate an even number of pixels.
	15:0	**Y Offset for V(Cr)** Format: U16 Row Offset in Pixels This field specifies the vertical offset in rows from the Surface Base Address to the start (origin) of the V(Cr) plane. This field is only used for PLANAR surface formats with Interleave Chroma disabled. This field is ignored by all video codec, only used by JPEG. **Programming Notes** For PLANAR_420 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs.

VEB_DI_IECP_COMMAND_SURFACE_CONTROL_BITS

<table>
<tr><th colspan="3">VEB_DI_IECP_COMMAND_SURFACE_CONTROL_BITS</th></tr>
<tr><td colspan="3">Source: VideoEnhancementCS
Size (in bits): 32
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:11</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>10:9</td><td>Tiled Resource Mode for Output Frame Surface Base Address
For Media Surfaces: This field specifies the tiled resource mode.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0h</td><td>TRMODE_NONE</td><td>No tiled resource</td></tr>
<tr><td>1h</td><td>TRMODE_TILEYF</td><td>4KB tiled resources</td></tr>
<tr><td>2h</td><td>TRMODE_TILEYS</td><td>64KB tiled resources</td></tr>
<tr><td>3h</td><td>Reserved</td><td></td></tr>
</table></td></tr>
<tr><td></td><td>8</td><td>Memory Compression Mode
Distinguishes Vertical from Horizontal compression.
<table>
<tr><th>Value</th><th>Name</th></tr>
<tr><td>0</td><td>Horizontal Compression Mode</td></tr>
</table></td></tr>
<tr><td></td><td>7</td><td>Memory Compression Enable
Format: Enable
Memory compression will be attempted for this surface.</td></tr>
<tr><td></td><td>6:1</td><td>Index to Memory Object Control State (MOCS) Tables
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.</td></tr>
<tr><td></td><td>0</td><td>Reserved</td></tr>
</table>

VEBOX_ACE_LACE_STATE

VEBOX_ACE_LACE_STATE

Source:	VideoEnhancementCS
Size (in bits):	416
Default Value:	0x00000068, 0x4C382410, 0x9C887460, 0xEBD8C4B0, 0x604C3824, 0xB09C8874, 0x0000D8C4, 0x04000400, 0x04000400, 0x04000400, 0x04000400, 0x04000400, 0x00000400

This state structure contains the IECP State Table Contents for ACE state.

DWord	Bit	Description
0	31:16	**Min_ACE_luma** Format: U16
	15:14	**LACE Single Histogram Set** This bit tells LACE which frames will be included in the histogram when the Deinterlacer is enabled. <table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>00b</td><td>Current</td><td>The histogram includes only the current frame.</td></tr><tr><td>01b</td><td>Previous</td><td>The histogram includes only the previous frame.</td></tr><tr><td>10b</td><td>Current + Previous</td><td>The histogram includes pixels from both the current and previous frame.</td></tr><tr><td>11b</td><td>Previous + Current</td><td>The histogram includes the previous frame followed by the current frame.</td></tr></table> **Programming Notes** When the Deinterlacer is disabled, this field must be 00b. If **DI Output Frames** is set to only output a single field then the histogram can not be collected on the disabled field. This Field must be set to 00b when DN/DI First Frame is set to 1
	13	**LACE Histogram Size** <table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>128-bin histogram</td></tr><tr><td>1</td><td>256-bin histogram</td></tr></table>
	12	**LACE Histogram Enable** Default Value: 0 This bit enables the collection of LACE histogram data. If this bit is 0 then only the ACE histogram will be collected. **Programming Notes** This bit must be set to "0"
	11:7	**Reserved**

VEBOX_ACE_LACE_STATE

DWord	Bit	Description
		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	**Reserved** Format: MBZ
	0	**ACE Enable** Format: Enable **Programming Notes** This bit must be set to "0"
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** 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.

VEBOX_ACE_LACE_STATE

	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.
	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

VEBOX_ACE_LACE_STATE		
		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.
	7:0	**B5** Default Value: 116 Format: U8 The value of the bias for point 5 in PWL.
6	31:16	**Reserved** Format: MBZ

VEBOX_ACE_LACE_STATE		
	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** Default Value: 1024 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** Default Value: 1024 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** Default Value: 1024 Format: U1.10 The value of the slope for point 3 in PWL The default is 1024/1024
	15:11	**Reserved** Format: MBZ
	10:0	**S2** Default Value: 1024

VEBOX_ACE_LACE_STATE

		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** Default Value: 1024 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** Default Value: 1024 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** Default Value: 1024 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** Default Value: 1024 Format: U1.10 The default is 1024/1024
11	31:27	**Reserved**

VEBOX_ACE_LACE_STATE

		Format: MBZ
	26:16	**S9** Default Value: 1024 Format: U1.10 The value of the slope for point 9 in PWL The default is 1024/1024
	15:11	**Reserved** Format: MBZ
	10:0	**S8** Default Value: 1024 Format: U1.10 The value of the slope for point 8 in PWL The default is 1024/1024
12	31:16	**Max_ACE_luma** Format: U16 The maximum luma for which ACE correction will be used.
	15:11	**Reserved** Format: MBZ
	10:0	**S10** Default Value: 1024 Format: U1.10 The value of the slope for point 10 in PWL.

VEBOX_ALPHA_AOI_STATE

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

This state structure contains the IECP State Table Contents for Fixed Alpha State and Area of Interest State.

DWord	Bit	Description
0	31:18	**Reserved** Format: MBZ
	17	**Full Image Histogram** Default Value: 0 Format: Enable Used to ignore the area of interest for a histogram across the full image. This applies to all statistics that are affected by AOI (Area of Interest).
	16	**Alpha from State Select** Format: U1 Enumerated type Value / Name: 0 — alpha is taken from message 1 — alpha is taken from state **Programming Notes** If the input format does not have alpha available and the output format provides alpha, this bit should be set to 1. This should be 0 when Alpha Plane Enable is 1.
	15:0	**Color Pipe Alpha** Format: U16 **Programming Notes** The 8 MSB of this field will be used for output formats that have 8-bits of alpha.
1	31:30	**Reserved** Format: MBZ
	29:16	**AOI Max X** Default Value: 0 Format: U14 Area of Interest Minimum X - The ACE histogram and Skin Tone Detection statistic gathering will occur within the MinX/MinY to MaxX/MaxY area (inclusive). AOI must intersect the frame such that at least 1 pixel is in the AOI.

VEBOX_ALPHA_AOI_STATE

DWord	Bit	Description
		The Area of Interest applies to the RGB Histogram and the White/Gray point sums as well. **Programming Notes** **This value must be a multiple of 4 minus 1.**
	15:14	**Reserved** Format: MBZ
	13:0	**AOI Min X** Default Value: 0 Format: U14 **Programming Notes** **This value must be a multiple of 4.**
2	31:30	**Reserved** Format: MBZ
	29:16	**AOI Max Y** Default Value: 0 Format: U14 **Programming Notes** **This value must be a multiple of 4 minus 1.**
	15:14	**Reserved** Format: MBZ
	13:0	**AOI Min Y** Default Value: 0 Format: U14 **Programming Notes** **This value must be a multiple of 4.**

VEBOX_CAPTURE_PIPE_STATE

VEBOX_CAPTURE_PIPE_STATE

Source:	VideoEnhancementCS
Size (in bits):	224
Default Value:	0x8511FF23, 0xAA64AFAA, 0xE6FD4000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

This command contains variables for controlling Demosaic and the White Balance Statistics.

DWord	Bit	Description
0	31:30	**DirMap_Scale** Default Value: 2 Format: U2
	29:24	**Good Pixel Threshold** Format: U6 The difference threshold between adjacent pixels for a pixel to be considered "good". Value | Name 5h | **[Default]**
	23	**Reserved** Format: MBZ
	22:20	**Shift Min Cost** Default Value: 1h Format: U3 The amount to shift the H2/V2 versions of min_cost.
	19:16	**Green Imbalance Threshold** Default Value: 1h Format: U4
	15:8	**Average Color Threshold** Format: U8 The threshold between two colors in a pixel for the Avg interpolation to be considered. Value | Name FFh | **[Default]** **Programming Notes** Must be set to 255.
	7:6	**Reserved** Format: U2
	5:0	**Good Pixel Neighbor Threshold** Default Value: 23h

VEBOX_CAPTURE_PIPE_STATE		
		Format: U6 Number of comparisons with neighbor pixels which pass before a pixel is considered good.
1	31:28	**Scale For Min Cost** Default Value: Ah The amount to scale the min_cost difference during the confidence check.
	27:24	**Good Intesity Threshold** Default Value: Ah Format: U4
	23:16	**Bad Color Threshold 1** Default Value: 64h Format: U8 Color value threshold used during the bad pixel check.
	15:8	**Bad Color Threshold 2** Default Value: AFh Format: U8 Color value threshold used during the bad pixel check.
	7:4	**Number Big Pixel Threshold** Default Value: Ah Format: U4 Number of comparisons with neighbor pixels which pass before a pixel is considered good.
	3:0	**Bad Color Threshold 3** Default Value: Ah Format: U4 Color value threshold used during the bad pixel check.
2	31:24	**Y Bright Value** Default Value: E6h The whitepoint threshold percentile in the Y histogram. Any pixel with Y value above this could be a whitepoint. This is the larger of the calculated Ybright value and the Ythreshold value, which is the minimum Y required to be considered a white point. **Programming Notes** "00000000" is appended to the LSBs before comparing with Y.
	23:16	**Y Outlier Value** Default Value: FDh The outlier threshold percentile in the Y histogram. Any pixel with Y value above this either

<table>
<tr><th colspan="3">VEBOX_CAPTURE_PIPE_STATE</th></tr>
<tr><td></td><td></td><td>clipped or an outlier in the image. These points will not be included in the white patch calculation.
<table><tr><th>Programming Notes</th></tr><tr><td>"00000000" is appended to the LSBs before comparing with Y.</td></tr></table></td></tr>
<tr><td></td><td>15:8</td><td>UV Threshold Value
The value denotes the maximum threshold of the ratio between U+V to Y can have to be considered a gray point.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>[255,0]</td><td></td><td>Encode a value from 255/256 to 0/256</td></tr>
<tr><td>64</td><td>[Default]</td><td>0.25 * 255 = 64</td></tr></table></td></tr>
<tr><td></td><td>7</td><td>Black Point Offset Red MSB</td></tr>
<tr><td></td><td>6</td><td>Black Point Offset Green Top MSB</td></tr>
<tr><td></td><td>5</td><td>Black Point Offset Blue MSB</td></tr>
<tr><td></td><td>4</td><td>Black Point Offset Green Bottom MSB</td></tr>
<tr><td></td><td>3</td><td>RGB Histogram Enable
Enables the collection of RGB Histograms for Auto-white balance correction and other uses.
<table><tr><th>Programming Notes</th></tr><tr><td>This bit can be set without White Balance enable being set.</td></tr></table></td></tr>
<tr><td></td><td>2</td><td>Vignette Correction Format
Defines what shift should be assumed for the Vignette Correction input values:
<table><tr><th>Value</th><th>Name</th></tr>
<tr><td>0</td><td>U8.8</td></tr>
<tr><td>1</td><td>U4.12</td></tr></table></td></tr>
<tr><td></td><td>1</td><td>Black Point Correction Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table></td></tr>
<tr><td></td><td>0</td><td>White Balance Correction Enable
<table><tr><td>Format:</td><td>Enable</td></tr></table>
<table><tr><th>Programming Notes</th></tr><tr><td>RGB Histogram enable must be set if this bit is set.</td></tr></table></td></tr>
<tr><td>3</td><td>31:16</td><td>Black Point Offset Red
<table><tr><td>Default Value:</td><td>0</td></tr><tr><td>Format:</td><td>U16</td></tr></table>
Value subtracted from Red pixels of Bayer pattern - combined with MSB to form a 2's complement signed number.</td></tr>
<tr><td></td><td>15:0</td><td>Black Point Offset Green Top
<table><tr><td>Default Value:</td><td>0</td></tr><tr><td>Format:</td><td>U16</td></tr></table></td></tr>
</table>

VEBOX_CAPTURE_PIPE_STATE		
		Value subtracted from the top Green pixels of Bayer pattern (X=1, Y=0 for Bayer Pattern #1) - combined with MSB to form a 2's complement signed number.
4	31:16	**Black Point Offset Blue** Default Value: 0 Format: U16 Value subtracted from Blue pixels of Bayer pattern - Combine with MSB to form a 2's complement signed number.
	15:0	**Black Point Offset Green Bottom** Default Value: 0 Format: U16 Value subtracted from the bottom Green pixels of Bayer pattern (X=0, Y=1 for Bayer Pattern #1) - combined with MSB to form a 2's complement signed number.
5	31:16	**White Balance Red Correction** Format: U4.12 The correction factor multiplied by the Red pixels of the Bayer pattern.
	15:0	**White Balance Green Top Correction** Format: U4.12 The correction factor multiplied by the top Green pixels of the Bayer pattern(X=1, Y=0 for Bayer Pattern #1).
6	31:16	**White Balance Blue Correction** Format: U4.12 The correction factor multiplied by the Blue pixels of the Bayer pattern.
	15:0	**White Balance Green Bottom Correction** Format: U4.12 The correction factor multiplied by the bottom Green pixels of the Bayer pattern (X=0, Y=1 for Bayer Pattern #1)

VEBOX_CCM_STATE

VEBOX_CCM_STATE		
Source:	VideoEnhancementCS	
Size (in bits):	480	
Default Value:	0x00004750, 0x0000AE80, 0x00000470, 0x00000220, 0x001FFCC0, 0x0000D230, 0x00000A80, 0x001FFF40, 0x0000D6A0, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000	

This state structure contains the IECP State Table Contents for the Color Correction Matrix State.

DWord	Bit	Description
0	31	**Color Correction Matrix Enable** Format: Enable This bit enables the Color Correction Matrix. **Programming Notes** Single Pipe IECP Enable must also be set if this bit is enabled. This bit must be set to "0"
	30:21	**Reserved** Format: MBZ
	20:0	**C1** Default Value: 0004750h = 18256/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
1	31:21	**Reserved** Format: MBZ
	20:0	**C0** Default Value: 000AE80h = 44672/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
2	31:21	**Reserved** Format: MBZ
	20:0	**C3** Default Value: 0000470h = 1136/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
3	31:21	**Reserved** Format: MBZ
	20:0	**C2**

VEBOX_CCM_STATE

		Default Value: 0000220h = 544/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
4	31:21	**Reserved** Format: MBZ
	20:0	**C5** Default Value: 1FFCC0h = -832/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
5	31:21	**Reserved** Format: MBZ
	20:0	**C4** Default Value: 000D230h = 53808/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
6	31:21	**Reserved** Format: MBZ
	20:0	**C7** Default Value: 0000A80h = 2688/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
7	31:21	**Reserved** Format: MBZ
	20:0	**C6** Default Value: 1FFF40h = -192/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
8	31:21	**Reserved** Format: MBZ
	20:0	**C8** Default Value: 000D6A0h = 54944/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
9	31:17	**Reserved** Format: MBZ

VEBOX_CCM_STATE		
	16:0	**Offset_in_R** Default Value: 0 Format: S16 The input offset for red component.
10	31:17	**Reserved** Format: MBZ
	16:0	**Offset_in_G** Default Value: 0 Format: S16 The input offset for green component.
11	31:17	**Reserved** Format: MBZ
	16:0	**Offset_in_B** Default Value: 0 Format: S16 The input offset for blue component.
12	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_R** Default Value: 0 Format: S16 The output offset for red component.
13	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_G** Default Value: 0 Format: S16 The output offset for green component.
14	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_B** Default Value: 0 Format: S16 The output offset for blue component.

VEBOX_Ch_Dir_Filter_Coefficient

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_CSC_STATE

VEBOX_CSC_STATE	
Source:	VideoEnhancementCS
Size (in bits):	384
Default Value:	0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000

This state structure contains the IECP State Table Contents for CSC state.

DWord	Bit	Description
0	31	**Transform Enable** Format: Enable **Programming Notes** This bit must be set to "0"
	29:19	**Reserved** Format: MBZ
	18:0	**C0** Default Value: 10000h or 1.0 Format: S2.16 2's complement Transform coefficient.
1	31:19	**Reserved** Format: MBZ
	18:0	**C1** Default Value: 0 Format: S2.16 2's complement Transform coefficient.
2	31:19	**Reserved** Format: MBZ
	18:0	**C2** Default Value: 0 Format: S2.16 2's complement Transform coefficient.
3	31:19	**Reserved** Format: MBZ
	18:0	**C3**

VEBOX_CSC_STATE		
		Default Value: 0 Format: S2.16 2's complement Transform coefficient.
4	31:19	**Reserved** Format: MBZ
	18:0	**C4** Default Value: 10000h or 1.0 Format: S2.16 2's complement Transform coefficient.
5	31:19	**Reserved** Format: MBZ
	18:0	**C5** Default Value: 0 Format: S2.16 2's complement Transform coefficient.
6	31:19	**Reserved** Format: MBZ
	18:0	**C6** Default Value: 0 Format: S2.16 2's complement Transform coefficient.
7	31:19	**Reserved** Format: MBZ
	18:0	**C7** Default Value: 0 Format: S2.16 2's complement Transform coefficient.
8	31:19	**Reserved** Format: MBZ
	18:0	**C8** Default Value: 10000h or 1.0 Format: S2.16 2's complement Transform coefficient. The offset value is multiplied by 2 before being added to the output.

VEBOX_CSC_STATE

9	31:16	**Offset Out 1** Default Value: 0 Format: S15 2's Complement Offset in for Y/R. The offset value is multiplied by 2 before being added to the output.
	15:0	**Offset in 1** Default Value: 0 Format: S15 2's Complement Offset in for Y/R. The offset value is multiplied by 2 before being added to the output.
10	31:16	**Offset Out 2** Default Value: 0 Format: S15 2's Complement Offset out for U/G. The offset value is multiplied by 2 before being added to the output.
	15:0	**Offset in 2** Default Value: 0 Format: S15 2's Complement Offset out for U/G. The offset value is multiplied by 2 before being added to the output.
11	31:16	**Offset Out 3** Default Value: 0 Format: S15 2's Complement Offset out for V/B. The offset value is multiplied by 2 before being added to the output.
	15:0	**Offset in 3** Default Value: 0 Format: S15 2's Complement Offset out for V/B. The offset value is multiplied by 2 before being added to the output.

VEBOX_DNDI_STATE

VEBOX_DNDI_STATE		
Source:	VideoEnhancementCS	
Size (in bits):	1440	
Default Value:	0x00000000, 0x80000000, 0x00000400, 0x00000000, 0x00000000, 0x00000000, 0x000000A0, 0x00000000, 0x0800F000, 0x0F025800, 0x00007800, 0x2D000000, 0x00000254, 0x00000000, 0x00000400, 0x05FF05FF, 0x00000000, 0x005064A5, 0x00000000, 0x00000000, 0x1314640F, 0x00000000, 0xFF804020	

This state table is used by the *Denoise and Deinterlacer functions*

DWord	Bit	Description
0	31:17	**Denoise STAD Threshold** Format: U15 Threshold for denoise sum of temporal absolute differences.
	16:8	**Reserved** Format: MBZ
	7:0	**Denoise Maximum History** Format: U8 Maximum allowed value for denoise history. Value: [128,240] — Name: (none)
1	31:28	**Denoise History increase** Amount that denoise_history is increased by. MAX:15 Value: 8h — Name: **[Default]** — Description: (none) Value: 15 — Name: (none) — Description: Maximum Allowed
	27:23	**Denoise Moving Pixel Threshold** Format: U5 Value: [0,16] — Name: (none)
	22:12	**Reserved** Format: MBZ
	11:0	**Denoise ASD Threshold** Format: U12

<table>
<tr><th colspan="3">VEBOX_DNDI_STATE</th></tr>
<tr><td></td><td></td><td>Threshold for denoise absolute sum of differences.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,1023]</td><td></td></tr></table></td></tr>
<tr><td>2</td><td>31:20</td><td>Temporal Difference Threshold
<table><tr><td>Format:</td><td>U12</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>0 < (Temporal Difference Threshold - Low Temporal Difference Threshold) <= 256 except when both thresholds are set to 0.</td></tr></table></td></tr>
<tr><td></td><td>19:11</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>10:5</td><td>Initial Denoise History
<table><tr><td>Default Value:</td><td>32</td></tr><tr><td>Format:</td><td>U6</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>Initial value for Denoise history for both Luma and Chroma</td></tr></table></td></tr>
<tr><td></td><td>4:0</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td>3</td><td>31:20</td><td>Low Temporal Difference Threshold
<table><tr><td>Format:</td><td>U12</td></tr></table>0 < (Temporal Difference Threshold - Low Temporal Difference Threshold) <= 256 except when both thresholds are set to 0.</td></tr>
<tr><td></td><td>19:11</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>10</td><td>Progressive DN
<table><tr><td>Format:</td><td>Enable</td></tr></table>Indicates that the denoise algorithm should assume progressive input when filtering neighboring pixels. This bit must be set if the input to Denoise is RGB.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>DN assumes interlaced video and filters alternate lines together</td></tr><tr><td>1</td><td></td><td>DN assumes progressive video and filters neighboring lines together</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>DI Enable must be disabled when this field is enabled.</td></tr></table></td></tr>
<tr><td></td><td>9:2</td><td>Hot Pixel Count Luma
<table><tr><td>Format:</td><td>U8</td></tr></table>Number of neighboring pixels different more than Hot Pixel Threshold before a pixel is</td></tr>
</table>

VEBOX_DNDI_STATE

DWord	Bit	Description
		considered hot. **Value** / **Name** [0,8] / **Programming Notes** 0 will cause all pixels to be considered hot and will perform a median filter on the entire image.
	1:0	**Reserved** Format: MBZ
4	31:20	**Denoise Threshold for Sum of Complexity Measure Luma** Format: U12
	19:12	**Hot Pixel Threshold Luma** Format: U8 Threshold for a difference from the value of a neighboring pixel. Is shifted up to 16-bits before compare.
	11:0	**Block Noise Estimate Noise Threshold** Format: U12 Threshold for noise maximum/minimum. **Value** / **Name** [0,4095] /
5	31:17	**Chroma Denoise STAD Threshold** Format: U15 Threshold for denoise sum of temporal absolute differences.
	16	**Reserved** Format: MBZ
	15:8	**Hot Pixel Threshold Chroma U** Format: U8 Threshold for a difference from the value of a neighboring pixel. Is shifted up to 16-bits before compare.
	7:0	**Hot Pixel Count Chroma U** Format: U8 Number of neighboring pixels different more than **Hot Pixel Threshold** before a pixel is considered hot
6	31:20	**Chroma Temporal Difference Threshold** Format: U12

VEBOX_DNDI_STATE

DWord	Bit	Description
		0 < (Chroma Temporal Difference Threshold - Chroma Low Temporal Difference Threshold) <=256 except when both thresholds are set to 0
	19:12	**Reserved** Format: MBZ
	11:1	**Block Noise Estimate Edge Threshold** Default Value: 80 Threshold for detecting an edge in block noise estimate.
	0	**Chroma Denoise Enable** Format: Enable **Value** | **Name** | **Description** 0 | | The U and V channels will be passed to the next stage after DN unchanged. 1 | | The U and V chroma channels will be denoise filtered.
7	31:20	**Chroma Low Temporal Difference Threshold** Format: U12 0 < (Chroma Temporal Difference Threshold - Chroma Low Temporal Difference Threshold) <= 256 except when both thresholds are set to 0
	19:16	**Reserved** Format: MBZ
	15:8	**Hot Pixel Threshold Chroma V** Format: U8 Threshold for a difference from the value of a neighboring pixel. Is shifted up to 16-bits before compare.
	7:0	**Hot Pixel Count Chroma V** Format: U8 Number of neighboring pixels different more than **Hot Pixel Threshold** before a pixel is considered hot
8	31:29	**Reserved** Format: MBZ
	28:24	**Chroma Denoise Moving Pixel Threshold** Format: U5
	23:12	**Chroma Denoise ASD Threshold** Format: U12 Threshold for denoise absolute sum of differences.

VEBOX_DNDI_STATE		
	11:0	**Chroma Denoise Threshold for Sum of Complexity Measure**
9	31:30	**Reserved** Format: MBZ
	29:25	**DnY_Wr5[4:0]** Weight to be applied when: th4 <= (difference in luma, Bayer or RGB value)
	24:20	**DnY_Wr4[4:0]** Weight to be applied when: th3 <= (difference in luma, Bayer or RGB value) < th4
	19:15	**DnY_Wr3[4:0]** Weight to be applied when: th2 <= (difference in luma, Bayer or RGB value) < th3
	14:10	**DnY_Wr2[4:0]** Weight to be applied when: th1 <= (difference in luma, Bayer or RGBvalue) < th2
	9:5	**DnY_Wr1[4:0]** Weight to be applied when: th0 <= (difference in luma, Bayer or RGB value) < th1
	4:0	**DnY_Wr0[4:0]** Weight to be applied when: (difference in luma, Bayer or RGB value) < th0
10	31:29	**Reserved** Format: MBZ
	28:16	**DnY_thmax[12:0]** Maximum threshold value for luma, Bayer or RGB
	15:13	**Reserved** Format: MBZ
	12:0	**DnY_thmin[12:0]** Minimum threshold value
11	31:29	**Reserved** Format: MBZ
	28:16	**DnY_prt5[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnY_dyn_thmin[12:0]** Minimum Dynamic threshold value
12	31:29	**Reserved** Format: MBZ
	28:16	**DnY_prt4[12:0]** Multiplied by thrscale and then used as the threshold for comparing the luma or RGB differences.
	15:13	**Reserved** Format: MBZ
	12:0	**DnY_prt3[12:0]**
13	31:29	**Reserved**

VEBOX_DNDI_STATE

		Format: MBZ
	28:16	**DnY_prt2[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnY_prt1[12:0]**
14	31:29	**Reserved** Format: MBZ
	28:16	**DnY_prt0[12:0]**
	15	**Reserved** Format: MBZ
	14:10	**DnY_wd22[4:0]** Weight to be applied to the 4 luma, Bayer or RGB pixels that are at X±2 and Y±2
	9:5	**DnY_wd21[4:0]** Weight to be applied to the 4 luma, Bayer or RGB pixels that are at X±1 and Y±2
	4:0	**DnY_wd20[4:0]** Weight to be applied to the 2 luma, Bayer or RGB pixels that are at X and Y±2
15	31:30	**Reserved** Format: MBZ
	29:25	**DnY_wd12[4:0]** Weight to be applied to the 4 luma, Bayer or RGB pixels that are at X±2 and Y±1
	24:20	**DnY_wd11[4:0]** Weight to be applied to the 4 luma, Bayer or RGB pixels that are at X±1 and Y±1
	19:15	**DnY_wd10[4:0]** Weight to be applied to the 2 luma, Bayer or RGB pixels that are at X and Y±1
	14:10	**DnY_wd02[4:0]** Weight to be applied to the 2 luma, Bayer or RGB pixels that are at X±2 and Y
	9:5	**DnY_wd01[4:0]** Weight to be applied to the 2 luma, Bayer or RGB pixels that are at X±1 and Y
	4:0	**DnY_wd00[4:0]** Weight to be applied to the 1 luma, Bayer or RGB pixels that are at X and Y
16	31:30	**Reserved** Format: MBZ
	29:25	**DnU_Wr5[4:0]** Weight to be applied when: th4 <= (difference in chroma U value)
	24:20	**DnU_Wr4[4:0]** Weight to be applied when: th3 <= (difference in chroma U value) < th4
	19:15	**DnU_Wr3[4:0]** Weight to be applied when: th2 <= (difference in chroma U value) < th3
	14:10	**DnU_Wr2[4:0]**

VEBOX_DNDI_STATE		
		Weight to be applied when: th1 <= (difference in chroma Uvalue) < th2
	9:5	**DnU_Wr1[4:0]** Weight to be applied when: th0 <= (difference in chroma U value) < th1
	4:0	**DnU_Wr0[4:0]** Weight to be applied when: (difference in chroma U value) < th0
17	31:29	**Reserved** Format: MBZ
	28:16	**DnU_thmax[12:0]** Maximum threshold value for chroma U
	15:13	**Reserved** Format: MBZ
	12:0	**DnU_thmin[12:0]** Minimum threshold value
18	31:29	**Reserved** Format: MBZ
	28:16	**DnU_prt5[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnU_dyn_thmin[12:0]** Minimum Dynamic threshold value.
19	31:29	**Reserved** Format: MBZ
	28:16	**DnU_prt4[12:0]** Multiplied by thrscale and then used as the threshold for comparing chroma U differences.
	15:13	**Reserved** Format: MBZ
	12:0	**DnU_prt3[12:0]**
20	31:29	**Reserved** Format: MBZ
	28:16	**DnU_prt2[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnU_prt1[12:0]**
21	31:29	**Reserved** Format: MBZ
	28:16	**DnU_prt0[12:0]**
	15	**Reserved**

VEBOX_DNDI_STATE

		Format: MBZ
	14:10	**DnU_wd22[4:0]** Weight to be applied to the 4 chroma U pixels that are at X±2 and Y±2
	9:5	**DnU_wd21[4:0]** Weight to be applied to the 4 chroma U pixels that are at X±1 and Y±2
	4:0	**DnU_wd20[4:0]** Weight to be applied to the 2 chroma U pixels that are at X and Y±2
22	31:30	**Reserved** Format: MBZ
	29:25	**DnU_wd12[4:0]** Weight to be applied to the 4 chroma U pixels that are at X±2 and Y±1
	24:20	**DnU_wd11[4:0]** Weight to be applied to the 4 chroma U pixels that are at X±1 and Y±1
	19:15	**DnU_wd10[4:0]** Weight to be applied to the 2 chroma U pixels that are at X and Y±1
	14:10	**DnU_wd02[4:0]** Weight to be applied to the 2 chroma U pixels that are at X±2 and Y
	9:5	**DnU_wd01[4:0]** Weight to be applied to the 2 chroma U pixels that are at X±1 and Y
	4:0	**DnU_wd00[4:0]** Weight to be applied to the 1 chroma U pixels that are at X and Y
23	31:30	**Reserved** Format: MBZ
	29:25	**DnV_Wr5[4:0]** Weight to be applied when: th4 <= (difference in chroma V value)
	24:20	**DnV_Wr4[4:0]** Weight to be applied when: th3 <= (difference in chroma V value) < th4
	19:15	**DnV_Wr3[4:0]** Weight to be applied when: th2 <= (difference in chroma V value) < th3
	14:10	**DnV_Wr2[4:0]** Weight to be applied when: th1 <= (difference in chroma V value) < th2
	9:5	**DnV_Wr51[4:0]** Weight to be applied when: th0 <= (difference in chroma V value) < th1
	4:0	**DnV_Wr0[4:0]** Weight to be applied when: (difference in chroma V value) < th0
24	31:29	**Reserved** Format: MBZ
	28:16	**DnV_thmax[12:0]** Maximum threshold value for chroma V
	15:13	**Reserved**

VEBOX_DNDI_STATE		
		Format: MBZ
	12:0	**DnV_thmin[12:0]** Minimum threshold value
25	31:29	**Reserved** Format: MBZ
	28:16	**DnV_prt5[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnV_dyn_thmin[12:0]** Minimum Dynamic threshold value.
26	31:29	**Reserved** Format: MBZ
	28:16	**DnV_prt4[12:0]** Multiplied by thrscale and then used as the threshold for comparing chroma V differences.
	15:13	**Reserved** Format: MBZ
	12:0	**DnV_prt3[12:0]**
27	31:29	**Reserved** Format: MBZ
	28:16	**DnV_prt2[12:0]**
	15:13	**Reserved** Format: MBZ
	12:0	**DnV_prt1[12:0]**
28	31:29	**Reserved** Format: MBZ
	28:16	**DnV_prt0[12:0]**
	15	**Reserved** Format: MBZ
	14:10	**DnV_wd22[4:0]** Weight to be applied to the 4 chroma V pixels that are at X±2 and Y±2
	9:5	**DnV_wd21[4:0]** Weight to be applied to the 4 chroma V pixels that are at X±1 and Y±2
	4:0	**DnV_wd20[4:0]** Weight to be applied to the 2 chroma V pixels that are at X and Y±2
29	31:30	**Reserved** Format: MBZ
	29:25	**DnV_wd12[4:0]**

VEBOX_DNDI_STATE		
		Weight to be applied to the 4 chroma V pixels that are at X±2 and Y±1
	24:20	**DnV_wd11[4:0]** Weight to be applied to the 4 chroma V pixels that are at X±1 and Y±1
	19:15	**DnV_wd10[4:0]** Weight to be applied to the 2 chroma V pixels that are at X and Y±1
	14:10	**DnV_wd02[4:0]** Weight to be applied to the 2 chroma V pixels that are at X±2 and Y
	9:5	**DnV_wd01[4:0]** Weight to be applied to the 2 chroma V pixels that are at X±1 and Y
	4:0	**DnV_wd00[4:0]** Weight to be applied to the 1 chroma V pixels that are at X and Y
30	31:17	**Eight Direction Edge Threshold** Default Value: 1024 Format: U15 Threshold to determine an edge in eight directional edge detector
	16:7	**Valid Pixel Threshold** Default Value: 480 Format: U10
	6:0	**Reserved** Format: MBZ
31	31:19	**Small Sobel Threshold** Default Value: 480 Format: U13 Threshold for weak Sobel response
	18:6	**Large Sobel Threshold** Default Value: 2400 Format: U13 Threshold for strong Sobel response **Programming Notes** Large Sobel Threshold > Small Sobel Threshold
	5:0	**Small Sobel Count Threshold** Format: U6 Threshold for number of pixels in a block that have weak Sobel response (Default: 6)
32	31:26	**Median Sobel Count Threshold** Format: U6 Threshold for number of pixels in a block that have regular Sobel response (Default: 40)

VEBOX_DNDI_STATE

<table>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td>25:20</td><td>Large Sobel Count Threshold
Format: U6
Threshold for number of pixels in a block that have strong Sobel response (Default: 6)</td></tr>
<tr><td></td><td>19:6</td><td>Block Sigma Diff Threshold
Default Value: 480
Format: U14
Threshold for the difference between maximum and minimum sigma within a block</td></tr>
<tr><td></td><td>5:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>33</td><td>31:19</td><td>Max Sobel Threshold
Default Value: 1440
Format: U13</td></tr>
<tr><td></td><td>18:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>34</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>15:13</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>12:10</td><td>STMM C2
Format: U3
Bias for divisor in STMM equation.<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>[0,7]</td><td></td><td>Representing values [1,8]</td></tr></table></td></tr>
<tr><td></td><td>9:6</td><td>Content Adaptive Threshold Slope
Format: U4
Determines the slope of the Content Adaptive Threshold.<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>9</td><td>[Default]</td><td>CAT_slope value = 10</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>+1 added internally to get CAT_slope.</td></tr></table></td></tr>
<tr><td></td><td>5:2</td><td>SAD Tight Threshold
Default Value: 5</td></tr>
</table>

VEBOX_DNDI_STATE

DWord	Bit	Description
		Format: U4
	1:0	**Smooth MV Threshold** Format: U2
35	31	**STMM Blending Constant Select** Format: U1 Value | Name | Description 0 | | Use the blending constant for small values of STMM for stmm_md_th 1 | | Use the blending constant for large values of STMM for stmm_md_th
	30:24	**Blending constant across time for large values of STMM** Format: U7
	23:16	**Blending constant across time for small values of STMM** Format: U8
	15:14	**Reserved** Format: MBZ
	13:8	**Multiplier for VECM** Format: U6 Determines the strength of the vertical edge complexity measure.
	7:0	**Maximum STMM** Format: U8 Largest allowed STMM in blending equations.
36	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).

VEBOX_DNDI_STATE		
		<table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>Shift by 6</td></tr><tr><td>1</td><td>Shift by 7</td></tr><tr><td>2</td><td>Shift by 8</td></tr><tr><td>3</td><td>Reserved</td></tr></table>
	19:16	**STMM Output Shift** <table><tr><td>Format:</td><td>U4</td></tr></table> <table><tr><th>Value</th><th>Name</th></tr><tr><td>[0,15]</td><td></td></tr></table> <table><tr><th>Programming Notes</th></tr><tr><td>The value of this field must satisfy the following equation: stmm_max - stmm_min = 2 ^ stmm_output_shift</td></tr></table>
	15:12	**ChromaTDM_WT** <table><tr><td>Default Value:</td><td>0</td></tr><tr><td>Format:</td><td>U2.2</td></tr></table>
	11:8	**LumaTDM_WT** <table><tr><td>Default Value:</td><td>4</td></tr><tr><td>Format:</td><td>U2.2</td></tr></table>
	7:0	**FMD Temporal Difference Threshold** <table><tr><td>Format:</td><td>U8</td></tr></table>
37	31:28	**Reserved** <table><tr><td>Format:</td><td>MBZ</td></tr></table>
	27:24	**Deltabit value for SHCM** <table><tr><td>Format:</td><td>U4</td></tr></table> <table><tr><th>Value</th><th>Name</th></tr><tr><td>5</td><td>[Default]</td></tr><tr><td>[0,8]</td><td>Range</td></tr></table>
	23:16	**Coring Threshold for SHCM** <table><tr><td>Default Value:</td><td>255</td></tr><tr><td>Format:</td><td>U8</td></tr></table>
	15:12	**Reserved** <table><tr><td>Format:</td><td>MBZ</td></tr></table>
	11:8	**Deltabit value for SVCM** <table><tr><td>Format:</td><td>U4</td></tr></table>

VEBOX_DNDI_STATE

DWord	Bit	Description
		Value: 5 — Name: **[Default]**; Value: [0,8] — Name: Range
	7:0	**Coring Threshold for SVCM** Default Value: 255 Format: U8
38	31:24	**FMD #1 Vertical Difference Threshold** Format: U8
	23:16	**FMD #2 Vertical Difference Threshold** Format: U8
	15:14	**CAT Threshold** Default Value: 0 Format: U2
	13:8	**FMD Tear Threshold** Format: U6
	7	**MCDI Enable** Use Motion Compensated Deinterlace algorithm. **Programming Notes** This bit is Ignored if DI Enable is off.
	6:4	**Reserved** Format: MBZ
	3	**DN/DI Top First** Format: Enable Indicates the top field is first in sequence, otherwise bottom is first. Value 0 — Name: (none) — Description: Bottom field occurs first in sequence Value 1 — Name: (none) — Description: Top field occurs first in sequence
	2:0	**Reserved** Format: MBZ
39	31:26	**Reserved** Format: MBZ
	25	**FasterCovergence** Default Value: 0 Format: U1 For synthetic content faster convergence to current STMM value is preferred.

VEBOX_DNDI_STATE

	24	**Luma Smaller Window for TDM** Format: U1
	23	**Chroma Smaller Window for TDM** Format: U1
	22:19	**Neighbor Pixel Threshold** Default Value: 10 Format: U4
	18	**Reserved** Format: MBZ
	17:16	**Progressive Cadence Reconstruction For 2nd Field Of Previous Frame** Format: U2

Value	Name	Description
0	Deinterlace	
1	Put together with previous field in sequence	1st field of previous frame
2	Put together with next field in sequence	1st field of current frame

Programming Notes
Deflicker can be enabled only in De-interlace mode and not in Cadence construction mode.

	15:10	**MC Pixel Consistency Threshold** Default Value: 25 Format: U6
	9:8	**Progressive Cadence Reconstruction for 1st Field of Current Frame** Format: U2

Value	Name	Description
0	Deinterlace	
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

Programming Notes
Deflicker can be enabled only in De-interlace mode and not if either fields are in Cadence construction mode.

	7:4	**SAD THB** Default Value: 10 Format: U4
	3:0	**SAD THA**

VEBOX_DNDI_STATE

DWord	Bit	Description
		Default Value: 5 Format: U4
40	31:24	**SAD_WT[3]** Format: U8 Value / Name: 192 — Default for Natural 38 — Default for Synthetic
	23:16	**SAD_WT[2]** Format: U8 Value / Name: 179 — Default for Natural 25 — Default for Synthetic
	15:8	**SAD_WT[1]** Format: U8 Value / Name: 166 — Default for Natural 12 — Default for Synthetic
	7:0	**SAD_WT[0]** Format: U8 Value / Name: 0 — Default for Natural and Synthetic
41	31:24	**Coring Threshold for Chroma SAD calculation** Default Value: 0 Format: U8
	23:16	**Coring Threshold for Luma SAD calculation** Default Value: 0 Format: U8
	15:8	**SAD_WT[6]** Format: U8 Value / Name: 217 — Default for Natural 64 — Default for Synthetic

VEBOX_DNDI_STATE		
	7:0	**SAD_WT[4]** Format: U8 Value / Name: 218 — Default for Natural 90 — Default for Synthetic
42	31	**Reserved** Format: MBZ
	30	**Bypass Deflicker** Format: U1
	29	**PAR_UseSyntheticContentMedian** Default Value: 0 Format: U1
	28	**PAR_LocalCheck** Default Value: 1 Format: U1
	27	**PAR_SyntheticContentCheck** Default Value: 0 Format: U1
	26:24	**PAR_DirectionCheckTh** Default Value: 3 Format: U3
	23:16	**PAR_TearingLowThreshold** Default Value: 20 Format: U8
	15:8	**PAR_TearingHighThreshold** Default Value: 100 Format: U8
	7:0	**PAR_DiffCheckSlackThreshold** Default Value: 15 Format: U8
43	31:24	**LPFWtLUT[3]** Default Value: 0 Format: U8
	23:16	**LPFWtLUT[2]** Default Value: 0

VEBOX_DNDI_STATE			
		Format:	U8
	15:8	**LPFWtLUT[1]**	
		Default Value:	0
		Format:	U8
	7:0	**LPFWtLUT[0]**	
		Default Value:	0
		Format:	U8
44	31:24	**LPFWtLUT[7]**	
		Default Value:	255
		Format:	U8
	23:16	**LPFWtLUT[6]**	
		Default Value:	128
		Format:	U8
	15:8	**LPFWtLUT[5]**	
		Default Value:	64
		Format:	U8
	7:0	**LPFWtLUT[4]**	
		Default Value:	32
		Format:	U8

VEBOX_Filter_Coefficient

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)

VEBOX_FORWARD_GAMMA_CORRECTION_STATE

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

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE
0x00000000, 0x00000000,

VEBOX_FORWARD_GAMMA_CORRECTION_STATE

0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000

DWord	Bit	Description
0..2047	63:0	**PRGB Corrected Value** Format: VEBOX_RGB_TO_GAMMA_CORRECTION

Programming Notes

Order in which the values are stored:

Interleaves									
0	256	1	257	2	258	...	...	255	511
512	768	513	769	514	770	...	...	767	1023

Point 0-255, 256-511 are interleaved first followed by interleaving the next set of 512points, interleaving between points 512-767, 768-1023.

VEBOX_FRONT_END_CSC_STATE

VEBOX_FRONT_END_CSC_STATE		
Source:	VideoEnhancementCS	
Size (in bits):	384	
Default Value:	0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000	
This state structure contains the IECP State Table Contents for Front-end CSC state.		
DWord	**Bit**	**Description**
0	31	**Front End CSC Transform Enable** Format: Enable **Programming Notes** **Single Pipe IECP Enable** must also be set if this is enabled. This bit must be set to "0"
	30:19	**Reserved** Format: MBZ
	18:0	**FECSC C0: Transform coefficient** Default Value: 10000h or 1.0 Format: S2.16
1	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C1: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
2	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C2: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
3	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C3: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
4	31:19	**Reserved** Format: MBZ

VEBOX_FRONT_END_CSC_STATE

	18:0	**FECSC C4: Transform coefficient** Default Value: 10000h or 1.0 Format: S2.16
5	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C5: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
6	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C6: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
7	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C7: Transform coefficient** Default Value: 0 or 0.0 Format: S2.16
8	31:19	**Reserved** Format: MBZ
	18:0	**FECSC C8: Transform coefficient** Default Value: 10000h or 1.0 Format: S2.16
9	31:16	**FEC SC Offset out 1: Offset out for Y/R** Default Value: 0 Format: S15 The offset value is multiplied by 2 before being added to the output.
	15:0	**FEC SC Offset in 1: Offset in for Y/R** Default Value: 0 Format: S15 The offset value is multiplied by 2 before being added to the output.
10	31:16	**FEC SC Offset out 2: Offset out for U/G** Default Value: 0 Format: S15 The offset value is multiplied by 2 before being added to the output.
	15:0	**FEC SC Offset in 2: Offset out for U/G** Default Value: 0

VEBOX_FRONT_END_CSC_STATE

		Format: S15 The offset value is multiplied by 2 before being added to the output.
11	31:16	**FEC SC Offset out 3: Offset out for V/B** Default Value: 0 Format: S15 The offset value is multiplied by 2 before being added to the output.
	15:0	**FEC SC Offset in 3: Offset out for V/B** Default Value: 0 Format: S15 The offset value is multiplied by 2 before being added to the output.

VEBOX_GAMUT_CONTROL_STATE

VEBOX_GAMUT_CONTROL_STATE

Source:	VideoEnhancementCS
Size (in bits):	576
Default Value:	0xDA004750, 0x0000AE80, 0x00000470, 0x00000220, 0x001FFCC0, 0x0000D230, 0x00000A80, 0x001FFF40, 0x0000D6A0, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x0CD2911F, 0xB0000334, 0x00000000

<table>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:23</td><td>A(r)
Default Value: 436
Format: U9
Gain_factor_R (default: 436, preferred range: 256-511).</td></tr>
<tr><td></td><td>22</td><td>Global Mode Enable
Format: U1
The gain factor derived from state CM(w).
<table><tr><th>Value</th><th>Name</th></tr><tr><td>0</td><td>Advance Mode</td></tr><tr><td>1</td><td>Basic Mode</td></tr></table></td></tr>
<tr><td></td><td>21</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>20:0</td><td>C1
Default Value: 0004750h = 18256/65536
Format: S4.16
Coefficient of 3x3 Transform matrix.</td></tr>
<tr><td>1</td><td>31:22</td><td>CM(w)
Format: U10
WeightingFactorForGain_factor (only enabled when the GlobalModeEnable is on).</td></tr>
<tr><td></td><td>21</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>20:0</td><td>C0
Default Value: 000AE80h = 44672/65536
Format: S4.16
Coefficient of 3x3 Transform matrix.</td></tr>
<tr><td>2</td><td>31:22</td><td>CM(s)
Format: U2.8</td></tr>
</table>

VEBOX_GAMUT_CONTROL_STATE

		AccurateColorComponentScaling (default: 640/256, preferred range: [512-1023]/256).
	21	**Reserved** Format: MBZ
	20:0	**C3** Default Value: 0000470h = 1136/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
3	31:25	**A(g)** Format: U7 Gain_factor_G (default: 26/256, preferred range: [26-127]/256).
	24:21	**Reserved** Format: MBZ
	20:0	**C2** Default Value: 0000220h = 544/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
4	31:25	**A(b)** Format: U7 Gain_factor_B (default: 26/256, preferred range: [26-127]/256).
	24:21	**Reserved** Format: MBZ
	20:0	**C5** Default Value: 1FFCC0h = -832/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
5	31:22	**R(s)** Format: U2.8 RedScaling (default: 768/256, preferred range: [512-1023]/256).
	21	**Reserved** Format: MBZ
	20:0	**C4** Default Value: 000D230h = 53808/65536

VEBOX_GAMUT_CONTROL_STATE

		Format: S4.16 Coefficient of 3x3 Transform matrix.
6	31:24	**CM(i)** Format: U0.8 AccurateColorComponentOffset (default: 192/256, preferred range: [0-192]/256).
	23:21	**Reserved** Format: MBZ
	20:0	**C7** Default Value: 0000A80h = 2688/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
7	31:24	**R(i)** Format: U0.8 RedOffset (default: 128/256, preferred range: [0-128]/256).
	23:21	**Reserved** Format: MBZ
	20:0	**C6** Default Value: 1FFF40h = -192/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
8	31:21	**Reserved** Format: MBZ
	20:0	**C8** Default Value: 00D6A0h = 54944/65536 Format: S4.16 Coefficient of 3x3 Transform matrix.
9	31:17	**Reserved** Format: MBZ
	16:0	**Offset_in_R** Default Value: 0 Format: S16 The input offset for red component.

VEBOX_GAMUT_CONTROL_STATE		
10	31:17	**Reserved** Format: MBZ
	16:0	**Offset_in_G** Default Value: 0 Format: S16 The input offset for green component.
11	31:17	**Reserved** Format: MBZ
	16:0	**Offset_in_B** Default Value: 0 Format: S16 The input offset for blue component.
12	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_R** Default Value: 0 Format: S16 The output offset for red component.
13	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_G** Default Value: 0 Format: S16 The output offset for green component.
14	31:17	**Reserved** Format: MBZ
	16:0	**Offset_out_B** Default Value: 0 Format: S16 The output offset for blue component.
15	31	**Reserved** Format: MBZ
	30	**FullRangeMappingEnable**

VEBOX_GAMUT_CONTROL_STATE

		Format: U1 **Value** / **Name** 0 — Basic Mode **[Default]** 1 — Advance Mode
	29:20	**d(in,default)** Default Value: 205 Format: U10 InnerTriangleMappingLength.
	19:10	**d(out, default)** Default Value: 164 Format: U10 OuterTriangleMappingLength.
	9:0	**d1(out)** Default Value: 287 Format: U10 OuterTriangleMappingLengthBelow.
16	31	**xvYccDecEncEnable** Format: U1 This bit is valid only when ColorGamutCompressionnEnable is on. **Value** / **Name** 1 — Both xvYcc decode and xvYcc encode are enabled **[Default]** 0 — To disable both xvYcc decode and xvYcc encode
	30:28	**CompressionLineShift** Format: U3 . **Value** / **Name** 3 — **[Default]** 0,4 —
	27:10	**Reserved** Format: MBZ
	9:0	**d1(in)** Default Value: 820 Format: U10 InnerTriangleMappingLengthBelow.

VEBOX_GAMUT_CONTROL_STATE

17	31:30	**GCC BasicModeSelection** Format: U2 **Value** | **Name** | **Description** 00b | Default **[Default]** | 01b | Scaling Factor | Used along with Dword66 Bits 28:11 10b | Single Axis Gamma Correction | Used along with Dword67 Bit 29 11b | Scaling factor with fixed luma | Used along with Dword37 Bits 28:11
	29	**LumaChromaOnlyCorrection** Format: U1 **Value** | **Name** 0 | Luma Only Correction **[Default]** 1 | Chorma Only Correction
	28:25	**Reserved** Format: MBZ
	24:11	**BasicModeScalingFactor** Format: U2.12 Used when FullRangeMappingEnable is in basic mode and base mode selection bit is set to scaling factor.
	10:1	**Reserved** Format: MBZ
	0	**Cpi Override** Format: U1 **Value** | **Name** 0 | **[Default]** 1 | Override Cpi calculation

VEBOX_IECP_STATE

VEBOX_IECP_STATE	
Source:	VideoEnhancementCS
Size (in bits):	3680
Default Value:	0x9A6E39F0, 0x00000000, 0x00000068, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0xDCDCDC00, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x01000001, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00004750, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00010000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0xDA004750, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000

DWord	Bit	Description
0..28	927:0	**STD/STE State** Format: **VEBOX_STD_STE_STATE** For description of this state, refer to *STD/STE State Section.*
29..41	415:0	**ACE State** Format: **VEBOX_ACE_LACE_STATE** For description of this state, refer to *ACE State Section.*
42..52	351:0	**TCC State** Format: **VEBOX_TCC_STATE** For description of this state, refer to *TCC State Section.*
53..54	63:0	**ProcAmp State** Format: **VEBOX_PROCAMP_STATE** For description of this state, refer to *ProcAmp State Section.*
55..66	383:0	**CSC State**

VEBOX_IECP_STATE

		Format: **VEBOX_CSC_STATE** For description of this state, refer to *CSC State Section*.
67..69	95:0	**Alpha/AOI State** Format: **VEBOX_ALPHA_AOI_STATE** For description of this state, refer to *Alpha State Section*.
70..84	479:0	**CCM State** Format: **VEBOX_CCM_STATE** For description of this state, refer to *CCM State Section*.
85..96	383:0	**Front-end CSC** Format: **VEBOX_FRONT_END_CSC_STATE** For description of this state, refer to *Front-end CSC State Section*.
97..114	575:0	**Gamut_STATE** Format: **VEBOX_GAMUT_CONTROL_STATE**

VEBOX_PROCAMP_STATE

VEBOX_PROCAMP_STATE		
Source:	VideoEnhancementCS	
Size (in bits):	64	
Default Value:	0x01000001, 0x01000000	

This state structure contains the IECP State Table Contents for ProcAmp state.

DWord	Bit	Description
0	31:28	**Reserved** Format: MBZ
	27:17	**Contrast** Default Value: 80h = 1.0 in fixed point U4.7 Format: U4.7 Contrast magnitude.
	16:13	**Reserved** Format: MBZ
	12:1	**Brightness** Default Value: 0 or 0.0 Format: S7.4 2's complement Brightness magnitude.
	0	**PROCAMP Enable** Default Value: 1 Format: Enable **Programming Notes** This bit must be set to "0"
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.

VEBOX_RGB_TO_GAMMA_CORRECTION

VEBOX_RGB_TO_GAMMA_CORRECTION		
Source:	VideoEnhancementCS	
Size (in bits):	64	
Default Value:	0x00000000, 0x00000000	
Color depth is 16 bits.		
DWord	**Bit**	**Description**
0..1	63:48	**B-ch Corrected Value** Default Value: 0h Format: U16
	47:32	**G-ch Corrected Value** Default Value: 0h Format: U16
	31:16	**R-ch Corrected Value** Default Value: 0h Format: U16
	15:0	**Pixel Value** Default Value: 0h Format: U16 **Programming Notes** N indicates the index into the table. Pixel value 0 and Pixel Value 1023 should be always programmed to 0 and 0xFFFF respectively.

intel

VEBOX_Scalar_State

VEBOX_Scalar_State		
Source:	BSpec	
Size (in bits):	960	
Default Value:	0x00000000, 0x00000020, 0x0008A003, 0x03FF0708, 0x26190C00, 0x3C804033	

DWord	Bit	Description
0..1	63:0	**Table Y-ch X-dir Filter Coefficient[0,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
2..3	63:0	**Table Y-ch Y-dir Filter Coefficient[0,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
4..5	63:0	**Table Y-ch X-dir Filter Coefficient[1,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
6..7	63:0	**Table Y-ch Y-dir Filter Coefficient[1,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
8..9	63:0	**Table UV-ch X-dir Filter Coefficient[0,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
10..11	63:0	**Table UV-ch Y-dir Filter Coefficient[0,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
12..13	63:0	**Table UV-ch X-dir Filter Coefficient[1,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
14..15	63:0	**Table UV-ch Y-dir Filter Coefficient[1,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
16..17	63:0	**Table UV-ch X-dir Filter Coefficient[2,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
18..19	63:0	**Table UV-ch Y-dir Filter Coefficient[2,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
20..21	63:0	**Table UV-ch X-dir Filter Coefficient[3,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
22..23	63:0	**Table UV-ch Y-dir Filter Coefficient[3,7:0]** Format: **VEBOX_Ch_Dir_Filter_Coefficient**
24	31:24	**Default Sharpness Level** Format: U8

VEBOX_Scalar_State

<table>
<tr><td></td><td></td><td>When adaptive scaling is off, determines the balance between sharp and smooth scalers.<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td>Smooth [Default]</td><td>Contribute 1 from the smooth scalar</td></tr><tr><td>255</td><td>Sharp</td><td>Contribute 1 from the sharp scalar</td></tr></table></td></tr>
<tr><td></td><td>23:16</td><td>Max Derivative 4 Pixels<table><tr><td>Format:</td><td>U8</td></tr></table>Used in adaptive filtering to specify the lower boundary of the smooth 4 pixel area.</td></tr>
<tr><td></td><td>15:8</td><td>Max Derivative 8 Pixels<table><tr><td>Format:</td><td>U8</td></tr></table>Used in adaptive filtering to specify the lower boundary of the smooth 8 pixel area.</td></tr>
<tr><td></td><td>7</td><td>Disable Adaptive Filter<table><tr><td>Format:</td><td>U1</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>When Adaptive filter is disabled the Default Sharpness Level is used to control the adaptive filtering.</td></tr></table></td></tr>
<tr><td></td><td>6:4</td><td>Transition Area with 4 Pixels<table><tr><td>Format:</td><td>U3</td></tr></table>Used in adaptive filtering to specify the width of the transition area for the 4 pixel calculation.</td></tr>
<tr><td></td><td>3</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>2:0</td><td>Transition Area with 8 Pixels<table><tr><td>Format:</td><td>U3</td></tr></table>Used in adaptive filtering to specify the width of the transition area for the 8 pixel calculation.</td></tr>
<tr><td>25</td><td>31:8</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>7:0</td><td>Edge_Th<table><tr><td>Default Value:</td><td>20h</td></tr><tr><td>Format:</td><td>U8</td></tr></table></td></tr>
<tr><td>26</td><td>31:22</td><td>Reserved<table><tr><td>Format:</td><td>MBZ</td></tr></table></td></tr>
<tr><td></td><td>21:18</td><td>DirDiag Threshold<table><tr><td>Format:</td><td>U4</td></tr></table></td></tr>
</table>

VEBOX_Scalar_State

		Value / **Name**: 2 = Default **[Default]**; [0, 2] = Valid Range
17:14	**Dir2 Threshold**	

Value	Name
2	Default **[Default]**
[0, 2]	Valid Range

17:14 Dir2 Threshold

Format:	U4

Value	Name
2	Default **[Default]**
[0, 2]	Valid Range

13 Mode

Default Value:	1h
Format:	U1

12 Disable Chroma Channel Diagonal Interpolation

Default Value:	0h
Format:	U1

Programming Notes
The Chroma Diagonal interpolation should be disabled when the input format is 420/422.

11:10 Chroma Co-sited Horizontal position

Format:	U2

Value	Name
0h	**[Default]**
[0, 2]	Valid Range

Programming Notes
Valid only when Chroma Diagonal interpolation is disabled. This is used only in 420/422 input surface formats. Should be set to 0 otherwise.

9:8 Chroma Co-sited Vertical position

Format:	U2

Value	Name
0h	**[Default]**
[0, 2]	Valid Range

Programming Notes
Valid only when Chroma Diagonal interpolation is disabled. This is used only in 420input

VEBOX_Scalar_State

		surface formats. Should be set to 0 otherwise.
	7:0	**Num_Trans_Th** Default Value: 3 Format: U8
27	31:16	**Vertical_TH** Default Value: 1023 Format: U16
	11:0	**Tearing_TH** Default Value: 708h Format: U12
28	31:24	**SAD_WT[3]** Default Value: 38 Format: U8
	23:16	**SAD_WT[2]** Default Value: 25 Format: U8
	15:8	**SAD_WT[1]** Default Value: 12 Format: U8
	7:0	**SAD_WT[0]** Default Value: 0 Format: U8
29	31:24	**D5_TH** Default Value: 60 Format: U8
	23:16	**Complexity_TH** Default Value: 128 Format: U8
	15:8	**SAD_WT[5]** Default Value: 64 Format: U8
	7:0	**SAD_WT[4]** Default Value: 51

VEBOX_Scalar_State			
		Format:	U8

VEBOX_STD_STE_STATE

VEBOX_STD_STE_STATE		
Source:	VideoEnhancementCS	
Size (in bits):	928	
Default Value:	0x9A6E39F0, 0x400D3C65, 0x000C9180, 0xFE2F2E00, 0x0003FFFF, 0x00140000, 0xD82E0640, 0x8285ECEC, 0x07FB8282, 0x00000000, 0x02117000, 0xA38FEC96, 0x0100C8C8, 0x003A6871, 0x01478000, 0x0007C300, 0x1291F008, 0x00094855, 0x1C1BD100, 0x03802008, 0x0002A980, 0x00080180, 0x0007CFFB, 0x18D1F07C, 0x000800BD, 0x1C080100, 0x03800000, 0x0008012B, 0x0008012B	

This state structure contains the state used by the STD/STE 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 / Name: 0: Output Pixels 1: Output STD Decisions
	1	**STE Enable**

VEBOX_STD_STE_STATE

		Format: Enable **Programming Notes** This bit must be set to "0"
	0	**STD Enable** Format: Enable **Programming Notes** This needs to be enabled if 'STD Score Output' is enabled. This bit must be set to "0"
1	31	**STD Score Output** Format: Enable
	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 Defines rectangle margin.
	17:10	**Cos(α)** Default Value: 79 Format: S0.7 2's complement The default is 79/128
	9:8	**Reserved** Format: MBZ
	7:0	**Sin(α)** Default Value: 101 Format: S0.7 2's complement The default is 101/128
2	31:21	**Reserved**

VEBOX_STD_STE_STATE		
		Format: MBZ
	20:13	**Diamond_alpha** Default Value: 100 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 Rhombus center shift in the hue-direction, relative to the rectangle center.
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

<table>
<tr><th colspan="3">VEBOX_STD_STE_STATE</th></tr>
<tr><td></td><td>17:13</td><td>Y_Slope_2
Default Value: 31
Format: U2.3
Slope between points Y3 and Y4.
The default is 31/8</td></tr>
<tr><td></td><td>12:8</td><td>Y_Slope_1
Default Value: 31
Format: U2.3
Slope between points Y1 and Y2.
The default is 31/8</td></tr>
<tr><td></td><td>7:0</td><td>Y_point_4
Default Value: 255
Format: U8
Fourth point of the Y piecewise linear membership function.</td></tr>
<tr><td>5</td><td>31:16</td><td>INV_Skin_types_margin
Format: U0.16
1/(2* Skin_types_margin)
<table><tr><th>Value</th><th>Name</th></tr><tr><td>20</td><td>Skin_Type_margin [Default]</td></tr><tr><td>1638</td><td></td></tr></table></td></tr>
<tr><td></td><td>15:0</td><td>INV_Margin_VYL
Format: U0.16
1 / Margin_VYL 1/ Margin_VYL = 3300/65536</td></tr>
<tr><td>6</td><td>31:24</td><td>P1L
Default Value: 216
Format: U8
Y Point 1 of the lower part of the detection PWLF.</td></tr>
<tr><td></td><td>23:16</td><td>P0L
Default Value: 46
Format: U8
Y Point 0 of the lower part of the detection PWLF.</td></tr>
<tr><td></td><td>15:0</td><td>INV_Margin_VYU
Default Value: 1600</td></tr>
</table>

VEBOX_STD_STE_STATE

		Format: U0.16 1 / Margin_VYU = 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** 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** Default Value: 7FBh 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

VEBOX_STD_STE_STATE

		V Bias 2 of the lower part of the detection PWLF.
9	31:22	**Reserved** Format: MBZ
	21:11	**S2L** Default Value: 0 Format: S2.8 2's complement The default is 0/256
	10:0	**S1L** Default Value: 0 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.
	18:11	**P0U** Default Value: 46 Format: U8 Y Point 0 of the upper part of the detection PWLF.
	10:0	**S3L** Default Value: 0 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**

VEBOX_STD_STE_STATE

DWord	Bit	Description
		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** Default Value: 256 Format: S2.8 2's complement Slope 0 of the upper part of the detection PWLF. The default is 256/256
	15:8	**B3U** Format: U8 V Bias 3 of the upper part of the detection PWLF. Value / Name: 200 **[Default]** 140
	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** Default Value: 74Dh Format: S2.8 2's complement

VEBOX_STD_STE_STATE

		Slope 2 of the upper part of the detection PWLF. The default is -179/256
	10:0	**S1U** Default Value: 113 Format: S2.8 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 Restrict Skin_types_thresh >= Skin_types_margin > 0 Restrict (Skin_types_thresh + Skin_types_margin) <= 255
	19:12	**Skin_types_thresh** Default Value: 120 Format: U8 Skin types Y margin Restrict Skin_types_thresh >= Skin_types_margin > 0 Restrict (Skin_types_thresh + Skin_types_margin) <= 255
	11	**Skin_Types_Enable** Default Value: 0 Disable Format: Enable Treat differently bright and dark skin types
	10:0	**S3U** Default Value: 0 Format: S2.8 2's complement 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

VEBOX_STD_STE_STATE

<table>
<tr><td></td><td></td><td>
<table><tr><td>First bias for the saturation PWLF (bright skin).</td></tr><tr><td>The default numerical value is -8/4</td></tr></table>
<table><tr><th>Value</th><th>Name</th></tr><tr><td>3F8h</td><td></td></tr></table>
</td></tr>
<tr><td></td><td>20:14</td><td>SATP3
<table><tr><td>Default Value:</td><td>31</td></tr><tr><td>Format:</td><td>S6 2's complement</td></tr></table>
Third point for the saturation PWLF (bright skin).</td></tr>
<tr><td></td><td>13:7</td><td>SATP2
<table><tr><td>Default Value:</td><td>6</td></tr><tr><td>Format:</td><td>S6 2's complement</td></tr></table>
Second point for the saturation PWLF (bright skin).</td></tr>
<tr><td></td><td>6:0</td><td>SATP1
<table><tr><td>Format:</td><td>S6 2's complement</td></tr></table>
First point for the saturation PWLF (bright skin).
The default numerical valueis -6/64.
<table><tr><th>Value</th><th>Name</th></tr><tr><td>7Ah</td><td></td></tr></table>
</td></tr>
<tr><td>16</td><td>31</td><td>Reserved
<table><tr><td>Format:</td><td>MBZ</td></tr></table>
</td></tr>
<tr><td></td><td>30:20</td><td>SATS0
<table><tr><td>Default Value:</td><td>297</td></tr><tr><td>Format:</td><td>U3.8</td></tr></table>
<table><tr><td>Zeroth slope for the saturation PWLF (bright skin)</td></tr><tr><td>The default is 297/256</td></tr></table>
</td></tr>
<tr><td></td><td>19:10</td><td>SATB3
<table><tr><td>Default Value:</td><td>124</td></tr><tr><td>Format:</td><td>S7.2 2's complement</td></tr></table>
<table><tr><td>Third bias for the saturation PWLF (bright skin)</td></tr><tr><td>The default is 124/4</td></tr></table>
</td></tr>
<tr><td></td><td>9:0</td><td>SATB2
<table><tr><td>Default Value:</td><td>8</td></tr><tr><td>Format:</td><td>S7.2 2's complement</td></tr></table>
</td></tr>
</table>

VEBOX_STD_STE_STATE		
		Second bias for the saturation PWLF (bright skin) The default is 8/4
17	31:22	**Reserved** Format: MBZ
	21:11	**SATS2** Default Value: 297 Format: U3.8 Second slope for the saturation PWLF (bright skin) The default is 297/256
	10:0	**SATS1** Default Value: 85 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** Default Value: 7Ah -6 Format: S6 2's complement First point for the hue PWLF (bright skin)
	10:0	**SATS3** Default Value: 256 Format: U3.8 Third slope for the saturation PWLF (bright skin) The default is 256/256

VEBOX_STD_STE_STATE		
19	31:30	**Reserved** Format: MBZ
	29:20	**HUEB3** Default Value: 56 Format: S7.2 2's complement Third bias for the hue PWLF (bright skin) The default is 56/4
	19:10	**HUEB2** Default Value: 8 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 Value / Name: 8 — **[Default]** 0xf8
20	31:22	**Reserved** Format: MBZ
	21:11	**HUES1** Default Value: 85 Format: U3.8 First slope for the hue PWLF (bright skin) The default is 85/256
	10:0	**HUES0** Default Value: 384 Format: U3.8 Zeroth slope for the hue PWLF (bright skin) The default is 384/256
21	31:22	**Reserved**

VEBOX_STD_STE_STATE		
		Format: MBZ
	21:11	**HUES3** Default Value: 256 Format: U3.8 Third slope for the hue PWLF (bright skin) The default is 256/256
	10:0	**HUES2** Default Value: 384 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** Default Value: 0 Format: S7.2 2's complement First bias for the saturation PWLF (dark skin) 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** Default Value: 7Bh Format: S6 2's complement First point for the saturation PWLF (dark skin) Default Value: -5
23	31	**Reserved** Format: MBZ

VEBOX_STD_STE_STATE

	30:20	**SATS0_DARK** Default Value: 397 Format: U3.8 Zeroth slope for the saturation PWLF (dark skin) The default is 397/256
	19:10	**SATB3_DARK** Default Value: 124 Format: S7.2 2's complement Third bias for the saturation PWLF (dark skin) The default is 124/4
	9:0	**SATB2_DARK** Default Value: 124 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** Default Value: 256 Format: U3.8 Second slope for the saturation PWLF (dark skin) The default is 256/256
	10:0	**SATS1_DARK** Default Value: 189 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**

VEBOX_STD_STE_STATE

		Default Value: 2 Format: S6 2's complement Second point for the hue PWLF (dark skin).
	17:11	**HUEP1_DARK** Default Value: 0 Format: S6 2's complement First point for the hue PWLF (dark skin).
	10:0	**SATS3_DARK** Default Value: 256 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** Default Value: 56 Format: S7.2 2's complement Third bias for the hue PWLF (dark skin). The default is 56/4
	19:10	**HUEB2_DARK** Default Value: 0 Format: S7.2 2's complement Second bias for the hue PWLF (dark skin). The default is 0/4
	9:0	**HUEB1_DARK** Default Value: 0 Format: S7.2 2's complement First bias for the hue PWLF (dark skin). The default is 0/4
27	31:22	**Reserved** Format: MBZ
	21:11	**HUES1_DARK**

VEBOX_STD_STE_STATE		
		Default Value: 256 Format: U3.8 First slope for the hue PWLF (dark skin). The default is 256/256
	10:0	**HUES0_DARK** Format: U3.8 Zeroth slope for the hue PWLF (dark skin). The default is 299/256 **Value** / **Name** 299 — **[Default]** 256
28	31:22	**Reserved** Format: MBZ
	21:11	**HUES3_DARK** Default Value: 256 Format: U3.8 Third slope for the hue PWLF (dark skin). The default is 256/256
	10:0	**HUES2_DARK** Default Value: 299 Format: U3.8 Second slope for the hue PWLF (dark skin). The default is 299/256

VEBOX_TCC_STATE

VEBOX_TCC_STATE

Source: VideoEnhancementCS
Size (in bits): 352
Default Value: 0xDCDCDC00, 0xDCDCDC00, 0x1E34CC91, 0x3E3CCE91, 0x02E80195, 0x0197046B, 0x01790174, 0x00096000, 0x00000000, 0x03030000, 0x009201C0

This state structure contains the IECP State Table Contents for TCC state.

DWord	Bit	Description
0	31:24	**SatFactor3** Format: U1.7 The saturation factor for yellow. The default is 220/128 Value / Name: 220 — **[Default]** 160
	23:16	**SatFactor2** Format: U1.7 The saturation factor for red. The default is 220/128 Value / Name: 220 — **[Default]** 160
	15:8	**SatFactor1** Format: U1.7 The saturation factor for magenta. The default is 220/128 Value / Name: 220 — **[Default]** 160
	7	**TCC Enable** Format: Enable

VEBOX_TCC_STATE

<table>
<tr><td></td><td></td><td>Programming Notes
This bit must be set to "0"</td></tr>
<tr><td></td><td>6:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>1</td><td>31:24</td><td>SatFactor6
Format: U1.7

The saturation factor for blue.
The default is 220/128

<table><tr><th>Value</th><th>Name</th></tr><tr><td>220</td><td>[Default]</td></tr><tr><td>160</td><td></td></tr></table></td></tr>
<tr><td></td><td>23:16</td><td>SatFactor5
Format: U1.7

The saturation factor for cyan.
The default is 220/128

<table><tr><th>Value</th><th>Name</th></tr><tr><td>220</td><td>[Default]</td></tr><tr><td>160</td><td></td></tr></table></td></tr>
<tr><td></td><td>15:8</td><td>SatFactor4
Format: U1.7

The saturation factor for green.
The default is 220/128

<table><tr><th>Value</th><th>Name</th></tr><tr><td>220</td><td>[Default]</td></tr><tr><td>160</td><td></td></tr></table></td></tr>
<tr><td></td><td>7:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>2</td><td>31:30</td><td>Reserved
Format: MBZ</td></tr>
<tr><td></td><td>29:20</td><td>BaseColor3
Default Value: 483</td></tr>
</table>

VEBOX_TCC_STATE

		Format: U10 Base Color 3 - this value must be greater than BaseColor2
	19:10	**BaseColor2** Default Value: 307 Format: U10 Base Color 2 - this value must be greater than BaseColor1
	9:0	**BaseColor1** Default Value: 145 Format: U10 Base Color 1
3	31:30	**Reserved** Format: MBZ
	29:20	**BaseColor6** Default Value: 995 Format: U10 Base Color 6 - this value must be greater than BaseColor5
	19:10	**BaseColor5** Default Value: 819 Format: U10 Base Color 5 - this value must be greater than BaseColor4
	9:0	**BaseColo4** Default Value: 657 Format: U10 Base Color 4 - this value must be greater than BaseColor3
4	31:16	**ColorTransitSlope23** Default Value: 744 Format: U0.16 The calculation result of 1 / (BC3 - BC2) [1/62]
	15:0	**ColorTransitSlope2** Default Value: 405 Format: U0.16 The calculation result of 1 / (BC2 - BC1) [1/57]

VEBOX_TCC_STATE

5	31:16	**ColorTransitSlope45** Default Value: 407 Format: U0.16 The calculation result of 1 / (BC5 - BC4) [1/57]
	15:0	**ColorTransitSlope34** Default Value: 1131 Format: U0.16 The calculation result of 1 / (BC4 - BC3) [1/61]
6	31:16	**ColorTransitSlope61** Default Value: 377 Format: U0.16 The calculation result of 1 / (BC1 - BC6) [1/62]
	15:0	**ColorTransitSlope56** Default Value: 372 Format: U0.16 The calculation result of 1 / (BC6 - BC5) [1/62]
7	31:22	**ColorBias3** Default Value: 0 Format: U2.8 Color bias for BaseColor3.
	21:12	**ColorBias2** Default Value: 150 Format: U2.8 Color bias for BaseColor2. The default is 150/256
	11:2	**ColorBias1** Default Value: 0 Format: U2.8 Color bias for BaseColor1.
	1:0	**Reserved** Format: MBZ
8	31:22	**ColorBias6**

VEBOX_TCC_STATE

		Default Value: 0 Format: U2.8 Color bias for BaseColor6.
	21:12	**ColorBias5** 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**

VEBOX_TCC_STATE		
		Format: MBZ
	2:0	**STE Slope Bits** Default Value: 0 Format: U3 Skin tone pixels enhancement slope bits.
10	31:16	**Inv_UVMaxColor** Default Value: 146 Format: U16 1 / UVMaxColor. Used for the SFs2 calculation.
	15:9	**Reserved** Format: MBZ
	8:0	**UVMaxColor** Default Value: 448 Format: U9 The maximum absolute value of the legal UV pixels. Used for the SFs2 calculation.

VEBOX_VERTEX_TABLE

VEBOX_VERTEX_TABLE	
Source:	VideoEnhancementCS
Size (in bits):	16384
Default Value:	0x00000000, 0x00000000,

VEBOX_VERTEX_TABLE

0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x00000000

DWord	Bit	Description

VEBOX_VERTEX_TABLE		
0..511	16383:0	**VertexTableEntry** Format: VEBOX_VERTEX_TABLE_ENTRY[512]

VEBOX_VERTEX_TABLE_ENTRY

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

DWord	Bit	Description
0	31:28	**Reserved** Format: MBZ
	27:16	**Vertex table entry 0 - Lv (12 bits)** Value: 100h-ED6h; Name: ; Description: Range for Vertices BT601 and BT709
	15:12	**Reserved** Format: MBZ
	11:0	**Vertex table entry 0 - Cv (12 bits)** Value: 400h-A00h; Name: ; Description: Range for Vertices BT601 and BT709

VECS Hardware-Detected Error Bit Definitions

<table>
<tr><th colspan="3">VECS Hardware-Detected Error Bit Definitions</th></tr>
<tr><td colspan="3">Source: VideoEnhancementCS
Size (in bits): 16
Default Value: 0x00000000</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="4">0</td><td>15:3</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>2</td><td>Command Privilege Violation Error
This bit is set if a command classified as privileged is parsed in a non-privileged batch buffer. The command will be converted to a NOOP and parsing will continue.</td></tr>
<tr><td>1</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>0</td><td>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:
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>1</td><td></td><td>Instruction Error detected</td></tr></table><table><tr><th>Programming Notes</th></tr><tr><td>This error indications cannot be cleared except by reset (i.e., it is a fatal error).</td></tr></table></td></tr>
</table>

VERTEX_BUFFER_STATE

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.

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	**Reserved** Format: MBZ
	24:23	**Reserved** Format: MBZ
	22:16	**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** If set, the Buffer Starting Address field is used to update the state of this buffer. If clear, that field is ignored and the previously-programmed value is maintained.
	13	**Null Vertex Buffer** Format: Enable This field enabled causes any fetch for vertex data to return 0. **Programming Notes** VERTEX_BUFFER_STATE.Null Vertex Buffer must be set when the VERTEX_BUFFER_STATE.Buffer Size is 0x0.
	12	**Reserved** Format: MBZ
	11:0	**Buffer Pitch** Format: U12 Count of bytes

VERTEX_BUFFER_STATE

DWord	Bit	Description
		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**: [0,4095] — **Name**: — **Description**: 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..2	63:0	**Buffer Starting Address** Format: GraphicsAddress[63:0] This field contains the byte-aligned Graphics Address LSBs 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 wrapped indices are subject to Max Index checking (see below).
3	31:0	**Buffer Size** Format: U32 Count of bytes This field specifies the size of the buffer in bytes. Vertex element accesses which straddle or go past the end of the buffer will return 0's for all elements.Note that BufferSize=0 indicates that there is no valid data in the buffer. **Value**: [0, FFFFFFFFh] — **Name**:

VERTEX_ELEMENT_STATE

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 one to four 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 (new) 3DSTATE_VF_SGVS command is used to specify optional insertion of VertexID and/or InstanceID into the input vertex data, logically following the processing of the VERTEX_ELEMENT_STATE structures. The VFCOMP_STORE_VID/IID encodings are no longer available in VERTEX_ELEMENT_STATE.
- When SourceElementFormat is set to one of the *64*_PASSTHRU formats, 64-bit components are stored in the URB without any conversion. In this case, vertex elements must be written as 128 or 256 bits, with VFCOMP_STORE_0 being used to pad the output as required. E.g., if R64_PASSTHRU is used to copy a 64-bit Red component into the URB, Component 1 must be specified as VFCOMP_STORE_0 (with Components 2,3 set to VFCOMP_NOSTORE) in order to output a 128-bit vertex element, or Components 1-3 must be specified as VFCOMP_STORE_0 in order to output a 256-bit vertex element. Likewise, use of R64G64B64_PASSTHRU requires Component 3 to be specified as VFCOMP_STORE_0 in order to output a 256-bit vertex element.
- When SourceElementFormat is set to one of the *64*_PASSTHRU formats then VFCOMP_STORE_SRC must be used for every valid component.
- Any SourceElementFormat of *64*_PASSTHRU cannot be used with an element which has edge flag enabled.

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

Software shall not attempt to disable any components (via 3DSTATE_VF_COMPONENT_PACKING) for elements associated with 256-bit SURFACE_FORMATs.

DWord	Bit	Description
0	31:26	**Vertex Buffer Index** Format: U6 This field specifies which vertex buffer the element is sourced from. **Value** | **Name**

VERTEX_ELEMENT_STATE

[0,32]	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	Name	Description
1h	TRUE	this vertex element is used in vertex assembly
0h	FALSE	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 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.

VERTEX_ELEMENT_STATE

DWord	Bit	Description
		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: [0,2047] — Name: (none) **Programming Notes** See note on 64-bit float alignment in Buffer Starting Address.
1	31	**Reserved** Format: MBZ
	30:28	**Component 0 Control** Format: 3D_Vertex_Component_Control Refer to the 3D_Vertex_Component_Control table below
	27	**Reserved** Format: MBZ
	26:24	**Component 1 Control** Format: 3D_Vertex_Component_Control Refer to the 3D_Vertex_Component_Control table below
	23	**Reserved** Format: MBZ
	22:20	**Component 2 Control** Format: 3D_Vertex_Component_Control Refer to the 3D_Vertex_Component_Control table below
	19	**Reserved** Format: MBZ
	18:16	**Component 3 Control** Format: 3D_Vertex_Component_Control Refer to the 3D_Vertex_Component_Control table below
	15:8	**Reserved** Format: MBZ

VERTEX_ELEMENT_STATE		
	7:0	**Reserved** Format: | MBZ

Vertical Line Stride Override Message Descriptor Control Field

MDC_VLSO - Vertical Line Stride Override Message Descriptor Control Field

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

DWord	Bit	Description
0	2	**Vertical Line Stride Override** Format: Enable If set, override the Vertical Line Stride and Vertical Line Stride Offset fields in the surface state with the fields below.
	1	**Vertical Line Stride** Format: U1 Specifies number of lines (0 or 1) to skip between logically adjacent lines - provides support of interleaved (field) surfaces as textures.
	0	**Vertical Line Stride Offset** Format: U1 Specifies the offset of the initial line from the beginning of the buffer. Ignored when Override VerticalLine Stride is 0.

VideoDecoder Interrupt Vector

VIDEODECODER_INTR_VEC - VideoDecoder Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
DWord	**Bit**	**Description**
0	15	**Catastrophic Error** This interrupt signals that a unrecoverable error (for e.g encountered fault when accessing a page mapped in Global GTT) during the engine processing. When Memory interface signals this error, the Command Streamer will stop parsing any more instructions. Scheduler is expected to reset the engine to evict the context
	14:12	**Reserved**
	11	**VCS Wait On Semaphore** Exec-List Scheduling: Set when MI_SEMAPHORE_WAIT command is un-successful and when "Inhibit Synchronous Context Switch" is set. Scheduler can use this interrupt to preempt the context waiting on semaphore wait. Ring Buffer Scheduling: Set when MI_SEMAPHORE_WAIT command is un-successful.
	10	**Reserved**
	9	**Reserved**
	8	**VCS Context Switch Interrupt** Set when a context switch has just occurred. **Execlist Enable bit** needs to be set for this interrupt to occur.
	7	**Legacy Context Per Process Page Fault Interrupt** Fault interrupt is generated by GA fabric, not by the CS This interrupt is for handling Legacy context PP GTT Page Fault.
	6	**VCS Watchdog Counter Expired** Set when the VCS timeout counter has reached the timeout thresh-hold value.
	5	**Reserved**
	4	**VCS MI Flush DW Notify** 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	**VCS Error Interrupt** 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 Blitter Instruction Parser encounters an error while parsing an

VIDEODECODER_INTR_VEC - VideoDecoder Interrupt Vector		
		instruction.
	2:1	**Reserved**
	0	**VCS1 MI User Interrupt** This status bit is set when an MI_USER_INTERRUPT instruction is executed on the Video 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.

VideoEnhancement Interrupt Vector

VIDEOENHANCE_INTR_VEC - VideoEnhancement Interrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	

DWord	Bit	Description
0	15	**Catastrophic Error** This interrupt signals that a unrecoverable error (for e.g encountered fault when accessing a page mapped in Global GTT) during the engine processing. When Memory interface signals this error, the Command Streamer will stop parsing any more instructions. Scheduler is expected to reset the engine to evict the context
	14:12	**Reserved**
	11	**VECS Wait On Semaphore**
	10	**Reserved**
	9	**Reserved**
	8	**VECS Context Switch Interrupt**
	7	**Legacy Context Per Process Page Fault Interrupt** Fault interrupt is generated by GA fabric, not by the CS This interrupt is for handling Legacy context PPTGTT Page Fault.
	6	**VECS Watchdog Counter Expired**
	5	**Reserved**
	4	**VECS MI Flush DW Notify**
	3	**VECS Error Interrupt**
	2:1	**Reserved**
	0	**VECS MI User Interrupt**

VP8 Encoder StreamOut Format

VP8 Encoder StreamOut Format		
Source:	VideoCS	
Size (in bits):	128	
Default Value:	0x00000000, 0x00000000, 0x00000000, 0x00000000	
DWord	**Bit**	**Description**
0	31:24	**MbY** Format: U8
	23:16	**MbX** Format: U8
	15:8	**MbClock16** Format: U8
	7:3	**Reserved** Format: MBZ
	2	**MbRcFlag** Format: U1
	1	**MBLevelInterMBConformanceFlag** Format: U1
	0	**MBLevelIntraMBConformanceFlag** Format: U1
1	31:29	**Reserved** Format: MBZ
	28:16	**MB_Residual_BitCount** Format: U13
	15:13	**Reserved** Format: MBZ
	12:0	**MB_Total_BitCount** Format: U13
2	31:25	**Reserved** Format: MBZ
	24:0	**Cbp** Format: U25
3	31	**Reserved** Format: MBZ
	30	**LastMbFlag**

VP8 Encoder StreamOut Format

		Format:	U1
	29	**IntraMBFlag**	
		Format:	U1
	28:24	**MBType5Bits**	
		Format:	U5
	23:19	**Reserved**	
		Format:	MBZ
	18	**QindexClampHigh**	
		Format:	U1
	17	**QindexClampLow**	
		Format:	U1
	16	**CoeffClampStatus**	
		Format:	U1
	15:0	**Reserved**	
		Format:	MBZ

WDBoxOAInterrupt Vector

WDOA_INTR_VEC - WDBoxOAInterrupt Vector		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	

DWord	Bit	Description
0	15:13	**Reserved**
	12	**Performance Monitoring Buffer Half-Full Interrupt** For internal trigger (timer event based) reporting, this interrupt is generated if the report buffer crosses the half full limit.
	7:6	**RESERVED**
	5	**WDBox 2 Status Interrupt**
	4	**WDBox 2 End of Frame Interrupt**
	3:2	**Reserved**
	1	**WDBox 1 Status Interrupt**
	0	**WDBox 1 End of Frame Interrupt**

WDE Packetization Parameters0

<table>
<tr><th colspan="3">WDE Packetization Parameters0</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 224
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">Please note that DW0-9, correspond to DW80-89 of WiGig Parameters.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>1</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>2</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>3..4</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>5</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>6..8</td><td>31:0</td><td>Reserved
Format: MBZ</td></tr>
<tr><td rowspan="3">9</td><td>31:24</td><td>GFX WNIC METADATA offset in CL
Offset relatives to 4K/8K TFD entry base address.
Programming Notes
This field should be set to 63 CLs (4032B) or 127 CLs (8132B) respective to 4K or 8KB GFX_WNIC_SHARED_DATABUFFER_PACKSIZE.</td></tr>
<tr><td>23:12</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>11:0</td><td>WNIC_TAIL_PTR_TRANSMIT_RATE
This field specifies the transmission rate at which the WDBOX sends the updated tail pointer message to the WNIC device.
<table>
<tr><th>Value</th><th>Name</th><th>Description</th></tr>
<tr><td>0</td><td></td><td>Illegal</td></tr>
<tr><td>1</td><td></td><td>Indicates a tail pointer message is sent for each TFD output by WDbox.</td></tr>
<tr><td>2-4095</td><td></td><td>WDbox shall send a tail pointer/interrupt at the granularity programmed.
A final/last tail pointer update message is sent at the end of each frame to flush incomplete/remaining data
i.e., a program value of 2 implies a single tail pointer message is sent for every two TFD packages or 2K buffers written into the circular buffer.</td></tr>
</table></td></tr>
</table>

WDE Packetization Parameters1

<table>
<tr><th colspan="3">WDE Packetization Parameters1</th></tr>
<tr><td colspan="3">Source: BSpec
Size (in bits): 192
Default Value: 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000</td></tr>
<tr><td colspan="3">Please note that DW0-5, correspond to DW90-95 of WiGig Parameters.</td></tr>
<tr><th>DWord</th><th>Bit</th><th>Description</th></tr>
<tr><td rowspan="2">0</td><td>31:16</td><td>GFX_WNIC_SHARED_DATABUFFER_STRIDE
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.
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
This field can be programmed to 4K or 8K Buffer stride size. Each TFD entry can only contain one WiGig WDE packet.
The maximum size of a WDE packet is 7920 bytes for 8K stride (this is the maximum size of the packets that go into the WiGig MAC (max MSDU size)) and 4032 for 4K stride.</td></tr>
<tr><td>15:0</td><td>Reserverd
Format: MBZ</td></tr>
<tr><td rowspan="2">1</td><td>31:16</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>15:0</td><td>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.
<table><tr><th>Value</th><th>Name</th><th>Description</th></tr><tr><td>0</td><td></td><td>Timer Timeout is Disabled</td></tr><tr><td>1-FFFFh</td><td></td><td>Number of Clock to wait before flushing the output queue</td></tr></table></td></tr>
<tr><td rowspan="3">2</td><td>31:28</td><td>Reserved
Format: MBZ</td></tr>
<tr><td>27:16</td><td>Start 2k byte offset position for AV multiplexer
Format: MBZ
This field should be set to zero in functional mode (Reserved) Reserved.</td></tr>
<tr><td>15:12</td><td>Reserved
Format: MBZ</td></tr>
</table>

WDE Packetization Parameters1		
	11:0	**Max Buffer pointer Minus 1** Format: U12 This field specifies the number TFD entries / Size of the Circular buffer. For example a value of 63 indicates 64 TFD entries. A value of 4095 indicates 4096 TFD entries.
3	31:16	**Reserved** Format: MBZ
	15	**Interrupt Direction Mode** Master Interrupt register in SA for WGBOX interrupt messages is located at 2005_020Ch upper 16 bits [31:16]. Value | Name | Description 0 | Host | Interrupt message is sent to SA (HOST) 1 | Reserved |
	14:2	**Reserved** Format: MBZ
	1	**TFD head pointer update interrupt message enable** If set, WGBOX will send interrupt message after updating TFD head pointer to WNIC.
	0	**EOF Interrupt Message Enable** If set, WGBOX will send interrupt message at the end of the frame to SA as indication of frame completion.
4	31:24	**Reserved** Format: MBZ
	23:0	**Video Packet ID Header Parameter** This field specifies the static program fields in MPEG/WDE header for each Video packet. **Programming Notes** • Bit 23 - Reserved. • Bit 22:16 - This field specifies the WDE PESXF stream ID (STIDEXT) (7-bit). The valid range for AVC video data is 0x48 to 0x4F. • Bit 15:8 - This field specifies the WDE subheader Stream ID (8-bit). • Bit 7:0 - This field specifies the WDE subheader field Program Number (8-bit).
5	31:0	**Reserved** Format: MBZ

WD Interrupt Bit Definition

WD Interrupt Bit Definition		
Source:	BSpec	
Size (in bits):	16	
Default Value:	0x00000000	
The WD Interrupt Registers all share the same bit definitions from this table.		

DWord	Bit	Description
0	15:9	**Reserved** Format: MBZ
	8	**Reserved**
	7	**WD_Frame_Complete** This event occurs when WD capture fully completes a frame.
	6	**WD_GTT_Fault** This event occurs when a GTT fault is detected.
	5	**WD_Vblank** This event occurs at the start of the WD internal vertical blank.
	4	**WD_Capture_sync** This event occurs when WD counter reached the programmed frame time interval.
	3	**WD_Capturing** This event occurs when WD capture starts to capture pixels.
	2	**WD_Writes_Complete** This event occurs when WD capture data writes complete for the current frame, before the data has been flushed to memory. WD Frame Complete should be used to find when the captured data can be accessed.
	1	**WD_TG_Late_Run** This event occurs when capsync for the next frame occurred before WD completed capturing all the pixels in the previous frame.
	0	**WD_WDBOX_Late_Run** This event occurs when capsync for the next frame occurred before the wdbox frame completion message was received for the previous frame.

WiGig Parameters

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

DWord	Bit	Description
0..9	31:0	**Encoder Control State Parameters0** For the description of this structure, please refer to *Encoder Control State Parameters0* in the following section.
10..19	31:0	**Encoder Base Address Parameters1** For the description of this structure, please refer to *Encoder Base Address Parameters1* in the following sections.
20..29	31:0	**Encoder Base Address Parameters2** For the description of this structure, please refer to *Encoder Base Address Parameters2* in the following sections.
30..39	31:0	**Encoder Base Address Parameters3** For the description of this structure, please refer to *Encoder Base Address Parameters3* in the following sections.
40..49	31:0	**Encoder Base Address Parameters4** For the description of this structure, please refer to *Encoder Base Address Parameters4* in the following sections.
50..59	31:0	**Encoder State Parameters5**

WiGig Parameters		
		For the description of this structure, please refer to *Encoder State Parameters5* in the following sections.
60..61	31:0	**Encoder State Parameters6** For the description of this structure, please refer to *Encoder State Parameters6* in the following sections.
62..63	31:0	**Display Source Parameters** For the description of this structure, please refer to *Display Source Parameters* in the following sections.
64..69	31:0	**Motion Decision Setting Parameters0** For the description of this structure, please refer to *Motion Decision Setting Parameters0* in the following section.
70..79	31:0	**Motion Decision Setting Parameters1** For the description of this structure, please refer to *Motion Decision Setting Parameters1* in the following section.
80..89	31:0	**Packetization Parameters0** for the description of this structure, please refer to *Packetization Parameters0* in the following section.
90..95	31:0	**Packetization Parameters1** for the description of this structure, please refer to *Packetization Parameters1* in the following section.
96..99	31:0	**Static Frame Control Parameters** For the description of this structure, please refer to *Control State Parameters0* in the following section.
100..109	31:0	**MBHRD State Parameters1** For the description of this structure, please refer to *MBHRD State Parameters1* in the following sections.
110..119	31:0	**MBHRD State Parameters2** For the description of this structure, please refer to *MBHRD State Parameters2* in the following sections.
120..127	31:0	**MBHRD State Parameters3** For the description of this structure, please refer to *MBHRD State Parameters3* in the following sections.