



Intel® Open Source HD Graphics and Intel Iris™ Graphics

Programmer's Reference Manual

For the 2014-2015 Intel Core™ Processors, Celeron™ Processors
and Pentium™ Processors based on the "Broadwell" Platform

Volume 2a: Command Reference: Instructions (Command Opcodes)

October 2015, Revision 1.1



Creative Commons License

You are free to Share - to copy, distribute, display, and perform the work under the following conditions:

- **Attribution.** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- **No Derivative Works.** You may not alter, transform, or build upon this work.

Notices and Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

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.

The products described in this document 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.

Implementations of the I2C bus/protocol may require licenses from various entities, including Philips Electronics N.V. and North American Philips Corporation.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.

Copyright © 2015, Intel Corporation. All rights reserved.



Table of Contents

3DPRIMITIVE	1
3DSTATE_AA_LINE_PARAMETERS	6
3DSTATE_BINDING_TABLE_EDIT_DS	8
3DSTATE_BINDING_TABLE_EDIT_GS	10
3DSTATE_BINDING_TABLE_EDIT_HS	12
3DSTATE_BINDING_TABLE_EDIT_PS	14
3DSTATE_BINDING_TABLE_EDIT_VS	16
3DSTATE_BINDING_TABLE_POINTERS_DS	18
3DSTATE_BINDING_TABLE_POINTERS_GS	20
3DSTATE_BINDING_TABLE_POINTERS_HS	22
3DSTATE_BINDING_TABLE_POINTERS_PS	24
3DSTATE_BINDING_TABLE_POINTERS_VS	26
3DSTATE_BINDING_TABLE_POOL_ALLOC	28
3DSTATE_BLEND_STATE_POINTERS	30
3DSTATE_CC_STATE_POINTERS	32
3DSTATE_CHROMA_KEY	34
3DSTATE_CLEAR_PARAMS	36
3DSTATE_CLIP	38
3DSTATE_CONSTANT_DS	45
3DSTATE_CONSTANT_GS	47
3DSTATE_CONSTANT_HS	49
3DSTATE_CONSTANT_PS	51
3DSTATE_CONSTANT_VS	53
3DSTATE_DEPTH_BUFFER	55
3DSTATE_DRAWING_RECTANGLE	62
3DSTATE_DS	65
3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC	74
3DSTATE_DX9_CONSTANTB_PS	76
3DSTATE_DX9_CONSTANTB_VS	78
3DSTATE_DX9_CONSTANTF_PS	80
3DSTATE_DX9_CONSTANTF_VS	82
3DSTATE_DX9_CONSTANTI_PS	84

3DSTATE_DX9_CONSTANTI_VS	86
3DSTATE_DX9_GENERATE_ACTIVE_PS.....	88
3DSTATE_DX9_GENERATE_ACTIVE_VS	90
3DSTATE_DX9_LOCAL_VALID_PS.....	92
3DSTATE_DX9_LOCAL_VALID_VS.....	94
3DSTATE_GATHER_CONSTANT_DS.....	96
3DSTATE_GATHER_CONSTANT_GS.....	99
3DSTATE_GATHER_CONSTANT_HS.....	102
3DSTATE_GATHER_CONSTANT_PS	105
3DSTATE_GATHER_CONSTANT_VS	108
3DSTATE_GATHER_POOL_ALLOC	111
3DSTATE_GS	113
3DSTATE_HIER_DEPTH_BUFFER.....	125
3DSTATE_HS	128
3DSTATE_INDEX_BUFFER.....	136
3DSTATE_LINE_STIPPLE	138
3DSTATE_MONOFILTER_SIZE.....	140
3DSTATE_MULTISAMPLE.....	142
3DSTATE_POLY_STIPPLE_OFFSET	145
3DSTATE_POLY_STIPPLE_PATTERN.....	147
3DSTATE_PS_BLEND	148
3DSTATE_PS.....	151
3DSTATE_PS_EXTRA	160
3DSTATE_PUSH_CONSTANT_ALLOC_DS	164
3DSTATE_PUSH_CONSTANT_ALLOC_GS	166
3DSTATE_PUSH_CONSTANT_ALLOC_HS	168
3DSTATE_PUSH_CONSTANT_ALLOC_PS.....	170
3DSTATE_PUSH_CONSTANT_ALLOC_VS.....	172
3DSTATE_RASTER	174
3DSTATE_SAMPLE_MASK	180
3DSTATE_SAMPLE_PATTERN	182
3DSTATE_SAMPLE_PATTERN	189
3DSTATE_SAMPLER_PALETTE_LOAD0	196
3DSTATE_SAMPLER_PALETTE_LOAD1	197



3DSTATE_SAMPLER_STATE_POINTERS_DS	199
3DSTATE_SAMPLER_STATE_POINTERS_GS	200
3DSTATE_SAMPLER_STATE_POINTERS_HS	201
3DSTATE_SAMPLER_STATE_POINTERS_PS	202
3DSTATE_SAMPLER_STATE_POINTERS_VS	203
3DSTATE_SBE	204
3DSTATE_SBE_SWIZ	208
3DSTATE_SCISSOR_STATE_POINTERS	210
3DSTATE_SF	211
3DSTATE_SO_BUFFER	216
3DSTATE_SO_DECL_LIST	219
3DSTATE_STENCIL_BUFFER	222
3DSTATE_STREAMOUT	225
3DSTATE_TE	231
3DSTATE_URB_DS	235
3DSTATE_URB_GS	237
3DSTATE_URB_HS	239
3DSTATE_URB_VS	241
3DSTATE_VERTEX_BUFFERS	243
3DSTATE_VERTEX_ELEMENTS	245
3DSTATE_VF	247
3DSTATE_VF_INSTANCING	249
3DSTATE_VF_SGVS	251
3DSTATE_VF_STATISTICS	254
3DSTATE_VF_TOPOLOGY	255
3DSTATE_VIEWPORT_STATE_POINTERS_CC	256
3DSTATE_VIEWPORT_STATE_POINTERS_SF_CLIP	257
3DSTATE_VS	258
3DSTATE_WM_CHROMAKEY	268
3DSTATE_WM_DEPTH_STENCIL	269
3DSTATE_WM	274
3DSTATE_WM_HZ_OP	281
A64 Byte Scattered Write MSD	287
A64 Dword Scattered Read MSD	289

A64 Dword Scattered Write MSD..... 291

A64 Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD293

A64 Dword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD 295

A64 Dword SIMD4x2 Untyped Atomic Integer Ternary with Return Data Operation MSD297

A64 Dword SIMD4x2 Untyped Atomic Integer Ternary Write Only Operation MSD 299

A64 Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD301

A64 Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD 303

A64 Dword Untyped Atomic Integer Binary with Return Data Operation MSD 305

A64 Dword Untyped Atomic Integer Binary Write Only Operation MSD..... 307

A64 Dword Untyped Atomic Integer Ternary with Return Data Operation MSD 309

A64 Dword Untyped Atomic Integer Ternary Write Only Operation MSD 311

A64 Dword Untyped Atomic Integer Unary with Return Data Operation MSD 313

A64 Dword Untyped Atomic Integer Unary Write Only Operation MSD 315

A64 Hword Block Read MSD..... 317

A64 Hword Block Write MSD..... 319

A64 Oword Block Read MSD..... 321

A64 Oword Block Write MSD..... 323

A64 Oword Dual Block Read MSD 325

A64 Oword Dual Block Write MSD 327

A64 Oword Unaligned Block Read MSD 329

A64 Qword Scattered Write MSD 331

A64 Qword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD333

A64 Qword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD 335

A64 Qword SIMD4x2 Untyped Atomic Integer Ternary with Return Data Operation MSD337

A64 Qword SIMD4x2 Untyped Atomic Integer Ternary Write Only Operation MSD 339

A64 Qword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD341

A64 Qword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD 343

A64 Qword Untyped Atomic Integer Binary with Return Data Operation MSD 345

A64 Qword Untyped Atomic Integer Binary Write Only Operation MSD 347

A64 Qword Untyped Atomic Integer Ternary with Return Data Operation MSD 349

A64 Qword Untyped Atomic Integer Ternary Write Only Operation MSD 351

A64 Qword Untyped Atomic Integer Unary with Return Data Operation MSD 353

A64 Qword Untyped Atomic Integer Unary Write Only Operation MSD 355

A64 Untyped Surface Read MSD 357



A64 Untyped Surface Write MSD	358
Addition	359
Addition with Carry	361
Arithmetic Shift Right	362
Average	364
Bit Field Extract	365
Bit Field Insert 1	368
Bit Field Insert 2	370
Bit Field Reverse	373
Branch Converging	374
Branch Diverging	376
Break	378
Byte Scattered Read MSD	380
Byte Scattered Write MSD	382
Call	384
Call Absolute	386
Compare	388
Compare NaN	390
Conditional Select	392
Conditional Send Message	395
Constant Cache Oword Block Read MSD	397
Constant Cache Oword Unaligned Block Read MSD	399
Continue	400
Count Bits Set	402
Dot Product 2	403
Dot Product 3	405
Dot Product 4	407
Dot Product Homogeneous	409
Dword Atomic Counter Binary with Return Data Operation MSD	411
Dword Atomic Counter Binary Write Only Operation MSD	412
Dword Atomic Counter Unary with Return Data Operation MSD	413
Dword Atomic Counter Unary Write Only Operation MSD	414
Dword Scattered Write MSD	415
Dword SIMD4x2 Atomic Counter Binary with Return Data Operation MSD	417

Dword SIMD4x2 Atomic Counter Binary Write Only Operation MSD 418

Dword SIMD4x2 Atomic Counter Unary with Return Data Operation MSD 419

Dword SIMD4x2 Atomic Counter Unary Write Only Operation MSD 420

Dword SIMD4x2 Typed Atomic Integer Binary with Return Data Operation MSD..... 421

Dword SIMD4x2 Typed Atomic Integer Binary Write Only Operation MSD 422

Dword SIMD4x2 Typed Atomic Integer Ternary with Return Data Operation MSD 423

Dword SIMD4x2 Typed Atomic Integer Ternary Write Only Operation MSD 424

Dword SIMD4x2 Typed Atomic Integer Unary with Return Data Operation MSD 425

Dword SIMD4x2 Typed Atomic Integer Unary Write Only Operation MSD 426

Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD 427

Dword SIMD4x2 Untyped Atomic Integer Ternary with Return Data Operation MSD ... 428

Dword SIMD4x2 Untyped Atomic Integer Ternary Write Only Operation MSD 429

Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD 430

Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD 431

Dword Typed Atomic Integer Binary with Return Data Operation MSD 432

Dword Typed Atomic Integer Binary Write Only Operation MSD 433

Dword Typed Atomic Integer Ternary with Return Data Operation MSD 434

Dword Typed Atomic Integer Ternary Write Only Operation MSD..... 435

Dword Typed Atomic Integer Unary with Return Data Operation MSD..... 436

Dword Typed Atomic Integer Unary Write Only Operation MSD..... 437

Else 438

End If..... 440

Extended Math Function 442

Find First Bit from LSB Side..... 445

Find First Bit from MSB Side 446

Fraction..... 448

Goto 449

GPGPU_CSR_BASE_ADDRESS 451

GPGPU_WALKER 453

Halt..... 457

HI8DS Render Target Write MSD 459

If 462

Illegal 464

Integer Subtraction with Borrow 465



Join	466
Jump Indexed	468
Leading Zero Detection	470
Line	471
Linear Interpolation	473
LO8DS Render Target Write MSD	476
Logic And	479
Logic Not	480
Logic Or	481
Logic Xor	483
MEDIA_CURBE_LOAD	485
MEDIA_INTERFACE_DESCRIPTOR_LOAD	487
MEDIA_OBJECT	489
MEDIA_OBJECT_GRPID	494
MEDIA_OBJECT_PRT	498
MEDIA_OBJECT_WALKER	500
MEDIA_STATE_FLUSH	507
MEDIA_VFE_STATE	509
Media Block Read MSD	517
Media Block Write MSD	518
Media Transpose Read MSD	519
Memory Fence MSD	520
MFC_AVC_PAK_OBJECT	522
MFC_MPEG2_PAK_OBJECT	524
MFC_MPEG2_SLICEGROUP_STATE	526
MFD_AVC_BSD_OBJECT	534
MFD_AVC_DPB_STATE	536
MFD_AVC_PICID_STATE	539
MFD_AVC_SLICEADDR	541
MFD_IT_OBJECT	543
MFD_JPEG_BSD_OBJECT	546
MFD_MPEG2_BSD_OBJECT	549
MFD_VC1_BSD_OBJECT	551
MFD_VC1_LONG_PIC_STATE	554

MFD_VC1_SHORT_PIC_STATE	569
MFD_VP8_BSD_OBJECT	579
MFV_AVC_DIRECTMODE_STATE	585
MFV_AVC_IMG_STATE	592
MFV_AVC_REF_IDX_STATE	612
MFV_AVC_SLICE_STATE	615
MFV_AVC_WEIGHTOFFSET_STATE	628
MFV_BSP_BUF_BASE_ADDR_STATE	630
MFV_DBK_OBJECT	638
MFV_FQM_STATE	648
MFV_IND_OBJ_BASE_ADDR_STATE	650
MFV_JPEG_HUFF_TABLE_STATE	665
MFV_JPEG_PIC_STATE	667
MFV_MPEG2_PIC_STATE	672
MFV_PAK_INSERT_OBJECT	687
MFV_PIPE_BUF_ADDR_STATE	691
MFV_PIPE_MODE_SELECT	713
MFV_QM_STATE	720
MFV_STATE_POINTER	722
MFV_STITCH_OBJECT	724
MFV_SURFACE_STATE	726
MFV_VC1_DIRECTMODE_STATE	734
MFV_VC1_PRED_PIPE_STATE	739
MFV_VP8_PAK_OBJECT	745
MFV_VP8_PIC_STATE	747
MFV_WAIT	776
MI_ARB_CHECK	777
MI_ARB_CHECK	778
MI_ARB_CHECK	779
MI_ARB_CHECK	780
MI_ARB_ON_OFF	781
MI_ATOMIC	783
MI_BATCH_BUFFER_END	789
MI_BATCH_BUFFER_END	790



MI_BATCH_BUFFER_END	791
MI_BATCH_BUFFER_END	792
MI_BATCH_BUFFER_START	793
MI_BATCH_BUFFER_START	796
MI_BATCH_BUFFER_START	799
MI_BATCH_BUFFER_START	801
MI_CLFLUSH	804
MI_CONDITIONAL_BATCH_BUFFER_END	806
MI_CONDITIONAL_BATCH_BUFFER_END	808
MI_CONDITIONAL_BATCH_BUFFER_END	810
MI_CONDITIONAL_BATCH_BUFFER_END	812
MI_COPY_MEM_MEM	814
MI_COPY_MEM_MEM	817
MI_COPY_MEM_MEM	819
MI_COPY_MEM_MEM	822
MI_DISPLAY_FLIP	825
MI_FLUSH_DW	830
MI_FLUSH_DW	834
MI_FLUSH_DW	838
MI_LOAD_REGISTER_IMM	841
MI_LOAD_REGISTER_IMM	843
MI_LOAD_REGISTER_IMM	845
MI_LOAD_REGISTER_IMM	847
MI_LOAD_REGISTER_MEM	849
MI_LOAD_REGISTER_REG	852
MI_LOAD_SCAN_LINES_EXCL	854
MI_LOAD_SCAN_LINES_EXCL	856
MI_LOAD_SCAN_LINES_INCL	858
MI_LOAD_SCAN_LINES_INCL	860
MI_LOAD_URB_MEM	862
MI_MATH	864
MI_MATH	865
MI_MATH	866
MI_MATH	867



MI_NOOP	868
MI_NOOP	869
MI_NOOP	870
MI_NOOP	871
MI_PREDICATE	872
MI_REPORT_HEAD	874
MI_REPORT_HEAD	875
MI_REPORT_HEAD	876
MI_REPORT_HEAD	877
MI_REPORT_PERF_COUNT	878
MI_RS_CONTEXT	880
MI_RS_CONTROL	881
MI_RS_STORE_DATA_IMM	883
MI_SEMAPHORE_SIGNAL	885
MI_SEMAPHORE_WAIT	888
MI_SEMAPHORE_WAIT	892
MI_SEMAPHORE_WAIT	895
MI_SEMAPHORE_WAIT	899
MI_SEMAPHORE_WAIT	902
MI_SET_CONTEXT	905
MI_SET_PREDICATE	908
MI_STORE_DATA_IMM	910
MI_STORE_DATA_IMM	912
MI_STORE_DATA_IMM	915
MI_STORE_DATA_IMM	918
MI_STORE_DATA_INDEX	921
MI_STORE_DATA_INDEX	923
MI_STORE_DATA_INDEX	925
MI_STORE_DATA_INDEX	927
MI_STORE_REGISTER_MEM	929
MI_STORE_URB_MEM	932
MI_SUSPEND_FLUSH	934
MI_SUSPEND_FLUSH	935
MI_SUSPEND_FLUSH	936



MI_SUSPEND_FLUSH	937
MI_TOPOLOGY_FILTER	938
MI_UPDATE_GTT	939
MI_UPDATE_GTT	940
MI_UPDATE_GTT	942
MI_UPDATE_GTT	943
MI_URB_ATOMIC_ALLOC	945
MI_URB_CLEAR	946
MI_USER_INTERRUPT	948
MI_USER_INTERRUPT	949
MI_USER_INTERRUPT	950
MI_USER_INTERRUPT	951
MI_WAIT_FOR_EVENT.....	952
MI_WAIT_FOR_EVENT.....	956
Move	962
Move Indexed.....	964
Multiply	966
Multiply Accumulate	968
Multiply Accumulate High	969
Multiply Add.....	971
Multiply Add for Macro	974
No Operation.....	977
Oword Block Read MSD.....	978
Oword Block Write MSD.....	980
Oword Dual Block Write MSD	981
Oword Unaligned Block Read MSD	982
PIPE_CONTROL.....	983
PIPELINE_SELECT	992
Plane	994
REP16 Render Target Write MSD.....	996
Return	999
Round Down.....	1001
Round to Nearest or Even	1002
Round to Zero	1003

Round Up	1004
Scattered Move	1005
Scratch Block Read MSD	1007
Scratch Block Write MSD	1009
Select	1011
Send Message	1013
Shift Left	1016
Shift Right	1018
SIMD8 Render Target Write MSD	1019
SIMD16 Render Target Write MSD	1022
STATE_BASE_ADDRESS	1025
STATE_PREFETCH	1033
STATE_SIP	1035
Sum of Absolute Difference 2	1036
Sum of Absolute Difference Accumulate 2	1037
SWTESS_BASE_ADDRESS	1039
Typed Surface Read MSD	1041
Typed Surface Write MSD	1042
Untyped Surface Write MSD	1043
URB Hword Dual Block Read MSD	1044
URB Hword Dual Block Write MSD	1046
URB Oword Block Write MSD	1048
URB Oword Dual Block Read MSD	1050
URB Oword Dual Block Write MSD	1052
VEBOX_STATE	1054
VEBOX_SURFACE_STATE	1060
Wait Notification	1067
While	1069
XY_COLOR_BLT	1071
XY_FULL_BLT	1074
XY_FULL_IMMEDIATE_PATTERN_BLT	1078
XY_FULL_MONO_PATTERN_BLT	1081
XY_FULL_MONO_PATTERN_MONO_SRC_BLT	1085
XY_FULL_MONO_SRC_BLT	1089



XY_FULL_MONO_SRC_IMMEDIATE_PATTERN_BLT	1093
XY_MONO_PAT_BLT	1096
XY_MONO_PAT_FIXED_BLT	1099
XY_MONO_SRC_COPY_BLT	1103
XY_MONO_SRC_COPY_IMMEDIATE_BLT	1106
XY_PAT_BLT	1109
XY_PAT_BLT_IMMEDIATE	1112
XY_PAT_CHROMA_BLT	1115
XY_PAT_CHROMA_BLT_IMMEDIATE	1118
XY_PIXEL_BLT	1121
XY_SCANLINES_BLT	1122
XY_SETUP_BLT	1123
XY_SETUP_CLIP_BLT	1126
XY_SETUP_MONO_PATTERN_SL_BLT	1127
XY_SRC_COPY_BLT	1130
XY_SRC_COPY_CHROMA_BLT	1133
XY_TEXT_BLT	1137
XY_TEXT_IMMEDIATE_BLT	1139

3DPRIMITIVE

3DPRIMITIVE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DPRIMITIVE command is used to submit 3D primitives to be processed by the 3D pipeline. Typically the processing results in rendering pixel data into the render targets, but this is not required. The parameters passed in this command are forwarded to the Vertex Fetch function. The Vertex Fetch function will use this information to generate vertex data structures and store them in the URB. These vertices are then passed down the 3D pipeline.</p>			
Programming Notes			
<p>If the threads spawned by this command are required to observe memory writes performed by threads spawned from a previous command, software must precede this command with a command that performs a (preferably pipelined) memory flush (e.g., 3D_PIPECONTROL).</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	3h 3DPRIMITIVE
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		0h 3DPRIMITIVE	
Format:		OpCode	
15	Reserved		
14	Project:	BDW	
	Format:	MBZ	
13	Project:	BDW	
	Format:	MBZ	
12	Project:	BDW	
	Format:	MBZ	
11	Project:	BDW	
	Format:	MBZ	

3DPRIMITIVE						
10	<p>Indirect Parameter Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>If set, the values in DW 2-5 are ignored and replaced by the current values of the corresponding 3DPRIM_xxx MMIO registers:</p> <ul style="list-style-type: none"> • 3DPRIM_VERTEX_COUNT (instead of DW2: VertexCountPerInstance) • 3DPRIM_START_VERTEX (instead of DW3: StartVertexLocation) • 3DPRIM_INSTANCE_COUNT (instead of DW4: InstanceCount) • 3DPRIM_START_INSTANCE (instead of DW5: StartInstanceLocation) • 3DPRIM_BASE_VERTEX (instead of DW6: BaseVertexLocation) <p>Indirect Parameter Enable and End Offset Enable shall not be ENABLED at the same time, or behavior is UNDEFINED.</p>	Format:	Enable			
	Format:	Enable				
	<p>9 UAV Coherency Required</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>SW will be required to set this bit if there is the possibility of sharing a UAV from a previous 3DPRIMITVE command. If set, this command may cause a flush due to UAV coherency requirements. If none of the shaders have UAV access enabled, then this bit is ignored.</p>	Project:	BDW	Format:	U1	
	Project:	BDW				
Format:	U1					
<p>8 Predicate Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>If set, this command is executed (or not) depending on the current value of the MI Predicate internal state bit. This command is ignored only if PredicateEnable is set and the Predicate state bit is 0.</p>	Format:	Enable				
Format:	Enable					
<p>7:0 DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>5h Excludes DWord (0,1)</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	5h Excludes DWord (0,1)	Project:	BDW	Format:	=n Total Length - 2
Default Value:	5h Excludes DWord (0,1)					
Project:	BDW					
Format:	=n Total Length - 2					
1	<p>31:10 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					

3DPRIMITIVE										
9	<p>End Offset Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>If set, the Vertex Count Per Instance field is IGNORED, and the 3DPRIM_END_OFFSET register is used to indirectly specify the vertex count by defining the amount of valid data in VB0. The following restrictions apply:</p> <ul style="list-style-type: none"> • VB0 must be enabled for use • VertexAccessType = SEQUENTIAL • Start Vertex Location = 0 • Start Instance Location = 0 • Base Vertex Location = 0 <p>Vertices are output until EndOffset is reached or exceeded in VB0. If EndOffset is reached or exceeded within the data associated with a vertex, that vertex is considered incomplete and will not be output. Partial objects will be discarded (as is normally done). If clear, End Offset is ignored. Indirect Parameter Enable and End Offset Enable must not be ENABLED at the same time, or behavior is UNDEFINED.</p>	Format:	Enable							
Format:	Enable									
8	<p>Vertex Access Type</p> <p>This field specifies how data held in vertex buffers marked as VERTEXDATA is accessed by Vertex Fetch.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">SEQUENTIAL</td> <td>VERTEXDATA buffers are accessed sequentiallyRequiref if End Offset Enable is ENABLED.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">RANDOM</td> <td>VERTEXDATA buffers are accessed randomly via an index obtained from the Index Buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	SEQUENTIAL	VERTEXDATA buffers are accessed sequentiallyRequiref if End Offset Enable is ENABLED.	1h	RANDOM	VERTEXDATA buffers are accessed randomly via an index obtained from the Index Buffer.
Value	Name	Description								
0h	SEQUENTIAL	VERTEXDATA buffers are accessed sequentiallyRequiref if End Offset Enable is ENABLED.								
1h	RANDOM	VERTEXDATA buffers are accessed randomly via an index obtained from the Index Buffer.								
7:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
5:0	<p>Primitive Topology Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Format:</td> <td>3D_Prim_Topo_Type [BDW] See table below for encoding, see 3D Overview for diagrams and general comments</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="text-align: center; background-color: #e1eef6;">Description</th> </tr> </thead> <tbody> <tr> <td>This field specifies the topology type of 3D primitive generated by this command. Note that a single primitive topology (list/strip/fan/etc.) can contain a number of basic objects (lines, triangles, etc.).</td> </tr> <tr> <td>This field is ignored. The topology type is specified via the 3DSTATE_VF_TOPOLOGY command.</td> </tr> </tbody> </table>	Format:	3D_Prim_Topo_Type [BDW] See table below for encoding, see 3D Overview for diagrams and general comments	Description	This field specifies the topology type of 3D primitive generated by this command. Note that a single primitive topology (list/strip/fan/etc.) can contain a number of basic objects (lines, triangles, etc.).	This field is ignored. The topology type is specified via the 3DSTATE_VF_TOPOLOGY command.				
Format:	3D_Prim_Topo_Type [BDW] See table below for encoding, see 3D Overview for diagrams and general comments									
Description										
This field specifies the topology type of 3D primitive generated by this command. Note that a single primitive topology (list/strip/fan/etc.) can contain a number of basic objects (lines, triangles, etc.).										
This field is ignored. The topology type is specified via the 3DSTATE_VF_TOPOLOGY command.										

		3DPRIMITIVE							
2	31:0	Vertex Count Per Instance <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U32 Count of vertices</td> </tr> </table> <p>This field specifies how many vertices are to be generated for each instance of the primitive topology. If End Offset Enable is clear: Format = U32 count of vertices Range = [0, 2³²-1] (upper limit probably constrained by VB size) Ignored if End Offset Enable or Indirect Parameter Enable is ENABLED.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> <tr> <td colspan="2"> <ul style="list-style-type: none"> This per-instance value should specify a valid number of vertices for the primitive topology type. E.g., for 3DPRIM_TRILIST_ADJ, this field should specify a multiple of 6 vertices. However, in cases where too few or too many vertices are provided, the unused vertices will be silently discarded by the pipeline. A 0 value in this field effectively makes the command a 'no-operation'. </td> </tr> </table>		Format:	U32 Count of vertices	Programming Notes		<ul style="list-style-type: none"> This per-instance value should specify a valid number of vertices for the primitive topology type. E.g., for 3DPRIM_TRILIST_ADJ, this field should specify a multiple of 6 vertices. However, in cases where too few or too many vertices are provided, the unused vertices will be silently discarded by the pipeline. A 0 value in this field effectively makes the command a 'no-operation'. 	
Format:	U32 Count of vertices								
Programming Notes									
<ul style="list-style-type: none"> This per-instance value should specify a valid number of vertices for the primitive topology type. E.g., for 3DPRIM_TRILIST_ADJ, this field should specify a multiple of 6 vertices. However, in cases where too few or too many vertices are provided, the unused vertices will be silently discarded by the pipeline. A 0 value in this field effectively makes the command a 'no-operation'. 									
3	31:0	Start Vertex Location <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U32 structure index</td> </tr> </table> <p>This field specifies the "starting vertex" for each instance. This allows skipping over part of the vertices in a buffer if, for example, a previous 3DPRIMITIVE command had already drawn the primitives associated with the earlier entries. For SEQUENTIAL access, this field specifies, for each instance, a starting structure index into the vertex buffers For RANDOM access, this field specifies, for each instance, a starting index into the Index Buffer.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> <tr> <td colspan="2"> <ul style="list-style-type: none"> Access of any data outside of the valid extent of a vertex or index buffer will return the value 0 (i.e., appears as if the data stored at the invalid location was 0). Must be set to 0 if End Offset Enable is ENABLED. Ignored if Indirect Parameter Enable is ENABLED </td> </tr> </table>		Format:	U32 structure index	Programming Notes		<ul style="list-style-type: none"> Access of any data outside of the valid extent of a vertex or index buffer will return the value 0 (i.e., appears as if the data stored at the invalid location was 0). Must be set to 0 if End Offset Enable is ENABLED. Ignored if Indirect Parameter Enable is ENABLED 	
Format:	U32 structure index								
Programming Notes									
<ul style="list-style-type: none"> Access of any data outside of the valid extent of a vertex or index buffer will return the value 0 (i.e., appears as if the data stored at the invalid location was 0). Must be set to 0 if End Offset Enable is ENABLED. Ignored if Indirect Parameter Enable is ENABLED 									
4	31:0	Instance Count <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U32 Count of instances</td> </tr> </table> <p>This field specifies the number of instances by which the primitive topology is to be regenerated. A value of 0 indicates "no instances" (no-op operation). A value of 1 effectively specifies "non-instanced" operation, though vertex buffers will still be used to provide instance data, if so programmed. Ignored if Indirect Parameter Enable is ENABLED.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <thead> <tr> <th style="width: 70%; text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, FFFFFFFFh]</td> <td></td> </tr> </tbody> </table>		Format:	U32 Count of instances	Value	Name	[0, FFFFFFFFh]	
Format:	U32 Count of instances								
Value	Name								
[0, FFFFFFFFh]									

3DPRIMITIVE				
5	31:0	Start Instance Location		
		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U32 structure index</td> </tr> </table>	Format:	U32 structure index
		Format:	U32 structure index	
		Description		
<p>This field specifies the "starting instance" for the command as an initial structure index into Vertex Buffers for vertex elements with InstancingEnable set.</p> <p>Subsequent instances will access sequential instance data structures, as controlled by the Instance Data Step Rate.</p>				
6	31:0	Base Vertex Location		
		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>S31 index structure bias</td> </tr> </table>	Format:	S31 index structure bias
		Format:	S31 index structure bias	
		Programming Notes		
<ul style="list-style-type: none"> • Access of any data outside of the valid extent of a vertex or index buffer will return the value 0 (i.e., appears as if the data stored at the invalid location was 0). • Must be set to 0 if End Offset Enable is ENABLED. • Ignored if Indirect Parameter Enable is ENABLED. 				

3DSTATE_AA_LINE_PARAMETERS

3DSTATE_AA_LINE_PARAMETERS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_AA_LINE_PARAMS command is used to specify the slope and bias terms used in the improved alpha coverage computation (specifically for DX WHQL compliance). Note that in these devices the coverage values passed to PS threads are full U0.8 values, versus [DevBWR, DevCLN] where U0.4 values are passed.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		0Ah 3DSTATE_AA_LINE_PARAMETERS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	1h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:24	AA Point Coverage Bias	
		Project:	BDW
		Format:	U0.8
	This field specifies the bias term to be used in the aa coverage computation for edges 0 and 3.		
	23:16	AA Coverage Bias	
Project:		All	
Format:		U0.8	
This field specifies the bias term to be used in the aa coverage computation for edges 0 and 3.			



3DSTATE_AA_LINE_PARAMETERS						
	15:8	<p>AA Point Coverage Slope</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the slope term to be used in the aa coverage computation for edges 0 and 3. If this field is zero, the Windower will revert to legacy aa line coverage computation (though still output expanded U0.8 coverage values).</p>	Project:	BDW	Format:	U0.8
	Project:	BDW				
Format:	U0.8					
7:0	<p>AA Coverage Slope</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the slope term to be used in the aa coverage computation for edges 0 and 3. If this field is zero, the Windower will revert to legacy aa line coverage computation (though still output expanded U0.8 coverage values).</p>	Project:	All	Format:	U0.8	
Project:	All					
Format:	U0.8					
2	31:24	<p>AA Point Coverage EndCap Bias</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the bias term to be used in the aa coverage computation for edges 1 and 2.</p>	Project:	BDW	Format:	U0.8
	Project:	BDW				
	Format:	U0.8				
	23:16	<p>AA Coverage EndCap Bias</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the bias term to be used in the aa coverage computation for edges 1 and 2.</p>	Project:	All	Format:	U0.8
Project:	All					
Format:	U0.8					
15:8	<p>AA Point Coverage EndCap Slope</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the slope term to be used in the aa coverage computation for edges 1 and 2.</p>	Project:	BDW	Format:	U0.8	
Project:	BDW					
Format:	U0.8					
7:0	<p>AA Coverage EndCap Slope</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.8</td> </tr> </table> <p>This field specifies the slope term to be used in the aa coverage computation for edges 1 and 2.</p>	Project:	All	Format:	U0.8	
Project:	All					
Format:	U0.8					

3DSTATE_BINDING_TABLE_EDIT_DS

3DSTATE_BINDING_TABLE_EDIT_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command edits the binding table for DS.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		46h 3DSTATE_BINDING_TABLE_EDIT_DS	
Format:		OpCode	
15:9	Reserved		
	Format:	MBZ	
8:0	DWord Length		
	Format:	=n	
	Value	Name	
	0h	DWORD_COUNT_n [Default]	
	0h - 100h	Range	
1	31:16	Binding Table Block Clear	
		Format:	U16
	<p>Each bit in this field corresponds to a 16 entry block of the binding table. Bit 0 of this field corresponds to entries 0-15, bit 1 to 16-31, and so on. When a bit is set it clears the corresponding bind table entries to 0. (effectively disabling them). The clear is applied before the individual binding table entries contained in this message are applied. When this bit is clear then the corresponding 16 entry block is not cleared.</p>		
15:2	Reserved		
Format:	MBZ		



3DSTATE_BINDING_TABLE_EDIT_DS																	
	1:0	Binding Table Edit Target Specifies which core should respond to this 3DSTATE_BINDING_TABLE_EDIT_DS command:															
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>11b</td> <td>All Cores</td> <td>All cores should respond to this command</td> </tr> <tr> <td>10b</td> <td>Core 1</td> <td>Only Core1 should respond to this command</td> </tr> <tr> <td>01b</td> <td>Core 0</td> <td>Only Core0 should respond to this command</td> </tr> <tr> <td>00b</td> <td>Reserved</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	11b	All Cores	All cores should respond to this command	10b	Core 1	Only Core1 should respond to this command	01b	Core 0	Only Core0 should respond to this command	00b	Reserved	Reserved
		Value	Name	Description													
		11b	All Cores	All cores should respond to this command													
		10b	Core 1	Only Core1 should respond to this command													
01b	Core 0	Only Core0 should respond to this command															
00b	Reserved	Reserved															
2..n	31:0	Entry [n] Format: BINDING_TABLE_EDIT_ENTRY															

3DSTATE_BINDING_TABLE_EDIT_GS

3DSTATE_BINDING_TABLE_EDIT_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command edits the binding table for GS.;			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		44h 3DSTATE_BINDING_TABLE_EDIT_GS	
Format:		OpCode	
15:9	Reserved		
	Format:	MBZ	
8:0	DWord Length		
	Format:	=n	
	Value	Name	
	0h	DWORD_COUNT_n [Default]	
	0h - 100h	Range	
1	31:16	Binding Table Block Clear	
		Format:	U16
	<p>Each bit in this field corresponds to a 16 entry block of the binding table. Bit 0 of this field corresponds to entries 0-15, bit 1 to 16-31, and so on. When a bit is set it clears the corresponding bind table entries to 0. (effectively disabling them). The clear is applied before the individual binding table entries contained in this message are applied. When this bit is clear then the corresponding 16 entry block is not cleared.</p>		
15:2	Reserved		
	Format:	MBZ	



3DSTATE_BINDING_TABLE_EDIT_GS																	
	1:0	Binding Table Edit Target Specifies which core should respond to this 3DSTATE_BINDING_TABLE_EDIT_GS command:															
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>11b</td> <td>All Cores</td> <td>All cores should respond to this command</td> </tr> <tr> <td>10b</td> <td>Core 1</td> <td>Only Core1 should respond to this command</td> </tr> <tr> <td>01b</td> <td>Core 0</td> <td>Only Core0 should respond to this command</td> </tr> <tr> <td>00b</td> <td>Reserved</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	11b	All Cores	All cores should respond to this command	10b	Core 1	Only Core1 should respond to this command	01b	Core 0	Only Core0 should respond to this command	00b	Reserved	Reserved
		Value	Name	Description													
		11b	All Cores	All cores should respond to this command													
		10b	Core 1	Only Core1 should respond to this command													
01b	Core 0	Only Core0 should respond to this command															
00b	Reserved	Reserved															
2..n	31:0	Entry [n] Format: BINDING_TABLE_EDIT_ENTRY															

3DSTATE_BINDING_TABLE_EDIT_HS

3DSTATE_BINDING_TABLE_EDIT_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command edits the binding table for HS.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		45h 3DSTATE_BINDING_TABLE_EDIT_HS	
Format:		OpCode	
15:9	Reserved		
	Format:	MBZ	
8:0	DWord Length		
	Format:	=n	
	Value	Name	
	0h	DWORD_COUNT_n [Default]	
	0h - 100h	Range	
1	31:16	Binding Table Block Clear	
		Format:	U16
	<p>Each bit in this field corresponds to a 16 entry block of the binding table. Bit 0 of this field corresponds to entries 0-15, bit 1 to 16-31, and so on. When a bit is set it clears the corresponding bind table entries to 0. (effectively disabling them). The clear is applied before the individual binding table entries contained in this message are applied. When this bit is clear then the corresponding 16 entry block is not cleared.</p>		
15:2	Reserved		
Format:	MBZ		



3DSTATE_BINDING_TABLE_EDIT_HS																	
	1:0	<p>Binding Table Edit Target Specifies which core should respond to this 3DSTATE_BINDING_TABLE_EDIT_HS command:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">11b</td> <td style="text-align: center;">All Cores</td> <td>All cores should respond to this command</td> </tr> <tr> <td style="text-align: center;">10b</td> <td style="text-align: center;">Core 1</td> <td>Only Core1 should respond to this command</td> </tr> <tr> <td style="text-align: center;">01b</td> <td style="text-align: center;">Core 0</td> <td>Only Core0 should respond to this command</td> </tr> <tr> <td style="text-align: center;">00b</td> <td style="text-align: center;">Reserved</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	11b	All Cores	All cores should respond to this command	10b	Core 1	Only Core1 should respond to this command	01b	Core 0	Only Core0 should respond to this command	00b	Reserved	Reserved
		Value	Name	Description													
		11b	All Cores	All cores should respond to this command													
		10b	Core 1	Only Core1 should respond to this command													
		01b	Core 0	Only Core0 should respond to this command													
00b	Reserved	Reserved															
2..n	31:0	<p>Entry [n]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>BINDING_TABLE_EDIT_ENTRY</td> </tr> </table>	Format:	BINDING_TABLE_EDIT_ENTRY													
Format:	BINDING_TABLE_EDIT_ENTRY																

3DSTATE_BINDING_TABLE_EDIT_PS

3DSTATE_BINDING_TABLE_EDIT_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command edits the binding table for PS.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	47h 3DSTATE_BINDING_TABLE_EDIT_PS	
	Format:	OpCode	
15:9	Reserved		
	Format:	MBZ	
8:0	DWord Length		
	Format:	=n	
	Value	Name	
	0h	DWORD_COUNT_n [Default]	
	0h - 100h	Range	
1	31:16	Binding Table Block Clear	
		Format:	U16
	<p>Each bit in this field corresponds to a 16 entry block of the binding table. Bit 0 of this field corresponds to entries 0-15, bit 1 to 16-31, and so on. When a bit is set it clears the corresponding bind table entries to 0. (effectively disabling them). The clear is applied before the individual binding table entries contained in this message are applied. When this bit is clear then the corresponding 16 entry block is not cleared.</p>		
15:2	Reserved		
	Format:	MBZ	



3DSTATE_BINDING_TABLE_EDIT_PS																	
	1:0	Binding Table Edit Target Specifies which core should respond to this 3DSTATE_BINDING_TABLE_EDIT_PS command:															
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>11b</td> <td>All Cores</td> <td>All cores should respond to this command</td> </tr> <tr> <td>10b</td> <td>Core 1</td> <td>Only Core1 should respond to this command</td> </tr> <tr> <td>01b</td> <td>Core 0</td> <td>Only Core0 should respond to this command</td> </tr> <tr> <td>00b</td> <td>Reserved</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	11b	All Cores	All cores should respond to this command	10b	Core 1	Only Core1 should respond to this command	01b	Core 0	Only Core0 should respond to this command	00b	Reserved	Reserved
		Value	Name	Description													
		11b	All Cores	All cores should respond to this command													
		10b	Core 1	Only Core1 should respond to this command													
01b	Core 0	Only Core0 should respond to this command															
00b	Reserved	Reserved															
2..n	31:0	Entry [n] Format: BINDING_TABLE_EDIT_ENTRY															

3DSTATE_BINDING_TABLE_EDIT_VS

3DSTATE_BINDING_TABLE_EDIT_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command edits the binding table for VS. The 3DSTATE_BINDING_TABLE_EDIT_VS is a variable length command.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	43h 3DSTATE_BINDING_TABLE_EDIT_VS	
	Format:	OpCode	
15:9	Reserved		
	Format:	MBZ	
8:0	DWord Length		
	Format:	=n	
	Value	Name	
	0h	DWORD_COUNT_n [Default]	
	0h - 100h	Range	
1	31:16	Binding Table Block Clear	
		Format:	U16
	Each bit in this field corresponds to a 16 entry block of the binding table. Bit 0 of this field corresponds to entries 0-15, bit 1 to 16-31, and so on. When a bit is set it clears the corresponding bind table entries to 0. (affectively disabling them). The clear is applied before the individual binding table entries contained in this message are applied. When this bit is clear then the corresponding 16 entry block is not cleared.		
15:2	Reserved		
	Format:	MBZ	



3DSTATE_BINDING_TABLE_EDIT_VS																	
	1:0	<p>Binding Table Edit Target Specifies which core should respond to this 3DSTATE_BINDING_TABLE_EDIT_VS command:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">11b</td> <td style="text-align: center;">All Cores</td> <td>All cores should respond to this command</td> </tr> <tr> <td style="text-align: center;">10b</td> <td style="text-align: center;">Core 1</td> <td>Only Core1 should respond to this command</td> </tr> <tr> <td style="text-align: center;">01b</td> <td style="text-align: center;">Core 0</td> <td>Only Core0 should respond to this command</td> </tr> <tr> <td style="text-align: center;">00b</td> <td style="text-align: center;">Reserved</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	11b	All Cores	All cores should respond to this command	10b	Core 1	Only Core1 should respond to this command	01b	Core 0	Only Core0 should respond to this command	00b	Reserved	Reserved
		Value	Name	Description													
		11b	All Cores	All cores should respond to this command													
		10b	Core 1	Only Core1 should respond to this command													
		01b	Core 0	Only Core0 should respond to this command													
00b	Reserved	Reserved															
2..n	31:0	<p>Entry [n]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>BINDING_TABLE_EDIT_ENTRY</td> </tr> </table>	Format:	BINDING_TABLE_EDIT_ENTRY													
Format:	BINDING_TABLE_EDIT_ENTRY																

3DSTATE_BINDING_TABLE_POINTERS_DS

3DSTATE_BINDING_TABLE_POINTERS_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BINDING_TABLE_POINTERS_DS command is used to define the location of fixed functions' BINDING_TABLE_STATE. Only some of the fixed functions utilize binding tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		28h 3DSTATE_BINDING_TABLE_POINTERS_DS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_BINDING_TABLE_POINTERS_DS							
15:5	<p>Pointer to DS Binding Table</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled</td> </tr> </table> <p>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: If HW Binding Table is disabled, the offset is relative to Surface State Base Address and the alignment is 32B. If HW Binding Table is enabled the offset is relative to the Binding Table Pool Base Address and the alignment is 64B.</p>	Project:	BDW	Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled	Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled
Project:	BDW						
Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled						
Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled						
4:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						

3DSTATE_BINDING_TABLE_POINTERS_GS

3DSTATE_BINDING_TABLE_POINTERS_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BINDING_TABLE_POINTERS_GS command is used to define the location of fixed functions' BINDING_TABLE_STATE. Only some of the fixed functions utilize binding tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		29h 3DSTATE_BINDING_TABLE_POINTERS_GS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_BINDING_TABLE_POINTERS_GS							
15:5	<p>Pointer to GS Binding Table</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled</td> </tr> </table> <p>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: If HW Binding Table is disabled, the offset is relative to Surface State Base Address and the alignment is 32B. If HW Binding Table is enabled the offset is relative to the Binding Table Pool Base Address and the alignment is 64B.</p>	Project:	BDW	Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled	Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled
Project:	BDW						
Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled						
Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled						
4:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						

3DSTATE_BINDING_TABLE_POINTERS_HS

3DSTATE_BINDING_TABLE_POINTERS_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BINDING_TABLE_POINTERS_HS command is used to define the location of fixed functions' BINDING_TABLE_STATE. Only some of the fixed functions utilize binding tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		27h 3DSTATE_BINDING_TABLE_POINTERS_HS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_BINDING_TABLE_POINTERS_HS							
15:5	<p>Pointer to HS Binding Table</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled</td> </tr> </table> <p>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: If HW Binding Table is disabled, the offset is relative to Surface State Base Address and the alignment is 32B. If HW Binding Table is enabled the offset is relative to the Binding Table Pool Base Address and the alignment is 64B.</p>	Project:	BDW	Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled	Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled
Project:	BDW						
Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled						
Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled						
4:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						

3DSTATE_BINDING_TABLE_POINTERS_PS

3DSTATE_BINDING_TABLE_POINTERS_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BINDING_TABLE_POINTERS_PS command is used to define the location of fixed functions' BINDING_TABLE_STATE. Only some of the fixed functions utilize binding tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		2Ah 3DSTATE_BINDING_TABLE_POINTERS_PS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_BINDING_TABLE_POINTERS_PS		
15:5	Pointer to PS Binding Table	
	Project:	BDW
	Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled
	Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled
<p>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: If HW Binding Table is disabled, the offset is relative to Surface State Base Address and the alignment is 32B. If HW Binding Table is enabled the offset is relative to the Binding Table Pool Base Address and the alignment is 64B.</p>		
4:0	Reserved	
	Project:	All
	Format:	MBZ

3DSTATE_BINDING_TABLE_POINTERS_VS

3DSTATE_BINDING_TABLE_POINTERS_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BINDING_TABLE_POINTERS_VS command is used to define the location of fixed functions' BINDING_TABLE_STATE. Only some of the fixed functions utilize binding tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		26h 3DSTATE_BINDING_TABLE_POINTERS_VS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_BINDING_TABLE_POINTERS_VS							
15:5	<p>Pointer to VS Binding Table</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled</td> </tr> <tr> <td>Format:</td> <td>SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled</td> </tr> </table> <p>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: If HW Binding Table is disabled, the offset is relative to Surface State Base Address and the alignment is 32B. If HW Binding Table is enabled the offset is relative to the Binding Table Pool Base Address and the alignment is 64B.</p>	Project:	BDW	Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled	Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled
Project:	BDW						
Format:	SurfaceStateOffset[15:5]BINDING_TABLE_STATE*256 When HW binding table is disabled						
Format:	SurfaceStateOffset[16:6]BINDING_TABLE_STATE*256 When HW-generated binding table is enabled						
4:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						

3DSTATE_BINDING_TABLE_POOL_ALLOC

DWord		Bit	Description
<p>3DSTATE_BINDING_TABLE_POOL_ALLOC</p> <p>Project: BDW Source: RenderCS Length Bias: 2</p> <p>This command sets up the binding table pool for HW generated binding tables.</p>			
Programming Notes			
<p>When RS is enabled due to a MI_RS_CONTROL or MI_BATCH_BUFFER_START with RS enable bit set, driver must reprogram the 3DSTATE_BINDING_TABLE_POOL_ALLOC to ensure the resource streamer and render engine are in sync with the programming with the command. Otherwise there could be cases where RS sees that the Binding Table Pool is disabled while the render pipeline sees the binding table is enabled in the case the 3DSTATE_BINDING_TABLE_POOL_ALLOC was enabled while RS was off.</p>			
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	19h 3DSTATE_BINDING_TABLE_POOL_ALLOC
		Format:	OpCode
	15:8	Reserved	
		Project:	All
Format:		MBZ	
7:0	DWord Length		
	Project:	All	
	Format:	=n	
	Value	Name	
	2h	DWORD_COUNT_n [Default]	

3DSTATE_BINDING_TABLE_POOL_ALLOC																					
1.2 Project: BDW	63:12	Binding Table Pool Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:12]BindingTablePool</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Project:	BDW	Format:	GraphicsAddress[63:12]BindingTablePool															
	Project:	BDW																			
	Format:	GraphicsAddress[63:12]BindingTablePool																			
	11	Binding Table Pool Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>When this bit is set it enables HW generation of binding tables. When this bit is cleared it disables HW generation of binding tables.</p>	Project:	BDW	Format:	U1															
	Project:	BDW																			
Format:	U1																				
10	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW																		
Project:	BDW																				
9:7	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW																		
Project:	BDW																				
6:0	Surface Object Control State <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state for this surface.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th colspan="3" style="text-align: center;">Programming Notes</th> </tr> <tr> <td colspan="3">Bit 2 is not programmable and is always zero.</td> </tr> </table>	Project:	BDW	Format:	MEMORY_OBJECT_CONTROL_STATE	Programming Notes			Bit 2 is not programmable and is always zero.												
Project:	BDW																				
Format:	MEMORY_OBJECT_CONTROL_STATE																				
Programming Notes																					
Bit 2 is not programmable and is always zero.																					
3 Project: BDW	31:12	Binding Table Pool Buffer Size <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U20</td> </tr> </table> <p>This field specifies the size of the buffer in 4K pages. Any access which straddle or go past the end of the buffer will return 0.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,1048575]</td> <td></td> <td></td> </tr> <tr> <td>0</td> <td>No Valid Data</td> <td>There is no valid data in the buffer</td> </tr> </tbody> </table> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th colspan="3" style="text-align: center;">Restriction</th> </tr> <tr> <td colspan="3">Programming size of zero is illegal in the case that the pool is enabled.</td> </tr> </table>	Project:	BDW	Format:	U20	Value	Name	Description	[0,1048575]			0	No Valid Data	There is no valid data in the buffer	Restriction			Programming size of zero is illegal in the case that the pool is enabled.		
	Project:	BDW																			
	Format:	U20																			
	Value	Name	Description																		
[0,1048575]																					
0	No Valid Data	There is no valid data in the buffer																			
Restriction																					
Programming size of zero is illegal in the case that the pool is enabled.																					
11	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																
Project:	BDW																				
Format:	MBZ																				
10:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																
Project:	BDW																				
Format:	MBZ																				

3DSTATE_BLEND_STATE_POINTERS

3DSTATE_BLEND_STATE_POINTERS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_BLEND_STATE_POINTERS command is used to set up the pointers to the color calculator state.			
Programming Notes			
When the BLEND_STATE pointer changes but not the CC_STATE pointer, driver needs to force a CC_STATE pointer change to improve blend performance in pixel backend.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		24h 3DSTATE_BLEND_STATE_POINTERS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:6	Blend State Pointer	
		Project:	All
		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 .			



3DSTATE_BLEND_STATE_POINTERS					
	5:1	Reserved			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:
	Project:	All			
	Format:	MBZ			
0	Blend State Pointer Valid				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table>	Project:	BDW	Format:	Enable
	Project:	BDW			
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

3DSTATE_CC_STATE_POINTERS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_CC_STATE_POINTERS command is used to set up the pointers to the color calculator state.			
Programming Notes			
When the CC_STATE pointer changes but not the BLEND_STATE pointer, driver needs to force a BLEND_STATE pointer change in order to improve blend performance in the pixel backend.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	0Eh 3DSTATE_CC_STATE_POINTERS	
	Format:	OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Project:	All	
	Format:	=n	
1	31:6	Color Calc State Pointer	
		Project:	All
		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 .	



3DSTATE_CC_STATE_POINTERS		
	5:1	Reserved
		Project: All
		Format: MBZ
	0	Color Calc State Pointer Valid
		Project: BDW
		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 possibility of the CC state changing between context switches.

3DSTATE_CHROMA_KEY

3DSTATE_CHROMA_KEY			
Project:	DevBWR+		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_CHROMA_KEY instruction is used to program texture color/chroma-key key values. A table containing four set of values is supported. The ChromaKey Index sampler state variable is used to select which table entry is associated with the map. Texture chromakey functions are enabled and controlled via use of the ChromaKey Enable texture sampler state variable. Texture Color Key (keying on a paletted texture index) is not supported.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 3h GFXPIPE Format: Opcode	
	28:27	Command SubType	
		Default Value: 3h GFXPIPE_3D Format: Opcode	
	26:24	3D Command Opcode	
		Default Value: 1h 3DSTATE_NONPIPELINED Format: Opcode	
	23:16	3D Command Sub Opcode	
Default Value: 04h 3DSTATE_CHROMA_KEY Format: Opcode			
15:8	Reserved		
	Project: All Format: MBZ		
7:0	DWord Length		
	Default Value: 2h Excludes DWord (0,1) Format: =n Total Length - 2		
1	31:30	ChromaKey Table Index	
		Project: All Format: U2 index	
		Selects which entry in the ChromaKey table is to be loaded	
		Value	Name
	[0,3]		



3DSTATE_CHROMA_KEY																
29:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ											
Project:	All															
Format:	MBZ															
2	<p>31:0 ChromaKey Low Value This field specifies the "low" (minimum) value of the chroma key range. Texel samples are considered "matching the key" if each component of the texel falls within the (inclusive) chroma range. See ChromaKey High Value for further format, programming info.</p>															
3	<p>31:0 ChromaKey High Value This field specifies the "high" (maximum) value of the chroma key range. Texel samples are considered "matching the key" if each component of the texel falls within the (inclusive) chroma range.</p> <p style="text-align: center;">Programming Notes</p> <p>ChromaKey values are specified using 8-bit channels. When using surface formats with less than 8 bits per channel, the device will expand channels by replicating the required number of MSBs into the LSBs of each channel. Software must account for this conversion when it programs Chromakey Low/High Values (e.g., by performing the same replication).</p> <p>For channels that do not exist in the actual surface (e.g., Alpha channel for non-ARGB maps), software must explicitly program full range high/low values (High=FFh, Low=0h for formats using unsigned chroma key values, High=7Fh, Low=FFh for formats using sign magnitude chroma key values) in order to effectively remove the comparison of that field from the ChromaKey function.</p> <p>For channels in SNORM format in the surface format, the value in the high/low value for that channel is interpreted in sign magnitude format. Negative zero value is not supported (use positive zero instead). For channels with mixed UNORM/SNORM formats (i.e. R5G5_SNORM_B6_UNORM), the ChromaKey is programmed as if all channels are SNORM.</p> <p>YUV ChromaKey will use an interpolated chrominance value from the map for comparison to the chroma key values for those texels without chrominance due to downsampling. The chrominance value used is the average of values to the left and right of the texel in question.</p> <p>It is UNDEFINED to program any component of the ChromaKey High Value to be less than the corresponding component of ChromaKey Low Value.</p> <p>Format = interpreted according to associated texel format "class":</p> <p>Only the surface formats listed as supported for chroma key in the surface formats table can be used with this feature. Use of any other surface format with chroma key enabled is UNDEFINED.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="text-align: left;">Surface Format</th> <th>31:24</th> <th>23:15</th> <th>16:8</th> <th>7:0</th> </tr> </thead> <tbody> <tr> <td>ARGB and BC (DXT) formats</td> <td>A</td> <td>R</td> <td>G</td> <td>B</td> </tr> <tr> <td>YCrCb formats</td> <td>A</td> <td>Cr</td> <td>Y</td> <td>Cb</td> </tr> </tbody> </table>	Surface Format	31:24	23:15	16:8	7:0	ARGB and BC (DXT) formats	A	R	G	B	YCrCb formats	A	Cr	Y	Cb
Surface Format	31:24	23:15	16:8	7:0												
ARGB and BC (DXT) formats	A	R	G	B												
YCrCb formats	A	Cr	Y	Cb												

3DSTATE_CLEAR_PARAMS

3DSTATE_CLEAR_PARAMS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
Description		
<p>This command defines the depth clear value delivered as a pipelined state command. However, the state change pipelining isn't completely transparent (see restriction below).</p> <p>HW will internally manage the draining pipe and flushing of the caches when this command is issued. The PIPE_CONTROL restrictions are removed.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED Format: OpCode
	23:16	3D Command Sub Opcode
Default Value: 04h 3DSTATE_CLEAR_PARAMS Format: OpCode		
15:8	Reserved	
	Format: MBZ	
7:0	Dword Length	
	Default Value: 1h Excludes Dword (0,1) Format: =n Total Length - 2	
1	31:0	Depth Clear Value
		Project: BDW Format: IEEE_Float
	<p>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.</p>	
	Programming Notes	
<p>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.</p>		



3DSTATE_CLEAR_PARAMS				
2	31:1	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
0	<p>Depth Clear Value Valid</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Boolean</td> </tr> </table> <p>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.</p>	Format:	Boolean	
Format:	Boolean			

3DSTATE_CLIP

3DSTATE_CLIP			
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		12h 3DSTATE_CLIP	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	02h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31:21	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_CLIP			
20	Force User Clip Distance Cull Test Enable Bitmask		
	Project:	All	
	Format:	Enable	
	This field provides a work around override for the computation of SOL_INT::Render_Enable		
	Value	Name	Description
	0h	Normal	Clip_INT::User Clip Distance Cull Test Enable Bitmask normally
	1h	Force	Forces Clip_INT::User Clip Distance Cull Test Enable Bitmask to use the value in 3DSTATE_CLIP:: User Clip Distance Cull Test Enable Bitmask
	Vertex Sub Pixel Precision Select		
	Project:	All	
	Format:	U1	
Selects the number of fractional bits maintained in the vertex data			
Value	Name	Description	
0h	8 Bit	8 sub pixel precision bits maintained	
1h	4 Bit	4 sub pixel precision bits maintained	
18	Early Cull Enable		
	Project:	All	
	Format:	Enable	
This field is used to enable/disable the EarlyCull function.			
17	Force User Clip Distance Clip Test Enable Bitmask		
	Project:	All	
	Format:	Enable	
	This field provides a work around override for the computation of SOL_INT::Render_Enable.		
	Value	Name	Description
	0b	Normal	Clip_INT:: User Clip Distance Clip Test Enable Bitmask normally
1b	Force	Forces Clip_INT:: User Clip Distance Clip Test Enable Bitmask to use the value in 3DSTATE_CLIP::User Clip Distance Clip Test Enable Bitmask	
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.	
15:11	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_CLIP			
10	Clipper Statistics Enable		
	Project:	All	
	Format:	Enable	
	This bit controls whether Clip-unit-specific statistics register(s) can be incremented.		
	Value	Name	Description
	0h	Disable	CL_INVOCATIONS_COUNT cannot increment
	1h	Enable	CL_INVOCATIONS_COUNT can increment
	Reserved		
	Project:	BDW	
	Format:	MBZ	
7:0	User Clip Distance Cull Test Enable Bitmask		
	Project:	All	
	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.		
2	31	Clip Enable	
		Project:	All
		Format:	Enable
	Specifies whether the Clip function is enabled or disabled (pass-through).		
	30	API Mode	
		Project:	All
		Controls the definition of the NEAR clipping plane	
	Value	Name	Description
	0h	OGL	NEAR VP boundary == 0.0 (NDC)
	29	Reserved	
		Project:	All
		Format:	MBZ
	28	Viewport XY Clip Test Enable	
		Project:	All
		Format:	Enable
This field is used to control whether the Viewport X, Y extents are considered in VertexClipTest. See Tristrip Clipping subsection.			
27	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_CLIP		
26	Guardband Clip Test Enable	
	Project:	All
	Format:	Enable
<p>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.</p>		
25:24	Reserved	
	Project:	All
	Format:	MBZ
23:16	User Clip Distance Clip Test Enable Bitmask	
	Project:	All
	Format:	Enable[8]
<p>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.</p>		
15:13	Clip Mode	
	Project:	All
<p>This field specifies a general mode of the CLIP unit, when the CLIP unit is ENABLED.</p>		
	Value	Name Description
	0h	NORMAL TrivialAccept objects are passed down the pipeline, MustClip objects Clipped in the Fixed Function Clipper HW, TrivialReject and BAD objects are discarded
	1h	Reserved
	2h	Reserved
	3h	REJECT_ALL All objects are discarded
	4h	ACCEPT_ALL 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.
	5h-7h	Reserved
12:10	Reserved	
	Project:	All
	Format:	MBZ

3DSTATE_CLIP															
9	<p>Perspective Divide Disable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Disable</td> </tr> </table> <p>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).</p>	Project:	All	Format:	Disable										
Project:	All														
Format:	Disable														
8	<p>Non-Perspective Barycentric Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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 barycentric parameters are enabled in the Windower.</p>	Project:	All	Format:	Enable										
Project:	All														
Format:	Enable														
7:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ										
Project:	All														
Format:	MBZ														
5:4	<p>Triangle Strip/List Provoking Vertex Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>enumerated type</p> <p>This field selects which vertex of a triangle (in a triangle strip or list primitive) is considered the "provoking vertex".</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>0</td> </tr> <tr> <td>1h</td> <td>1</td> </tr> <tr> <td>2h</td> <td>2</td> </tr> <tr> <td>3h</td> <td>Reserved</td> </tr> </tbody> </table>	Project:	All	Format:	U2	Value	Name	0h	0	1h	1	2h	2	3h	Reserved
Project:	All														
Format:	U2														
Value	Name														
0h	0														
1h	1														
2h	2														
3h	Reserved														

3DSTATE_CLIP																
3	3:2	<p>Line Strip/List Provoking Vertex Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>enumerated type</p> <p>This field selects which vertex of a line (in a line strip or list primitive) is considered the "provoking vertex".</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>0</td> </tr> <tr> <td>1h</td> <td>1</td> </tr> <tr> <td>2h</td> <td>Reserved</td> </tr> <tr> <td>3h</td> <td>Reserved</td> </tr> </tbody> </table>	Project:	All	Format:	U2	Value	Name	0h	0	1h	1	2h	Reserved	3h	Reserved
	Project:	All														
	Format:	U2														
Value	Name															
0h	0															
1h	1															
2h	Reserved															
3h	Reserved															
1:0	<p>Triangle Fan Provoking Vertex Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>enumerated type</p> <p>This field selects which vertex of a triangle (in a triangle fan primitive) is considered the "provoking vertex".</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>0</td> </tr> <tr> <td>1h</td> <td>1</td> </tr> <tr> <td>2h</td> <td>2</td> </tr> <tr> <td>3h</td> <td>Reserved</td> </tr> </tbody> </table>	Project:	All	Format:	U2	Value	Name	0h	0	1h	1	2h	2	3h	Reserved	
Project:	All															
Format:	U2															
Value	Name															
0h	0															
1h	1															
2h	2															
3h	Reserved															
31:28	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ											
Project:	All															
Format:	MBZ															
27:17	<p>Minimum Point Width</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8.3 pixels</td> </tr> </table> <p>This value is used to clamp read-back PointWidth values.</p>	Project:	All	Format:	U8.3 pixels											
Project:	All															
Format:	U8.3 pixels															
16:6	<p>Maximum Point Width</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8.3 pixels</td> </tr> </table> <p>This value is used to clamp read-back PointWidth values.</p>	Project:	All	Format:	U8.3 pixels											
Project:	All															
Format:	U8.3 pixels															

3DSTATE_CLIP			
5	Force Zero RTA Index Enable		
	Project:	All	
	Format:	Enable	
	<p>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.</p>		
	4	Reserved	
		Project:	All
		Format:	MBZ
	3:0	Maximum VP Index	
		Project:	All
		Format:	U4-1 index value (# of viewports)
<p>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.</p>			



3DSTATE_CONSTANT_DS

3DSTATE_CONSTANT_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets pointers to the push constants for the DS unit. The constant data pointed to by this command is loaded into the DS unit's push constant buffer (PCB).			
Programming Notes			
[BDW] A 3DSTATE_GATHER_DS command must be dispatched along with any 3DSTATE_CONSTANT_DS command when Gather Pool is enabled within a batch buffer.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	1Ah 3DSTATE_CONSTANT_DS
		Format:	OpCode
15	Reserved		
	Project:	BDW	
	Format:	MBZ	
14:8	Constant Buffer Object Control State		
	Project:	BDW	
	Format:	MEMORY_OBJECT_CONTROL_STATE	
	Specifies the memory object control state for all constant buffers defined in this command.		
	Programming Notes		
	Constant Buffer Object Control State must be always programmed to zero.		

3DSTATE_CONSTANT_DS								
	7:0	DWord Length						
		Project: All						
		Format: =n Total Length - 2						
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">9h</td> <td>Excludes DWord (0,1) [Default]</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Value	Name	Project	9h	Excludes DWord (0,1) [Default]	BDW
		Value	Name	Project				
9h	Excludes DWord (0,1) [Default]	BDW						
1..10 Project: BDW	319:0	Constant Body						
		Project: BDW						
		Format: 3DSTATE_CONSTANT(Body)						
		See the 3DSTATE_CONSTANT(Body) format for the shared portion of the 3DSTATE_CONSTANT command for VS, HS, DS, and GS.						



3DSTATE_CONSTANT_GS

3DSTATE_CONSTANT_GS				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
<p>This command sets pointers to the push constants for the GS unit. The constant data pointed to by this command will be loaded into the GS unit's push constant buffer (PCB).</p>				
Programming Notes				
<p>[BDW]: A 3DSTATE_GATHER_GS command must be dispatched along with any 3DSTATE_CONSTANT_GS command when the Gather Pool is enabled within a batch buffer.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h GFXPIPE	
		Format:	OpCode	
	28:27	28:27	Command SubType	
			Default Value:	3h GFXPIPE_3D
			Format:	OpCode
	26:24	26:24	3D Command Opcode	
			Default Value:	0h 3DSTATE_PIPELINED
			Format:	OpCode
	23:16	23:16	3D Command Sub Opcode	
			Default Value:	16h 3DSTATE_CONSTANT_GS
			Format:	OpCode
15	15	Reserved		
		Project:	All	
		Format:	MBZ	
14:8	14:8	Constant Buffer Object Control State		
		Project:	BDW	
		Format:	MEMORY_OBJECT_CONTROL_STATE	
		<p>Specifies the memory object control state for all constant buffers defined in this command.</p>		
		Programming Notes		
<p>Constant Buffer Object Control State must be always programmed to zero.</p>				

3DSTATE_CONSTANT_GS										
	7:0	<p>DWord Length</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>9h</td> <td>Excludes DWord (0,1) [Default]</td> </tr> </tbody> </table>	Project:	All	Format:	=n Total Length - 2	Value	Name	9h	Excludes DWord (0,1) [Default]
Project:	All									
Format:	=n Total Length - 2									
Value	Name									
9h	Excludes DWord (0,1) [Default]									
1..10 Project: BDW	319:0	<p>Constant Body</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>3DSTATE_CONSTANT(Body)</td> </tr> </table> <p>Following table is the shared portion of the 3DSTATE_CONSTANT command for VS, HS, DS, and GS</p>	Project:	BDW	Format:	3DSTATE_CONSTANT(Body)				
Project:	BDW									
Format:	3DSTATE_CONSTANT(Body)									



3DSTATE_CONSTANT_HS

3DSTATE_CONSTANT_HS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command sets pointers to the push constants for the HS unit. The constant data pointed to by this command is loaded into the HS unit's push constant buffer (PCB).</p>		
Programming Notes		
<p>A 3DSTATE_GATHER_HS command must be dispatched along with any 3DSTATE_CONSTANT_HS command when Gather Pool is enabled within a batch buffer.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
		Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D
		Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED
		Format: OpCode
	23:16	3D Command Sub Opcode
		Default Value: 19h 3DSTATE_CONSTANT_HS
		Format: OpCode
15	Reserved	
	Project: BDW	
	Format: MBZ	
14:8	Constant Buffer Object Control State	
	Project: BDW	
	Format: MEMORY_OBJECT_CONTROL_STATE	
	Specifies the memory object control state for all constant buffers defined in this command.	
	Programming Notes	
	Constant Buffer Object Control State must be always programmed to zero.	

3DSTATE_CONSTANT_HS			
	7:0	DWord Length	
		Project: All	
		Format: =n Total Length - 2	
		Value	Name
		9h	Excludes DWord (0,1) [Default]
1..10 Project: BDW	319:0	Constant Body	
		Project: BDW	
		Format: 3DSTATE_CONSTANT(Body)	
		Following table is the shared portion of the 3DSTATE_CONSTANT command for VS, HS, DS, and GS	



3DSTATE_CONSTANT_PS

DWord		Bit	Description
3DSTATE_CONSTANT_PS			
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
<p>This command sets pointers to the push constants for the PS unit. The constant data pointed to by this command is loaded into the PS unit's push constant buffer (PCB).</p>			
Programming Notes			
<p>[BDW]: A 3DSTATE_GATHER_PS command must be dispatched along with any 3DSTATE_CONSTANT_PS command when the Gather Pool is enabled within a batch buffer.</p>			
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
	Format:		OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	Format:		OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
	Format:		OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	17h 3DSTATE_CONSTANT_PS
	Format:		OpCode
15	Reserved		
	Project:	BDW	
Format:		MBZ	
14:8	Constant Buffer Object Control State		
	Project:	BDW	
	Format:	MEMORY_OBJECT_CONTROL_STATE	
	Specifies the memory object control state for all constant buffers defined in this command.		
Programming Notes			
Constant Buffer Object Control State must be always programmed to zero.			

3DSTATE_CONSTANT_PS			
	7:0	Dword Length	
		Project: All	
		Format: =n Total Length - 2	
		Value	Name
		9h	Excludes DWord (0,1) [Default]
1..10 Project: BDW	319:0	Constant Body	
		Project: BDW	
		Format: 3DSTATE_CONSTANT(Body)	
		Following table is the shared portion of the 3DSTATE_CONSTANT command for VS, HS, DS, and GS	



3DSTATE_CONSTANT_VS

DWord		Bit	Description
3DSTATE_CONSTANT_VS			
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
<p>This command sets pointers to the push constants for VS unit. The constant data pointed to by this command is loaded into the VS unit's push constant buffer (PCB).</p>			
Programming Notes			
<p>[BDW] A 3DSTATE_GATHER_VS command must be dispatched along with any 3DSTATE_CONSTANT_VS command when Gather Pool is enabled within a batch buffer.</p>			
0	31:29	Command Type Default Value: 3h GFXPIPE Format: OpCode	
	28:27	Command SubType Default Value: 3h GFXPIPE_3D Format: OpCode	
	26:24	3D Command Opcode Default Value: 0h 3DSTATE_PIPELINED Format: OpCode	
	23:16	3D Command Sub Opcode Default Value: 15h 3DSTATE_CONSTANT_VS Format: OpCode	
	15	Reserved Project: BDW Format: MBZ	
	14:8	Constant Buffer Object Control State Project: BDW Format: MEMORY_OBJECT_CONTROL_STATE Specifies the memory object control state for all constant buffers defined in this command.	
Programming Notes			
Constant Buffer Object Control State must be always programmed to zero.			

3DSTATE_CONSTANT_VS			
	7:0	DWord Length	
		Project: All	
		Format: =n Total Length - 2	
		Value	Name
		9h	Excludes DWord (0,1) [Default]
1..10 Project: BDW	319:0	Constant Body	
		Project: BDW	
		Format: 3DSTATE_CONSTANT(Body)	
		Following table is the shared portion of the 3DSTATE_CONSTANT command for VS, HS, DS, and GS	



3DSTATE_DEPTH_BUFFER

3DSTATE_DEPTH_BUFFER		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>The depth buffer surface state is delivered as a pipelined state packet. However, the state change pipelining isn't completely transparent (see restriction below).</p>		
<p>WM HW will internally manage the draining pipe and flushing of the caches when this commands is issued. The PIPE_CONTROL restrictions are removed.</p>		
Programming Notes		
<p>Note for validation teams. If the depth surface is backdoor initialized or written to directly by the CPU, the values placed in the Depth Surface must be within the valid numeric range permitted by the Viewport Min and Max documentation, which may change per API. Currently this is the numeric range of [0.0 ... 1.0] for DirectX and may in the future include +/- max floating point values; but not +/-Inf or any NaN code.</p>		
DWord	Bit	Description
0	31:2	Command Type
	9	Default Value: 3h GFXPIPE
		Format: OpCode
	28:2	Command SubType
	7	Default Value: 3h GFXPIPE_3D
		Format: OpCode
	26:2	3D Command Opcode
	4	Default Value: 0h 3DSTATE_PIPELINED
		Format: OpCode
	23:1	3D Command Sub Opcode
	6	Default Value: 5h 3DSTATE_DEPTH_BUFFER
		Format: OpCode
	15:8	Reserved
		Format: MBZ
7:0	DWord Length	
	Default Value: 6h Excludes Dword (0,1)	
	Format: =n	
	Excludes DWord(0,1)	

3DSTATE_DEPTH_BUFFER																									
1	31:2 9	<p>Surface Type</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>SURFTYPE_1D</td> <td>Defines a 1-dimensional map or array of maps</td> </tr> <tr> <td style="text-align: center;">1h</td> <td>SURFTYPE_2D</td> <td>Defines a 2-dimensional map or array of maps</td> </tr> <tr> <td style="text-align: center;">2h</td> <td>SURFTYPE_3D</td> <td>Defines a 3-dimensional (volumetric) map</td> </tr> <tr> <td style="text-align: center;">3h</td> <td>SURFTYPE_CUBE</td> <td>Defines a cube map</td> </tr> <tr> <td style="text-align: center;">4h-6h</td> <td>Reserved</td> <td></td> </tr> <tr> <td style="text-align: center;">7h</td> <td>SURFTYPE_NULL</td> <td>Defines a null surface</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; color: #0070C0;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">The Surface Type of the depth buffer must be the same as the Surface Type of the render target(s) (defined in SURFACE_STATE), unless either the depth buffer or render targets are SURFTYPE_NULL.</td> </tr> </tbody> </table>	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	4h-6h	Reserved		7h	SURFTYPE_NULL	Defines a null surface	Programming Notes	The Surface Type of the depth buffer must be the same as the Surface Type of the render target(s) (defined in SURFACE_STATE), unless either the depth buffer or render targets are SURFTYPE_NULL.
	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																						
	4h-6h	Reserved																							
	7h	SURFTYPE_NULL	Defines a null surface																						
	Programming Notes																								
	The Surface Type of the depth buffer must be the same as the Surface Type of the render target(s) (defined in SURFACE_STATE), unless either the depth buffer or render targets are SURFTYPE_NULL.																								
28	<p>Depth Write Enable</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>This field enables depth writes to the depth buffer surface. Both this field and the Depth Buffer Write Enable field in DEPTH_STENCIL_STATE must be enabled in order for depth writes to occur.</p>	Format:	Enable																						
Format:	Enable																								
27	<p>Stencil Write Enable</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>This field enables stencil writes to the depth buffer or stencil buffer surface, depending on where stencil is located. Both this field and the Stencil Buffer Write Enable field in DEPTH_STENCIL_STATE must be enabled in order for stencil writes to occur.</p>	Format:	Enable																						
Format:	Enable																								
26:2 4	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																						
Format:	MBZ																								
23	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>This field, when ENABLED, indicates when a surface is using corner texel-mode for depth surface. This bit changes how the size of each MIP when calculating the offset within a surface.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>Disable [Default]</td> <td>Corner Texel mode is not enabled.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td>Enable</td> <td>Corner Texel Mode is enabled.</td> </tr> </tbody> </table>	Project:	BDW	Format:	MBZ	Value	Name	Description	0h	Disable [Default]	Corner Texel mode is not enabled.	1h	Enable	Corner Texel Mode is enabled.											
Project:	BDW																								
Format:	MBZ																								
Value	Name	Description																							
0h	Disable [Default]	Corner Texel mode is not enabled.																							
1h	Enable	Corner Texel Mode is enabled.																							

3DSTATE_DEPTH_BUFFER																	
	22	Hierarchical Depth Buffer Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>If enabled, indicates that a hierarchical depth buffer is defined.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>If this field is enabled, the Software Tiled Rendering Mode must be NORMAL. This field must be disabled if Early Depth Test Enable is disabled OR if depth buffer surface type is NULL.</p>	Format:	Enable	Programming Notes												
	Format:	Enable															
	Programming Notes																
	21	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ													
Format:	MBZ																
20:1 8	Surface Format Specifies the format of the depth buffer. See Stencil Test Enable field in DEPTH_STENCIL_STATE field for restrictions on the use of some of these formats. <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr style="background-color: #e6f2ff;"> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <tr><td>0h</td><td>Reserved</td></tr> <tr><td>1h</td><td>D32_FLOAT</td></tr> <tr><td>2h</td><td>Reserved</td></tr> <tr><td>3h</td><td>D24_UNORM_X8_UINT</td></tr> <tr><td>4h</td><td>Reserved</td></tr> <tr><td>5h</td><td>D16_UNORM</td></tr> <tr><td>6h-7h</td><td>Reserved</td></tr> </tbody> </table>	Value	Name	0h	Reserved	1h	D32_FLOAT	2h	Reserved	3h	D24_UNORM_X8_UINT	4h	Reserved	5h	D16_UNORM	6h-7h	Reserved
Value	Name																
0h	Reserved																
1h	D32_FLOAT																
2h	Reserved																
3h	D24_UNORM_X8_UINT																
4h	Reserved																
5h	D16_UNORM																
6h-7h	Reserved																
17:0	Surface Pitch <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U18-1 Pitch in (Bytes-1)</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr style="background-color: #e6f2ff;"> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>[7Fh,3FFFFh]</td> <td></td> <td>corresponding to [128B, 256KB] also restricted to a multiple of 128B</td> </tr> </tbody> </table> <table border="1" style="width: 100%; background-color: #e6f2ff; margin-top: 10px;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>The pitch specified must be a multiple of the tile pitch, in the range [128B, 128KB].</p>	Format:	U18-1 Pitch in (Bytes-1)	Value	Name	Description	[7Fh,3FFFFh]		corresponding to [128B, 256KB] also restricted to a multiple of 128B	Programming Notes							
Format:	U18-1 Pitch in (Bytes-1)																
Value	Name	Description															
[7Fh,3FFFFh]		corresponding to [128B, 256KB] also restricted to a multiple of 128B															
Programming Notes																	
2..3 Project: BDW	63:0 Surface Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:0]DepthBuffer</td> </tr> </table> <p>This field specifies address of the buffer in mapped Graphics Memory. Graphics Address [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] = [47].</p> <table border="1" style="width: 100%; background-color: #e6f2ff; margin-top: 10px;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>The Depth Buffer can only be mapped to Main Memory (uncached).</p> <p>If the buffer is linear, the surface must be 64-byte aligned.</p>	Project:	BDW	Format:	GraphicsAddress[63:0]DepthBuffer	Programming Notes											
Project:	BDW																
Format:	GraphicsAddress[63:0]DepthBuffer																
Programming Notes																	

		3DSTATE_DEPTH_BUFFER																					
4	31:1 8	Height																					
	Format:		U14-1																				
This field specifies the height of the surface. If the surface is MIP-mapped, this field contains the height of the base MIP level.																							
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,0]</td> <td>Legal Range</td> <td>Must be zero</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')</td> </tr> <tr> <td>[0,16383]</td> <td>Legal Range</td> <td>Height of surface - 1 (y/v dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')</td> </tr> <tr> <td>[0,2047]</td> <td>Legal Range</td> <td>Height of surface - 1 (y/v dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')</td> </tr> <tr> <td>[0,16383]</td> <td>Legal Range</td> <td>y/v dimension</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')</td> </tr> </tbody> </table>				Value	Name	Description	Exists If	[0,0]	Legal Range	Must be zero	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')	[0,16383]	Legal Range	Height of surface - 1 (y/v dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')	[0,2047]	Legal Range	Height of surface - 1 (y/v dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')	[0,16383]	Legal Range	y/v dimension	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')
Value	Name	Description	Exists If																				
[0,0]	Legal Range	Must be zero	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')																				
[0,16383]	Legal Range	Height of surface - 1 (y/v dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')																				
[0,2047]	Legal Range	Height of surface - 1 (y/v dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')																				
[0,16383]	Legal Range	y/v dimension	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')																				
Programming Notes																							
The Height of the depth buffer must be the same as the Height of the render target(s) (defined in SURFACE_STATE), unless Surface Type is SURFTYPE_1D or SURFTYPE_2D with Depth = 0 (non-array) and LOD = 0 (non-mip mapped).																							
4	17:4	Width																					
	Format:		U14-1																				
This field specifies the width of the surface. If the surface is MIP-mapped, this field specifies the width of the base MIP level. The width is specified in units of pixels.																							
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>Legal Range</td> <td>Width of surface - 1 (x/u dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')</td> </tr> <tr> <td>[0,16383]</td> <td>Legal Range</td> <td>Width of surface - 1 (x/u dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')</td> </tr> <tr> <td>[0,2047]</td> <td>Legal Range</td> <td>Width of surface - 1 (x/u dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')</td> </tr> <tr> <td>[0,16383]</td> <td>Legal Range</td> <td>Width of surface - 1 (x/u dimension)</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')</td> </tr> </tbody> </table>				Value	Name	Description	Exists If	[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')	[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')	[0,2047]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')	[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')
Value	Name	Description	Exists If																				
[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_1D')																				
[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_2D')																				
[0,2047]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_3D')																				
[0,16383]	Legal Range	Width of surface - 1 (x/u dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]== 'SURFTYPE_CUBE')																				
Programming Notes																							
The Width specified by this 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 Height. The Width of the depth buffer must be the same as the Width of the render target(s) (defined in SURFACE_STATE), unless Surface Type is SURFTYPE_1D or SURFTYPE_2D with Depth = 0 (non-array) and LOD = 0 (non-mip mapped).																							

		3DSTATE_DEPTH_BUFFER	
	3:0	LOD	
		Format:	U4 for LOD units
		Value	Name
		[0,14]	
		Programming Notes	
		The LOD of the depth buffer must be the same as the LOD of the render target(s) (defined in SURFACE_STATE)	
5	31:2 1	Depth	
		Format:	U11-1
		This field specifies the total number of levels for a volume texture or the number of array elements allowed to be accessed starting at the Minimum Array Element for arrayed surfaces. If the volume texture is MIP-mapped, this field specifies the depth of the base MIP level.	
		Value	Name
		Description	Exists If
		[0,2047]	Legal Range
		Number of array elements - 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D')
		[0,2047]	Legal Range
		Number of array elements - 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')
		[0,2047]	Legal Range
		Depth of surface - 1 (r/z dimension)	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')
		[0,0]	Legal Range
		Must be zero	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_CUBE')
		Programming Notes	
		The Depth of the depth buffer must be the same as the Depth of the render target(s) (defined in SURFACE_STATE).	

3DSTATE_DEPTH_BUFFER																								
	20:1 0	Minimum Array Element <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">U11</td> </tr> </table> <p>For 1D and 2D Surfaces: This field indicates the minimum array element that can be accessed as part of this surface. The delivered array index is added to this field before being used to address the surface.</p> <p>For 3D Surfaces This field indicates the minimum 'R' coordinate on the LOD currently being rendered to. This field is added to the delivered array index before it is used to address the surface.</p> <p>For Other Surfaces This field is ignored</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,2047]]</td> <td>SURFTYPE_1D/2D</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D' Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')</td> </tr> <tr> <td>[0,2047]]</td> <td>SURFTYPE_3D</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')</td> </tr> </tbody> </table>	Format:	U11	Value	Name	Exists If	[0,2047]]	SURFTYPE_1D/2D	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D' Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')	[0,2047]]	SURFTYPE_3D	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')											
	Format:	U11																						
	Value	Name	Exists If																					
	[0,2047]]	SURFTYPE_1D/2D	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D' Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')																					
[0,2047]]	SURFTYPE_3D	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')																						
9:7	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ																					
Format:	MBZ																							
6:0	Depth Buffer Object Control State <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state for the depth buffer.</p>	Format:	MEMORY_OBJECT_CONTROL_STATE																					
Format:	MEMORY_OBJECT_CONTROL_STATE																							
6	31:2 6	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																		
	Project:	BDW																						
Format:	MBZ																							
25:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ																					
Format:	MBZ																							
For 1D and 2D Surfaces: This field must be set to the same value as the Depth field. For 3D Surfaces: This field	31:2 1	Render Target View Extent <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">U11-1</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 20%;">Description</th> <th style="width: 50%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,2047]</td> <td>Legal Range</td> <td>Number of array elements- 1</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D')</td> </tr> <tr> <td>[0,2047]</td> <td>Legal Range</td> <td>Number of array elements- 1</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')</td> </tr> <tr> <td>[0,2047]</td> <td>Legal Range</td> <td>To indication extent of [1,2048]</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')</td> </tr> <tr> <td>[0,0]</td> <td>Legal Range</td> <td>Must be zero</td> <td>(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_CUBE')</td> </tr> </tbody> </table>	Format:	U11-1	Value	Name	Description	Exists If	[0,2047]	Legal Range	Number of array elements- 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D')	[0,2047]	Legal Range	Number of array elements- 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')	[0,2047]	Legal Range	To indication extent of [1,2048]	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')	[0,0]	Legal Range	Must be zero	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_CUBE')
	Format:	U11-1																						
	Value	Name	Description	Exists If																				
	[0,2047]	Legal Range	Number of array elements- 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_1D')																				
	[0,2047]	Legal Range	Number of array elements- 1	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_2D')																				
[0,2047]	Legal Range	To indication extent of [1,2048]	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_3D')																					
[0,0]	Legal Range	Must be zero	(Structure[RENDER_SURFACE_STATE][Surface Type]='SURFTYPE_CUBE')																					
20:1 5	Reserved																							

3DSTATE_DEPTH_BUFFER						
indicates the extent of the accessible 'R' coordinate s minus 1 on the LOD currently being rendered to. For Other Surfaces This field is ignored.	14:0	Surface QPitch				
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>QPitch[16:2]</td> </tr> </table>	Format:	QPitch[16:2]		
		Format:	QPitch[16:2]			
		Description				
		This field specifies the distance in rows between array slices. It is used only in the following cases: <ul style="list-style-type: none"> • Surface Array is enabled <i>OR</i> • Number of Multisamples is not NUMSAMPLES_1 and Multisampled Surface Storage Format set to MSFMT_MSS <i>OR</i> • Surface Type is SURFTYPE_CUBE 				
		Other surface types: field is ignored				
<table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e6f2ff;"> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[4h, 1FFFCh]</td> <td></td> <td>in multiples of 4 (low 2 bits missing)</td> </tr> </tbody> </table>	Value	Name	Description	[4h, 1FFFCh]		in multiples of 4 (low 2 bits missing)
Value	Name	Description				
[4h, 1FFFCh]		in multiples of 4 (low 2 bits missing)				
		Programming Notes				
		Software must ensure that this field is set to a value sufficiently large that array slices in the surface do not overlap. Refer to the <i>Memory Data Formats</i> section for information on how surfaces are stored.				

3DSTATE_DRAWING_RECTANGLE

3DSTATE_DRAWING_RECTANGLE			
Project:	DevBWR+		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_DRAWING_RECTANGLE command is used to set the 3D drawing rectangle and related state.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	00h 3DSTATE_DRAWING_RECTANGLE
		Format:	OpCode
	15:14	Core Mode Select	
		Project:	BDW
Format:		U2	
Specifies which core this command will be considered valid and update based on the state in this command.			
Value		Description	
0h		Both cores are enabled and will update the state.	
1h		State will be updated in Core 0 only	
2h		State will be updated in Core 1 only	
3h			
13:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	2h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	



3DSTATE_DRAWING_RECTANGLE										
1	31:16	<p>Clipped Drawing Rectangle Y Min</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Color Buffer origin (upper left corner)</td> </tr> </table> <p>Specifies Ymin value of (inclusive) intersection of Drawing rectangle with the Color (Destination) Buffer, used for clipping. Pixels with Y coordinates less than Ymin will be clipped out.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,16383]</td> <td>Device ignores bits 31:30</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This value can be larger than Clipped Drawing Rectangle Y Max. If Ymin>Ymax, the clipped drawing rectangle is null, all polygons are discarded. If Ymin==Ymax, the clipped drawing rectangle is 1 pixel wide in the Y direction.</p>	Project:	All	Format:	U16 in Pixels from Color Buffer origin (upper left corner)	Value	Name	[0,16383]	Device ignores bits 31:30
	Project:	All								
Format:	U16 in Pixels from Color Buffer origin (upper left corner)									
Value	Name									
[0,16383]	Device ignores bits 31:30									
15:0	<p>Clipped Drawing Rectangle X Min</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Color Buffer origin (upper left corner)</td> </tr> </table> <p>Specifies Xmin value of (inclusive) intersection of Drawing rectangle with the Color (Destination) Buffer, used for clipping. Pixels with X coordinates less than Xmin will be clipped out.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,16383]</td> <td>Device ignores bits 15:14</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This value can be larger than Clipped Drawing Rectangle X Max. If Xmin>Xmax, the clipped drawing rectangle is null, all polygons are discarded. If Xmin==Xmax, the clipped drawing rectangle is 1 pixel wide in the X direction.</p>	Project:	All	Format:	U16 in Pixels from Color Buffer origin (upper left corner)	Value	Name	[0,16383]	Device ignores bits 15:14	
Project:	All									
Format:	U16 in Pixels from Color Buffer origin (upper left corner)									
Value	Name									
[0,16383]	Device ignores bits 15:14									
2	31:16	<p>Clipped Drawing Rectangle Y Max</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Color Buffer origin (upper left corner)</td> </tr> </table> <p>Specifies Ymax value of (inclusive) intersection of Drawing rectangle with the Color (Destination) Buffer, used for clipping. Pixels with coordinates greater than Ymax will be clipped out.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,16383]</td> <td>Device ignores bits 31:30</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This value can be less than Clipped Drawing Rectangle Y Min. If Ymax<Ymin, the clipped drawing rectangle is null, all polygons are discarded. If Ymin==Ymax, the clipped drawing rectangle is 1 pixel wide in the Y direction.</p>	Project:	All	Format:	U16 in Pixels from Color Buffer origin (upper left corner)	Value	Name	[0,16383]	Device ignores bits 31:30
	Project:	All								
Format:	U16 in Pixels from Color Buffer origin (upper left corner)									
Value	Name									
[0,16383]	Device ignores bits 31:30									

3DSTATE_DRAWING_RECTANGLE										
3	15:0	<p>Clipped Drawing Rectangle X Max</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Color Buffer origin (upper left corner)</td> </tr> </table> <p>Specifies Xmax value of (inclusive) intersection of Drawing rectangle with the Color (Destination) Buffer, used for clipping. Pixels with coordinates greater than Xmax will be clipped out.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>Device ignores bits 15:14</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This value can be less than Clipped Drawing Rectangle X Min. If Xmax<Xmin, the clipped drawing rectangle is null, all polygons are discarded. If Xmin==Xmax, the clipped drawing rectangle is 1 pixel wide in the X direction.</p>	Project:	All	Format:	U16 in Pixels from Color Buffer origin (upper left corner)	Value	Name	[0,16383]	Device ignores bits 15:14
	Project:	All								
Format:	U16 in Pixels from Color Buffer origin (upper left corner)									
Value	Name									
[0,16383]	Device ignores bits 15:14									
31:16	<p>Drawing Rectangle Origin Y</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>S15 in Pixels from Color Buffer origin (upper left corner).</td> </tr> </table> <p style="text-align: center;">Description</p> <p>Range: [-16384,16383] (Bit 31 should be a sign extension)</p> <p>Specifies Y origin of Drawing Rectangle (in whole pixels) relative to origin of the Color Buffer, used to map incoming (Draw Rectangle-relative) vertex positions to the Color Buffer space.</p>	Project:	All	Format:	S15 in Pixels from Color Buffer origin (upper left corner).					
Project:	All									
Format:	S15 in Pixels from Color Buffer origin (upper left corner).									
15:0	<p>Drawing Rectangle Origin X</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>S15 in Pixels from Color Buffer origin (upper left corner).</td> </tr> </table> <p style="text-align: center;">Description</p> <p>Range: [-16384,16383] (Bit 15 should be a sign extension)</p> <p>Specifies X origin of Drawing Rectangle (in whole pixels) relative to origin of the Color Buffer, used to map incoming (Draw Rectangle-relative) vertex positions to the Color Buffer space.</p>	Project:	All	Format:	S15 in Pixels from Color Buffer origin (upper left corner).					
Project:	All									
Format:	S15 in Pixels from Color Buffer origin (upper left corner).									

3DSTATE_DS

3DSTATE_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The state used by DS is defined with this inline state packet			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	1Dh 3DSTATE_DS	
	Format:	OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	7h Excludes DWord (0,1)	
	Project:	BDW	
	Format:	=n Total Length - 2	
1..2 Project: BDW	63:6	Kernel Start Pointer	
		Project:	BDW
		Format:	InstructionBaseOffset[63:6]Kernel
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	
		Project:	All
		Format:	MBZ

3DSTATE_DS																							
3	31	<p>Single Domain Point Dispatch</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U1 Enumerated Type</td> </tr> </table> <p>This field can be used to force single domain point SIMD4x2 DS threads. This field is ignored if SIMD8 Dispatch Enable is set.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Multiple</td> <td>Dual domain point SIMD4x2 thread dispatches are allowed.</td> </tr> <tr> <td>1h</td> <td>Single</td> <td>Single domain point SIMD4x2 thread dispatches are forced.</td> </tr> </tbody> </table> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Workaround</th> </tr> </table> <p>Workaround BDW: The Single Domain Point Dispatch must always be set to 0.</p>	Project:	BDW	Format:	U1 Enumerated Type	Value	Name	Description	0h	Multiple	Dual domain point SIMD4x2 thread dispatches are allowed.	1h	Single	Single domain point SIMD4x2 thread dispatches are forced.	Workaround							
	Project:	BDW																					
	Format:	U1 Enumerated Type																					
Value	Name	Description																					
0h	Multiple	Dual domain point SIMD4x2 thread dispatches are allowed.																					
1h	Single	Single domain point SIMD4x2 thread dispatches are forced.																					
Workaround																							
30	<p>Vector Mask Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U1 Enumerated Type</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Dmask</td> <td>The EU will use the Dispatch Mask (supplied by the DS 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> </tbody> </table> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </table> <p>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).</p>	Project:	BDW	Format:	U1 Enumerated Type	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								
Project:	BDW																						
Format:	U1 Enumerated Type																						
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																							
29:27	<p>Sampler Count</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 30%;">Name</th> <th>Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	U3	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
Project:	BDW																						
Format:	U3																						
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																					

3DSTATE_DS			
26	Reserved		
	Project:	BDW	
	Format:	MBZ	
	25:18	Binding Table Entry Count	
		Project:	BDW
		Format:	U8
		<p>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.</p> <p>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.</p>	
		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	
		Project:	BDW
	Format:	U1 Enumerated Type	
	Specifies the priority of the thread for dispatch: This field is ignored if DS Function Enable is DISABLED.		
	Value	Name	
	0h	Normal	
1h	High		
16	Floating Point Mode		
	Project:	BDW	
	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	
	0h	IEEE-754	
1h	Alternate		

3DSTATE_DS				
	15	Reserved	Project: BDW	Format: MBZ
	14	Accesses UAV	Project: BDW	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	Project: BDW	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	Project: BDW	Format: MBZ	
7	Software Exception Enable	Project: BDW	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	Project: BDW	Format: MBZ	
4..5 Project: BDW	63:10	Scratch Space Base Pointer	Project: BDW	Format: GeneralStateOffset[63:10]ScratchSpace
	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	Project: BDW	Format: MBZ

3DSTATE_DS											
	3:0	<p>Per-Thread Scratch Space</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U4 power of 2 Bytes over 1K Bytes</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td>indicating [1K Bytes, 2M Bytes]</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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.</p>	Project:	BDW	Format:	U4 power of 2 Bytes over 1K Bytes	Value	Name	[0,11]	indicating [1K Bytes, 2M Bytes]	
	Project:	BDW									
	Format:	U4 power of 2 Bytes over 1K Bytes									
	Value	Name									
	[0,11]	indicating [1K Bytes, 2M Bytes]									
6 Project: BDW	31:25	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
	Project:	BDW									
	Format:	MBZ									
	24:20	<p>Dispatch GRF Start Register For URB Data</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GRFRegister[4:0]</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> <td>indicating GRF [R0, R31]</td> </tr> </tbody> </table>	Project:	BDW	Format:	GRFRegister[4:0]	Value	Name	Description	[0,31]	
Project:	BDW										
Format:	GRFRegister[4:0]										
Value	Name	Description									
[0,31]		indicating GRF [R0, R31]									
19:18	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
Project:	BDW										
Format:	MBZ										
17:11	<p>Patch URB Entry Read Length</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U7</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>[0,64]</td> <td></td> </tr> </tbody> </table>	Project:	BDW	Format:	U7	Value	Name	[0,64]			
Project:	BDW										
Format:	U7										
Value	Name										
[0,64]											
10	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW									
Format:	MBZ										

3DSTATE_DS											
	9:4	<p>Patch URB Entry Read Offset</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,63]</td> <td></td> </tr> </tbody> </table>	Project:	BDW	Format:	U6	Value	Name	[0,63]		
	Project:	BDW									
Format:	U6										
Value	Name										
[0,63]											
	3:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
Project:	BDW										
Format:	MBZ										
7 Project: BDW	31	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
	Project:	BDW									
	Format:	MBZ									
	30	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
	Project:	BDW									
Format:	MBZ										
29:21	<p>Maximum Number of Threads</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U9-1 Thread Count</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,503]</td> <td></td> <td style="text-align: center;">indicating thread count of [1,504]</td> </tr> </tbody> </table>	Project:	BDW	Format:	U9-1 Thread Count	Value	Name	Description	[0,503]		indicating thread count of [1,504]
Project:	BDW										
Format:	U9-1 Thread Count										
Value	Name	Description									
[0,503]		indicating thread count of [1,504]									
20:11	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
Project:	BDW										
Format:	MBZ										
10	<p>Statistics Enable</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	BDW	Format:	Enable						
Project:	BDW										
Format:	Enable										



3DSTATE_DS			
9:5	Reserved		
	Project: BDW		
	Format: MBZ		
	Reserved		
	Project: BDW		
4	Format: MBZ		
	Reserved		
3	SIMD8 Dispatch Enable		
	Project: BDW		
2	Format: Enable		
	Compute W Coordinate Enable		
1	Project: BDW		
	Format: Disable		
<p>This field is used to specify how DS threads are dispatched. The setting of this field must agree with how the DS kernel was compiled. If ENABLED, SIMD8 DS thread dispatches are performed. The Single Domain Point Dispatch field is ignored. If DISABLED, SIMD4x2 thread dispatches are performed. The Single Domain Point Dispatch field can be used to force single domain point dispatches.</p>			
<p>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.</p>			
<p>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.</p>			
<table border="1" style="width: 100%;"> <tr> <td style="width: 50%; text-align: center;">Workaround</td> <td style="width: 50%; text-align: center;">Project</td> </tr> </table>		Workaround	Project
Workaround	Project		

3DSTATE_DS										
8 Project: BDW	0	Function Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </table> <p>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.</p>	Project:	BDW	Format:	Enable	Programming Notes			
	Project:	BDW								
	Format:	Enable								
	Programming Notes									
	31:28	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
Project:	BDW									
Format:	MBZ									
27	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW							
Project:	BDW									
26:21	Vertex URB Entry Output Read Offset <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB by SBE.</p> <table border="1" style="width: 100%; border-collapse: collapse; background-color: #e6f2ff;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,63]</td> <td></td> </tr> </tbody> </table>	Project:	BDW	Format:	U6	Value	Name	[0,63]		
Project:	BDW									
Format:	U6									
Value	Name									
[0,63]										
20:16	Vertex URB Entry Output Length <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>Specifies the amount of URB data written for each Vertex URB entry, in 256-bit register increments.</p> <table border="1" style="width: 100%; border-collapse: collapse; background-color: #e6f2ff;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[1,16]</td> <td></td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </table> <p>This length does not include the vertex header.</p>	Project:	BDW	Format:	U5	Value	Name	[1,16]		Programming Notes
Project:	BDW									
Format:	U5									
Value	Name									
[1,16]										
Programming Notes										
15:8	User Clip Distance Clip Test Enable Bitmask <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Mask[8]</td> </tr> </table> <p>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.</p>	Project:	BDW	Format:	Mask[8]					
Project:	BDW									
Format:	Mask[8]									



3DSTATE_DS					
7:0	<p>User Clip Distance Cull Test Enable Bitmask</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">BDW</td> </tr> <tr> <td>Format:</td> <td>Mask[8]</td> </tr> </table> <p>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.</p>	Project:	BDW	Format:	Mask[8]
Project:	BDW				
Format:	Mask[8]				

3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC

3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the Gather Pool for Gather Buffers.			
Programming Notes			
This command must only be programmed when resource streamer is enabled thru batch buffer start and MI_RS_CONTROL has not disabled resource streamer.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	GFXPIPE_3D		
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
23:16	3D Command Sub Opcode		
	Default Value:	1Bh 3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC	
	Format:	OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n	
	Total Length - 2		
	Value	Name	
2h	DWORD_COUNT_n [Default]		
1..2 Project: BDW	63:48	Reserved	
		Project:	BDW
		Format:	MBZ

3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC		
	47:13 Dx9 Constant Buffer Pool Base Address	
	Project: BDW	
	Format: GraphicsAddress[47:13]Dx9_Constant_Buffer_Pool	
	Specifies the base address of the Dx9 Constant Buffer pool.	
	12:11 Reserved	
	Project: BDW Format: MBZ	
	10 Dx9 Constant Buffer Pool Enable	
	Project: BDW	
	Format: Enable	
	When this bit is set it enables HW Dx9 constants buffers. When this bit is cleared it disables HW Dx9 constant buffers, the local bits for the constant buffers are cleared and the buffers will not be save or restored as part of context.	
	9:7 Reserved	
	Project: BDW Format: MBZ	
	6:0 Surface Object Control State	
	Project: BDW	
	Format: MEMORY_OBJECT_CONTROL_STATE	
	Specifies the memory object control state for this surface.	
	Programming Notes	
	Bit 2 is not programmable and is always zero.	
3 Project: BDW	31:13 Dx9 Constant Buffer Pool Buffer Size	
	Project: BDW	
	Format: U19	
	This field specifies the size of the buffer in 8K pages. Any access 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.	
	Restriction	
	Programming size of zero is illegal in the case that the pool is enabled.	
	12:0 Reserved	
	Project: BDW	
	Format: MBZ	

3DSTATE_DX9_CONSTANTB_PS

3DSTATE_DX9_CONSTANTB_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets a DX9 constant Boolean register for PS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTB_PS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	3Eh 3DSTATE_DX9_CONSTANTB_PS
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ
7:0	DWord Length		
	Project:	All	
	Format:	=n Total Length - 2	
	Value	Name	
	0h	[Default]	
0h-10h	Excludes DWord (0,1)		

3DSTATE_DX9_CONSTANTB_PS			
1	31:16	Reserved	
		Project: All	
		Format: MBZ	
	15	Global Constant Register	
		Project: All	
		Format: U1	
	When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.		
	14:4	Reserved	
		Project: All	
		Format: MBZ	
	3:0	Constant Register Index	
		Project: All	
Format: U4			
This field specifies the index of 1st boolean to be updated.			
2..n	31:0	Entry	
		Format: DX9_CONSTANTB_ENTRY	
The nth boolean to be updated.			

3DSTATE_DX9_CONSTANTB_VS

3DSTATE_DX9_CONSTANTB_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets a DX9 constant Boolean register for PS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTB_VS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		3Dh 3DSTATE_DX9_CONSTANTB_VS	
Format:		OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	
	0h	[Default]	
	0h-10h	Excludes DWord (0,1)	
1	31:16	Reserved	
		Format:	MBZ



3DSTATE_DX9_CONSTANTB_VS				
	15	<p>Global Constant Register</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U1</td> </tr> </table> <p>When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.</p>	Format:	U1
	Format:	U1		
	14:4	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
3:0	<p>Constant Register Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>This field specifies the index of 1st boolean to be updated.</p>	Format:	U4	
Format:	U4			
2..n	31:0	<p>Entry</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">DX9_CONSTANTB_ENTRY</td> </tr> </table> <p>The nth boolean to be updated.</p>	Format:	DX9_CONSTANTB_ENTRY
Format:	DX9_CONSTANTB_ENTRY			

3DSTATE_DX9_CONSTANTF_PS

3DSTATE_DX9_CONSTANTF_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets one or more DX9 constant float registers for PS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTF_PS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		3Ah 3DSTATE_DX9_CONSTANTF_PS	
Format:		OpCode	
15:11	Reserved		
	Format:	MBZ	
10:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	
	1h	Excludes DWord (0,1) [Default]	
	1h-400h	multiples of 4	
1	31:16	Reserved	
		Format:	MBZ



3DSTATE_DX9_CONSTANTF_PS				
	15	Global Constant Register <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U1</td> </tr> </table> <p>When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.</p>	Format:	U1
	Format:	U1		
	14:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
7:0	Constant Register Index <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U8</td> </tr> </table> <p>This field specifies the index of 1st 4 component float to be updated.</p>	Format:	U8	
Format:	U8			
2..n	127:0	Entry <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">DX9_CONSTANTF_ENTRY</td> </tr> </table> <p>The four components of the nth float to be updated.</p>	Format:	DX9_CONSTANTF_ENTRY
Format:	DX9_CONSTANTF_ENTRY			

3DSTATE_DX9_CONSTANTF_VS

3DSTATE_DX9_CONSTANTF_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets one or more DX9 constant float registers for VS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTF_VS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		39h 3DSTATE_DX9_CONSTANTF_VS	
Format:		OpCode	
15:11	Reserved		
	Format:	MBZ	
10:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	
	1h	Excludes DWord (0,1) [Default]	
	1h-400h	multiples of 4	
1	31:16	Reserved	
		Format:	MBZ

3DSTATE_DX9_CONSTANTF_VS				
	15	Global Constant Register <table border="1" style="width: 100%;"> <tr> <td style="width: 80%;">Format:</td> <td style="width: 20%;">U1</td> </tr> </table> <p>When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.</p>	Format:	U1
	Format:	U1		
	14:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 80%;">Format:</td> <td style="width: 20%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
7:0	Constant Register Index <table border="1" style="width: 100%;"> <tr> <td style="width: 80%;">Format:</td> <td style="width: 20%;">U8</td> </tr> </table> <p>This field specifies the index of 1st 4 component float to be updated.</p>	Format:	U8	
Format:	U8			
2..n	127:0	Entry <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">DX9_CONSTANTF_ENTRY</td> </tr> </table> <p>The four components of the nth float to be updated.</p>	Format:	DX9_CONSTANTF_ENTRY
Format:	DX9_CONSTANTF_ENTRY			

3DSTATE_DX9_CONSTANTI_PS

3DSTATE_DX9_CONSTANTI_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets one or more DX9 constant integer registers for PS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTI_PS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		3Ch 3DSTATE_DX9_CONSTANTI_PS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	Description
	1h	[Default]	Excludes DWord (0,1)
	0h-80h	multiples of 4	
1	31:16	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_DX9_CONSTANTI_PS					
	15	Global Constant Register			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.</p>	Project:	All	Format:
	Project:	All			
	Format:	U1			
	14:5	Reserved			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:
Project:	All				
Format:	MBZ				
4:0	Constant Register Index				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>This field specifies the index of 1st 4 component integer to be updated.</p>	Project:	All	Format:	U5
	Project:	All			
Format:	U5				
2..n	127:0	Entry <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>DX9_CONSTANTI_ENTRY</td> </tr> </table> <p>The four components of the nth float to be updated.</p>	Format:	DX9_CONSTANTI_ENTRY	
Format:	DX9_CONSTANTI_ENTRY				

3DSTATE_DX9_CONSTANTI_VS

3DSTATE_DX9_CONSTANTI_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets one or more DX9 constant integer registers for PS.			
Programming Notes			
<ul style="list-style-type: none"> The 3DSTATE_DX9_CONSTANTI_VS is a variable length command. Programming this command in batch buffer requires that all float, integer and boolean constants initialized prior to any commands or events that cause the constants to be written to memory. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h GFXPIPE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	3Bh 3DSTATE_DX9_CONSTANTI_VS	
	Format:	OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	Description
	0h	[Default]	Excludes DWord (0,1)
0h-80h	multiples of 4		
1	31:16	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_DX9_CONSTANTI_VS						
	15	Global Constant Register <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>When this bit is set the global constant register set will be updated. When this bit is clear the local constant register set will be updated.</p>	Project:	All	Format:	U1
	Project:	All				
	Format:	U1				
	14:5	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
	Format:	MBZ				
4:0	Constant Register Index <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>This field specifies the index of 1st 4 component integer to be updated.</p>	Project:	All	Format:	U5	
Project:	All					
Format:	U5					
2..n	127:0	Entry <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>DX9_CONSTANTI_ENTRY</td> </tr> </table> <p>The four components of the nth float to be updated.</p>	Format:	DX9_CONSTANTI_ENTRY		
Format:	DX9_CONSTANTI_ENTRY					

3DSTATE_DX9_GENERATE_ACTIVE_PS

3DSTATE_DX9_GENERATE_ACTIVE_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_DX9_GENERATE_ACTIVE_PS command is used to generate fixed functions' DX9 Constant Buffer. A DX9 Constant register is made active by writing it out to the constant buffer.</p>			
Programming Notes			
<p>Restriction: The global and local buffers are not initialized after reset. Any data written without being initialized will be undefined. DX9 constant buffers are written due to context save/restore or the Generate Active Command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		42h 3DSTATE_DX9_GENERATE_ACTIVE_PS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	0h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31:24	Reserved	
		Project:	All
		Format:	MBZ
	23:13	Pointer to PS Constant Buffer	

3DSTATE_DX9_GENERATE_ACTIVE_PS								
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>ConstantBufferOffset[23:13]BINDING_TABLE_STATE*</td> </tr> </table> <p>Specifies the 8KB aligned address offset of the PS function's Dx9 constant buffer. This offset is relative to the DX9 Constant buffer Base Address.</p>	Project:	All	Format:	ConstantBufferOffset[23:13]BINDING_TABLE_STATE*			
Project:	All							
Format:	ConstantBufferOffset[23:13]BINDING_TABLE_STATE*							
12	<p>DX9 Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>When this bit is set, the Resource Streamer will generate the PS constant buffer according to the DX9 rules:</p> <ol style="list-style-type: none"> Valid local register are made active. Global register becomes active, unless the corresponding local register is valid. Local register valids are reset. <p>When this bit is cleared, the Resource Streamer will generate the PS constant buffer according to the DX8 rules:</p> <ol style="list-style-type: none"> Global register become active. Local register valids are reset. <table border="1" style="background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </table> <p>In DX8 mode software will set all constants as globals, even ones locally defined within a shader.</p>	Project:	All	Format:	Enable	Format:	U1	Programming Notes
Project:	All							
Format:	Enable							
Format:	U1							
Programming Notes								
11	<p>Clamp Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>When this bit is set, the Resource Streamer will generate the PS constant buffer with the global values clamped to [-1,1]. When this bit is cleared, the Resource Streamer will generate the PS constant buffer without the global value clamped.</p> <table border="1" style="background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </table> <p>The clamping only affects the values written out to the constant buffer and not the on-die registers.</p>	Project:	All	Format:	Enable	Format:	U1	Programming Notes
Project:	All							
Format:	Enable							
Format:	U1							
Programming Notes								
10:8	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ			
Project:	BDW							
Format:	MBZ							
7:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ			
Project:	All							
Format:	MBZ							

3DSTATE_DX9_GENERATE_ACTIVE_VS

3DSTATE_DX9_GENERATE_ACTIVE_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_DX9_GENERATE_ACTIVE_VS command is used to generate fixed functions' DX9 Constant Buffer. A DX9 Constant register is made active by writing it out to the constant buffer.</p> <p>Programming Restriction: The global and local buffers are not initialized after reset. Any data written without being initialized will be undefined. DX9 constant buffers are written due to context save/restore or the Generate Active Command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
23:16	3D Command Sub Opcode		
	Default Value:	41h 3DSTATE_DX9_GENERATE_ACTIVE_VS	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Format:	=n Total Length - 2	
1	31:24	Reserved	
		Format:	MBZ
23:13	Pointer to VS Constant Buffer		
	Format:	ConstantBufferOffset[23:13]	
<p>Specifies the 8KB aligned address offset of the VS function's Dx9 constant buffer. This offset is relative to the DX9 Constant buffer Base Address.</p>			



3DSTATE_DX9_GENERATE_ACTIVE_VS					
12	<p>DX9 Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set, the Resource Streamer will generate the VS constant buffer according to the DX9 rules:</p> <ol style="list-style-type: none"> 1. Valid local register are made active. 2. Global register becomes active, unless the corresponding local register is valid. 3. Local register valids are reset. <p>When this bit is cleared, the Resource Streamer will generate the VS constant buffer according to the DX8 rules:</p> <ol style="list-style-type: none"> 1. Global register become active. 2. Local register valids are reset. <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>In DX8 mode software will set all constants as globals, even ones locally defined within a shader.</p>	Format:	Enable	Programming Notes	
Format:	Enable				
Programming Notes					
11	<p>Clamp Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set, the Resource Streamer will generate the VS constant buffer with the global values clamped to [-1,1]. When this bit is cleared, the Resource Streamer will generate the VS constant buffer without the global value clamped.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>The clamping only affects the values written out to the constant buffer and not the on-die registers.</p>	Format:	Enable	Programming Notes	
Format:	Enable				
Programming Notes					
10:8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
7:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ		
Format:	MBZ				

3DSTATE_DX9_LOCAL_VALID_PS

3DSTATE_DX9_LOCAL_VALID_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets the local valid bits for the DX9 Constant Buffer			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	40h 3DSTATE_DX9_LOCAL_VALID_PS
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	9h Excludes Dword (0,1)	
	Project:	BDW	
	Format:	=n Total Length - 2	
1..8	31:0	Local ConstantF Valid Bits	
		Project:	All
		Format:	U32
Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.			
9	31:0	Local ConstantI Valid Bits	
		Project:	BDW
		Format:	U32
Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.			



3DSTATE_DX9_LOCAL_VALID_PS		
10	31:16	Reserved
		Project: All
		Format: MBZ
	15:0	Local ConstantB Valid Bits
		Project: BDW
		Format: U16
Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.		

3DSTATE_DX9_LOCAL_VALID_VS

3DSTATE_DX9_LOCAL_VALID_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets the local valid bits for the DX9 Constant Buffer			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h GFXPIPE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	3Fh 3DSTATE_DX9_LOCAL_VALID_VS
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	9h Excludes DWord (0,1)	
	Project:	BDW	
	Format:	=n Total Length - 2	
1..8	31:0	Local ConstantF Valid Bits	
		Project:	All
		Format:	U32
Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.			
9	31:0	Local ConstantI Valid Bits	
		Project:	BDW
		Format:	U32
Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.			



3DSTATE_DX9_LOCAL_VALID_VS		
10 Project: BDW	31:16	Reserved
		Project: BDW
		Format: MBZ
	15:0	Local ConstantB Valid Bits
		Project: BDW
		Format: U16
		Each bit field when set indicates that the corresponding local register is valid. When the bit is clear it indicates the local register is invalid.

3DSTATE_GATHER_CONSTANT_DS

3DSTATE_GATHER_CONSTANT_DS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command uses the constant buffer binding table entries to reference constant buffer surface states for the DS unit. The constant data in these is gathered and packed according to a gather table contained in this command.</p>		
Programming Notes		
The HW generated binding table must be enabled to use this command.		
The constant buffer block (group of aligned 16 binding table entries) must be set before this command is issued.		
If the surface type is NULL, any fetch using the surface state base address is not bound by the size of the surface state and the fetch still occurs.		
The length of the gather table is derived from the total length of the command. The command length is in DWords, but the gather table entries are 16 bits in length. If there is an unused odd entry at the end of the command the channel mask should be set to all 0s.		
When a 3DSTATE_GATHER_CONSTANT_* command is used there must be a matching 3DSTATE_CONSTANT_*. Furthermore the 3DSTATE_CONSTANT_* must occur in the same order as the 3DSTATE_GATHER_CONSTANT_*. For example if a 3DSTATE_GATHER_CONSTANT_VS occurs before a 3DSTATE_GATHER_CONSTANT_PS, then the 3DSTATE_CONSTANT_VS must occur before the 3DSTATE_CONSTANT_PS.		
If Gather pool is enabled, there must be a corresponding 3DSTATE_GATHER_CONSTANT command with any 3DSTATE_CONSTANT for any particular shader. To avoid any update to the Gather pool, and yet program the 3DSTATE_CONSTANT for a particular shader, send a 3DSTATE_GATHER_CONSTANT command with all valid bits set to zero.		
<p>The following commands must be executed after any 3DSTATE_GATHER_CONSTANT_* command that has Constant Buffer Valid greater than zero:</p> <ul style="list-style-type: none"> •(N times, minimum number is 4) MI_RS_STORE_DATA_IMM –To force engine idle before executing the next instruction. Write must occur to address that will not corrupt memory: •Resource Streamer Flush = 1 •3DSTATE_GATHER_CONSTANT_* (Ensures correct timing of sync between resource streamer and render pipeline) •The Constant Buffer Valid field should be zero and the Dword length equal to 1h. •3DSTATE_CONSTANT_*: •All values match the previous 3DSTATE_CONSTANT_* 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
		Format: OpCode

3DSTATE_GATHER_CONSTANT_DS											
	26:24	3D Command Opcode Default Value: 0h 3DSTATE_PIPELINED Format: OpCode									
	23:16	3D Command Sub Opcode Default Value: 37h 3DSTATE_GATHER_CONSTANT_DS Format: OpCode									
	15:8	Reserved Format: MBZ									
	7:0	DWord Length Format: =n Total Length - 2 <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 50%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>DWORD_COUNT_n [Default]</td> <td>excludes DWords 0,1</td> </tr> <tr> <td>[1,128]</td> <td>Range</td> <td>1-128 Entries</td> </tr> </tbody> </table>	Value	Name	Description	1	DWORD_COUNT_n [Default]	excludes DWords 0,1	[1,128]	Range	1-128 Entries
	Value	Name	Description								
	1	DWORD_COUNT_n [Default]	excludes DWords 0,1								
	[1,128]	Range	1-128 Entries								
	1	31:16	Constant Buffer Valid Format: U16 This field specifies which of the 16 constant buffers are used in the push constant gather. If a bit is set it indicates the corresponding constant buffer is used. If a bit is clear it indicates the corresponding constant buffer is not used. If this field is zero it indicate that the gather buffer is not used.								
		15:12	Constant Buffer Binding Table Block Format: U4 This field specifies the 16 entry block constant buffer in the binding table. The constant buffer entry block must be aligned on a 16 entry boundary.								
		11:2	Reserved Format: MBZ								
1		Reserved Project: BDW Format: MBZ									
0		Reserved Format: MBZ									
2	31:23	Reserved Format: MBZ									

3DSTATE_GATHER_CONSTANT_DS						
	22:6	Gather Buffer Offset <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GatherBufferOffset[22:6]</td> </tr> </table> <p>This field specifies the offset of the gather buffer within the Gather Pool</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> </table> <p>SW increments the offset by the size of the gather buffer in 512 bit units for each gather buffer generated.</p>	Format:	GatherBufferOffset[22:6]	Programming Notes	
	Format:	GatherBufferOffset[22:6]				
	Programming Notes					
	5	Constant Buffer Dx9 Generate Stall <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set the resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization.</p>	Project:	BDW	Format:	Enable
	Project:	BDW				
Format:	Enable					
4	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
3	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
2:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
3..n	15:0 <table border="1" style="width: 100%;"> <tr> <td>Entry</td> <td></td> </tr> <tr> <td>Format:</td> <td>GATHER_CONSTANT_ENTRY</td> </tr> </table>	Entry		Format:	GATHER_CONSTANT_ENTRY	
Entry						
Format:	GATHER_CONSTANT_ENTRY					



3DSTATE_GATHER_CONSTANT_GS

3DSTATE_GATHER_CONSTANT_GS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command uses the constant buffer binding table entries to reference constant buffer surface states for GS unit. The constant data in these is gathered and packed according to a gather table contained in this command.</p>		
Programming Notes		
<p>The HW generated binding table must be enabled to use this command.</p>		
<p>The constant buffer block (group of aligned 16 binding table entries) must be set before this command is issued.</p>		
<p>If the surface type is NULL, any fetch using the surface state base address is not bound by the size of the surface state and the fetch still occurs.</p>		
<p>The length of the gather table is derived from the total length of the command. The command length is in DWords, but the gather table entries are 16 bits in length. If there is an unused odd entry at the end of the command the channel mask should be set to all 0s.</p>		
<p>When a 3DSTATE_GATHER_CONSTANT_* command is used there must be a matching 3DSTATE_CONSTANT_*. Furthermore the 3DSTATE_CONSTANT_* must occur in the same order as the 3DSTATE_GATHER_CONSTANT_*. For example if a 3DSTATE_GATHER_CONSTANT_VS occurs before a 3DSTATE_GATHER_CONSTANT_PS, then the 3DSTATE_CONSTANT_VS must occur before the 3DSTATE_CONSTANT_PS.</p>		
<p>If Gather pool is enabled, there must be a corresponding 3DSTATE_GATHER_CONSTANT command with any 3DSTATE_CONSTANT for any particular shader. To avoid any update to the Gather pool, and yet program the 3DSTATE_CONSTANT for a particular shader, send a 3DSTATE_GATHER_CONSTANT command with all valid bits set to zero.</p>		
<p>The following commands must be executed after any 3DSTATE_GATHER_CONSTANT_* command that has Constant Buffer Valid greater than zero: •(N times, minimum number is 4) MI_RS_STORE_DATA_IMM –To force engine idle before executing the next instruction. Write must occur to address that will not corrupt memory:</p> <ul style="list-style-type: none"> •Resource Streamer Flush = 1 •3DSTATE_GATHER_CONSTANT_* (Ensures correct timing of sync between resource streamer and render pipeline) •The Constant Buffer Valid field should be zero and the Dword length equal to 1h. •3DSTATE_CONSTANT_*: •All values match the previous 3DSTATE_CONSTANT_* 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
Format: OpCode		

3DSTATE_GATHER_CONSTANT_GS													
	26:24	3D Command Opcode <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h 3DSTATE_PIPELINED</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h 3DSTATE_PIPELINED	Format:	OpCode							
	Default Value:	0h 3DSTATE_PIPELINED											
	Format:	OpCode											
	23:16	3D Command Sub Opcode <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>35h 3DSTATE_GATHER_CONSTANT_GS</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	35h 3DSTATE_GATHER_CONSTANT_GS	Format:	OpCode							
	Default Value:	35h 3DSTATE_GATHER_CONSTANT_GS											
	Format:	OpCode											
	15:8	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 55%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>DWORD_COUNT_n [Default]</td> <td>excludes DWords 0,1</td> </tr> <tr> <td>[1,128]</td> <td>Range</td> <td>1-128 Entries</td> </tr> </tbody> </table>	Format:	=n	Value	Name	Description	1	DWORD_COUNT_n [Default]	excludes DWords 0,1	[1,128]	Range	1-128 Entries
	Format:	=n											
Value	Name	Description											
1	DWORD_COUNT_n [Default]	excludes DWords 0,1											
[1,128]	Range	1-128 Entries											
1	31:16	Constant Buffer Valid <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p>This field specifies which of the 16 constant buffers are used in the push constant gather. If a bit is set it indicates the corresponding constant buffer is used. If a bit is clear it indicates the corresponding constant buffer is not used. If this field is zero it indicate that the gather buffer is not used.</p>	Format:	U16									
	Format:	U16											
	15:12	Constant Buffer Binding Table Block <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>This field specifies the 16 entry block constant buffer in the binding table. The constant buffer entry block must be aligned on a 16 entry boundary.</p>	Format:	U4									
	Format:	U4											
	11:2	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
1	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW												
Format:	MBZ												
0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
2	31:23	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											

3DSTATE_GATHER_CONSTANT_GS						
	22:6	Gather Buffer Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GatherBufferOffset[22:6]</td> </tr> </table> <p>This field specifies the offset of the gather buffer within the Gather Pool.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> </table> <p>SW increments the offset by the size of the gather buffer in 512 bit units for each gather buffer generated.</p>	Format:	GatherBufferOffset[22:6]	Programming Notes	
	Format:	GatherBufferOffset[22:6]				
	Programming Notes					
	5	Constant Buffer Dx9 Generate Stall <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set the resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization.</p>	Project:	BDW	Format:	Enable
	Project:	BDW				
Format:	Enable					
4	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
3	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
2:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
3..n	15:0 Entry <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GATHER_CONSTANT_ENTRY</td> </tr> </table>	Format:	GATHER_CONSTANT_ENTRY			
Format:	GATHER_CONSTANT_ENTRY					

3DSTATE_GATHER_CONSTANT_HS

3DSTATE_GATHER_CONSTANT_HS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command uses the constant buffer binding table entries to reference constant buffer surface states for HS unit. The constant data in these is gathered and packed according to a gather table contained in this command.</p>		
Programming Notes		
<p>The HW generated binding table must be enabled to use this command.</p>		
<p>The constant buffer block (group of aligned 16 binding table entries) must be set before this command is issued.</p>		
<p>If the surface type is NULL, any fetch using the surface state base address is not bound by the size of the surface state and the fetch still occurs.</p>		
<p>The length of the gather table is derived from the total length of the command. The command length is in DWords, but the gather table entries are 16 bits in length. If there is an unused odd entry at the end of the command the channel mask should be set to all 0s.</p>		
<p>When a 3DSTATE_GATHER_CONSTANT_* command is used there must be a matching 3DSTATE_CONSTANT_*. Furthermore the 3DSTATE_CONSTANT_* must occur in the same order as the 3DSTATE_GATHER_CONSTANT_*. For example if a 3DSTATE_GATHER_CONSTANT_VS occurs before a 3DSTATE_GATHER_CONSTANT_PS, then the 3DSTATE_CONSTANT_VS must occur before the 3DSTATE_CONSTANT_PS.</p>		
<p>If Gather pool is enabled, there must be a corresponding 3DSTATE_GATHER_CONSTANT command with any 3DSTATE_CONSTANT for any particular shader. To avoid any update to the Gather pool, and yet program the 3DSTATE_CONSTANT for a particular shader, send a 3DSTATE_GATHER_CONSTANT command with all valid bits set to zero.</p>		
<p>The following commands must be executed after any 3DSTATE_GATHER_CONSTANT_* command that has Constant Buffer Valid greater than zero: •(N times, minimum number is 4) MI_RS_STORE_DATA_IMM –To force engine idle before executing the next instruction. Write must occur to address that will not corrupt memory:</p> <ul style="list-style-type: none"> •Resource Streamer Flush = 1 •3DSTATE_GATHER_CONSTANT_* (Ensures correct timing of sync between resource streamer and render pipeline) •The Constant Buffer Valid field should be zero and the Dword length equal to 1h. •3DSTATE_CONSTANT_*: •All values match the previous 3DSTATE_CONSTANT_* 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
Format: OpCode		

3DSTATE_GATHER_CONSTANT_HS											
	26:24	3D Command Opcode Default Value: 0h 3DSTATE_PIPELINED Format: OpCode									
	23:16	3D Command Sub Opcode Default Value: 36h 3DSTATE_GATHER_CONSTANT_HS Format: OpCode									
	15:8	Reserved Format: MBZ									
	7:0	DWord Length Format: =n Total Length - 2 <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>DWORD_COUNT_n [Default]</td> <td>excludes DWords 0,1</td> </tr> <tr> <td>[1,128]</td> <td>Range</td> <td>1-128 Entries</td> </tr> </tbody> </table>	Value	Name	Description	1	DWORD_COUNT_n [Default]	excludes DWords 0,1	[1,128]	Range	1-128 Entries
	Value	Name	Description								
	1	DWORD_COUNT_n [Default]	excludes DWords 0,1								
	[1,128]	Range	1-128 Entries								
	1	31:16	Constant Buffer Valid Format: U16 This field specifies which of the 16 constant buffers are used in the push constant gather. If a bit is set it indicates the corresponding constant buffer is used. If a bit is clear it indicates the corresponding constant buffer is not used. If this field is zero it indicate that the gather buffer is not used.								
		15:12	Constant Buffer Binding Table Block Format: U4 This field specifies the 16 entry block constant buffer in the binding table. The constant buffer entry block must be aligned on a 16 entry boundary.								
		11:2	Reserved Format: MBZ								
1		Reserved Project: BDW Format: MBZ									
		Reserved Format: MBZ									
2		31:23	Reserved Format: MBZ								

3DSTATE_GATHER_CONSTANT_HS		
	22:6	Gather Buffer Offset
		Format: GatherBufferOffset[22:6]
		This field specifies the offset of the gather buffer within the Gather Pool.
		Programming Notes
		SW increments the offset by the size of the gather buffer in 512 bit units for each gather buffer generated.
	5	Constant Buffer Dx9 Generate Stall
		Project: BDW
		Format: Enable
		When this bit is set the resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization.
	4	Reserved
		Project: All Format: MBZ
	3	Reserved
		Project: BDW Format: MBZ
	2:0	Reserved
		Project: All Format: MBZ
3..n	15:0	Entry
		Format: GATHER_CONSTANT_ENTRY



3DSTATE_GATHER_CONSTANT_PS

3DSTATE_GATHER_CONSTANT_PS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command uses the constant buffer binding table entries to reference constant buffer surface states for PS unit. The constant data in these is gathered and packed according to a gather table contained in this command.</p>		
Programming Notes		
<p>The HW generated binding table must be enabled to use this command.</p>		
<p>The constant buffer block (group of aligned 16 binding table entries) must be set before this command is issued.</p>		
<p>If the surface type is NULL, any fetch using the surface state base address is not bound by the size of the surface state and the fetch still occurs.</p>		
<p>The length of the gather table is derived from the total length of the command. The command length is in DWords, but the gather table entries are 16 bits in length. If there is an unused odd entry at the end of the command the channel mask should be set to all 0s.</p>		
<p>When a 3DSTATE_GATHER_CONSTANT_* command is used there must be a matching 3DSTATE_CONSTANT_*. Furthermore the 3DSTATE_CONSTANT_* must occur in the same order as the 3DSTATE_GATHER_CONSTANT_*. For example if a 3DSTATE_GATHER_CONSTANT_VS occurs before a 3DSTATE_GATHER_CONSTANT_PS, then the 3DSTATE_CONSTANT_VS must occur before the 3DSTATE_CONSTANT_PS.</p>		
<p>If Gather pool is enabled, there must be a corresponding 3DSTATE_GATHER_CONSTANT command with any 3DSTATE_CONSTANT for any particular shader. To avoid any update to the Gather pool, and yet program the 3DSTATE_CONSTANT for a particular shader, send a 3DSTATE_GATHER_CONSTANT command with all valid bits set to zero.</p>		
<p>The following commands must be executed after any 3DSTATE_GATHER_CONSTANT_* command that has Constant Buffer Valid greater than zero: •(N times, minimum number is 4) MI_RS_STORE_DATA_IMM –To force engine idle before executing the next instruction. Write must occur to address that will not corrupt memory:</p> <ul style="list-style-type: none"> •Resource Streamer Flush = 1 •3DSTATE_GATHER_CONSTANT_* (Ensures correct timing of sync between resource streamer and render pipeline) •The Constant Buffer Valid field should be zero and the Dword length equal to 1h. •3DSTATE_CONSTANT_*: •All values match the previous 3DSTATE_CONSTANT_* 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
Format: OpCode		

3DSTATE_GATHER_CONSTANT_PS													
	26:24	3D Command Opcode <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h 3DSTATE_PIPELINED</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h 3DSTATE_PIPELINED	Format:	OpCode							
	Default Value:	0h 3DSTATE_PIPELINED											
	Format:	OpCode											
	23:16	3D Command Sub Opcode <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>38h 3DSTATE_GATHER_CONSTANT_PS</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	38h 3DSTATE_GATHER_CONSTANT_PS	Format:	OpCode							
	Default Value:	38h 3DSTATE_GATHER_CONSTANT_PS											
	Format:	OpCode											
	15:8	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 55%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>DWORD_COUNT_n [Default]</td> <td>excludes DWords 0,1</td> </tr> <tr> <td>[1,128]</td> <td>Range</td> <td>1-128 Entries</td> </tr> </tbody> </table>	Format:	=n	Value	Name	Description	1	DWORD_COUNT_n [Default]	excludes DWords 0,1	[1,128]	Range	1-128 Entries
	Format:	=n											
Value	Name	Description											
1	DWORD_COUNT_n [Default]	excludes DWords 0,1											
[1,128]	Range	1-128 Entries											
1	31:16	Constant Buffer Valid <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p>This field specifies which of the 16 constant buffers are used in the push constant gather. If a bit is set it indicates the corresponding constant buffer is used. If a bit is clear it indicates the corresponding constant buffer is not used. If this field is zero it indicate that the gather buffer is not used.</p>	Format:	U16									
	Format:	U16											
	15:12	Constant Buffer Binding Table Block <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>This field specifies the 16 entry block constant buffer in the binding table. The constant buffer entry block must be aligned on a 16 entry boundary.</p>	Format:	U4									
	Format:	U4											
11:2	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW												
Format:	MBZ												
2	31:23	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
Format:	MBZ												

3DSTATE_GATHER_CONSTANT_PS						
	22:6	<p>Gather Buffer Offset</p> <table border="1"> <tr> <td>Format:</td> <td>GatherBufferOffset[22:6]</td> </tr> </table> <p>This field specifies the offset of the gather buffer within the Gather Pool.</p> <table border="1"> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> </table> <p>SW increments the offset by the size of the gather buffer in 512 bit units for each gather buffer generated.</p>	Format:	GatherBufferOffset[22:6]	Programming Notes	
	Format:	GatherBufferOffset[22:6]				
	Programming Notes					
	5	<p>Constant Buffer Dx9 Generate Stall</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set the resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization.</p>	Project:	BDW	Format:	Enable
	Project:	BDW				
Format:	Enable					
4	<p>Constant Buffer Dx9 Enable</p> <table border="1"> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set it indicates that the constant buffer is a HW generated Dx9 constant buffer. The resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization. Additionally the Dx9 constant buffers are a single buffer but larger than 4KB. Internally the HW will treat the DX9 buffer as 2 constant buffers. When this bit is enable only the 1st constant buffer valid bit is set. The 2nd constant buffer surface pointer will automatically be the 1st pointer + 4KB.</p>	Format:	Enable			
Format:	Enable					
3	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
2:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
3..n	<p>15:0 Entry</p> <table border="1"> <tr> <td>Format:</td> <td>GATHER_CONSTANT_ENTRY</td> </tr> </table>	Format:	GATHER_CONSTANT_ENTRY			
Format:	GATHER_CONSTANT_ENTRY					

3DSTATE_GATHER_CONSTANT_VS

3DSTATE_GATHER_CONSTANT_VS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command uses the constant buffer binding table entries to reference constant buffer surface states for VS unit. The constant data in these is gathered and packed according to a gather table contained in this command.</p>		
Programming Notes		
<p>The HW generated binding table must be enabled to use this command.</p>		
<p>The constant buffer block (group of aligned 16 binding table entries) must be set before this command is issued.</p>		
<p>If the surface type is NULL, any fetch using the surface state base address is not bound by the size of the surface state and the fetch still occurs.</p>		
<p>The length of the gather table is derived from the total length of the command. The command length is in DWords, but the gather table entries are 16 bits in length. If there is an unused odd entry at the end of the command the channel mask should be set to all 0s.</p>		
<p>When a 3DSTATE_GATHER_CONSTANT_* command is used there must be a matching 3DSTATE_CONSTANT_*. Furthermore the 3DSTATE_CONSTANT_* must occur in the same order as the 3DSTATE_GATHER_CONSTANT_*. For example if a 3DSTATE_GATHER_CONSTANT_VS occurs before a 3DSTATE_GATHER_CONSTANT_PS, then the 3DSTATE_CONSTANT_VS must occur before the 3DSTATE_CONSTANT_PS.</p>		
<p>If Gather pool is enabled, there must be a corresponding 3DSTATE_GATHER_CONSTANT command with any 3DSTATE_CONSTANT for any particular shader. To avoid any update to the Gather pool, and yet program the 3DSTATE_CONSTANT for a particular shader, send a 3DSTATE_GATHER_CONSTANT command with all valid bits set to zero.</p>		
<p>The following commands must be executed after any 3DSTATE_GATHER_CONSTANT_* command that has Constant Buffer Valid greater than zero:</p> <ul style="list-style-type: none"> •(N times, minimum number is 4) MI_RS_STORE_DATA_IMM –To force engine idle before executing the next instruction. Write must occur to address that will not corrupt memory: •Resource Streamer Flush = 1 •3DSTATE_GATHER_CONSTANT_* (Ensures correct timing of sync between resource streamer and render pipeline) •The Constant Buffer Valid field should be zero and the Dword length equal to 1h. •3DSTATE_CONSTANT_*: •All values match the previous 3DSTATE_CONSTANT_* 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
		Format: OpCode

3DSTATE_GATHER_CONSTANT_VS											
	26:24	3D Command Opcode Default Value: 0h 3DSTATE_PIPELINED Format: OpCode									
	23:16	3D Command Sub Opcode Default Value: 34h 3DSTATE_GATHER_CONSTANT_VS Format: OpCode									
	15:8	Reserved Format: MBZ									
	7:0	DWord Length Format: =n Total Length - 2 <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 55%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>DWORD_COUNT_n [Default]</td> <td>excludes DWords 0,1</td> </tr> <tr> <td>[1,128]</td> <td>Range</td> <td>1-128 Entries</td> </tr> </tbody> </table>	Value	Name	Description	0	DWORD_COUNT_n [Default]	excludes DWords 0,1	[1,128]	Range	1-128 Entries
	Value	Name	Description								
	0	DWORD_COUNT_n [Default]	excludes DWords 0,1								
	[1,128]	Range	1-128 Entries								
	1	31:16	Constant Buffer Valid Format: U16 This field specifies which of the 16 constant buffers are used in the push constant gather. If a bit is set it indicates the corresponding constant buffer is used. If a bit is clear it indicates the corresponding constant buffer is not used. If this field is zero it indicate that the gather buffer is not used.								
		15:12	Constant Buffer Binding Table Block Format: U4 This field specifies the 16 entry block constant buffer in the binding table. The constant buffer entry block must be aligned on a 16 entry boundary.								
		11:2	Reserved Format: MBZ								
1:0		Reserved Project: BDW Format: MBZ									
2	31:23	Reserved Format: MBZ									

3DSTATE_GATHER_CONSTANT_VS						
	22:6	Gather Buffer Offset <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GatherBufferOffset[22:6]</td> </tr> </table> <p>This field specifies the offset of the gather buffer within the Gather Pool.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> </table> <p>SW increments the offset by the size of the gather buffer in 512 bit units for each gather buffer generated.</p>	Format:	GatherBufferOffset[22:6]	Programming Notes	
	Format:	GatherBufferOffset[22:6]				
	Programming Notes					
	5	Constant Buffer Dx9 Generate Stall <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set the resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization.</p>	Project:	BDW	Format:	Enable
	Project:	BDW				
Format:	Enable					
4	Constant Buffer Dx9 Enable <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When this bit is set it indicates that the constant buffer is a HW generated Dx9 constant buffer. The resource streamer will wait for the Dx9 constant buffer generator to be done before issuing this command to ensure buffer synchronization. Additionally the Dx9 constant buffers are a single buffer but larger than 4KB. Internally the HW will treat the DX9 buffer as 2 constant buffers. When this bit is enable only the 1st constant buffer valid bit is set. The 2nd constant buffer surface pointer will automatically be the 1st pointer + 4KB.</p>	Format:	Enable			
Format:	Enable					
3	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
2:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
3..n	15:0 <table border="1" style="width: 100%;"> <tr> <td colspan="2">Entry</td> </tr> <tr> <td>Format:</td> <td>GATHER_CONSTANT_ENTRY</td> </tr> </table>	Entry		Format:	GATHER_CONSTANT_ENTRY	
Entry						
Format:	GATHER_CONSTANT_ENTRY					

3DSTATE_GATHER_POOL_ALLOC

3DSTATE_GATHER_POOL_ALLOC			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the Gather Pool for Gather Buffers.			
Programming Notes			
If the Gather Constant Pool is enabled and RS is enabled, then for each 3DSTATE_CONSTANT_* command there must be a corresponding 3DSTATE_GATHER_CONSTANT_* command. If gather pool is enabled, then Buffer 1 of the 3DSTATE_CONSTANT command address will be an offset from the Gather Pool Base Address.			
The gather constants can only be enabled if the binding table generator is also enabled. This command must only be programmed when resource streamer is enabled to parse commands within a batch buffer.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		1h 3DSTATE_NONPIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	1Ah 3DSTATE_GATHER_POOL_ALLOC	
	Format:	OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n	
	Value	Name	
	2h	DWORD_COUNT_n [Default]	
1..2 Project: BDW	63:12	Gather Pool Base Address	
		Project:	BDW
		Format:	GraphicsAddress[63:12]Gather_Pool
		GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].	

3DSTATE_GATHER_POOL_ALLOC																	
	<table border="1"> <tr> <td style="text-align: center;">11</td> <td>Gather Pool Enable</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> <tr> <td colspan="2"> <p>When this bit is set it enables HW gathering of push constants. When this bit is cleared it disables HW gathering of push constants.</p> </td> </tr> </table>	11	Gather Pool Enable	Project:	BDW	Format:	Enable	<p>When this bit is set it enables HW gathering of push constants. When this bit is cleared it disables HW gathering of push constants.</p>									
	11	Gather Pool Enable															
	Project:	BDW															
Format:	Enable																
<p>When this bit is set it enables HW gathering of push constants. When this bit is cleared it disables HW gathering of push constants.</p>																	
<table border="1"> <tr> <td style="text-align: center;">10:7</td> <td>Reserved</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	10:7	Reserved	Project:	BDW	Format:	MBZ											
10:7	Reserved																
Project:	BDW																
Format:	MBZ																
<table border="1"> <tr> <td style="text-align: center;">6:0</td> <td>Memory Object Control State</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> <tr> <td colspan="2"> <p>Specifies the memory object control state for this surface.</p> </td> </tr> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> <tr> <td colspan="2"> <p>Bit 2 is not programmable and is always zero.</p> </td> </tr> </table>	6:0	Memory Object Control State	Project:	BDW	Format:	MEMORY_OBJECT_CONTROL_STATE	<p>Specifies the memory object control state for this surface.</p>		Programming Notes		<p>Bit 2 is not programmable and is always zero.</p>						
6:0	Memory Object Control State																
Project:	BDW																
Format:	MEMORY_OBJECT_CONTROL_STATE																
<p>Specifies the memory object control state for this surface.</p>																	
Programming Notes																	
<p>Bit 2 is not programmable and is always zero.</p>																	
3 Project: BDW	<table border="1"> <tr> <td style="text-align: center;">31:12</td> <td>Gather Pool Buffer Size</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U20</td> </tr> <tr> <td colspan="2"> <p>This field specifies the size of the buffer in 4K pages. Any access which straddle or go past the end of the buffer will return undefined data. Note that BufferSize=0 indicates that there is no valid data in the buffer.</p> </td> </tr> <tr> <td colspan="2" style="text-align: center;">Value</td> </tr> <tr> <td colspan="2"> <p>[0,1048575]</p> </td> </tr> <tr> <td colspan="2" style="text-align: center;">Restriction</td> </tr> <tr> <td colspan="2"> <p>Programming size of zero is illegal in the case that the pool is enabled.</p> </td> </tr> </table>	31:12	Gather Pool Buffer Size	Project:	BDW	Format:	U20	<p>This field specifies the size of the buffer in 4K pages. Any access which straddle or go past the end of the buffer will return undefined data. Note that BufferSize=0 indicates that there is no valid data in the buffer.</p>		Value		<p>[0,1048575]</p>		Restriction		<p>Programming size of zero is illegal in the case that the pool is enabled.</p>	
	31:12	Gather Pool Buffer Size															
	Project:	BDW															
Format:	U20																
<p>This field specifies the size of the buffer in 4K pages. Any access which straddle or go past the end of the buffer will return undefined data. Note that BufferSize=0 indicates that there is no valid data in the buffer.</p>																	
Value																	
<p>[0,1048575]</p>																	
Restriction																	
<p>Programming size of zero is illegal in the case that the pool is enabled.</p>																	
<table border="1"> <tr> <td style="text-align: center;">11:0</td> <td>Reserved</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	11:0	Reserved	Project:	BDW	Format:	MBZ											
11:0	Reserved																
Project:	BDW																
Format:	MBZ																

3DSTATE_GS

3DSTATE_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Controls the GS stage hardware.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		11h 3DSTATE_GS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	8h Excludes DWord (0,1)	
	Format:	=n	
1..2	63:6	Kernel Start Pointer	
		Project:	All
		Format:	InstructionBaseOffset[63:6]Kernel
	This field specifies the starting location (1st GEN4 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	
Project:		All	
Format:		MBZ	

3DSTATE_GS																								
3	31	<p>Single Program Flow</p> <p>Specifies the initial condition of the kernel program as either a single program flow (SIMDn_{xm} with m = 1) or as multiple program flows (SIMDn_{xm} with m > 1). See CR0 description in ISA Execution Environment.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">Disable</td> <td>Single Program Flow disabled</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">Enable</td> <td>Single Program Flow enabled</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	Single Program Flow disabled	1h	Enable	Single Program Flow enabled													
	Value	Name	Description																					
	0h	Disable	Single Program Flow disabled																					
1h	Enable	Single Program Flow enabled																						
30	<p>Vector Mask Enable</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>U1 Enumerated Type</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">Dmask</td> <td>The EU will use the Dispatch Mask (supplied by the GS stage) for instruction execution.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">Vmask</td> <td>The EU will use the Vector Mask (derived from Dispatch Mask) for instruction execution.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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 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).</p>	Format:	U1 Enumerated Type	Value	Name	Description	0h	Dmask	The EU will use the Dispatch Mask (supplied by the GS stage) for instruction execution.	1h	Vmask	The EU will use the Vector Mask (derived from Dispatch Mask) for instruction execution.												
Format:	U1 Enumerated Type																							
Value	Name	Description																						
0h	Dmask	The EU will use the Dispatch Mask (supplied by the GS stage) for instruction execution.																						
1h	Vmask	The EU will use the Vector Mask (derived from Dispatch Mask) for instruction execution.																						
29:27	<p>Sampler Count</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>U3</td> </tr> </table> <p>Specifies how many samplers (in multiples of 4) the geometry shader kernel uses. Used only for prefetching the associated sampler state entries.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">No Samplers</td> <td>No Samplers used</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">1-4 Samplers</td> <td>Between 1 and 4 samplers used</td> </tr> <tr> <td style="text-align: center;">2h</td> <td style="text-align: center;">5-8 Samplers</td> <td>Between 5 and 8 samplers used</td> </tr> <tr> <td style="text-align: center;">3h</td> <td style="text-align: center;">9-12 Samplers</td> <td>Between 9 and 12 samplers used</td> </tr> <tr> <td style="text-align: center;">4h</td> <td style="text-align: center;">13-16 Samplers</td> <td>Between 13 and 16 samplers used</td> </tr> <tr> <td style="text-align: center;">5h-7h</td> <td style="text-align: center;">Reserved</td> <td></td> </tr> </tbody> </table>	Format:	U3	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	
Format:	U3																							
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	<p>Reserved</p>																							

3DSTATE_GS		
25:18	Binding Table Entry Count	
	Format:	U8
	<p>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.</p>	
	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	
	Project:	All
	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	
	Format:	MBZ
13	Illegal Opcode Exception Enable	
	Format:	Enable
	This bit gets loaded into EU CR0.1[12] (note the bit # difference). See <i>Exceptions and ISA Execution Environment</i> .	
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 <i>Exceptions and ISA Execution Environment</i> .	

3DSTATE_GS							
	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 <i>Exceptions and ISA Execution Environment</i> .						
	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).						
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[1,32]</td> <td></td> </tr> </tbody> </table>	Value	Name	[1,32]		
Value	Name						
[1,32]							
4..5	63:10	Scratch Space Base Pointer					
	Format: GeneralStateOffset[63:10]ScratchSpace 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.						
	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.						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%; text-align: center;">Value</th> <th style="width: 25%; text-align: center;">Name</th> <th style="width: 50%; text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,11]</td> <td></td> <td>indicating [1K Bytes, 2M Bytes]</td> </tr> </tbody> </table>		Value	Name	Description	[0,11]	
Value	Name	Description					
[0,11]		indicating [1K Bytes, 2M Bytes]					
6	31	Reserved					
		Project: All Format: MBZ					

3DSTATE_GS					
30:29	Reserved				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
28:23	Output Vertex Size				
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table>	Project:	All	Format:	U6
	Project:	All			
	Format:	U6			
<p>[0,63] indicating [1,64] 16B units</p> <p>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).</p>					
<table border="1"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> <tr> <td> <p>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.</p> </td> </tr> </table>	Programming Notes	<p>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.</p>			
Programming Notes					
<p>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.</p>					
22:17	Output Topology				
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3DPrimType</td> </tr> </table>	Project:	All	Format:	3DPrimType
	Project:	All			
Format:	3DPrimType				
<p>This field specifies the topology type (3DPrimType) to be associated with GS-thread output vertices (if any).</p>					
16:11	Vertex URB Entry Read Length				
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table>	Project:	All		
	Project:	All			
<p>Specifies the amount of URB data read and passed in the thread payload for each Vertex URB entry, in 256-bit register increments.</p> <table border="1"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> <tr> <td> <p>Programming Restriction: This field must be a non-zero value if Include Vertex Handles is cleared to zero.</p> </td> </tr> </table>	Programming Notes	<p>Programming Restriction: This field must be a non-zero value if Include Vertex Handles is cleared to zero.</p>			
Programming Notes					
<p>Programming Restriction: This field must be a non-zero value if Include Vertex Handles is cleared to zero.</p>					
10	Include Vertex Handles				
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Boolean</td> </tr> </table>	Project:	All	Format:	Boolean
	Project:	All			
	Format:	Boolean			
<p>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.</p>					
<table border="1"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> <tr> <td> <p>Programming Restriction: This field must be set if Vertex URB Entry Read Length is cleared to zero.</p> </td> </tr> </table>	Programming Notes	<p>Programming Restriction: This field must be set if Vertex URB Entry Read Length is cleared to zero.</p>			
Programming Notes					
<p>Programming Restriction: This field must be set if Vertex URB Entry Read Length is cleared to zero.</p>					

3DSTATE_GS											
9:4	<p>Vertex URB Entry Read Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>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.</p>	Project:	All	Format:	U6						
	Project:	All									
Format:	U6										
3:0	<p>Dispatch GRF Start Register For URB Data</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>Specifies the starting GRF register number for the URB portion (Constant + Vertices) of the thread payload.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,15]</td> <td></td> <td>indicating GRF [R0, R15]</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center; color: blue; font-weight: bold;">Programming Notes</p> <p>If Include Vertex Handles is enabled (pull or hybrid handles case), then For simd4x2: For DUAL_OBJECT dispatch mode this field should be: $((2 * \text{numVerticesPerObject}) + 8 - 1) / 8 + 1$ For SINGLE and DUAL_INSTANCE dispatch modes this field should be: $(\text{numVerticesPerObject} + 8 - 1) / 8 + 1$ If Include Primitive ID is set, then add 1 to the value obtained by using the above</p> <p>For simd8: For InstanceCount == 1: numVerticesPerObject + 2 For InstanceCount > 1: 3</p> </div>	Project:	All	Format:	U4	Value	Name	Description	[0,15]		indicating GRF [R0, R15]
	Project:	All									
Format:	U4										
Value	Name	Description									
[0,15]		indicating GRF [R0, R15]									
7	<p>31:24 Maximum Number of Threads</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U8/2-1 Thread Count</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>[3,251]</td> <td></td> <td>indicating thread count of [8,504]</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center; color: blue; font-weight: bold;">Programming Notes</p> <p>Note that this "Maximum Number of Threads" field is different from the other FF stages in that only an even number of threads.</p> </div>	Project:	BDW	Format:	U8/2-1 Thread Count	Value	Name	Description	[3,251]		indicating thread count of [8,504]
Project:	BDW										
Format:	U8/2-1 Thread Count										
Value	Name	Description									
[3,251]		indicating thread count of [8,504]									



3DSTATE_GS									
23:20	<p>Control Data Header Size</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>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. Software must ensure that the Control Data Header Size is sufficient to accommodate the maximum number of vertices output by the GS thread. It is UNDEFINED for a GS thread to report more output vertices than can be accommodated in a non-zero-sized header. (If the header size is zero, by definition neither cut nor StreamID bits are defined.)</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,8]</td> <td>32B Units</td> </tr> </tbody> </table>	Format:	U4	Value	Name	[0,8]	32B Units		
Format:	U4								
Value	Name								
[0,8]	32B Units								
19:15	<p>Instance Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td style="width: 60%;">U5-1 #Instances</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> <td>Indicating [1,31] instances</td> </tr> </tbody> </table>	Format:	U5-1 #Instances	Value	Name	Description	[0,31]		Indicating [1,31] instances
Format:	U5-1 #Instances								
Value	Name	Description							
[0,31]		Indicating [1,31] instances							
14:13	<p>Default Stream Id</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U2</td> </tr> </table> <p>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.</p>	Format:	U2						
Format:	U2								

3DSTATE_GS

		Dispatch Mode	
	Format:	U2	
	This field specifies how the GS unit dispatches multiple instances and/or multiple objects.		
	Value	Name	Description
	0h	Single	Each thread shades a single instance of one object.
	1h	Dual Instance	Each thread shades possibly two instances of one object. If the InstanceCount is odd, a trailing dispatch of only one instance will be made for each object received. Not recommended if InstanceCount = 1, assuming a kernel optimized for SINGLE or DUAL_OBJECT dispatch would outperform a kernel compiled for DUAL_INSTANCE but only passed one instance.
	2h	Dual Object	Each thread shades one instance of possibly two objects. The GS unit attempt to pair objects together into one dispatch, but under some circumstances only one object may be dispatched (as controlled by the DispatchMask generated by the GS unit). Not valid for objects with more than 16 vertices per object. Not valid if InstanceCount > 1 (more than one instance per object).
	3h	SIMD8	Each thread shades up to 8 different objects or (if InstanceCount > 1) 8 instances of a single object.
			Not valid for objects with more than 6 vertices per object.
	Programming Notes		
	The GS must be allocated at least two URB handles or behavior is UNDEFINED for Dual Instance or Dual Object mode.		
	At least 8 URB entries must be allocated in order to use SIMD8 DispatchMode.		
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



3DSTATE_GS											
9:5	Invocations Increment Value										
	Format:	U5									
	<p>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.</p>										
	<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,31]</td> <td></td> <td>indicating an increment of [1,32]</td> </tr> </tbody> </table>	Value	Name	Description	[0,31]		indicating an increment of [1,32]				
Value	Name	Description									
[0,31]		indicating an increment of [1,32]									
4	Include Primitive ID										
	Format:	Boolean									
<p>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.</p>											
3	Hint										
	Format:	U1									
<p>This state bit is simply passed in GS thread payloads for use by the GS kernel - it has no other impact on hardware operation.</p>											
2	Reorder Mode										
	<p>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).</p>										
	<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">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 style="text-align: center;">1h</td> <td style="text-align: center;">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> </tbody> </table>	Value	Name	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.	
	Value	Name	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.									
Programming Notes											
<p>Workaround: reorder mode must be set to REORDER_LEADING and reordering must be done in the Geometry shader.</p>											

3DSTATE_GS													
	1	<p>Discard Adjacency</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">Enable</td> </tr> </table> <p>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).</p> <p>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.</p>	Format:	Enable									
	Format:	Enable											
0	<p>Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">Enable</td> </tr> </table> <p>Specifies whether the GS stage is enabled or disabled (pass-through).</p>	Format:	Enable										
Format:	Enable												
8	31	<p>Control Data Format</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U1</td> </tr> </table> <p>This field specifies the format of the control data header (if any).</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <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 POINTLIST, or behavior is UNDEFINED.</td> </tr> </tbody> </table>	Format:	U1	Value	Name	Description	0h	CUT	The control data header contains cut bits.	1h	SID	The control data header contains StreamID bits. . Output Topology must be set to POINTLIST, or behavior is UNDEFINED.
Format:	U1												
Value	Name	Description											
0h	CUT	The control data header contains cut bits.											
1h	SID	The control data header contains StreamID bits. . Output Topology must be set to POINTLIST, or behavior is UNDEFINED.											
	30	<p>Static Output</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">Enable</td> </tr> </table> <p>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 at the very beginning of the output URB entry (see GS URB Entry section below).</p>	Format:	Enable									
Format:	Enable												
	29:27	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ									
Format:	MBZ												

3DSTATE_GS								
	<p>26:16 Static Output Vertex Count</p> <table border="1"> <tr> <td>Format:</td> <td>U11 Count of object vertices</td> </tr> </table> <p>If GSEnable is set and StaticOutput is set, this field specifies the total number of vertices output each GS shader invocation. If GSEnable is set and StaticOutput is clear (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. This field is then ignored. (See GS URB Entry below).</p>	Format:	U11 Count of object vertices					
	Format:	U11 Count of object vertices						
	<p>15:9 Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
<p>8:0 Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
Project:	BDW							
Format:	MBZ							
9	<p>31:28 Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ						
	<p>27 Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW					
	Project:	BDW						
	<p>26:21 Vertex URB Entry Output Read Offset</p> <table border="1"> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB by SBE.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,63]</td> <td></td> </tr> </tbody> </table>	Format:	U6	Value	Name	[0,63]		
Format:	U6							
Value	Name							
[0,63]								
<p>20:16 Vertex URB Entry Output Length</p> <table border="1"> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>Specifies the amount of URB data written for each Vertex URB entry, in 256-bit register increments.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,16]</td> <td></td> </tr> </tbody> </table> <table border="1"> <tr> <th style="text-align: center;">Programming Notes</th> </tr> <tr> <td>This length does not include the vertex header.</td> </tr> </table>	Format:	U5	Value	Name	[1,16]		Programming Notes	This length does not include the vertex header.
Format:	U5							
Value	Name							
[1,16]								
Programming Notes								
This length does not include the vertex header.								
<p>15:8 User Clip Distance Clip Test Enable Bitmask</p> <table border="1"> <tr> <td>Format:</td> <td>Enable[8]</td> </tr> </table> <p>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.</p>	Format:	Enable[8]						
Format:	Enable[8]							

3DSTATE_GS			
7:0	<p>User Clip Distance Cull Test Enable Bitmask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td style="width: 50%;">Enable[8]</td> </tr> </table> <p>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.</p>	Format:	Enable[8]
Format:	Enable[8]		



3DSTATE_HIER_DEPTH_BUFFER

3DSTATE_HIER_DEPTH_BUFFER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description			
<p>This command sets the surface state of the hierarchical depth buffer, delivered as a pipelined state command. However, the state change pipelining isn't completely transparent (see restriction below).</p> <p>WM HW will internally manage the draining pipe and flushing of the caches when this command is issued. The PIPE_CONTROL restrictions are removed.</p>			
Programming Notes			
<p>Restriction: Prior to changing Depth/Stencil Buffer state (i.e., any combination of 3DSTATE_DEPTH_BUFFER, 3DSTATE_CLEAR_PARAMS, 3DSTATE_STENCIL_BUFFER, 3DSTATE_HIER_DEPTH_BUFFER) SW must first issue a pipelined depth stall (PIPE_CONTROL with Depth Stall bit set, followed by a pipelined depth cache flush (PIPE_CONTROL with Depth Flush Bit set, followed by another pipelined depth stall (PIPE_CONTROL with Depth Stall Bit set), unless SW can otherwise guarantee that the pipeline from WM onwards is already flushed (e.g., via a preceding MI_FLUSH).</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	07h 3DSTATE_HIER_DEPTH_BUFFER
		Format:	OpCode
15:8	Reserved		
	Format:	MBZ	

3DSTATE_HIER_DEPTH_BUFFER									
	7:0	<p>Dword Length</p> <table border="1"> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>3h</td> <td>Excludes Dword (0,1) [Default]</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	3h	Excludes Dword (0,1) [Default]	
Format:	=n Total Length - 2								
Value	Name								
3h	Excludes Dword (0,1) [Default]								
1	31:25	<p>Hierarchical Depth Buffer Object Control State</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state for the hierarchical depth buffer.</p>	Project:	BDW	Format:	MEMORY_OBJECT_CONTROL_STATE			
	Project:	BDW							
	Format:	MEMORY_OBJECT_CONTROL_STATE							
	24:23	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ			
	Project:	BDW							
Format:	MBZ								
22	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
Project:	BDW								
Format:	MBZ								
21:17	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								
16:0	<p>Surface Pitch</p> <table border="1"> <tr> <td>Format:</td> <td>U17-1 Pitch in Bytes</td> </tr> </table> <p>This field specifies the pitch of the hierarchical depth buffer in (#Bytes - 1).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[127, 1FFFFh]</td> <td>corresponding to [128B, 128KB] also restricted to a multiple of 128B</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td>Since this surface is tiled, the pitch specified must be a multiple of the tile pitch, in the range [128B, 128KB].</td> </tr> </tbody> </table>	Format:	U17-1 Pitch in Bytes	Value	Name	[127, 1FFFFh]	corresponding to [128B, 128KB] also restricted to a multiple of 128B	Programming Notes	Since this surface is tiled, the pitch specified must be a multiple of the tile pitch, in the range [128B, 128KB].
Format:	U17-1 Pitch in Bytes								
Value	Name								
[127, 1FFFFh]	corresponding to [128B, 128KB] also restricted to a multiple of 128B								
Programming Notes									
Since this surface is tiled, the pitch specified must be a multiple of the tile pitch, in the range [128B, 128KB].									
2..3 Project: BDW	63:0	<p>Surface Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:0]HierarchicalDepthBuffer</td> </tr> </table> <p>This field specifies the address of the buffer in Graphics Memory.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td>The Hierarchical Depth Buffer can only be mapped to Main Memory (uncached).</td> </tr> </tbody> </table>	Project:	BDW	Format:	GraphicsAddress[63:0]HierarchicalDepthBuffer	Programming Notes	The Hierarchical Depth Buffer can only be mapped to Main Memory (uncached).	
Project:	BDW								
Format:	GraphicsAddress[63:0]HierarchicalDepthBuffer								
Programming Notes									
The Hierarchical Depth Buffer can only be mapped to Main Memory (uncached).									
4 Project: BDW	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ			
Project:	BDW								
Format:	MBZ								



3DSTATE_HIER_DEPTH_BUFFER	
14:0	Surface QPitch
	Project: BDW
	Format: QPitch[16:2]
	Description
	<p>This field specifies the distance in rows between array slices. It is used only in the following cases:</p> <ul style="list-style-type: none"> • Surface Array is enabled <i>OR</i> • Number of Multisamples is not NUMSAMPLES_1 and Multisampled Surface Storage Format set to MSFMT_MSS <i>OR</i> • Surface Type is SURFTYPE_CUBE
Value	Name
[4h, 1FFFCh]	

3DSTATE_HS

3DSTATE_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Controls the HS stage hardware.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		1Bh 3DSTATE_HS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n	
	Value	Name	
	7	Excludes DWord (0,1) [Default]	
1	31:30	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_HS			
29:27	Sampler Count		
	Project:	All	
	Format:	U3	
	Specifies how many samplers (in multiples of 4) the HS kernels use. 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	Reserved	
26	Reserved		
	Project:	All	
	Format:	MBZ	
25:18	Binding Table Entry Count		
	Project:	All	
	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.		
	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		
	Project:	BDW	
	Specifies the priority of the thread for dispatch		
	Value	Name	Description
	0h	Normal	Normal Priority
1h	High	High Priority	
16	Floating Point Mode		
	Project:	All	
	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	

3DSTATE_HS				
	15:14	Reserved	Project: All	Format: MBZ
	13	Illegal Opcode Exception Enable	Project: All	Format: Enable
	This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.			
	12	Software Exception Enable	Project: BDW	Format: Enable
	This bit gets loaded into EU CR01[13] (note the bit # difference). See Exceptions and ISA Execution Environment.			
	11:8	Reserved	Project: All	Format: MBZ
	7:0	Reserved	Project: BDW	Format: MBZ
	2	31	Enable	Project: All
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.				
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	Project: All	Format: Enable
This bit controls whether HS-unit-specific statistics register(s) will increment (for each patch).				



3DSTATE_HS		
28:27	Reserved	
	Project:	BDW
	Format:	MBZ
26:18	Reserved	
	Project:	All
	Format:	MBZ
17	Reserved	
	Project:	BDW
	Format:	MBZ
16:8	Maximum Number of Threads	
	Project:	BDW
	Format:	U9-1
<p>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.</p>		
	Value	Name
	[0,503]	indicating thread count of [1,504]
7:4	Reserved	
	Project:	All
	Format:	MBZ
3:0	Instance Count	
	Format:	U4-1
<p>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.</p>		
	Value	Description
	[0,15]	representing [1,16] instances
Programming Notes		
<p>A pipe_control with cs stall must be sent whenever the HS_STATE.InstanceCount changes from 0 (no instancing) to >0 (instancing) or when there is transition from HS_STATE.Enabled = false to (HS_STATE.Enabled = true && InstanceCount > 0).</p>		

3DSTATE_HS								
3.4 Project: BDW	63:6	Kernel Start Pointer						
		Project: BDW						
	Format: InstructionBaseOffset[63:6]Kernel							
	This field specifies the starting location (1st GEN 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							
	Project: BDW							
	Format: MBZ							
5.6 Project: BDW	63:10	Scratch Space Base Pointer						
		Project: BDW						
	Format: GeneralStateOffset[63:10]							
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> <td>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.</td> </tr> </tbody> </table>		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.
	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							
	Project: BDW							
Format: MBZ								
3:0	Per-Thread Scratch Space							
	Project: BDW							
	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.							
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td></td> <td>Indicating[1K Bytes, 2M Bytes</td> </tr> </tbody> </table>		Value	Name	Description	[0,11]		Indicating[1K Bytes, 2M Bytes	
Value	Name	Description						
[0,11]		Indicating[1K Bytes, 2M Bytes						
7 Project: BDW	31:29	Reserved						
		Project: BDW						
	Format: MBZ							
	28	Reserved						
Project: BDW								
Format: MBZ								

3DSTATE_HS			
27	Single Program Flow		
	Project:	BDW	
	Format:	Enable	
	Specifies the initial condition of the kernel program as either a single program flow (SIMDn _{xm} with m = 1) or as multiple program flows (SIMDn _{xm} with m > 1). See CR0 description in <i>ISA Execution Environment</i> .		
	Value	Name	Description
	0h	Reserved	
	1h	Enable	Single Program Flow Enabled
	26	Vector Mask Enable	
		Project:	BDW
		Format:	U1 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 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	
	Project:	BDW	
	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.		

3DSTATE_HS		
24	Include Vertex Handles	
	Project:	BDW
	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	
	Project:	BDW
	Format:	U5
	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.	
	Value	Name
[0,31]		indicating GRF [R0, R31]
18:17	Reserved	
	Project:	BDW
	Format:	MBZ
16:11	Vertex URB Entry Read Length	
	Project:	BDW
	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	
	Project:	BDW
	Format:	MBZ



3DSTATE_HS			
	9:4	Vertex URB Entry Read Offset	
		Project: BDW	
		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	
		Project: BDW	
		Format: MBZ	
	0	Reserved	
Project: BDW			
Format: MBZ			
8 Project: BDW	31:0	Reserved	
		Project: BDW	
		Format: MBZ	

3DSTATE_INDEX_BUFFER

3DSTATE_INDEX_BUFFER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>This command is used to specify the current IB state used by the VF function. At most one IB is defined and active at any given time. NOTES: The IB must be specified before any RANDOM 3D_PRIMITIVE commands are issued It is possible to have vertex elements source completely from generated ID values and therefore not require any Index Buffer accesses. In this case, VF function will simply ignore the Index Buffer state.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		0Ah 3DSTATE_INDEX_BUFFER	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	3h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:11	Reserved	
		Project:	All
		Format:	MBZ
	10	Reserved	
		Project:	BDW
	Format:	MBZ	

3DSTATE_INDEX_BUFFER			
	9:8	Index Format	
		Project: All	
		Format: U2 Enumerated type	
		This field specifies the data format of the index buffer. All index values are UNSIGNED.	
		Value	Name
		Project	
	0h	BYTE	All
	1h	WORD	All
	2h	DWORD	All
	7	Reserved	
Project: All			
Format: MBZ			
6:0	Memory Object Control State		
	Project: All		
	Format: MEMORY_OBJECT_CONTROL_STATE		
		Specifies the memory object control state for this index buffer.	
2..3	63:0	Buffer Starting Address	
		Project: All	
		Format: GraphicsAddress[63:0]Index_Buffer_Entry	
		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 base address to the starting structure within the buffer.	
		Programming Notes	
		Index Buffers can only be allocated in linear (not tiled) graphics memory.	
4	31:0	Buffer Size	
		Project: All	
		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_LINE_STIPPLE

3DSTATE_LINE_STIPPLE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_LINE_STIPPLE command is used to specify state variables used in the Line Stipple function.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		08h 3DSTATE_LINE_STIPPLE	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	1h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31	Modify Enable (Current Repeat Counter, Current Stipple Index)	
		Project:	All
		Format:	Enable
	Modify enable for Current Repeat Counter and Current Stipple Index fields.		
	Programming Notes		
	It is provided only for HW-generated commands as part of context save/restore. SW must initialize the current repeat counter, current stipple count fields if it sets this bit to enable. SW must set this bit to reset the stipple count.		



3DSTATE_LINE_STIPPLE				
	30	Reserved	Project: All	Format: MBZ
	29:21	Current Repeat Counter	Project: All	Format: U9
	This field sets the HW-internal repeat counter state. SW must initialize it to 1 if the modify enable is set.			
	20	Reserved	Project: All	Format: MBZ
	19:16	Current Stipple Index	Project: All	Format: U4
This field sets the HW-internal stipple pattern index. SW must initialize it to 0 if the modify enable is set.				
	15:0	Line Stipple Pattern	Project: All	Format: 16 bit mask Bit 15 = most significant bit, Bit 0 = least significant bit
	Specifies a pattern used to mask out bit specific pixels while rendering lines.			
2	31:15	Line Stipple Inverse Repeat Count	Project: All	Format: U1.16
	Range: [0.00390625, 1.0]			
	Specifies the inverse (truncated) of the repeat count for the line stipple function.			
	14:9	Reserved	Project: All	Format: MBZ
	8:0	Line Stipple Repeat Count	Project: All	Format: U9
Specifies the repeat count for the line stipple function.				
		Value	Name	
		[1, 256]		

3DSTATE_MONOFILTER_SIZE

3DSTATE_MONOFILTER_SIZE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This state specifies the size of the filter which is used when filtering in MAPFILTER_MONO mode.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	11h 3DSTATE_MONOFILTER_SIZE
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31:6	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_MONOFILTER_SIZE									
5:3	<p>Monochrome Filter Width</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> </table> <p>This field specifies the width of the monochrome filter. It is ignored if the monochrome filter is not enabled.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,7]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U3	Value	Name	[1,7]	
	Project:	All							
Format:	U3								
Value	Name								
[1,7]									
2:0	<p>Monochrome Filter Height</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> </table> <p>This field specifies the height of the monochrome filter. It is ignored if the monochrome filter is not enabled.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,7]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U3	Value	Name	[1,7]	
	Project:	All							
Format:	U3								
Value	Name								
[1,7]									

3DSTATE_MULTISAMPLE

3DSTATE_MULTISAMPLE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_MULTISAMPLE command is used to specify multisample state associated with the current render target/depth buffer.			
Programming Notes			
It is illegal to render to surfaces with multiple different values of the state fields in this command.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		0Dh 3DSTATE_MULTISAMPLE	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
	Dword Length		
7:0	Default Value:	0h	
	Project:	All	
	Format:	=n Total Length - 2	
	Excludes Dword (0,1)		
1	31:6	Reserved	
		Project:	All
		Format:	MBZ



3DSTATE_MULTISAMPLE		
5	Pixel Position Offset Enable	
	Project:	BDW
	Format:	U1
	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.	
4	Pixel Location	
	Project:	All
	Format:	U1
	This field specifies where the device evaluates "pixel" (vs. centroid or sample) values/attributes.	
	Value	Name
		Description
	0h	CENTER Use the pixel center (0.5, 0.5 offset)
	1h	UL_CORNER Use the pixel upper-left corner
	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.	BDW

3DSTATE_MULTISAMPLE				
3:1	Number of Multisamples			
	Project:	All		
	Format:	U3		
	This field specifies how many samples/pixel exist in all RTs and the Depth Buffer, as $\log_2(\#samples)$. This field is valid regardless of the setting of Multisample Rasterization Mode .			
	Value	Name	Description	Project
	0h	1	1 sample/pixel	All
	1h	2	2 samples/pixel	BDW
	2h	4	4 samples/pixel	All
	3h	8	8 samples/pixel	BDW
	5h-7h	Reserved		All
Programming Notes			Project	
The setting of this field must match the Number of Multisamples field in SURFACE_STATE of all bound render targets.				
0	Reserved			
	Project:	All		
	Format:	MBZ		

3DSTATE_POLY_STIPPLE_OFFSET

3DSTATE_POLY_STIPPLE_OFFSET			
Project:	DevBWR+		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_POLY_STIPPLE_OFFSET command is used to specify the origin of the repeated screen-space Polygon Stipple Pattern as an X, Y offset from the Color Buffer origin.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
23:16	3D Command Sub Opcode		
	Default Value:	06h 3DSTATE_POLY_STIPPLE_OFFSET	
15:8	Reserved		
	Project:	All	
7:0	Dword Length		
	Default Value:	0h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:13	Reserved	
		Project:	All
	Format:	MBZ	
	12:8	Polygon Stipple X Offset	
		Project:	All
Format:		U5	
Specifies a 5 bit x address offset in the poly stipple pattern			
	Value	Name	
	[0,31]		

3DSTATE_POLY_STIPPLE_OFFSET										
	7:5	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ				
	Project:	All								
Format:	MBZ									
	4:0	Polygon Stipple Y Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>Specifies a 5 bit y address offset in the poly stipple pattern</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U5	Value	Name	[0,31]	
Project:	All									
Format:	U5									
Value	Name									
[0,31]										

3DSTATE_POLY_STIPPLE_PATTERN

3DSTATE_POLY_STIPPLE_PATTERN			
Project:	DevBWR+		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_POLY_STIPPLE_PATTERN command is used to specify the 32x32 Polygon Stipple Pattern used in the Polygon Stipple function of the WM unit.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	07h 3DSTATE_POLY_STIPPLE_PATTERN
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	1Fh Excludes Dword (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1..32	31:0	Pattern Row	
		Project:	All
		Format:	32 bit mask Bit 31 = upper left corner, Bit 0 = upper right corner of first row. Specifies a pattern used by Polygon Stipple to mask out specific pixels of every 32x32 area rendered.

3DSTATE_PS_BLEND

3DSTATE_PS_BLEND						
Project:	BDW					
Source:	RenderCS					
Length Bias:	2					
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value: 3h GFXPIPE Format: OpCode				
	28:27	Command SubType				
		Default Value: 3h GFXPIPE_3D Format: OpCode				
	26:24	3D Command Opcode				
		Default Value: 0h 3DSTATE_PIPELINED Format: OpCode				
	23:16	3D Command Sub Opcode				
		Default Value: 4Dh 3DSTATE_PS_BLEND Format: OpCode				
	15:8	Reserved				
		Project: All Format: MBZ				
7:0	DWord Length					
	Format: =n					
	Total Length - 2					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>[Default]</td> <td>Excludes Dword (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	0h	[Default]
Value	Name	Description				
0h	[Default]	Excludes Dword (0,1)				
1	31	Alpha To Coverage Enable				
		Project: All 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				
		Project: All Format: Enable When set indicates the there is at least one non-null RT w/ at least one channel write enabled				

3DSTATE_PS_BLEND					
29	<p>Color Buffer Blend Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When set indicates that RT[0] has color buffer blend enabled.</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
28:24	<p>Source Alpha Blend Factor</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Color_Buffer_Blend_Factor</td> </tr> </table> <p>Indicates the "source factor" in alpha Color Buffer Blending stage for RT[0]</p>	Project:	All	Format:	3D_Color_Buffer_Blend_Factor
Project:	All				
Format:	3D_Color_Buffer_Blend_Factor				
23:19	<p>Destination Alpha Blend Factor</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Color_Buffer_Blend_Factor</td> </tr> </table> <p>Indicates the "destination factor" in alpha Color Buffer Blending stage for RT[0]</p>	Project:	All	Format:	3D_Color_Buffer_Blend_Factor
Project:	All				
Format:	3D_Color_Buffer_Blend_Factor				
18:14	<p>Source Blend Factor</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Color_Buffer_Blend_Factor</td> </tr> </table> <p>Indicates the "source factor" in Color Buffer Blending stage for RT[0]</p>	Project:	All	Format:	3D_Color_Buffer_Blend_Factor
Project:	All				
Format:	3D_Color_Buffer_Blend_Factor				
13:9	<p>Destination Blend Factor</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Color_Buffer_Blend_Factor</td> </tr> </table> <p>Indicates the "destination factor" in Color Buffer Blending stage for RT[0]</p>	Project:	All	Format:	3D_Color_Buffer_Blend_Factor
Project:	All				
Format:	3D_Color_Buffer_Blend_Factor				
8	<p>Alpha Test Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Indicates the AlphaTestEnable for RT[0]</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
7	<p>Independent Alpha Blend Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				

3DSTATE_PS_BLEND		
	6:0	Reserved
		Project: All
		Format: MBZ

3DSTATE_PS

3DSTATE_PS			
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		20h 3DSTATE_PS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0Ah Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1..2	63:6	Kernel Start Pointer 0	
		Project:	All
		Format:	InstructionBaseOffset[63:6]Kernel
Specifies the 64-byte aligned address offset of the first instruction in the kernel[0]. This pointer is relative to the Instruction Base Address .			
5:0	5:0	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_PS				
3	31	Single Program Flow		
	Project:		All	
	Single Program Flow (SPF) specifies the initial condition of the kernel program as either a single program flow (SIMDn _{xm} with m = 1) or as multiple program flows (SIMDn _{xm} with m > 1). See CR0 description in ISA Execution Environment.			
	Value	Name	Description	Project
	0h	Multiple	Multiple Program Flows	All
	1h	Single	Single Program Flows	All
	30	Vector Mask Enable		
	Project:		All	
	Format:		U1 Enumerated Type	
	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.			
Value	Name	Description	Project	
0h	Dmask	Channels are enabled based on the dispatch mask	All	
1h	Vmask	Channels are enabled based on the vector mask	All	
29:27	Sampler Count			
Project:		All		
Specifies how many samplers (in multiples of 4) the vertex shader 0 kernel uses. Used only for prefetching the associated sampler state entries.				
Value	Name	Description	Project	
[0,4]				
0h	No Samplers	no samplers used	All	
1h	1-4 Samplers	between 1 and 4 samplers used	All	
2h	5-8 Samplers	between 5 and 8 samplers used	All	
3h	9-12 Samplers	between 9 and 12 samplers used	All	
4h	13-16 Samplers	between 13 and 16 samplers used	All	
5h-7h		Reserved	All	
26	Single Precision Denormal Mode			
Project:		All		
Specifies the single precision denormal mode used by the dispatched thread.				
Value	Name	Description	Project	
0h	Flushed to Zero	Single Precision denormals are flushed to zero	All	
1h	Retained	Single Precision denormals are retained	All	

3DSTATE_PS				
25:18	Binding Table Entry Count			
	Project:		All	
	Description		Project	
	<p>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.</p> <p>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.</p>		BDW	
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			
	Project:		All	
	Specifies the priority of the thread for dispatch.			
	Value	Name	Description	Project
0h	Normal	Normal Priority	All	
1h	High	High Priority	All	
16	Floating Point Mode			
	Project:		All	
	Specifies the floating point mode used by the dispatched thread.			
	Value	Name	Description	Project
0h	IEEE-754	Use IEEE-754 rules	All	
1h	Alternate	Use alternate rules	All	
15:14	Rounding Mode			
	Project:		All	
	Specifies the rounding mode used by the dispatched thread.			
	Value	Name	Description	Project
	0h	RTNE	Round to Nearest Even	All
	1h	RU	Round toward +infinity	All
2h	RD	Round toward -infinity	All	
3h	RTZ	Round toward zero	All	

3DSTATE_PS						
	13	Illegal Opcode Exception Enable	Project: All	Format: Enable	This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.	
	12	Reserved	Project: All	Format: MBZ		
	11	Mask Stack Exception Enable	Project: All	Format: Enable	This bit gets loaded into EU CR0.1[12] (note the bit # difference). See Exceptions and ISA Execution Environment.	
	10:8	Reserved	Project: All	Format: MBZ		
	7	Software Exception Enable	Project: All	Format: Enable	This bit gets loaded into EU CR0.1[13] (note the bit # difference). See Exceptions and ISA Execution Environment.	
	6:0	Reserved	Project: All	Format: MBZ		
	4..5	63:10	Scratch Space Base Pointer	Project: All	Format: GeneralStateOffset[63:10]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.
		9:4	Reserved	Project: All	Format: MBZ	

3DSTATE_PS									
	3:0	Per Thread Scratch Space							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td>indicating [1k bytes, 2M bytes] in powers of two</td> </tr> </tbody> </table>		Project:	All	Format:	U4	Value	Name	[0,11]
Project:	All								
Format:	U4								
Value	Name								
[0,11]	indicating [1k bytes, 2M bytes] in powers of two								
6	31:23	Maximum Number of Threads Per PSD							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U8-2 Representing Thread Count</td> </tr> </table>		Project:	BDW	Format:	U8-2 Representing Thread Count		
		Project:	BDW						
		Format:	U8-2 Representing Thread Count						
		Description							
		Range = [0, 62] --> [2, 64] threads. Specifies the maximum number of simultaneous threads allowed to be active per PSD. Depending on the GT mode register, Number of PSDs in the system determine maximum number PS threads per configuration.							
		Has 2 PSDs therefore actual range for max PS threads is [4,128].							
		Has 3 PSDs therefore actual range for max PS threads is [6,192].							
		Has 6 PSDs therefore actual range for max PS threads is [12,384].							
		Project							
BDW:GT1									
BDW:GT2									
BDW:GT3									
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.									
	22:12	Reserved							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	All	Format:	MBZ			
Project:	All								
Format:	MBZ								
	11	Push Constant Enable							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>		Project:	All	Format:	Enable			
Project:	All								
Format:	Enable								
	10	Reserved							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	All	Format:	MBZ			
Project:	All								
Format:	MBZ								
	9	Reserved							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	BDW	Format:	MBZ			
Project:	BDW								
Format:	MBZ								

3DSTATE_PS		
8	Render Target Fast Clear Enable	
	Project: All	
	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.	
	7	Reserved
		Project: BDW
		Format: MBZ
	6	Render Target Resolve Enable
		Project: BDW
		Format: Enable
		This field is set to enable clear value resolve on non-multisampled render targets. See "Render Target Resolve" for restrictions on enabling this field.
	5	Reserved
Project: BDW		
Format: MBZ		

3DSTATE_PS																				
4:3	<p>Position XY Offset Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2 Enumerated Type</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>POSOFFSET_NONE</td> <td>No Position XY Offsets are included in the PS payload.</td> </tr> <tr> <td>1h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>2h</td> <td>POSOFFSET_CENTROID</td> <td>Position XY Offsets will be passed in the PS payload, and these will reflect the Centroid position(s).</td> </tr> <tr> <td>3h</td> <td>POSOFFSET_SAMPLE</td> <td>Position XY Offsets will be passed in the PS payload, and these will reflect the multisample position(s).</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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</p> <p>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 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).</p> <p>MSDISPMODE_PERSAMPLE is required in order to select POSOFFSET_SAMPLE.</p>	Project:	All	Format:	U2 Enumerated Type	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).
	Project:	All																		
	Format:	U2 Enumerated Type																		
	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).																	
	2	<p>32 Pixel Dispatch Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:	Enable														
		Project:	All																	
Format:		Enable																		
1	<p>16 Pixel Dispatch Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:	Enable															
	Project:	All																		
Format:	Enable																			

3DSTATE_PS					
0	8 Pixel Dispatch Enable				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:	Enable
	Project:	All			
	Format:	Enable			
<table border="1" style="width: 100%;"> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> <tr> <td>When Render Target Fast Clear Enable is ENABLED or Render Target Resolve Type = RESOLVE_PARTIAL or RESOLVE_FULL, this bit must be DISABLED.</td> <td style="text-align: center;">BDW</td> </tr> </table>	Programming Notes	Project	When Render Target Fast Clear Enable is ENABLED or Render Target Resolve Type = RESOLVE_PARTIAL or RESOLVE_FULL, this bit must be DISABLED.	BDW	
Programming Notes	Project				
When Render Target Fast Clear Enable is ENABLED or Render Target Resolve Type = RESOLVE_PARTIAL or RESOLVE_FULL, this bit must be DISABLED.	BDW				
7	31:23 Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All			
	Format:	MBZ			
	22:16 Dispatch GRF Start Register For Constant/Setup Data 0				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U7</td> </tr> </table> <p>Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[0].</p>	Format:	U7		
	Format:	U7			
	<table border="1" style="width: 100%;"> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> <tr> <td style="text-align: center;">[0,127]</td> <td></td> </tr> </table>	Value	Name	[0,127]	
	Value	Name			
	[0,127]				
	15 Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				
14:8 Dispatch GRF Start Register For Constant/Setup Data 1					
<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U7</td> </tr> </table> <p>Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[1].</p>	Project:	All	Format:	U7	
Project:	All				
Format:	U7				
<table border="1" style="width: 100%;"> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> <tr> <td style="text-align: center;">[0,127]</td> <td></td> </tr> </table>	Value	Name	[0,127]		
Value	Name				
[0,127]					
7 Reserved					
<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All				
Format:	MBZ				
6:0 Dispatch GRF Start Register For Constant/Setup Data 2					
<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U7</td> </tr> </table> <p>Specifies the starting GRF register number for the Constant/Setup portion of the thread payload for kernel[2].</p>	Project:	All	Format:	U7	
Project:	All				
Format:	U7				
<table border="1" style="width: 100%;"> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> <tr> <td style="text-align: center;">[0,127]</td> <td></td> </tr> </table>	Value	Name	[0,127]		
Value	Name				
[0,127]					

3DSTATE_PS		
8..9	63:6	Kernel Start Pointer 1 Project: All 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.
		Reserved Project: All Format: MBZ
	5:0	Kernel Start Pointer 2 Project: All 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 .
		Reserved Project: All Format: MBZ
10..11	63:6	Kernel Start Pointer 1 Project: All 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 .
		Reserved Project: All Format: MBZ
	5:0	Kernel Start Pointer 2 Project: All 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 .
		Reserved Project: All Format: MBZ

3DSTATE_PS_EXTRA

3DSTATE_PS_EXTRA			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		4fh 3DSTATE_PS_EXTRA	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31	Pixel Shader Valid	
		Project:	All
		Format:	Enable
	When set indicates a valid pixel shaderWhen this bit clear the rest of this command should also be clear.		
	30	Pixel Shader Does not write to RT	
Project:		All	
Format:		Enable	
When set indicates the pixel shader does not write to render target.			

3DSTATE_PS_EXTRA			
29	oMask Present to Render Target		
	Project:	All	
	Format:	Enable	
<p>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.</p>			
28	Pixel Shader Kills Pixel		
	Project:	All	
	Format:	Enable	
<p>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:</p> <ul style="list-style-type: none"> 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		
	Project:	All	
	Format:	U2 Enumerated Type	
<p>This field specifies the computed depth mode for the pixel shader.</p>			
	Value	Name	Description
	0h	PSCDEPTH_OFF	Pixel shader does not compute depth
	1h	PSCDEPTH_ON	Pixel shader computes depth with no guarantee as to its value
	2h	PSCDEPTH_ON_GE	Pixel shader computes depth and guarantees that oDepth >= SourceDepth
	3h	PSCDEPTH_ON_LE	Pixel shader computes depth and guarantees that oDepth <= SourceDepth
Programming Notes			
<p>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.</p>			
25	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_PS_EXTRA				
24	Pixel Shader Uses Source Depth			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:
Project:	All			
Format:	Enable			
23	Pixel Shader Uses Source W			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:
Project:	All			
Format:	Enable			
22	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
21:18	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
17	Removed			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
16:11	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:
Project:	All			
Format:	MBZ			
10	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
9	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW	
Project:	BDW			
8	Attribute Enable			
	<table border="1"> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Format:	Enable	
Format:	Enable			

3DSTATE_PS_EXTRA							
7	<p>Pixel Shader Disables Alpha To Coverage</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When set indicates the pixel shader AlphaToCoverage should be disabled due to oMask output. The setting of this bit is API dependent.</p>	Project:	All	Format:	Enable		
	Project:	All					
Format:	Enable						
6	<p>Pixel Shader Is Per Sample</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This bit, when ENABLED, indicates that the pixel shader is dispatched at the per sample shading rate. If the bit is DISABLED, the pixel shader is dispatched at the per pixel rate.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td colspan="2" style="text-align: center;">Programming Notes</td> </tr> </table> <p>This bit must NOT be set when PS is used to do clear MSRTs with Fast Clear Optimization Enabled.</p>	Project:	BDW	Format:	Enable	Programming Notes	
	Project:	BDW					
Format:	Enable						
Programming Notes							
5	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
	Project:	BDW					
Format:	MBZ						
4	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
	Project:	BDW					
Format:	MBZ						
3	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
	Project:	BDW					
Format:	MBZ						
2	<p>Pixel Shader Has UAV</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> <tr> <td>Format:</td> <td>U1 Enumerated Type</td> </tr> </table> <p>This field when set indicates that the pixel shader has a UAV attached to it.</p>	Project:	All	Format:	Enable	Format:	U1 Enumerated Type
	Project:	All					
	Format:	Enable					
Format:	U1 Enumerated Type						
1	<p>Pixel Shader Uses Input Coverage Mask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This bit, if ENABLED, indicates that the PS kernel requires the input coverage mask to be passed in the payload.</p>	Project:	BDW	Format:	Enable		
	Project:	BDW					
Format:	Enable						
0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
	Project:	BDW					
Format:	MBZ						

3DSTATE_PUSH_CONSTANT_ALLOC_DS

3DSTATE_PUSH_CONSTANT_ALLOC_DS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
This command sets up the URB configuration for DS Push Constant Buffer.		
Programming Notes		
Programming Restriction: <ul style="list-style-type: none"> The sum of the Constant Buffer Offset and the Constant Buffer Size may not exceed the maximum value of the Constant Buffer Size. The sum of the constant length programmed in 3DSTATE_CONSTANT_DS must be equal or smaller then the size of the allocated space in the URB including the buffering for half cachelines. See Push Constant URB Allocation section for more details. The 3DSTATE_CONSTANT_DS must be reprogrammed prior to the next 3DPRIMITIVE command after programming the 3DSTATE_PUSH_CONSTANT_ALLOC_DS. 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D
	Format: OpCode	
	26:24	3D Command Opcode
		Default Value: 1h 3DSTATE_NONPIPELINED
	Format: OpCode	
	23:16	3D Command Sub Opcode
		Default Value: 14h 3DSTATE_PUSH_CONSTANT_ALLOC_DS
	Format: OpCode	
15:8	Reserved	
	Project: All	
Format: MBZ		
7:0	DWord Length	
	Default Value: 0h Excludes DWord (0,1)	
	Project: All	
Format: =n Total Length - 2		
Programming Notes: Workaround : This command must be followed by a PIPE_CONTROL with CS Stall bit set.		

3DSTATE_PUSH_CONSTANT_ALLOC_DS							
1	31:21	Reserved					
		Project: BDW					
		Format: MBZ					
	20:16	Constant Buffer Offset					
		Project: BDW					
		Format: U5					
		Specifies the offset of the DS constant buffer into the URB.					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,31]</td> <td style="text-align: center;">(0KB - 31KB) Increments of 2KB</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Value	Name	Project	[0,31]	(0KB - 31KB) Increments of 2KB
	Value	Name	Project				
	[0,31]	(0KB - 31KB) Increments of 2KB	BDW				
	15:6	Reserved					
		Project: BDW					
Format: MBZ							
5:0	Constant Buffer Size						
	Project: BDW						
	Format: U6						
	Specifies the size of the DS constant buffer. This value will determine the amount of data the command stream can pre-fetch before the buffer is full. Value of zero is only valid when constants are not enabled for DS.						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,32]</td> <td style="text-align: center;">(0KB - 32KB) Increments of 2KB</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Value	Name	Project	[0,32]	(0KB - 32KB) Increments of 2KB	BDW
Value	Name	Project					
[0,32]	(0KB - 32KB) Increments of 2KB	BDW					

3DSTATE_PUSH_CONSTANT_ALLOC_GS

3DSTATE_PUSH_CONSTANT_ALLOC_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the URB configuration for GS Push Constant Buffer.			
Programming Notes			
<ul style="list-style-type: none"> The sum of the Constant Buffer Offset and the Constant Buffer Size may not exceed the maximum value of the Constant Buffer Size. The sum of the constant length programmed in 3DSTATE_CONSTANT_GS must be equal or smaller than the size of the allocated space in the URB including the buffering for half cachelines. The 3DSTATE_CONSTANT_GS must be reprogrammed prior to the next 3DPRIMITIVE command after programming the 3DSTATE_PUSH_CONSTANT_ALLOC_GS. 			
See Push Constant URB Allocation section for more details.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 3h GFXPIPE Format: OpCode	
	28:27	Command SubType	
		Default Value: 3h GFXPIPE_3D Format: OpCode	
	26:24	3D Command Opcode	
		Default Value: 1h 3DSTATE_NONPIPELINED Format: OpCode	
	23:16	3D Command Sub Opcode	
		Default Value: 15h 3DSTATE_PUSH_CONSTANT_ALLOC_GS Format: OpCode	
	15:8	Reserved	
		Project: All Format: MBZ	
	7:0	DWord Length	Format: =n
			Total Length - 2
Value		Name	Description
0h		3DSTATE_PUSH_CONSTANT_ALLOC_GS [Default]	Excludes DWord (0,1)

3DSTATE_PUSH_CONSTANT_ALLOC_GS												
1	31:21	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
	Format:	MBZ										
	20:16	Constant Buffer Offset <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>Specifies the offset of the GS constant buffer into the URB.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 20%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td>(0KB - 31KB) Increments of 2KB</td> <td>BDW</td> </tr> </tbody> </table>	Project:	BDW	Format:	U5	Value	Name	Project	[0,31]	(0KB - 31KB) Increments of 2KB	BDW
	Project:	BDW										
	Format:	U5										
	Value	Name	Project									
	[0,31]	(0KB - 31KB) Increments of 2KB	BDW									
	15:6	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
	Format:	MBZ										
	5:0	Constant Buffer Size <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>Specifies the size of the GS constant buffer. This value will determine the amount of data the command stream can pre-fetch before the buffer is full. Value of zero is only valid when constants are not enabled for GS.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 20%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,32]</td> <td>(0KB - 32KB) Increments of 2KB</td> <td>BDW</td> </tr> </tbody> </table>	Project:	BDW	Format:	U6	Value	Name	Project	[0,32]	(0KB - 32KB) Increments of 2KB	BDW
Project:	BDW											
Format:	U6											
Value	Name	Project										
[0,32]	(0KB - 32KB) Increments of 2KB	BDW										

3DSTATE_PUSH_CONSTANT_ALLOC_HS

3DSTATE_PUSH_CONSTANT_ALLOC_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the URB configuration for HS Push Constant Buffer.			
Programming Notes			
Programming Restriction: <ul style="list-style-type: none"> The sum of the Constant Buffer Offset and the Constant Buffer Size may not exceed the maximum value of the Constant Buffer Size. The sum of the constant length programmed in 3DSTATE_CONSTANT_HS must be equal or smaller then the size of the allocated space in the URB including the buffering for half cachelines. See Push Constant URB Allocation section for more details. The 3DSTATE_CONSTANT_HS must be reprogrammed prior to the next 3DPRIMITIVE command after programming the 3DSTATE_PUSH_CONSTANT_ALLOC_HS. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	13h 3DSTATE_PUSH_CONSTANT_ALLOC_HS
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	

3DSTATE_PUSH_CONSTANT_ALLOC_HS								
1	31:21	Reserved						
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
	Project:	BDW						
	Format:	MBZ						
	20:16	Constant Buffer Offset						
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table>		Project:	BDW	Format:	U5	
		Project:	BDW					
		Format:	U5					
	Specifies the offset of the HS constant buffer into the URB.							
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td>(0KB - 31KB) Increments of 2KB</td> <td>BDW</td> </tr> </tbody> </table>		Value	Name	Project	[0,31]	(0KB - 31KB) Increments of 2KB	BDW
	Value	Name	Project					
	[0,31]	(0KB - 31KB) Increments of 2KB	BDW					
15:6	Reserved							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	BDW	Format:	MBZ		
Project:	BDW							
Format:	MBZ							
5:0	Constant Buffer Size							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table>		Project:	BDW	Format:	U6		
	Project:	BDW						
	Format:	U6						
Specifies the size of the HS constant buffer. This value will determine the amount of data the command stream can pre-fetch before the buffer is full. Value of zero is only valid when constants are not enabled for HS.								
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,32]</td> <td>(0KB - 32KB) Increments of 2KB</td> <td>BDW</td> </tr> </tbody> </table>		Value	Name	Project	[0,32]	(0KB - 32KB) Increments of 2KB	BDW	
Value	Name	Project						
[0,32]	(0KB - 32KB) Increments of 2KB	BDW						

3DSTATE_PUSH_CONSTANT_ALLOC_PS

3DSTATE_PUSH_CONSTANT_ALLOC_PS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the URB configuration for PS Push Constant Buffer.			
Programming Notes			
Restriction: <ul style="list-style-type: none"> The sum of the Constant Buffer Offset and the Constant Buffer Size may not exceed the maximum value of the Constant Buffer Size. The sum of the constant length programmed in 3DSTATE_CONSTANT_PS must be equal or smaller then the size of the allocated space in the URB including the buffering for half cachelines. See Push Constant URB Allocation section for more details. The 3DSTATE_CONSTANT_PS must be reprogrammed prior to the next 3DPRIMITIVE command after programming the 3DSTATE_PUSH_CONSTANT_ALLOC_PS. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	16h 3DSTATE_PUSH_CONSTANT_ALLOC_PS
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ
7:0	Dword Length		
	Default Value:	0h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	

3DSTATE_PUSH_CONSTANT_ALLOC_PS							
1	31:21	Reserved					
		Project: BDW					
		Format: MBZ					
	20:16	Constant Buffer Offset					
		Project: BDW					
		Format: U5					
		Specifies the offset of the PS constant buffer into the URB.					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,31]</td> <td>(0KB - 31KB) Increments of 2KB</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Value	Name	Project	[0,31]	(0KB - 31KB) Increments of 2KB
	Value	Name	Project				
	[0,31]	(0KB - 31KB) Increments of 2KB	BDW				
	15:6	Reserved					
		Project: BDW					
Format: MBZ							
5:0	Constant Buffer Size						
	Project: BDW						
	Format: U6						
	Specifies the size of the PS constant buffer. This value will determine the amount of data the command stream can pre-fetch before the buffer is full. Value of zero is only valid when constants are not enabled for PS.						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,32]</td> <td>(0KB - 32KB) Increments of 2KB</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Value	Name	Project	[0,32]	(0KB - 32KB) Increments of 2KB	BDW
Value	Name	Project					
[0,32]	(0KB - 32KB) Increments of 2KB	BDW					

3DSTATE_PUSH_CONSTANT_ALLOC_VS

DWord		Bit	Description
3DSTATE_PUSH_CONSTANT_ALLOC_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command sets up the URB configuration for VS Push Constant Buffer.			
Programming Notes			
Programming Restriction:			
<ul style="list-style-type: none"> The sum of the Constant Buffer Offset and the Constant Buffer Size may not exceed the maximum value of the Constant Buffer Size. The sum of the constant length programmed in 3DSTATE_CONSTANT_VS must be equal or smaller than the size of the allocated space in the URB including the buffering for half cachelines. See Push Constant URB Allocation section for more details. The 3DSTATE_CONSTANT_VS must be reprogrammed prior to the next 3DPRIMITIVE command after programming the 3DSTATE_PUSH_CONSTANT_ALLOC_VS. 			
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	12h 3DSTATE_PUSH_CONSTANT_ALLOC_VS
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ
	7:0	DWord Length	
		Default Value:	0h Excludes DWord (0,1)
		Project:	All
		Format:	=n Total Length - 2

3DSTATE_PUSH_CONSTANT_ALLOC_VS			
1	31:21	Reserved	
		Project: BDW	
		Format: MBZ	
	20:16	Constant Buffer Offset	
		Project: BDW	
		Format: U5	
		Specifies the offset of the VS constant buffer into the URB.	
		Value	Name
	[0,31]	(0KB - 31KB) Increments of 2KB	BDW
	15:6	Reserved	
		Project: BDW	
		Format: MBZ	
5:0	Constant Buffer Size		
	Project: BDW		
	Format: U6		
	Specifies the size of the VS constant buffer. This value will determine the amount of data the command stream can pre-fetch before the buffer is full. Value of zero is only valid when constants are not enabled for VS.		
	Value	Name	Project
[0,32]	(0KB - 32KB) Increments of 2KB	BDW	

3DSTATE_RASTER

3DSTATE_RASTER			
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		50h 3DSTATE_RASTER	
Format:		OpCode	
15:8	Reserved		
	Project:	BDW	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	03h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31:28	Reserved	
		Project:	All
		Format:	MBZ
	27	Reserved	
		Project:	BDW
		Format:	MBZ
	26:24	Reserved	
		Project:	BDW
		Format:	MBZ

3DSTATE_RASTER

23:22	API Mode	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">All</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 45%;">Name</th> <th style="width: 20%;">Project</th> <th style="width: 10%;"></th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>DX9/OGL</td> <td>All</td> <td></td> </tr> <tr> <td>1h</td> <td>DX10.0</td> <td>All</td> <td></td> </tr> <tr> <td>2h</td> <td>DX10.1+</td> <td>All</td> <td></td> </tr> <tr> <td>3h</td> <td>Reserved</td> <td>All</td> <td></td> </tr> </tbody> </table>			Project:	All	Value	Name	Project		0h	DX9/OGL	All		1h	DX10.0	All		2h	DX10.1+	All		3h	Reserved	All															
Project:	All																																							
Value	Name	Project																																						
0h	DX9/OGL	All																																						
1h	DX10.0	All																																						
2h	DX10.1+	All																																						
3h	Reserved	All																																						
21	Front Winding	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">All</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 35%;">Name</th> <th style="width: 30%;">Description</th> <th style="width: 15%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Clockwise</td> <td>FRONTWINDING_CW</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Counter Clockwise [Default]</td> <td>FRONTWINDING_CCW</td> <td>All</td> </tr> </tbody> </table>			Project:	All	Value	Name	Description	Project	0h	Clockwise	FRONTWINDING_CW	All	1h	Counter Clockwise [Default]	FRONTWINDING_CCW	All																						
Project:	All																																							
Value	Name	Description	Project																																					
0h	Clockwise	FRONTWINDING_CW	All																																					
1h	Counter Clockwise [Default]	FRONTWINDING_CCW	All																																					
20:18	Forced Sample Count	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">Project:</td> <td style="width: 60%;">All</td> </tr> <tr> <td>Format:</td> <td>U3 Enumerated Type</td> </tr> </table> <p>This field specifies how many samples/pixel exist for RT Independent Rasterization</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> <th style="width: 10%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>NUMRASTSAMPLES_0</td> <td>No RT Independent Rasterization</td> <td>All</td> </tr> <tr> <td>1h</td> <td>NUMRASTSAMPLES_1</td> <td>1 rast-sample/pixel</td> <td>All</td> </tr> <tr> <td>2h</td> <td>NUMRASTSAMPLES_2</td> <td>2 rast-samples/pixel</td> <td>All</td> </tr> <tr> <td>3h</td> <td>NUMRASTSAMPLES_4</td> <td>4 rast-samples/pixel</td> <td>All</td> </tr> <tr> <td>4h</td> <td>NUMRASTSAMPLES_8</td> <td>8 rast-samples/pixel</td> <td>All</td> </tr> <tr> <td>5h</td> <td>NUMRASTSAMPLES_16</td> <td>16 rast-samples/pixel</td> <td>All</td> </tr> <tr> <td>6h-7h</td> <td>Reserved</td> <td></td> <td>All</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center; margin: 0;">Programming Notes</p> <p>When 3DSTATE_MULTISAMPLE::Number of Multisamples != NUMSAMPLES_1, this field must be either NUMRASTSAMPLES_0 or NUMRASTSAMPLES_1.</p> <p>When 3DSTATE_MULTISAMPLE::Number of Multisamples == NUMSAMPLES_1, this field must not be NUMRASTSAMPLES_1.</p> </div>			Project:	All	Format:	U3 Enumerated Type	Value	Name	Description	Project	0h	NUMRASTSAMPLES_0	No RT Independent Rasterization	All	1h	NUMRASTSAMPLES_1	1 rast-sample/pixel	All	2h	NUMRASTSAMPLES_2	2 rast-samples/pixel	All	3h	NUMRASTSAMPLES_4	4 rast-samples/pixel	All	4h	NUMRASTSAMPLES_8	8 rast-samples/pixel	All	5h	NUMRASTSAMPLES_16	16 rast-samples/pixel	All	6h-7h	Reserved		All
Project:	All																																							
Format:	U3 Enumerated Type																																							
Value	Name	Description	Project																																					
0h	NUMRASTSAMPLES_0	No RT Independent Rasterization	All																																					
1h	NUMRASTSAMPLES_1	1 rast-sample/pixel	All																																					
2h	NUMRASTSAMPLES_2	2 rast-samples/pixel	All																																					
3h	NUMRASTSAMPLES_4	4 rast-samples/pixel	All																																					
4h	NUMRASTSAMPLES_8	8 rast-samples/pixel	All																																					
5h	NUMRASTSAMPLES_16	16 rast-samples/pixel	All																																					
6h-7h	Reserved		All																																					

3DSTATE_RASTER

	17:16	Cull Mode <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td colspan="3">All</td> </tr> <tr> <td>Format:</td> <td colspan="3">3D_CullMode</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 60%;">Description</th> <th style="width: 15%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>CULLMODE_BOTH</td> <td>All triangles are discarded (i.e., no triangle objects are drawn)</td> <td>All</td> </tr> <tr> <td>1h</td> <td>CULLMODE_NONE [Default]</td> <td>No triangles are discarded due to orientation</td> <td>All</td> </tr> <tr> <td>2h</td> <td>CULLMODE_FRONT</td> <td>Triangles with a front-facing orientation are discarded</td> <td>All</td> </tr> <tr> <td>3h</td> <td>CULLMODE_BACK</td> <td>Triangles with a back-facing orientation are discarded</td> <td>All</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th colspan="4" style="text-align: center; background-color: #e1eef6;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td colspan="4">Orientation determination is based on the setting of the Front Winding state.</td> </tr> </tbody> </table>			Project:	All			Format:	3D_CullMode			Value	Name	Description	Project	0h	CULLMODE_BOTH	All triangles are discarded (i.e., no triangle objects are drawn)	All	1h	CULLMODE_NONE [Default]	No triangles are discarded due to orientation	All	2h	CULLMODE_FRONT	Triangles with a front-facing orientation are discarded	All	3h	CULLMODE_BACK	Triangles with a back-facing orientation are discarded	All	Programming Notes				Orientation determination is based on the setting of the Front Winding state.			
Project:	All																																							
Format:	3D_CullMode																																							
Value	Name	Description	Project																																					
0h	CULLMODE_BOTH	All triangles are discarded (i.e., no triangle objects are drawn)	All																																					
1h	CULLMODE_NONE [Default]	No triangles are discarded due to orientation	All																																					
2h	CULLMODE_FRONT	Triangles with a front-facing orientation are discarded	All																																					
3h	CULLMODE_BACK	Triangles with a back-facing orientation are discarded	All																																					
Programming Notes																																								
Orientation determination is based on the setting of the Front Winding state.																																								
	15	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td colspan="3">All</td> </tr> <tr> <td>Format:</td> <td colspan="3">MBZ</td> </tr> </table>			Project:	All			Format:	MBZ																														
Project:	All																																							
Format:	MBZ																																							
	14	Force Multisampling <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td colspan="3">All</td> </tr> </table> <p>This field provides a work around override for the computation of SF_INT::Multisample Rasterization Mode and WM_INT::Multisample Rasterization Mode.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 70%;">Description</th> <th style="width: 10%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Normal</td> <td>Multisampling mode is computed by HW according to formula for signal SF_INT::Multisample Rasterization Mode and WM_INT::Multisample Rasterization Mode in vol2a.11 3D Pipeline Windower [BDW] > Windower Pipelined State > 3DSTATE_WM > 3DSTATE_WM [BDW].</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Force</td> <td>Forces the DX Multisampling mode to be used directly</td> <td>All</td> </tr> </tbody> </table>			Project:	All			Value	Name	Description	Project	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 vol2a.11 3D Pipeline Windower [BDW] > Windower Pipelined State > 3DSTATE_WM > 3DSTATE_WM [BDW].	All	1h	Force	Forces the DX Multisampling mode to be used directly	All																				
Project:	All																																							
Value	Name	Description	Project																																					
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 vol2a.11 3D Pipeline Windower [BDW] > Windower Pipelined State > 3DSTATE_WM > 3DSTATE_WM [BDW].	All																																					
1h	Force	Forces the DX Multisampling mode to be used directly	All																																					
	13	Smooth Point Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td colspan="3">BDW</td> </tr> <tr> <td>Format:</td> <td colspan="3">Enable</td> </tr> </table> <p>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.</p>			Project:	BDW			Format:	Enable																														
Project:	BDW																																							
Format:	Enable																																							

3DSTATE_RASTER			
12	DX Multisample Rasterization Enable		
	Project:	All	
	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		
	Project:	All	
	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	Project
	0h	MSRASTMODE_ OFF_PIXEL	All
	1h	MSRASTMODE_ OFF_PATTERN	All
	2h	MSRASTMODE_ ON_PIXEL	All
	3h	MSRASTMODE_ ON_PATTERN	All
	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		
	Project:	All	
	Format:	Enable	
	Enables computation and application of Global Depth Offset for SOLID objects.		
8	Global Depth Offset Enable Wireframe		
	Project:	All	
	Format:	Enable	
	Enables computation and application of Global Depth Offset when triangles are rendered in WIREFRAME mode.		

3DSTATE_RASTER

7	Global Depth Offset Enable Point		
Project:		All	
Format:		Enable	
<p>Enables computation and application of Global Depth Offset when triangles are rendered in POINT mode.</p>			
6:5	Front Face Fill Mode		
Project:		All	
Format:		U2 enumerated type	
<p>This state controls how front-facing triangle and rectangle objects are rendered.</p>			
Value	Name	Description	Project
0h	SOLID	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.	All
1h	WIREFRAME	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).	All
2h	POINT	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).	All
3h	Reserved		All
4:3	Back Face Fill Mode		
Project:		All	
Format:		U2 enumerated type	
<p>This state controls how back-facing triangle and rectangle objects are rendered.</p>			
Value	Name	Description	Project
0h	SOLID	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.	All
1h	WIREFRAME	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).	All
2h	POINT	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).	All
3h	Reserved		All

3DSTATE_RASTER							
	2	<p>Antialiasing Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables "alpha-based" line antialiasing.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>This field must be disabled if any of the render targets have integer (UINT or SINT) surface format.</p>	Project:	All	Format:	Enable	Programming Notes
	Project:	All					
	Format:	Enable					
Programming Notes							
1	<p>Scissor Rectangle Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Enables operation of Scissor Rectangle.</p>	Project:	All	Format:	Enable		
Project:	All						
Format:	Enable						
0	<p>Viewport Z Clip Test Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field is used to control whether the Viewport Z extents (near, far) are considered in VertexClipTest.</p>	Project:	BDW	Format:	Enable		
Project:	BDW						
Format:	Enable						
2	31:0	<p>Global Depth Offset Constant</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>IEEE_Float</td> </tr> </table> <p>Specifies the constant term in the Global Depth Offset function.</p>	Format:	IEEE_Float			
Format:	IEEE_Float						
3	31:0	<p>Global Depth Offset Scale</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>IEEE_Float</td> </tr> </table> <p>Specifies the scale term used in the Global Depth Offset function.</p>	Format:	IEEE_Float			
Format:	IEEE_Float						
4	31:0	<p>Global Depth Offset Clamp</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>IEEE_Float</td> </tr> </table> <p>Specifies the clamp term used in the Global Depth Offset function.</p>	Format:	IEEE_Float			
Format:	IEEE_Float						

3DSTATE_SAMPLE_MASK

3DSTATE_SAMPLE_MASK			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	18h 3DSTATE_SAMPLE_MASK	
	Format:	OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	0h Excludes Dword (0,1)	
	Format:	=n Total Length - 2	
1	31:16	Reserved	
		Project:	BDW
		Format:	MBZ



3DSTATE_SAMPLE_MASK									
15:0	<p>Sample Mask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>16 bit mask Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_MULTISAMPLE)</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> <tr> <td colspan="2"> <ul style="list-style-type: none"> • 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. </td> </tr> </table>	Project:	BDW	Format:	16 bit mask Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_MULTISAMPLE)	Programming Notes		<ul style="list-style-type: none"> • 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. 	
Project:	BDW								
Format:	16 bit mask Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_MULTISAMPLE)								
Programming Notes									
<ul style="list-style-type: none"> • 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. 									

3DSTATE_SAMPLE_PATTERN

3DSTATE_SAMPLE_PATTERN			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_SAMPLE_PATTERN command is used to specify the sample offsets for all multisample sample modes. The set of offset used will be selected based on the multisample mode. This is non-pipelined state.</p>			
Programming Notes			
<p>When programming the sample offsets (for NUMSAMPLES_4 or _8 and MSRASTMODE_xxx_PATTERN), the order of the samples 0 to 3 (or 7 for 8X, or 15 for 16X) must have monotonically increasing distance from the pixel center. This is required to get the correct centroid computation in the device.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		1Ch 3DSTATE_SAMPLE_PATTERN	
Format:		OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	7	
	Format:	=n Total Length - 2	
	Excludes Dword (0,1)		
1..4 Project: BDW	31:0	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_SAMPLE_PATTERN		
5	31:28	8x Sample7 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 7 relative to the UL pixel origin for 8x mode.
		Range: [0,0.9375]
	27:24	8x Sample7 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 7 relative to the UL pixel origin for 8x mode.
		Range: [0,0.9375]
	23:20	8x Sample6 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 6 relative to the UL pixel origin for 8x mode.
		Range: [0,0.9375]
	19:16	8x Sample6 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 6 relative to the UL pixel origin for 8x mode.
		Range: [0,0.9375]
15:12	8x Sample5 X Offset	
	Project: All	
	Format: U0.4	
	Subpixel X offset of Sample 5 relative to the UL pixel origin for 8x mode.	
	Range: [0,0.9375]	
11:8	8x Sample5 Y Offset	
	Project: All	
	Format: U0.4	
	Subpixel Y offset of Sample 5 relative to the UL pixel origin for 8x mode.	
	Range: [0,0.9375]	

3DSTATE_SAMPLE_PATTERN						
	7:4	8x Sample4 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 4 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
Format:	U0.4					
3:0	8x Sample4 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 4 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					
6	31:28	8x Sample3 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 3 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	27:24	8x Sample3 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 3 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
Project:	All					
Format:	U0.4					
23:20	8x Sample2 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 2 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					
19:16	8x Sample2 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 2 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					

3DSTATE_SAMPLE_PATTERN						
	15:12	8x Sample1 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 1 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	11:8	8x Sample1 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 1 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	7:4	8x Sample0 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 0 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	3:0	8x Sample0 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 0 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
7	31:28	4x Sample3 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 3 relative to the UL pixel origin for 4x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
Format:	U0.4					
27:24	4x Sample3 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 3 relative to the UL pixel origin for 4x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					

3DSTATE_SAMPLE_PATTERN		
	23:20	4x Sample2 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 2 relative to the UL pixel origin for 4x mode.
		Range: [0,0.9375]
	19:16	4x Sample2 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 2 relative to the UL pixel origin for 4x mode.
		Range: [0,0.9375]
	15:12	4x Sample1 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 1 relative to the UL pixel origin for 4x mode.
		Range: [0,0.9375]
	11:8	4x Sample1 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 1 relative to the UL pixel origin for 4x mode.
		Range: [0,0.9375]
7:4	4x Sample0 X Offset	
	Project: All	
	Format: U0.4	
	Subpixel X offset of Sample 0 relative to the UL pixel origin for 4x mode.	
	Range: [0,0.9375]	
3:0	4x Sample0 Y Offset	
	Project: All	
	Format: U0.4	
	Subpixel Y offset of Sample 0 relative to the UL pixel origin for 4x mode.	
	Range: [0,0.9375]	

3DSTATE_SAMPLE_PATTERN			
8	31:24	Reserved	
		Project: All	
		Format: MBZ	
	23:20	1x Sample0 X Offset	
		Project: All	
		Format: U0.4	
		Subpixel X offset of Sample 0 relative to the UL pixel origin for 1x mode.	
		Range: [0,0.9375]	
	19:16	1x Sample0 Y Offset	
		Project: All	
		Format: U0.4	
		Subpixel Y offset of Sample 0 relative to the UL pixel origin for 1x mode.	
		Range: [0,0.9375]	
	15:12	2x Sample1 X Offset	
		Project: All	
Format: U0.4			
Subpixel X offset of Sample 1 relative to the UL pixel origin for 2x mode.			
Range: [0,0.9375]			
11:8	2x Sample1 Y Offset		
	Project: All		
	Format: U0.4		
	Subpixel Y offset of Sample 1 relative to the UL pixel origin for 2x mode.		
	Range: [0,0.9375]		
7:4	2x Sample0 X Offset		
	Project: All		
	Format: U0.4		
	Subpixel X offset of Sample 0 relative to the UL pixel origin for 2x mode.		
	Range: [0,0.9375]		

3DSTATE_SAMPLE_PATTERN		
	3:0	2x Sample0 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 0 relative to the UL pixel origin for 2x mode.
		Range: [0,0.9375]



3DSTATE_SAMPLE_PATTERN

3DSTATE_SAMPLE_PATTERN			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_SAMPLE_PATTERN command is used to specify the sample offsets for all multisample sample modes. The set of offset used will be selected based on the multisample mode. This is non-pipelined state.</p>			
Programming Notes			
<p>When programming the sample offsets (for NUMSAMPLES_4 or _8 and MSRASTMODE_XXX_PATTERN), the order of the samples 0 to 3 (or 7 for 8X) must have monotonically increasing distance from the pixel center. This is required to get the correct centroid computation in the device.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	1Ch 3DSTATE_SAMPLE_PATTERN
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ
	7:0	Dword Length	
		Default Value:	3
		Project:	All
Format:		=n Total Length - 2	
Excludes Dword (0,1)			

3DSTATE_SAMPLE_PATTERN					
1	31:28	8x Sample7 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 7 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	27:24	8x Sample7 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 7 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	23:20	8x Sample6 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 6 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	19:16	8x Sample6 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 6 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	15:12	8x Sample5 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 5 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	11:8	8x Sample5 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 5 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]



3DSTATE_SAMPLE_PATTERN						
	7:4	8x Sample4 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 4 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
Format:	U0.4					
3:0	8x Sample4 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 4 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					
2	31:28	8x Sample3 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 3 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	27:24	8x Sample3 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 3 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4
Project:	All					
Format:	U0.4					
23:20	8x Sample2 X Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel X offset of Sample 2 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					
19:16	8x Sample2 Y Offset <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> Subpixel Y offset of Sample 2 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					

3DSTATE_SAMPLE_PATTERN		
	15:12	8x Sample1 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 1 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	11:8	8x Sample1 Y Offset
		Project: All
		Format: U0.4
		Subpixel Y offset of Sample 1 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
	7:4	8x Sample0 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 0 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]
3:0	8x Sample0 Y Offset	
	Project: All	
	Format: U0.4	
	Subpixel Y offset of Sample 0 relative to the UL pixel origin for 8x mode. Range: [0,0.9375]	
3	31:28	4x Sample3 X Offset
		Project: All
		Format: U0.4
		Subpixel X offset of Sample 3 relative to the UL pixel origin for 4x mode. Range: [0,0.9375]
	27:24	4x Sample3 Y Offset
		Project: All
Format: U0.4		
Subpixel Y offset of Sample 3 relative to the UL pixel origin for 4x mode. Range: [0,0.9375]		

3DSTATE_SAMPLE_PATTERN						
	23:20	4x Sample2 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 2 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]
	19:16	4x Sample2 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 2 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]
	15:12	4x Sample1 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 1 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]
	11:8	4x Sample1 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 1 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]
	7:4	4x Sample0 X Offset	Project: All	Format: U0.4	Subpixel X offset of Sample 0 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]
	3:0	4x Sample0 Y Offset	Project: All	Format: U0.4	Subpixel Y offset of Sample 0 relative to the UL pixel origin for 4x mode.	Range: [0,0.9375]

3DSTATE_SAMPLE_PATTERN						
4	31:24	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
	Format:	MBZ				
	23:20	1x Sample0 X Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel X offset of Sample 0 relative to the UL pixel origin for 1x mode. Range: [0,0.9375]</p>	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	19:16	1x Sample0 Y Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel Y offset of Sample 0 relative to the UL pixel origin for 1x mode. Range: [0,0.9375]</p>	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
	15:12	2x Sample1 X Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel X offset of Sample 1 relative to the UL pixel origin for 2x mode. Range: [0,0.9375]</p>	Project:	All	Format:	U0.4
	Project:	All				
	Format:	U0.4				
11:8	2x Sample1 Y Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel Y offset of Sample 1 relative to the UL pixel origin for 2x mode. Range: [0,0.9375]</p>	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					
7:4	2x Sample0 X Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel X offset of Sample 0 relative to the UL pixel origin for 2x mode. Range: [0,0.9375]</p>	Project:	All	Format:	U0.4	
Project:	All					
Format:	U0.4					



3DSTATE_SAMPLE_PATTERN					
3:0	<p>2x Sample0 Y Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.4</td> </tr> </table> <p>Subpixel Y offset of Sample 0 relative to the UL pixel origin for 2x mode.</p> <p>Range: [0,0.9375]</p>	Project:	All	Format:	U0.4
Project:	All				
Format:	U0.4				

3DSTATE_SAMPLER_PALETTE_LOAD0

3DSTATE_SAMPLER_PALETTE_LOAD0		
Project:	DevBWR+	
Source:	RenderCS	
Length Bias:	2	
Description		
<p>The 3DSTATE_SAMPLER_PALETTE_LOAD0 instruction is used to load 32-bit values into the first texture palette. The texture palette is used whenever a texture with a paletted format (containing "Px [palette0]") is referenced by the sampler.</p> <p>This instruction is used to load all or a subset of the 256 entries of the first palette. Partial loads always start from the first (index 0) entry.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: Opcode	
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D
	Format: Opcode	
	26:24	3D Command Opcode
Default Value: 1h 3DSTATE_NONPIPELINED		
Format: Opcode		
23:16	3D Command Sub Opcode	
	Default Value: 02h 3DSTATE_SAMPLER_PALETTE_LOAD0	
Format: Opcode		
15:8	Reserved	
	Project: All	
Format: MBZ		
7:0	DWord Length	
	Format: =n	
	Total Length = 1 + entryCount - 2	
	Value	Name
[0,255]	Range	1-256 Entries
1..n	31:0	Entry
		Format: PALETTE_ENTRY



3DSTATE_SAMPLER_PALETTE_LOAD1

3DSTATE_SAMPLER_PALETTE_LOAD1			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The 3DSTATE_SAMPLER_PALETTE_LOAD1 instruction is used to load 32-bit values into the second texture palette. The second texture palette is used whenever a texture with a paletted format (containing "Px...[palette1]") is referenced by the sampler. This instruction is used to load all or a subset of the 256 entries of the second palette. Partial loads always start from the first (index 0) entry.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	26:24	3D Command Opcode	
		Default Value:	1h 3DSTATE_NONPIPELINED
23:16	3D Command Sub Opcode		
	Default Value:	0Ch 3DSTATE_SAMPLER_PALETTE_LOAD1	
15:8	Reserved		
	Project:	All	
7:0	Format:	MBZ	
	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
1..n	31:24	Palette Alpha[0:N-1]	
		Project:	All
		Format:	U8
	Alpha channel loaded into the Nth entry of the texture color palette.		
	23:16	Palette Red[0:N-1]	
Project:		All	
Format:		U8	
Alpha channel loaded into the Nth entry of the texture color palette.			

3DSTATE_SAMPLER_PALETTE_LOAD1						
	15:8	Palette Green[0:N-1] <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>Alpha channel loaded into the Nth entry of the texture color palette.</p>	Project:	All	Format:	U8
	Project:	All				
Format:	U8					
	7:0	Palette Blue[0:N-1] <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>Alpha channel loaded into the Nth entry of the texture color palette.</p>	Project:	All	Format:	U8
Project:	All					
Format:	U8					

3DSTATE_SAMPLER_STATE_POINTERS_DS

3DSTATE_SAMPLER_STATE_POINTERS_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_SAMPLER_STATE_POINTERS_DS command is used to define the location of DS SAMPLER_STATE table. Only some of the fixed functions utilize sampler state tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		2Dh 3DSTATE_SAMPLER_STATE_POINTERS_DS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Format:	=n	
1	31:5	Pointer to DS Sampler State	
		Project:	All
		Format:	DynamicStateOffset[31:5]SAMPLER_STATE*16
	Specifies the 32-byte aligned address offset of the DS function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.		
	4:0	Reserved	
Project:		All	
Format:		MBZ	

3DSTATE_SAMPLER_STATE_POINTERS_GS

3DSTATE_SAMPLER_STATE_POINTERS_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_SAMPLER_STATE_POINTERS_GS command is used to define the location of GS SAMPLER_STATE table. Only some of the fixed functions utilize sampler state tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	2Eh 3DSTATE_SAMPLER_STATE_POINTERS_GS	
	Format:	OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Format:	=n	
1	31:5	Pointer to GS Sampler State	
		Project:	All
		Format:	DynamicStateOffset[31:5]SAMPLER_STATE*16
	Specifies the 32-byte aligned address offset of the GS function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.		
4:0	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_SAMPLER_STATE_POINTERS_HS

3DSTATE_SAMPLER_STATE_POINTERS_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_SAMPLER_STATE_POINTERS_HS command is used to define the location of HS SAMPLER_STATE table. Only some of the fixed functions utilize sampler state tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:		OpCode	
23:16	3D Command Sub Opcode		
	Default Value:	2Ch 3DSTATE_SAMPLER_STATE_POINTERS_HS	
	Format:	OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Format:	=n	
1	31:5	Pointer to HS Sampler State	
		Project:	All
		Format:	DynamicStateOffset[31:5]SAMPLER_STATE*16
	Specifies the 32-byte aligned address offset of the HS function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.		
	4:0	Reserved	
Project:		All	
Format:		MBZ	

3DSTATE_SAMPLER_STATE_POINTERS_PS

3DSTATE_SAMPLER_STATE_POINTERS_PS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>The 3DSTATE_SAMPLER_STATE_POINTERS_PS command is used to define the location of PS SAMPLER_STATE table. Only some of the fixed functions utilize sampler state tables.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED Format: OpCode
	23:16	3D Command Sub Opcode
Default Value: 2Fh 3DSTATE_SAMPLER_STATE_POINTERS_PS Format: OpCode		
15:8	Reserved	
	Project: All Format: MBZ	
7:0	DWord Length	
	Default Value: 0h DWORD_COUNT_n Format: =n	
1	31:5	Pointer to PS Sampler State
		Project: All Format: DynamicStateOffset[31:5]SAMPLER_STATE*16 Specifies the 32-byte aligned address offset of the PS function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.
	4:0	Reserved
		Project: All Format: MBZ

3DSTATE_SAMPLER_STATE_POINTERS_VS

3DSTATE_SAMPLER_STATE_POINTERS_VS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The 3DSTATE_SAMPLER_STATE_POINTERS_VS command is used to define the location of VS SAMPLER_STATE table. Only some of the fixed functions utilize sampler state tables.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		2Bh 3DSTATE_SAMPLER_STATE_POINTERS_VS	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
	Format:	=n	
1	31:5	Pointer to VS Sampler State	
		Project:	All
		Format:	DynamicStateOffset[31:5]SAMPLER_STATE*16
	Specifies the 32-byte aligned address offset of the VS function's SAMPLER_STATE table. This offset is relative to the Dynamic State Base Address.		
	4:0	Reserved	
Project:		All	
Format:		MBZ	

3DSTATE_SBE

3DSTATE_SBE					
Project:		BDW			
Source:		RenderCS			
Length Bias:		2			
DWord	Bit	Description			
0	31:29	Command Type			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>3h GFXPIPE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	3h GFXPIPE	Format:
	Default Value:	3h GFXPIPE			
	Format:	OpCode			
	28:27	Command SubType			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>3h GFXPIPE_3D</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	3h GFXPIPE_3D	Format:
	Default Value:	3h GFXPIPE_3D			
Format:	OpCode				
26:24	3D Command Opcode				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h 3DSTATE_PIPELINED</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h 3DSTATE_PIPELINED	Format:	OpCode
Default Value:	0h 3DSTATE_PIPELINED				
Format:	OpCode				
23:16	3D Command Sub Opcode				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>1Fh 3DSTATE_SBE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	1Fh 3DSTATE_SBE	Format:	OpCode
Default Value:	1Fh 3DSTATE_SBE				
Format:	OpCode				
15:8	Reserved				
Format:		MBZ			
7:0	DWord Length				
	Default Value:	02h Excludes DWord (0,1)			
	Project:	BDW			
	Format:	=n			
		Total Length - 2			
1	31:30	Reserved			
		Format:	MBZ		
29	Force Vertex URB Entry Read Length				
	Project:	BDW			
	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.			

3DSTATE_SBE			
28	Force Vertex URB Entry Read Offset		
	Project:	BDW	
	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.		
27:22	Number of SF Output Attributes		
	Project:	BDW	
	Format:	U6 count of attributes	
	Specifies the number of vertex attributes passed from the SF stage to the WM stage (does not include Position).		
		Value	
		Name	
		[0,32]	
21	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.			
20	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).		
	Value	Name	Description
	0h	UPPERLEFT	Top Left = (0,0,0,1)Bottom Left = (0,1,0,1)Bottom Right = (1,1,0,1)
1h	LOWERLEFT	Top Left = (0,1,0,1)Bottom Left = (0,0,0,1)Bottom Right = (1,0,0,1)	
19	Primitive ID Override Component W		
	Project:	BDW	
	Format:	Enable	
If set, the W component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.			
18	Primitive ID Override Component Z		
	Project:	BDW	
	Format:	Enable	
If set, the Z component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.			

3DSTATE_SBE							
17	<p>Primitive ID Override Component Y</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set, the Y component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</p>	Project:	BDW	Format:	Enable		
	Project:	BDW					
	Format:	Enable					
	16	<p>Primitive ID Override Component X</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set, the X component of output attribute selected by Primitive ID Override Attribute Select is overridden with the Primitive ID.</p>	Project:	BDW	Format:	Enable	
		Project:	BDW				
	Format:	Enable					
15:11	<p>Vertex URB Entry Read Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U5</td> </tr> </table> <p>Specifies the amount of URB data read for each Vertex URB entry, in 256-bit register increments.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,16]</td> <td></td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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. $read_length = \text{ceiling}((\text{max_source_attr} + 1) / 2)$</p>	Format:	U5	Value	Name	[1,16]	
	Format:	U5					
	Value	Name					
	[1,16]						
10:5	<p>Vertex URB Entry Read Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>Specifies the offset (in 256-bit units) at which Vertex URB data is to be read from the URB.</p>	Project:	BDW				
	Project:	BDW					
4:0	<p>Primitive ID Override Attribute Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>Specifies which attribute is overridden w/ the Primitive ID</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 80%;">Programming Notes</th> <th style="width: 20%;">Project</th> </tr> </thead> <tbody> <tr> <td>Set all Primitive ID Override Component Select X/Y/Z/W to 0 to indicate there is no Primitive ID override.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	BDW	Programming Notes	Project	Set all Primitive ID Override Component Select X/Y/Z/W to 0 to indicate there is no Primitive ID override.	BDW
	Project:	BDW					
	Programming Notes	Project					
Set all Primitive ID Override Component Select X/Y/Z/W to 0 to indicate there is no Primitive ID override.	BDW						
2	31:0	<p>Point Sprite Texture Coordinate Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>Enable[32]</td> </tr> </table> <p>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.</p>	Format:	Enable[32]			
		Format:	Enable[32]				



3DSTATE_SBE				
3	31:0	<p>Constant Interpolation Enable</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Format:</td> <td>Enable[32]</td> </tr> </table> <p>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.</p>	Format:	Enable[32]
Format:	Enable[32]			

3DSTATE_SBE_SWIZ

3DSTATE_SBE_SWIZ		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
		Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D
		Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED
		Format: OpCode
	23:16	3D Command Sub Opcode
Default Value: 51h 3DSTATE_SBE_SWIZ		
Format: OpCode		
15:8	Reserved	
	Format: MBZ	
7:0	DWord Length	
	Default Value: 9h Excludes DWord (0,1)	
	Format: =n	
	Total Length - 2	
1..8	15:0	Attribute
		Format: SF_OUTPUT_ATTRIBUTE_DETAIL
9..10	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	

3DSTATE_SBE_SWIZ		
	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
	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

3DSTATE_SCISSOR_STATE_POINTERS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
The 3DSTATE_SCISSOR_STATE_POINTERS command is used to define the location of the indirect SCISSOR_RECT state.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED Format: OpCode
	23:16	3D Command Sub Opcode
Default Value: 0Fh 3DSTATE_SCISSOR_STATE_POINTERS Format: OpCode		
15:8	Reserved	
	Project: All Format: MBZ	
7:0	DWord Length	
	Default Value: 0h DWORD_COUNT_n Format: =n	
1	31:5	Scissor Rect Pointer
		Project: All 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
		Project: All Format: MBZ



3DSTATE_SF

3DSTATE_SF		
Project:		BDW
Source:		RenderCS
Length Bias:		2
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D Format: OpCode
	26:24	3D Command Opcode
		Default Value: 0h 3DSTATE_PIPELINED Format: OpCode
	23:16	3D Command Sub Opcode
Default Value: 13h 3DSTATE_SF Format: OpCode		
15:8	Reserved	
	Project: BDW Format: MBZ	
7:0	DWord Length	
	Default Value: 2h Excludes DWord (0,1) Format: =n	
	Total Length - 2	
1	31:30	Reserved
		Format: MBZ
29:12	29:12	Reserved
		Project: BDW Format: MBZ

3DSTATE_SF		
11	Legacy Global Depth Bias Enable	
	Format:	Enable
	<p>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.</p>	
	Programming Notes	
	<p>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.</p>	
10	Statistics Enable	
	Project:	All
	Format:	Enable
	<p>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.</p>	
	Programming Notes	
<p>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.</p>		
9:2	Reserved	
	Format:	MBZ
1	Viewport Transform Enable	
	Format:	Enable
<p>This bit controls the Viewport Transform function.</p>		
0	Reserved	
	Format:	MBZ
2	31:29	Reserved
		Format:
28	Reserved	

3DSTATE_SF																			
27:18	<p>Line Width</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U3.7</td> </tr> </table> <p>Range: [0.0, 7.9921875]</p> <p>Range: [0.0, 2047.9921875]</p> <p>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).</p> <p style="text-align: center;">Programming Notes</p> <p>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.</p>	Project:	BDW	Format:	U3.7														
	Project:	BDW																	
	Format:	U3.7																	
	17:16	<p>Line End Cap Antialiasing Region Width</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field specifies the distances over which the coverage of anti-aliased line end caps are computed.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Format:	U2	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
		Format:	U2																
		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																
15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																
	Format:	MBZ																	
14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																
	Format:	MBZ																	
13	<p>Reserved</p>																		
12	<p>Reserved</p>																		
11:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																
	Format:	MBZ																	
3	31	<p>Last Pixel Enable</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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).</p> <p style="text-align: center;">Programming Notes</p> <p>Last pixel is applied to all lines of a LINELIST, and only the last line of a LINESSTRIP.</p>	Format:	Enable															
		Format:	Enable																

3DSTATE_SF		
30:29	Triangle Strip/List Provoking Vertex Select	
	Format:	0-based vertex index
	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
28:27	Line Strip/List Provoking Vertex Select	
	Project:	All
	Format:	0-based vertex index
	Selects which vertex of a line (in a line strip or list primitive) is considered the "provoking vertex".	
	Value	Name
0h	0	Vertex 0
1h	1	Vertex 1
2h	Reserved	Reserved
3h	Reserved	Reserved
26:25	Triangle Fan Provoking Vertex Select	
	Format:	0-based vertex index
	Selects which vertex of a triangle (in a triangle fan primitive) is considered the "provoking vertex".	
	Value	Name
	0h	0
24:15	Reserved	
	Format:	MBZ
14	AA Line Distance Mode	
	Format:	U1
	This bit controls the distance computation for antialiased lines.	
	Value	Name
1h	AALINEDISTANCE_TRUE	True distance computation. This is the normal setting which should yield WHQL compliance.

3DSTATE_SF			
13	Smooth Point Enable		
	Format:	Enable	
	Double Buffer Armed By:	Enables logic to draw smooth OGL Points	
	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	Disable
	1h	Enable	4 sub pixel precision bits maintained
	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_SO_BUFFER

3DSTATE_SO_BUFFER		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
Programming Notes		Project
Foreach SO Buffer, the 3DSTATE_SO_BUFFER must only be sent once prior to each 3DPRIMITIVE command.		BDW
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
		Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D Format: OpCode
	26:24	3D Command Opcode
Default Value: 1h 3DSTATE_NONPIPELINED Format: OpCode		
23:16	3D Command Sub Opcode	
	Default Value: 18h 3DSTATE_SO_BUFFER Format: OpCode	
15:8	Reserved	
	Format: MBZ	
7:0	DWord Length	
	Default Value: 6h Excludes DWord (0,1)	
	Format: =n	
	Total Length - 2	
1	31	SO Buffer Enable
		Format: Enable
	<p>If set, stream output to SO Buffer is enabled, if 3DSTATE_STREAMOUT::SO Function ENABLE is also enabled..If clear, the SO Buffer is considered "not bound" and effectively treated as a zero-length buffer for the purposes of SO output and overflow detection. If an enabled stream's Stream to Buffer Selects includes this buffer it is by definition an overflow condition. That stream will cause no writes to occur, and only SO_PRIM_STORAGE_NEEDED[<stream>] will increment.</p>	
30:29	SO Buffer Index	
	Format: U2 Specifies which of the four SO Buffers is being defined.	

3DSTATE_SO_BUFFER		
	28:22	SO Buffer Object Control State Format: MEMORY_OBJECT_CONTROL_STATE Specifies the memory object control state for the SO buffer.
	21	Stream Offset Write Enable Format: Enable When set, this field allows the hardware to write SO_WRITE_OFFSET[Buffer#] as specified in the Stream Offset field. Programming Notes The field is operates irrespective of whether SO Buffer Enable is set or clear.
	20	Stream Output Buffer Offset Address Enable Format: Enable When set, this field allows the hardware to read/write the stream output buffer offset as specified in the "Stream Output Buffer Offset Address" field. Programming Notes The field is operates irrespective of whether SO Buffer Enable is set or clear.
	19:12	Reserved Format: MBZ
	11:0	Reserved Format: MBZ
2..3	63:48	Reserved Format: MBZ
	47:2	Surface Base Address Format: GraphicsAddress[47:2]SurfaceBase This field specifies the starting DWord address of the buffer in Graphics Memory.
	1:0	Reserved Format: MBZ
4	31:30	Reserved Format: MBZ
	29:0	Surface Size Format: U30-1 This field specifies the size of buffer in number DWords minus 1 of the buffer in Graphics Memory.
5..6	63:48	Reserved Format: MBZ

3DSTATE_SO_BUFFER											
	47:2	<p>Stream Output Buffer Offset Address</p> <table border="1"> <tr> <td>Format:</td> <td>GraphicsAddress[47:2]OutputBuffer</td> </tr> </table> <p>This field specifies the high 16 bits of address of the buffer in Graphics Memory where the Stream Output Buffer Offset is stored when all the data has been written. It is also used to fetch the stream Output buffer Offset when needed.</p>	Format:	GraphicsAddress[47:2]OutputBuffer							
	Format:	GraphicsAddress[47:2]OutputBuffer									
1:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
7	31:0	<p>Stream Offset</p> <p>This field specifies the Offset in stream output buffer to start at, or whether to append to the end of an existing buffer. The Offset must be DWORD aligned. If Stream Offset is equal to 0xFFFFFFFF then load the value at the Stream Output Buffer Offset address into SO_WRITE_OFFSET[Buffer#]. Otherwise, SO_WRITE_OFFSET[Buffer#] = Stream Offset.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>FFFFFFFFh</td> <td></td> <td>Load the value at the Stream Output Buffer Offset address into MMIO_SO_OFFSET[Buffer#].</td> </tr> <tr> <td>xxxxxxxx xxxxxxxx xxxxxxxx xxxxxx00b</td> <td></td> <td>MMIO_SO_OFFSET[Buffer#] = Stream Offset</td> </tr> </tbody> </table>	Value	Name	Description	FFFFFFFFh		Load the value at the Stream Output Buffer Offset address into MMIO_SO_OFFSET[Buffer#].	xxxxxxxx xxxxxxxx xxxxxxxx xxxxxx00b		MMIO_SO_OFFSET[Buffer#] = Stream Offset
Value	Name	Description									
FFFFFFFFh		Load the value at the Stream Output Buffer Offset address into MMIO_SO_OFFSET[Buffer#].									
xxxxxxxx xxxxxxxx xxxxxxxx xxxxxx00b		MMIO_SO_OFFSET[Buffer#] = Stream Offset									

3DSTATE_SO_DECL_LIST

3DSTATE_SO_DECL_LIST						
Project:	BDW					
Source:	RenderCS					
Length Bias:	2					
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value: 3h GFXPIPE Format: OpCode				
	28:27	Command SubType				
		Default Value: 3h GFXPIPE_3D Format: OpCode				
	26:24	3D Command Opcode				
		Default Value: 1h 3DSTATE_NONPIPELINED Format: OpCode				
	23:16	3D Command Sub Opcode				
Default Value: 17h 3DSTATE_SO_DECL_LIST Format: OpCode						
15:9	Reserved					
8:0	DWord Length					
	Format: =n Total Length - 2					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[1,257]</td> <td>Excludes DWORD (0,1) 0-128 Entries</td> <td>Value = 2 * (# of SO_DECL quads) + 1</td> </tr> </tbody> </table>	Value	Name	Description	[1,257]	Excludes DWORD (0,1) 0-128 Entries
Value	Name	Description				
[1,257]	Excludes DWORD (0,1) 0-128 Entries	Value = 2 * (# of SO_DECL quads) + 1				
1	31:16	Reserved				
		Format: MBZ				
	15:12	Stream to Buffer Selects [3]				
11:8	Stream to Buffer Selects [2]					
	Format: U4 bitmask Identifies to which SO Buffers stream 2 outputs. See Stream To Buffer Selects [0] field description.					

3DSTATE_SO_DECL_LIST												
	<p>7:4 Stream to Buffer Selects [1]</p> <table border="1"> <tr> <td>Format:</td> <td>U4 bitmask</td> </tr> </table> <p>Identifies to which SO Buffers stream 1 outputs. See Stream To Buffer Selects [0] field description.</p>	Format:	U4 bitmask									
	Format:	U4 bitmask										
<p>3:0 Stream to Buffer Selects [0]</p> <table border="1"> <tr> <td>Format:</td> <td>U4 bitmask</td> </tr> </table> <p>Identifies to which SO Buffers stream 0 outputs (irrespective of whether those buffers are enabled via 3DSTATE_STREAMOUT). Software is required to scan the SO_DECL list in order to provide this summary information. Note: For "inactive" streams, software must program this field to all zero (no buffers written to) and the corresponding Num Entries field to zero (no valid SO_DECLs).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>1xxx</td> <td>SO Buffer 3</td> </tr> <tr> <td>x1xx</td> <td>SO Buffer 2</td> </tr> <tr> <td>xx1x</td> <td>SO Buffer 1</td> </tr> <tr> <td>xxx1</td> <td>SO Buffer 0</td> </tr> </tbody> </table>	Format:	U4 bitmask	Value	Name	1xxx	SO Buffer 3	x1xx	SO Buffer 2	xx1x	SO Buffer 1	xxx1	SO Buffer 0
Format:	U4 bitmask											
Value	Name											
1xxx	SO Buffer 3											
x1xx	SO Buffer 2											
xx1x	SO Buffer 1											
xxx1	SO Buffer 0											
2	<p>31:24 Num Entries [3]</p> <table border="1"> <tr> <td>Format:</td> <td>U8 #entries</td> </tr> </table> <p>Specifies the number of valid SO_DECL entries for Stream 3. (See notes in Num Entries [0] field description).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,128]</td> <td>entries</td> </tr> </tbody> </table>	Format:	U8 #entries	Value	Name	[0,128]	entries					
	Format:	U8 #entries										
	Value	Name										
[0,128]	entries											
<p>23:16 Num Entries [2]</p> <table border="1"> <tr> <td>Format:</td> <td>U8 #entries</td> </tr> </table> <p>Specifies the number of valid SO_DECL entries for Stream 2. (See notes in Num Entries [0] field description).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,128]</td> <td>entries</td> </tr> </tbody> </table>	Format:	U8 #entries	Value	Name	[0,128]	entries						
Format:	U8 #entries											
Value	Name											
[0,128]	entries											
<p>15:8 Num Entries [1]</p> <table border="1"> <tr> <td>Format:</td> <td>U8 #entries</td> </tr> </table> <p>Specifies the number of valid SO_DECL entries for Stream 1. (See notes in Num Entries [0] field description).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,128]</td> <td>entries</td> </tr> </tbody> </table>	Format:	U8 #entries	Value	Name	[0,128]	entries						
Format:	U8 #entries											
Value	Name											
[0,128]	entries											

3DSTATE_SO_DECL_LIST					
	7:0	Num Entries [0]			
		Format: U8 #entries			
		Specifies the number of valid SO_DECL entries for Stream 0. Note that the SO_DECLs are programmed in groups of four (one SO_DECL for each of the four streams). Therefore the number of 2-DWord groups of SO_DECLs supplied in this command is derived from the stream(s) with the most valid SO_DECLs. The NumEntries value specific to each stream will indicate how many SO_DECLs are valid for that particular stream. Any trailing invalid SO_DECLs supplied for streams with fewer valid SO_DECLs will be ignored. It is legal to specify Num Entries = 0 for all four streams simultaneously. In this case there will be no SO_DECLs included in the command (only DW 0-2). Note that all Stream to Buffer Selects bits must be zero in this case (as no streams produce output).			
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,128]</td> <td style="text-align: center;">entries</td> </tr> </tbody> </table>	Value	Name	[0,128]
Value	Name				
[0,128]	entries				
3..n	63:0	Entry			
		Format: SO_DECL_ENTRY			

3DSTATE_STENCIL_BUFFER

3DSTATE_STENCIL_BUFFER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description		Project	
This command sets the surface state of the separate stencil buffer, delivered as a pipelined state command. However, the state change pipelining isn't completely transparent (see restriction below).			
WM HW will internally manage the draining pipe and flushing of the caches when this command is issued. The PIPE_CONTROL restrictions are removed.		BDW	
Programming Notes		Project	
Restriction: Prior to changing Depth/Stencil Buffer state (i.e., any combination of 3DSTATE_DEPTH_BUFFER, 3DSTATE_CLEAR_PARAMS, 3DSTATE_STENCIL_BUFFER, 3DSTATE_HIER_DEPTH_BUFFER) SW must first issue a pipelined depth stall (PIPE_CONTROL with Depth Stall bit set, followed by a pipelined depth cache flush (PIPE_CONTROL with Depth Flush Bit set, followed by another pipelined depth stall (PIPE_CONTROL with Depth Stall Bit set), unless SW can otherwise guarantee that the pipeline from WM onwards is already flushed (e.g., via a preceding MI_FLUSH).			
The stencil buffer is always Tile-W		BDW	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	06h 3DSTATE_STENCIL_BUFFER
		Format:	OpCode
15:8	Reserved		
	Format:	MBZ	

3DSTATE_STENCIL_BUFFER							
	7:0	Dword Length					
		Format: =n Total Length - 2					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 20%;">Project</th> </tr> </thead> <tbody> <tr> <td>3h</td> <td>Excludes Dword (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	Project	3h	Excludes Dword (0,1) [Default]
Value	Name	Project					
3h	Excludes Dword (0,1) [Default]	BDW					
1	31	Stencil Buffer Enable					
		Project: BDW					
		Format: U1					
		When set indicates that there is a valid stencil buffer.					
	Programming Notes						
	This bit should be "0" if Depth buffer surface format is D16_UNORM OR Depth buffer surface type is NULL.						
	30:29	Reserved					
	Format: MBZ						
	28:22	Stencil Buffer Object Control State					
	Project: BDW						
Format: MEMORY_OBJECT_CONTROL_STATE							
Specifies the memory object control state for the stencil buffer.							
21	Reserved						
	Project: BDW						
Format: MBZ							
20:17	Reserved						
	Format: MBZ						
16:0	Surface Pitch						
	Format: U17-1 Pitch in Bytes						
	This field specifies the pitch of the stencil buffer in (#Bytes - 1).						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>[127, 1FFFFh]</td> <td></td> <td>corresponding to [128B, 128KB]also restricted to a multiple of 128B</td> </tr> </tbody> </table>	Value	Name	Description	[127, 1FFFFh]		corresponding to [128B, 128KB]also restricted to a multiple of 128B
	Value	Name	Description				
	[127, 1FFFFh]		corresponding to [128B, 128KB]also restricted to a multiple of 128B				
Programming Notes							
<p>Since this surface is tiled, the pitch specified must be a multiple of the tile pitch, in the range [128B, 128KB].</p> <p>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).</p>							

3DSTATE_STENCIL_BUFFER							
2..3 Project: BDW	63:0	Surface Base Address					
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:0]Stencil_Buffer</td> </tr> </table>	Project:	BDW	Format:	GraphicsAddress[63:0]Stencil_Buffer	
		Project:	BDW				
		Format:	GraphicsAddress[63:0]Stencil_Buffer				
		This field specifies the address of the buffer in Graphics Memory.					
Programming Notes							
The Stencil Buffer can only be mapped to Main Memory (uncached).							
4 Project: BDW	31:15	Reserved					
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
	Project:	BDW					
	Format:	MBZ					
	14:0	Surface QPitch					
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>QPitch[16:2]</td> </tr> </table>	Project:	BDW	Format:	QPitch[16:2]	
		Project:	BDW				
		Format:	QPitch[16:2]				
		Description					
		This field specifies the distance in rows between array slices. It is used only in the following cases:					
<ul style="list-style-type: none"> • Surface Array is enabled <i>OR</i> • Number of Multisamples is not NUMSAMPLES_1 and Multisampled Surface Storage Format set to MSFMT_MSS <i>OR</i> • Surface Type is SURFTYPE_CUBE 							
Project							
BDW							
<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>[4h,1FFFCh]</td> <td></td> <td>in multiples of 4 (low 2 bits missing)</td> </tr> </tbody> </table>		Value	Name	Description	[4h,1FFFCh]		in multiples of 4 (low 2 bits missing)
Value	Name	Description					
[4h,1FFFCh]		in multiples of 4 (low 2 bits missing)					
Programming Notes							
This field must be set to an integer multiple of 8 (QPitch[2] MBZ) 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.							

3DSTATE_STREAMOUT

3DSTATE_STREAMOUT			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command contains pipelined state required by the SOL unit.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		1Eh 3DSTATE_STREAMOUT	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	3h Excludes DWord (0,1)	
	Project:	BDW	
	Format:	=n Total Length - 2	
1	31	SO Function Enable	
		Project:	All
		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.			

3DSTATE_STREAMOUT				
30	API Rendering Disable			
	Project:	BDW		
	Format:	U1		
	<p>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.</p>			
Programming Notes				
<p>The SOL unit generates an SOL_INT::Render_Enable which ultimately controls whether rendering occurs or not.</p>				
29	Reserved			
	Project:	All		
	Format:	MBZ		
28:27	Render Stream Select			
	Project:	All		
	Format:	U2		
	Description			Project
	<p>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.</p>			
	<p>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.</p>			BDW
26	Reorder Mode			
	Project:	All		
	<p>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.</p>			
	Value	Name	Description	Project
	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.	All
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.	All	



3DSTATE_STREAMOUT																								
	25	SO Statistics Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This bit controls whether StreamOutput statistics register(s) can be incremented.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 70%;">Description</th> <th style="width: 10%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers cannot increment.</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers can increment.</td> <td>All</td> </tr> </tbody> </table>	Project:	All	Format:	Enable	Value	Name	Description	Project	0h	Disable	SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers cannot increment.	All	1h	Enable	SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers can increment.	All						
	Project:	All																						
	Format:	Enable																						
	Value	Name	Description	Project																				
	0h	Disable	SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers cannot increment.	All																				
	1h	Enable	SO_NUM_PRIMS_WRITTEN[0..3] and SO_PRIM_STORAGE_NEEDED[0..3] registers can increment.	All																				
	24:23	Force Rendering <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field provides a work around override for the computation of SOL_INT::Render_Enable</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 70%;">Description</th> <th style="width: 10%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Normal</td> <td>SOL_INT::Render_Enable is computed normally</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Reseved</td> <td></td> <td>All</td> </tr> <tr> <td>2h</td> <td>Force_Off</td> <td>Forces the rendering to be disabled.</td> <td>All</td> </tr> <tr> <td>3h</td> <td>Force_on</td> <td>Forces the rendering to be enabled.</td> <td>All</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	Description	Project	0h	Normal	SOL_INT::Render_Enable is computed normally	All	1h	Reseved		All	2h	Force_Off	Forces the rendering to be disabled.	All	3h	Force_on	Forces the rendering to be enabled.	All
	Project:	BDW																						
	Value	Name	Description	Project																				
	0h	Normal	SOL_INT::Render_Enable is computed normally	All																				
1h	Reseved		All																					
2h	Force_Off	Forces the rendering to be disabled.	All																					
3h	Force_on	Forces the rendering to be enabled.	All																					
22:21	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																			
Project:	BDW																							
Format:	MBZ																							
20:12	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ																			
Project:	All																							
Format:	MBZ																							
11:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																			
Project:	BDW																							
Format:	MBZ																							
7:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ																			
Project:	All																							
Format:	MBZ																							
2	31:30	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ																		
	Project:	All																						
Format:	MBZ																							
29	Stream 3 Vertex Read Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1 count of 256-bit units</td> </tr> </table> <p>Specifies amount of data to skip over before reading back Stream 3 vertex data. (See Stream 0 Vertex Read Offset)</p>	Project:	All	Format:	U1 count of 256-bit units																			
Project:	All																							
Format:	U1 count of 256-bit units																							

3DSTATE_STREAMOUT		
28:24	Stream 3 Vertex Read Length	
	Project:	All
	Format:	U5-1 count of 256-bit units
(See Stream 0 Vertex Read Length)		
23:22	Reserved	
	Project:	All
	Format:	MBZ
21	Stream 2 Vertex Read Offset	
	Project:	All
	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	
	Project:	All
	Format:	U5-1 count of 256-bit units
15:14	Reserved	
	Project:	All
	Format:	MBZ
13	Stream 1 Vertex Read Offset	
	Project:	All
	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	
	Project:	All
	Format:	U5-1 count of 256-bit units
(See Stream 0 Vertex Read Length)		
7:6	Reserved	
	Project:	All
	Format:	MBZ

		3DSTATE_STREAMOUT	
	5	Stream 0 Vertex Read Offset	
		Project:	All
		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	
		Project:	All
		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).	
3 Project: BDW	31:28	Reserved	
		Project:	BDW
		Format:	MBZ
	27:16	Buffer 1 Surface Pitch	
		Project:	BDW
15:12		Reserved	
		Project:	BDW
		Format:	MBZ
	11:0	Buffer 0 Surface Pitch	
		Project:	BDW
		Format:	U12 pitch in Bytes
		This field specifies the pitch of the SO buffer in #Bytes.	
		Value	Name
		[0,2048]	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.	
4 Project: BDW	31:28	Reserved	
		Project:	BDW
		Format:	MBZ

3DSTATE_STREAMOUT		
	27:16	Buffer 3 Surface Pitch
		Project: BDW
	Format: U12	
	15:12	Reserved
		Project: BDW
	Format: MBZ	
11:0	Buffer 2 Surface Pitch	
	Project: BDW	
Format: U12		

3DSTATE_TE

3DSTATE_TE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
For [BDW], the state used by TE is defined with this inline state packet.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	1Ch 3DSTATE_TE
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	2h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:20	Reserved	
		Project:	All
		Format:	MBZ
	19	Reserved	
		Project:	BDW
		Format:	MBZ
	18:16	Reserved	
		Project:	BDW
		Format:	MBZ

3DSTATE_TE				
	15:14	Reserved		
		Project:	All	
		Format:	MBZ	
	13:12	Partitioning		
		Project:	All	
		Format:	U2	
		This field specifies how edges are partitioned based on tessellation factor.		
		Value	Name	Description
				Project
		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.
			All	
			All	
			All	
11:10	Reserved			
	Project:	All		
	Format:	MBZ		
9:8	Output Topology			
	Project:	All		
	Format:	U2		
	This field specifies which primitive types are to be output.			
	Value	Name	Description	
			Project	
	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.	
			All	
			All	
			All	
7:6	Reserved			
	Project:	All		
	Format:	MBZ		

3DSTATE_TE				
5:4	TE Domain			
	Project:		All	
	Format:		U2	
	This field specifies which type of domain is to be tessellated.			
	Value	Name	Description	Project
	0h	QUAD	2D (U, V) domain is tessellated	All
	1h	TRI	Triangular (U, V, W) domain is tessellated	All
	2h	ISOLINE	2D (U, V) domain is tessellated.	All
	3	Reserved		
		Project:		All
Format:		MBZ		
2:1	TE Mode			
	Project:		All	
	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	Project
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.	All	
1h	SW_TESS	Software Tessellation Mode. The TE unit will pass down HS-thread-generated tessellated domain points instead of generating them itself from TessFactors. The TE unit will read the Domain Point Count and Domain Point Buffer Starting Address fields from the patch header, and if the count is 0 it will consider the patch culled and discard it. Otherwise the address is used to start fetching DOMAIN_POINT structures from memory and passing them down the pipeline to DS.	BDW	
0	TE Enable			
	Project:		All	
	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			
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.				

3DSTATE_TE												
2	31:0	<p>Maximum Tessellation Factor Odd</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>IEEE_Float</td> </tr> </table> <p>This field specifies the maximum TessFactor for ODD_FRACTIONAL partitioning when in HW_TESS mode.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[427c0000h,427c0000h]</td> <td style="text-align: center;">63 [Default]</td> <td>Per API Spec, For normal operation software should set this value to 63.0</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Note that ISOLINE's LineDensity TF is always subjected to INTEGER partitioning regardless of the Partitioning state.</p>	Project:	All	Format:	IEEE_Float	Value	Name	Description	[427c0000h,427c0000h]	63 [Default]	Per API Spec, For normal operation software should set this value to 63.0
Project:	All											
Format:	IEEE_Float											
Value	Name	Description										
[427c0000h,427c0000h]	63 [Default]	Per API Spec, For normal operation software should set this value to 63.0										
3	31:0	<p>Maximum Tessellation Factor Not Odd</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>IEEE_Float</td> </tr> </table> <p>This field specifies the maximum TessFactor for EVEN_FRACTIONAL or INTEGER partitioning when in HW_TESS mode.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[42800000h,42800000h]</td> <td style="text-align: center;">64 [Default]</td> <td>Per API Spec, For normal operation software should set this value to 64.0</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Note that ISOLINE's LineDensity TF is always subjected to INTEGER partitioning regardless of the Partitioning state.</p>	Project:	All	Format:	IEEE_Float	Value	Name	Description	[42800000h,42800000h]	64 [Default]	Per API Spec, For normal operation software should set this value to 64.0
Project:	All											
Format:	IEEE_Float											
Value	Name	Description										
[42800000h,42800000h]	64 [Default]	Per API Spec, For normal operation software should set this value to 64.0										

3DSTATE_URB_DS

3DSTATE_URB_DS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description			
<p>This command may not overlap with the push constants in the URB defined by the 3DSTATE_PUSH_CONSTANT_ALLOC_VS, 3DSTATE_PUSH_CONSTANT_ALLOC_DS, 3DSTATE_PUSH_CONSTANT_ALLOC_HS, and 3DSTATE_PUSH_CONSTANT_ALLOC_GS commands.</p>			
<p>The URB Starting Address and Number of URB Entries fields shall be programmed as if there is only one slice enabled. When more than one slice is enabled, hardware will (a) recompute the actual URB Starting Address based on the number of enabled slices and (b) multiply the Number of URB Entries by the number of enabled slices. Software shall ensure that the values programmed do not exceed the URB capacity of a single slice. Refer to the L3 allocation and programming guide for valid URB configurations.</p>			
Project			
BDW			
Programming Notes			
<p>When programming DS URB state for the RCS 3D pipe, 3DSTATE_URB_VS, 3DSTATE_URB_HS, and 3DSTATE_URB_GS must also be programmed in order for the programming of this state to be valid.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	32h 3DSTATE_URB_DS
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_URB_DS																
	7:0	DWord Length														
		Default Value:	0h DWORD_COUNT_n													
		Project:	All													
		Format:	=n													
1	31:25	DS URB Starting Address														
		Project:	BDW													
		Format:	U7													
		Offset from the start of the URB memory where DS starts its allocation, specified in multiples of 8 KB.														
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 15%;">Project</th> <th style="width: 55%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] == 1</td> </tr> <tr> <td>[4,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] GT 1</td> </tr> </tbody> </table>				Value	Name	Project	Exists If	[0,48]		BDW	Device[SliceCount] == 1	[4,48]		BDW	Device[SliceCount] GT 1
	Value	Name	Project	Exists If												
	[0,48]		BDW	Device[SliceCount] == 1												
	[4,48]		BDW	Device[SliceCount] GT 1												
	24:16	DS URB Entry Allocation Size														
		Project:	All													
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).																
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,9]</td> <td></td> </tr> </tbody> </table>				Value	Name	[0,9]										
Value	Name															
[0,9]																
15:0	DS Number of URB Entries															
	Project:	All														
	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.		BDW													
	Project															
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,1536]</td> <td></td> <td>BDW</td> </tr> </tbody> </table>			Value	Name	Project	[0,1536]		BDW							
	Value	Name	Project													
	[0,1536]		BDW													
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

3DSTATE_URB_GS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description			
This command may not overlap with the push constants in the URB defined by the 3DSTATE_PUSH_CONSTANT_ALLOC_VS, 3DSTATE_PUSH_CONSTANT_ALLOC_DS, 3DSTATE_PUSH_CONSTANT_ALLOC_HS, and 3DSTATE_PUSH_CONSTANT_ALLOC_GS commands.			
The URB Starting Address and Number of URB Entries fields shall be programmed as if there is only one slice enabled. When more than one slice is enabled, hardware will (a) recompute the actual URB Starting Address based on the number of enabled slices and (b) multiply the Number of URB Entries by the number of enabled slices. Software shall ensure that the values programmed do not exceed the URB capacity of a single slice. Refer to the L3 allocation and programming guide for valid URB configurations			
Project			
BDW			
Programming Notes			
When programming GS URB state for the RCS 3D pipe, 3DSTATE_URB_VS, 3DSTATE_URB_HS, and 3DSTATE_URB_DS must also be programmed in order for the programming of this state to be valid.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	33h 3DSTATE_URB_GS
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_URB_GS																
	7:0	DWord Length														
		Default Value:	0h DWORD_COUNT_n													
		Project:	All													
		Format:	=n													
1	31:25	GS URB Starting Address														
		Project:	BDW													
		Format:	U7													
	Offset from the start of the URB memory where GS starts its allocation, specified in multiples of 8 KB.															
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 20%;">Project</th> <th style="width: 40%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] == 1</td> </tr> <tr> <td>[4,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] GT 1</td> </tr> </tbody> </table>				Value	Name	Project	Exists If	[0,48]		BDW	Device[SliceCount] == 1	[4,48]		BDW	Device[SliceCount] GT 1
	Value	Name	Project	Exists If												
	[0,48]		BDW	Device[SliceCount] == 1												
	[4,48]		BDW	Device[SliceCount] GT 1												
	24:16	GS URB Entry Allocation Size														
		Project:	All													
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	15:0	GS Number of URB Entries														
		Project:	All													
	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).															
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,960]</td> <td></td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Project	[0,960]		BDW						
	Value	Name	Project													
	[0,960]		BDW													
	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"															

3DSTATE_URB_HS

3DSTATE_URB_HS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description			
<p>This command may not overlap with the push constants in the URB defined by the 3DSTATE_PUSH_CONSTANT_ALLOC_VS, 3DSTATE_PUSH_CONSTANT_ALLOC_DS, 3DSTATE_PUSH_CONSTANT_ALLOC_HS, and 3DSTATE_PUSH_CONSTANT_ALLOC_GS commands.</p>			
<p>The URB Starting Address and Number of URB Entries fields shall be programmed as if there is only one slice enabled. When more than one slice is enabled, hardware will (a) recompute the actual URB Starting Address based on the number of enabled slices and (b) multiply the Number of URB Entries by the number of enabled slices. Software shall ensure that the values programmed do not exceed the URB capacity of a single slice. Refer to the L3 allocation and programming guide for valid URB configurations</p>			
Project			
BDW			
Programming Notes			
<p>When programming HS URB state for the RCS 3D pipe, 3DSTATE_URB_VS, 3DSTATE_URB_DS, and 3DSTATE_URB_GS must also be programmed in order for the programming of this state to be valid.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	31h 3DSTATE_URB_HS
		Format:	OpCode
15:8	Reserved		
	Project:	All	
	Format:	MBZ	

3DSTATE_URB_HS																
	7:0	DWord Length														
		Default Value:	0h DWORD_COUNT_n													
		Project:	All													
		Format:	=n													
1	31:25	HS URB Starting Address														
		Project:	BDW													
		Format:	U7													
		Offset from the start of the URB memory where HS starts its allocation, specified in multiples of 8 KB.														
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 20%;">Project</th> <th style="width: 40%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] == 1</td> </tr> <tr> <td>[4,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] GT 1</td> </tr> </tbody> </table>				Value	Name	Project	Exists If	[0,48]		BDW	Device[SliceCount] == 1	[4,48]		BDW	Device[SliceCount] GT 1
	Value	Name	Project	Exists If												
	[0,48]		BDW	Device[SliceCount] == 1												
	[4,48]		BDW	Device[SliceCount] GT 1												
	24:16	HS URB Entry Allocation Size														
		Project:	All													
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															
	Project:	All														
	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"															
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,504]</td> <td></td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Project	[0,504]		BDW							
Value	Name	Project														
[0,504]		BDW														



3DSTATE_URB_VS

3DSTATE_URB_VS			
Project:	BDW		
Source:	RenderCS, PositionCS		
Length Bias:	2		
Description			
VS URB Entry Allocation Size equal to 4(5 512-bit URB rows) may cause performance to decrease due to banking in the URB. Element sizes of 16 to 20 should be programmed with six 512-bit URB rows.			
This command may not overlap with the push constants in the URB defined by the 3DSTATE_PUSH_CONSTANT_ALLOC_VS, 3DSTATE_PUSH_CONSTANT_ALLOC_DS, 3DSTATE_PUSH_CONSTANT_ALLOC_HS, and 3DSTATE_PUSH_CONSTANT_ALLOC_GS commands.			
The offset and size should be programmed as if there is only one slice enabled. Hardware will grow the size based on the slice configuration. Software shall ensure that the values programmed do not exceed the URB capacity of one slice. Refer to the L3 allocation and programming guide for valid URB configurations.			
Programming Notes			
When programming VS URB state for the RCS 3D pipe, 3DSTATE_URB_HS, 3DSTATE_URB_DS, and 3DSTATE_URB_GS must also be programmed in order for the programming of this state to be valid.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	30h 3DSTATE_URB_VS
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_URB_VS																
	7:0	DWord Length														
		Default Value:	0h DWORD_COUNT_n													
		Project:	All													
		Format:	=n													
1	31:25	VS URB Starting Address														
		Project:	BDW													
		Format:	U7													
		Offset from the start of the URB memory where VS starts its allocation, specified in multiples of 8 KB.														
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 20%;">Project</th> <th style="width: 40%;">Exists If</th> </tr> </thead> <tbody> <tr> <td>[0,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] == 1</td> </tr> <tr> <td>[4,48]</td> <td></td> <td>BDW</td> <td>Device[SliceCount] GT 1</td> </tr> </tbody> </table>				Value	Name	Project	Exists If	[0,48]		BDW	Device[SliceCount] == 1	[4,48]		BDW	Device[SliceCount] GT 1
	Value	Name	Project	Exists If												
	[0,48]		BDW	Device[SliceCount] == 1												
	[4,48]		BDW	Device[SliceCount] GT 1												
	24:16	VS URB Entry Allocation Size														
		Project:	All													
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 border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[0,9]</td> <td></td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Project	[0,9]		BDW							
Value	Name	Project														
[0,9]		BDW														
Programming Notes																
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.																
15:0	VS Number of URB Entries															
	Project:	All														
	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 border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[64,2560]</td> <td></td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Project	[64,2560]		BDW						
	Value	Name	Project													
	[64,2560]		BDW													
	Programming Notes			Project												
	Programming Restriction: VS Number of URB Entries must be divisible by 8 if the VS URB Entry Allocation Size is less than 9 512-bit URB entries."2:0" = reserved "000b"															
When tessellation is enabled, the VS Number of URB Entries must be greater than or equal to 192.			BDW													



3DSTATE_VERTEX_BUFFERS

3DSTATE_VERTEX_BUFFERS			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Description			
This command is used to specify VB state used by the VF function.			
[BDW]: Can specify from 1 to 33 VBs.	BDW		
The VertexBufferID field within a VERTEX_BUFFER_STATE structure indicates the specific VB. If a VB definition is not included in this command, its associated state is left unchanged and is available for use if previously defined.			
Programming Notes			
It is possible to have individual vertex elements sourced completely from generated ID values and therefore not require any vertex buffer accesses for that vertex element. In this case, VF function will simply ignore the VB state associated with that vertex element. If all enabled vertex elements have this characteristic, no VBs are required to process 3DPRIMITIVE commands. For example, this might arise when the user wants to perform all data lookups in the first shader, so only generated index values need to be passed down to it. In this extreme case, SW would not need to program any VB state, and therefore not need to issue any 3DSTATE_VERTEX_BUFFERS commands.			
For any 3DSTATE_VERTEX_BUFFERS command, at least one VERTEX_BUFFER_STATE structure must be included.			
VERTEX_BUFFER_STATE structures are 4 DWords for both VERTEXDATA buffers and INSTANCEDATA buffers.			
Inclusion of partial VERTEX_BUFFER_STATE structures is UNDEFINED.			
The order in which VBs are defined within this command can be arbitrary, though a vertex buffer must be defined only once in any given command (otherwise operation is UNDEFINED).			
The VF cache (and the VFR cache if POSH is enabled) needs to be invalidated before binding and then using Vertex Buffers that overlap with any previously bound Vertex Buffer (at a 64B granularity) since the last invalidation.			
A VF cache invalidate is performed by setting the "VF Cache Invalidation Enable" bit in PIPE_CONTROL.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	03h GFXPIPE
		Format:	Opcode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	Opcode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	Opcode

3DSTATE_VERTEX_BUFFERS									
	23:16	3D Command Sub Opcode							
		Default Value: 08h 3DSTATE_VERTEX_BUFFERS							
		Format: Opcode							
	15:8	Reserved							
	7:0	DWord Length							
Format: =n									
n = 4b-1 (where b = # of buffer states included)									
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">3</td> <td>DWORD_COUNT_n [Default]</td> <td></td> </tr> <tr> <td style="text-align: center;">[3,131]</td> <td>1-33 Buffers</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>		Value	Name	Project	3	DWORD_COUNT_n [Default]		[3,131]	1-33 Buffers
Value	Name	Project							
3	DWORD_COUNT_n [Default]								
[3,131]	1-33 Buffers	BDW							
1..n	127:0	Vertex Buffer State							
		Format: VERTEX_BUFFER_STATE							



3DSTATE_VERTEX_ELEMENTS

3DSTATE_VERTEX_ELEMENTS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
Description		Project
This is a variable-length command used to specify the active vertex elements. Each VERTEX_ELEMENT_STATE structure contains a Valid bit which determines which elements are used.		BDW
[BDW]: Up to 34 elements.		BDW
Programming Notes		Project
[BDW]: At least one VERTEX_ELEMENT_STATE structure must be included.		BDW
The 3DSTATE_VERTEX_ELEMENTS must not be programmed more than once before each 3DPRIMITIVE command.		BDW
Inclusion of partial VERTEX_ELEMENT_STATE structures is UNDEFINED.		
[BDW]: SW must ensure that at least one vertex element is defined prior to issuing a 3DPRIMITIVE command, or operation is UNDEFINED.		BDW
[BDW]: There are no 'holes' allowed in the destination vertex: NOSTORE components must be overwritten by subsequent components unless they are the trailing DWords of the vertex. Software must explicitly chose some value (probably 0) to be written into DWords that would otherwise be 'holes'.		BDW
[BDW]: Within a VERTEX_ELEMENT_STATE structure, if a Component Control field is set to something other than VFCOMP_STORE_SRC, no higher-numbered Component Control fields may be set to VFCOMP_STORE_SRC. In other words, only trailing components can be set to something other than VFCOMP_STORE_SRC.		BDW
[BDW]: See additional restrictions listed in the command fields and VERTEX_ELEMENT_STATE description.		BDW
[BDW]: Element[0] must be valid.		BDW
[BDW]: All elements must be valid from Element[0] to the last valid element. (I.e. if Element[2] is valid then Element[1] and Element[0] must also be valid).		BDW
[BDW]: The pitch between elements packed in the URB will always be 128 bits.		BDW
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 03h GFXPIPE
	Format: Opcode	
	28:27	Command SubType
Default Value: 3h 3D		
Format: Opcode		

3DSTATE_VERTEX_ELEMENTS				
	26:24	3D Command Opcode		
		Default Value:	0h 3DSTATE_PIPELINED	
	Format:		Opcode	
	23:16	3D Command Sub Opcode		
		Default Value:	09h 3DSTATE_VERTEX_ELEMENTS	
	Format:		Opcode	
	15:8	Reserved		
	7:0	DWord Length		
		Format:		=n
		Vertex Element Count = (DWord Count + 1) / 2		
Value		Name	Description	
Project				
1	DWORD_COUNT_n [Default]	excludes DWords 0,1		
[1,67]	Range	1-34 Elements	BDW	
1..n	63:0	Element		
Format:		VERTEX_ELEMENT_STATE [BDW] [BDW]		

3DSTATE_VF

3DSTATE_VF			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command is used to set various state variables in the VF stage.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	0Ch 3DSTATE_VF
		Format:	OpCode
	15:13	Reserved	
		Project:	All
		Format:	MBZ
	12	Reserved	
		Project:	BDW
	11	Reserved	
		Project:	BDW
		Format:	MBZ
	10	Reserved	
		Project:	BDW
		Format:	MBZ
	9	Reserved	
Project:		BDW	
Format:		MBZ	

3DSTATE_VF								
	8	<p>Indexed Draw Cut Index Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If ENABLED, vertex indices in RANDOM 3DPRIMITIVE commands are compared to the Cut Index (specified below). When the vertex index matches the Cut Index any previous topology is terminated. If DISABLED, vertex indices are not compared to the Cut Index and are used strictly as indices into vertex buffers. This field can only be enabled for certain primitive topology types. Refer to the table later in this section for details.</p>	Project:	All	Format:	Enable		
	Project:	All						
Format:	Enable							
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h Excludes DWord (0,1)</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0h Excludes DWord (0,1)	Project:	All	Format:	=n Total Length - 2
Default Value:	0h Excludes DWord (0,1)							
Project:	All							
Format:	=n Total Length - 2							
1	31:0	<p>Cut Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> </table> <p>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.</p>	Project:	All				
Project:	All							

3DSTATE_VF_INSTANCING

3DSTATE_VF_INSTANCING			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command is used to control the "instancing" state associated with a specific vertex element.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		49h 3DSTATE_VF_INSTANCING	
Format:		OpCode	
15:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	
	1h	Excludes DWord (0,1) [Default]	
	43h	Context Restore	
1	31:9	Reserved	
		Format:	MBZ

3DSTATE_VF_INSTANCING		
8	Instancing Enable	
	Format: Enable	
	Value	Name
	Description	
0h	Disabled	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 the Vertex Access Type of the 3DPRIMITIVE command. Instance Data Step Rate is ignored for this vertex element.
1h	Enabled	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	Name
[0,33]		
2	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 3DPRIMITIVE 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

3DSTATE_VF_SGVS		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command is used to control the insertion of the VertexID and InstanceID System-Generated Values (SGVs) into an input Vertex URB Entry (VUE) (available as input to a VS thread). VertexID and InstanceID insertion can be individually controlled. The insertion locations are specified as 128-bit element locations (starting at the beginning of the VUE) and the 32-bit component within those specified elements. The SGV values can be inserted either (a) within a valid vertex element (in which case the value overwrites the value specified via 3DSTATE_VERTEX_ELEMENTS) or (b) beyond the last valid vertex element written to the URB. This permits some orthogonality between the programming of vertex elements (which typically is known at draw time) and programming of SGV insertion (which is associated with the shader). There are some restrictions however (see below). If an SGV is inserted beyond the last valid vertex element, zeroes are first inserted in the VUE after the last valid vertex element up to and including the vertex element receiving an SGV. If both of the SGVs are enabled for insertion, the zeroes will extend to the last (largest index) vertex element receiving an SGV. Then the SGV(s) are inserted.</p>		
Programming Notes		
<p>Programming Restrictions:</p> <ul style="list-style-type: none"> It is INVALID to store both the VertexID and InstanceID in the same element/component location within the VUE. The states programmed by this command overwrite the state programmed by any previous commands. I.e., VertexID and InstanceID (if enabled) can only be inserted in one component of a vertex. It is INVALID to insert an SGV value past the end of the VUE entry (as determined by VS URB Entry Allocation Size) or past the 33rd vertex element. Therefore the programming of VS URB Entry Allocation Size needs to comprehend any SGV insertion requirements. It is INVALID to use this command to overwrite any portion of a 64-bit vertex element component. It is INVALID to use this command to overwrite a EdgeFlag vertex element component or any vertex element beyond it. 		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
	Format: OpCode	
	28:27	Command SubType
Default Value: 3h GFXPIPE_3D		
Format: OpCode		

3DSTATE_VF_SGVS				
1	26:24	3D Command Opcode		
		Default Value:	0h 3DSTATE_PIPELINED	
		Format:	OpCode	
	23:16	3D Command Sub Opcode		
		Default Value:	4Ah 3DSTATE_VF_SGVS	
		Format:	OpCode	
	15:8	Reserved		
		Project:	All	
		Format:	MBZ	
	7:0	DWord Length		
		Default Value:	0h Excludes DWord (0,1)	
		Project:	All	
Format:		=n Total Length - 2		
31	InstancelD Enable			
	Project:	All		
	Format:	Boolean		
	Value	Name	Description	Project
	0h	Disabled	InstancelD is not inserted	All
	1h	Enabled	InstancelD is inserted	All
	30:29	InstancelD Component Number		
		Project:	All	
		If InstancelD Enable is ENABLED, this field specifies the 32-bit component location (within the 4-component VUE) where it is inserted.		
		If InstancelD Enable is DISABLED, this field is ignored.		
Value		Name	Description	Project
0		COMP_0	If enabled, InstancelD is inserted in component 0 (.x)	All
1	COMP_1	If enabled, InstancelD is inserted in component 1 (.y)	All	
2	COMP_2	If enabled, InstancelD is inserted in component 2 (.z)	All	
3	COMP_3	If enabled, InstancelD is inserted in component 3 (.w)	All	
28:22	Reserved			
	Project:	All		
	Format:	MBZ		

3DSTATE_VF_SGVS				
21:16	InstanceID Element Offset			
	Project:	All		
	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			
	Project:	All		
	Format:	Boolean		
	Value	Name	Description	Project
	0h	Disabled	VertexID is not inserted	All
1h	Enabled	VertexID is inserted	All	
14:13	VertexID Component Number			
	Project:	All		
	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	Project
	0	COMP_0	If enabled, VertexID is inserted in component 0 (.x)	All
1	COMP_1	If enabled, VertexID is inserted in component 1 (.y)	All	
2	COMP_2	If enabled, VertexID is inserted in component 2 (.z)	All	
3	COMP_3	If enabled, VertexID is inserted in component 3 (.w)	All	
12:6	Reserved			
	Project:	All		
	Format:	MBZ		
5:0	VertexID Element Offset			
	Project:	All		
	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_STATISTICS

3DSTATE_VF_STATISTICS					
Project:	DevBWR+				
Source:	RenderCS				
Length Bias:	1				
<p>The VF stage tracks two pipeline statistics, the number of vertices fetched and the number of objects generated. VF will increment the appropriate counter for each when statistics gathering is enabled by issuing the 3DSTATE_VF_STATISTICS command with the [Statistics Enable] bit set.</p>					
DWord	Bit	Description			
0	31:29	Command Type			
		Default Value:	3h GFXPIPE		
		Format:	Opcode		
	28:27	Command SubType			
		Format:	Opcode		
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>GFXPIPE_SINGLE_DW [Default]</td> </tr> </tbody> </table>	Value	Name	1h
	Value	Name			
	1h	GFXPIPE_SINGLE_DW [Default]			
	26:24	3D Command Opcode			
		Default Value:	0h 3DSTATE_PIPELINED		
Format:		Opcode			
		GFXPIPE[28:27 = 1h, 26:24 = 0h, 23:16 = 0Bh] (Pipelined, Single DWord)			
23:16	3D Command Sub Opcode				
	Default Value:	0Bh 3DSTATE_VF_STATISTICS			
	Format:	Opcode			
		GFXPIPE[28:27 = 1h, 26:24 = 0h, 23:16 = 0Bh] (Pipelined, Single DWord)			
15:1	Reserved				
	Format:	MBZ			
0	Statistics Enable				
	Format:	Enable			
		If ENABLED, VF will increment the pipeline statistics counters IA_VERTICES_COUNT and IA_PRIMITIVES_COUNT for each vertex fetched and each object output, respectively, for 3DPRIMITIVE commands issued subsequently. If DISABLED, these counters will not be incremented for subsequent 3DPRIMITIVE commands.			

3DSTATE_VF_TOPOLOGY

3DSTATE_VF_TOPOLOGY		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
This command specifies the VF stage's Topology state which can be used to override the Primitive Topology Type in subsequent 3DPRIMITIVE commands.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h GFXPIPE
		Format: OpCode
	28:27	Command SubType
		Default Value: 3h GFXPIPE_3D
		Format: OpCode
	26:24	3D Command Opcode
Default Value: 0h 3DSTATE_PIPELINED		
Format: OpCode		
23:16	3D Command Sub Opcode	
	Default Value: 4Bh 3DSTATE_VF_TOPOLOGY	
	Format: OpCode	
15:8	Reserved	
	Project: All	
	Format: MBZ	
7:0	DWord Length	
	Default Value: 0h Excludes DWord (0,1)	
	Project: All	
	Format: =n Total Length - 2	
1	31:6	Reserved
		Project: All
		Format: MBZ
	5:0	Primitive Topology Type
Project: All		
Format: 3D_Prim_Topo_Type		
This field specifies the VF stage's Topology state.		

3DSTATE_VIEWPORT_STATE_POINTERS_CC

DWord		Bit	Description
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
The 3DSTATE_VIEWPORT_STATE_POINTERS_CC command is used to define the location of fixed functions' viewport state table.			
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
	Format:	OpCode	
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	Format:	OpCode	
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:	OpCode		
23:16	3D Command Sub Opcode		
	Default Value:	23h 3DSTATE_VIEWPORT_STATE_POINTERS_CC	
Format:	OpCode		
15:8	Reserved		
	Project:	All	
Format:	MBZ		
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
Format:	=n		
1	31:5	CC Viewport Pointer	
		Project:	All
	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		
	Project:	All	
Format:	MBZ		

3DSTATE_VIEWPORT_STATE_POINTERS_SF_CLIP

DWord		Bit	Description
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
The 3DSTATE_VIEWPORT_STATE_POINTERS_CLIP command is used to define the location of fixed functions' viewport state table.			
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
	Format:	OpCode	
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	Format:	OpCode	
	26:24	3D Command Opcode	
Default Value:		0h 3DSTATE_PIPELINED	
Format:	OpCode		
23:16	3D Command Sub Opcode		
	Default Value:	21h 3DSTATE_VIEWPORT_STATE_POINTERS_SF_CLIP	
Format:	OpCode		
15:8	Reserved		
	Project:	All	
Format:	MBZ		
7:0	DWord Length		
	Default Value:	0h DWORD_COUNT_n	
Format:	=n		
1	31:6	SF Clip Viewport Pointer	
		Project:	All
	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		
	Project:	BDW	
Format:	MBZ		

3DSTATE_VS

3DSTATE_VS			
Project:	BDW		
Source:	RenderCS, PositionCS		
Length Bias:	2		
Description		Project	
This command specifies most of the state used by the Vertex Shader (VS) stage.		BDW	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
23:16	3D Command Sub Opcode		
	Default Value:	10h 3DSTATE_VS	
15:8	Reserved		
	Project:	All	
7:0	7:0	DWord Length	
		Default Value:	7h Excludes DWord (0,1)
		Project:	All
		Format:	=n Total Length - 2
1..2	63:6	Kernel Start Pointer	
		Project:	All
		Format:	InstructionBaseOffset[63:6]Kernel
<p>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.</p>			

3DSTATE_VS				
	5:0	Reserved		
		Project:	All	
		Format:	MBZ	
3	31	Single Vertex Dispatch		
		Project:	BDW	
	Format:	U1 Enumerated Type		
	When this bit is set, SIMD4x2 VS threads will only process a single vertex, otherwise SIMD4x2 threads will process either one or two vertices. This field is ignored if SIMD8 Dispatch Enable is set.			
	Value	Name	Description	Project
	0h	Multiple	Dual vertex SIMD4x2 thread dispatches are allowed.	All
	1h	Single	Single vertex SIMD4x2 thread dispatches are forced.	All
	30	Vector Mask Enable		
		Project:	All	
	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.			
Value	Name	Description	Project	
0h	Dmask	The EU will use the Dispatch Mask (supplied by the VS stage) for instruction execution.	All	
1h	Vmask	The EU will use the Vector Mask (derived from the Dispatch Mask) for instruction execution.	All	
Programming Notes				
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).				

3DSTATE_VS																									
29:27	Sampler Count																								
	Project: All																								
	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.																								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>No Samplers</td> <td>no samplers used</td> <td>All</td> </tr> <tr> <td>1h</td> <td>1-4 Samplers</td> <td>between 1 and 4 samplers used</td> <td>All</td> </tr> <tr> <td>2h</td> <td>5-8 Samplers</td> <td>between 5 and 8 samplers used</td> <td>All</td> </tr> <tr> <td>3h</td> <td>9-12 Samplers</td> <td>between 9 and 12 samplers used</td> <td>All</td> </tr> <tr> <td>4h</td> <td>13-16 Samplers</td> <td>between 13 and 16 samplers used</td> <td>All</td> </tr> </tbody> </table>	Value	Name	Description	Project	0h	No Samplers	no samplers used	All	1h	1-4 Samplers	between 1 and 4 samplers used	All	2h	5-8 Samplers	between 5 and 8 samplers used	All	3h	9-12 Samplers	between 9 and 12 samplers used	All	4h	13-16 Samplers	between 13 and 16 samplers used	All
	Value	Name	Description	Project																					
	0h	No Samplers	no samplers used	All																					
	1h	1-4 Samplers	between 1 and 4 samplers used	All																					
	2h	5-8 Samplers	between 5 and 8 samplers used	All																					
3h	9-12 Samplers	between 9 and 12 samplers used	All																						
4h	13-16 Samplers	between 13 and 16 samplers used	All																						
26	Reserved																								
	Project: All																								
	Format: MBZ																								
25:18	Binding Table Entry Count																								
	Project: All																								
	Format: U8																								
	<table border="1"> <thead> <tr> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>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.</td> <td></td> </tr> <tr> <td>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.</td> <td>BDW</td> </tr> </tbody> </table>	Description	Project	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.	BDW																		
	Description	Project																							
	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.	BDW																							
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>[0,255]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0,255]																					
	Value	Name																							
	[0,255]																								
<table border="1"> <thead> <tr> <th>Programming Notes</th> <th>Project</th> </tr> </thead> <tbody> <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> <td>BDW</td> </tr> </tbody> </table>	Programming Notes	Project	When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.	BDW																					
Programming Notes	Project																								
When HW binding table bit is set, it is assumed that the Binding Table Entry Count field will be generated at JIT time.	BDW																								

3DSTATE_VS				
17	Thread Dispatch Priority			
	Project:	All		
	Format:	U1 Enumerated Type		
	Specifies the priority of the thread for dispatch: This field is ignored if VS Function Enable is DISABLED.			
	Value	Name	Description	Project
0h	Normal	Normal Priority	All	
1h	High	High Priority	All	
16	Floating Point Mode			
	Project:	All		
	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	Project
0h	IEEE-754	Use IEEE-754 Rules	All	
1h	Alternate	Use Alternate Rules	All	
15:14	Reserved			
	Project:	All		
	Format:	MBZ		
13	Illegal Opcode Exception Enable			
	Project:	All		
	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			Project
	This field must not be set when VS Function Enable is disabled.			
Workaround: If the vertex shader is the last shader to have UAV access, a PIPE_CONTROL with CS_STALL must be sent before the 3dprimitive using the UAV access.			BDW	
11:8	Reserved			
	Project:	All		
	Format:	MBZ		

3DSTATE_VS											
	7	<p>Software Exception Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p>	Project:	All	Format:	Enable					
	Project:	All									
Format:	Enable										
	6:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ					
Project:	All										
Format:	MBZ										
4..5	63:10	<p>Scratch Space Base Pointer</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GeneralStateOffset[63:10]ScratchSpace</td> </tr> </table> <p>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.</p>	Project:	All	Format:	GeneralStateOffset[63:10]ScratchSpace					
		Project:	All								
		Format:	GeneralStateOffset[63:10]ScratchSpace								
9:4	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All										
Format:	MBZ										
3:0	<p>Per-Thread Scratch Space</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U4 power of 2 Bytes over 1K Bytes</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td></td> <td>Indicating [1K Bytes, 2M Bytes]</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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.</p>	Project:	All	Format:	U4 power of 2 Bytes over 1K Bytes	Value	Name	Description	[0,11]		Indicating [1K Bytes, 2M Bytes]
Project:	All										
Format:	U4 power of 2 Bytes over 1K Bytes										
Value	Name	Description									
[0,11]		Indicating [1K Bytes, 2M Bytes]									

3DSTATE_VS												
6	31:25	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
	Format:	MBZ										
	24:20	<p>Dispatch GRF Start Register For URB Data</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> <td>indicating GRF [R0, R31]</td> </tr> </tbody> </table>	Project:	All	Format:	U5	Value	Name	Description	[0,31]		indicating GRF [R0, R31]
	Project:	All										
	Format:	U5										
	Value	Name	Description									
	[0,31]		indicating GRF [R0, R31]									
	19:17	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
	Project:	All										
	Format:	MBZ										
16:11	<p>Vertex URB Entry Read Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U6</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>[1,63]</td> <td></td> <td>if SIMD8 dispatch disabled</td> </tr> <tr> <td>[0,15]</td> <td></td> <td>if SIMD8 dispatch enabled</td> </tr> </tbody> </table>	Format:	U6	Value	Name	Description	[1,63]		if SIMD8 dispatch disabled	[0,15]		if SIMD8 dispatch enabled
Format:	U6											
Value	Name	Description										
[1,63]		if SIMD8 dispatch disabled										
[0,15]		if SIMD8 dispatch enabled										
10	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ							
Project:	All											
Format:	MBZ											
9:4	<p>Vertex URB Entry Read Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,63]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U6	Value	Name	[0,63]				
Project:	All											
Format:	U6											
Value	Name											
[0,63]												

3DSTATE_VS					
7	3:0	Reserved			
		Project:	All		
		Format:	MBZ		
	31:23	Maximum Number of Threads			
		Project:	BDW		
		Format:	U9-1 Thread count		
	<p>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.</p>				
		Value	Name	Description	Project
		[0,503]		indicating thread count of [1,504]	BDW
	22	Reserved			
	Project:	BDW			
	Format:	MBZ			
21:13	Reserved				
	Project:	All			
	Format:	MBZ			
12:11	Reserved				
	Project:	BDW			
	Format:	MBZ			
10	Statistics Enable				
	Project:	All			
	Format:	Enable			
<p>If ENABLED, the VS stage will perform statistics gathering. See the Statistics Gathering subsection.</p> <p>If DISABLED, statistics information associated with the VS stage will be left unchanged.</p>					
9	Reserved				
	Project:	BDW			
	Format:	MBZ			
8:3	Reserved				
	Project:	All			
	Format:	MBZ			

3DSTATE_VS						
2	<p>SIMD8 Dispatch Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <p>If ENABLED, SIMD8 VS thread dispatches are performed. The Single Vertex Dispatch field is ignored.</p> <p>If DISABLED, SIMD4x2 thread dispatches are performed. The Single Vertex Dispatch field can be used to force single-vertex dispatches.</p>	Project:	All	Format:	Enable	
	Project:	All				
	Format:	Enable				
<p>Vertex Cache Disable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Disable</td> </tr> </table> <p>This bit controls the operation of the Vertex Cache. This field is always used.</p> <p>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.</p> <p>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.</p> <p>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).</p> <p>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.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>See the Vertex Caching subsection for details on implicit Vertex Cache disabling.</td> </tr> </table>	Project:	All	Format:	Disable	Programming Notes	See the Vertex Caching subsection for details on implicit Vertex Cache disabling.
Project:	All					
Format:	Disable					
Programming Notes						
See the Vertex Caching subsection for details on implicit Vertex Cache disabling.						
<p>Function Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>This bit determines whether or not the VS stage spawns VS threads, which comprises the bulk of the VS stage functionality.</p> <p>If ENABLED, VS threads may be spawned to process VF-generated vertices before the resulting vertices are passed down the pipeline.</p> <p>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.</p>	Format:	Enable				
Format:	Enable					
8	<p>31:28 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
	Project:	All				
Format:	MBZ					
<p>27 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
Project:	BDW					
Format:	MBZ					

3DSTATE_VS									
26:21	<p>Vertex URB Entry Output Read Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,63]</td> <td></td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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.</p> <p>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)</p>	Project:	All	Format:	U6	Value	Name	[0,63]	
Project:	All								
Format:	U6								
Value	Name								
[0,63]									
20:16	<p>Vertex URB Entry Output Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,16]</td> <td></td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>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).</p>	Project:	All	Format:	U5	Value	Name	[1,16]	
Project:	All								
Format:	U5								
Value	Name								
[1,16]									
15:8	<p>User Clip Distance Clip Test Enable Bitmask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>mask[8]</td> </tr> </table> <p>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.</p> <p>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.</p>	Project:	All	Format:	mask[8]				
Project:	All								
Format:	mask[8]								



3DSTATE_VS					
7:0	<p>User Clip Distance Cull Test Enable Bitmask</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>mask[8]</td> </tr> </table> <p>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.</p> <p>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.</p>	Project:	All	Format:	mask[8]
Project:	All				
Format:	mask[8]				

3DSTATE_WM_CHROMAKEY

3DSTATE_WM_CHROMAKEY			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		4Ch 3DSTATE_WM_CHROMAKEY	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	Dword Length		
	Default Value:	0h Excludes Dword (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31	ChromaKey Kill Enable	
		Project:	All
		Format:	Enable
	If ENABLED, indicates that at least one of the attached samplers has ChromaKeyKill enabled.		
	30:0	Reserved	
Project:		All	
Format:		MBZ	

3DSTATE_WM_DEPTH_STENCIL

3DSTATE_WM_DEPTH_STENCIL			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command replaces the indirect state DEPTH_STENCIL_STATE with an inline state command.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		4Eh 3DSTATE_WM_DEPTH_STENCIL	
Format:		OpCode	
15:8	Reserved		
	Project:	BDW	
	Format:	MBZ	
7:0	Dword Length	Project:	All
		Format:	=n
		Total Length - 2	
	Value	Name	Project
	01h	Excludes Dword (0,1) [Default]	BDW
1	31:29	Stencil Fail Op	
		Project:	All
		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.			

3DSTATE_WM_DEPTH_STENCIL					
28:26	<p>Stencil Pass Depth Fail Op</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Stencil_Operation</td> </tr> </table> <p>This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes but the depth pass fails.</p>	Project:	All	Format:	3D_Stencil_Operation
Project:	All				
Format:	3D_Stencil_Operation				
25:23	<p>Stencil Pass Depth Pass Op</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Stencil_Operation</td> </tr> </table> <p>This field specifies the operation to perform on the Stencil Buffer when the (front face) stencil test passes but the depth test passes.</p>	Project:	All	Format:	3D_Stencil_Operation
Project:	All				
Format:	3D_Stencil_Operation				
22:20	<p>Backface Stencil Test Function</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Compare_Function</td> </tr> </table>	Project:	All	Format:	3D_Compare_Function
Project:	All				
Format:	3D_Compare_Function				
19:17	<p>Backface Stencil Fail Op</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Stencil_Operation</td> </tr> </table>	Project:	All	Format:	3D_Stencil_Operation
Project:	All				
Format:	3D_Stencil_Operation				
16:14	<p>Backface Stencil Pass Depth Fail Op</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Stencil_Operation</td> </tr> </table> <p>This field specifies the operation to perform on the Stencil Buffer when the stencil test passes but the depth pass fails.</p>	Project:	All	Format:	3D_Stencil_Operation
Project:	All				
Format:	3D_Stencil_Operation				
13:11	<p>Backface Stencil Pass Depth Pass Op</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Stencil_Operation</td> </tr> </table> <p>This field specifies the operation to perform on the Stencil Buffer when the stencil test passes and the depth pass passes (or is disabled).</p>	Project:	All	Format:	3D_Stencil_Operation
Project:	All				
Format:	3D_Stencil_Operation				
10:8	<p>Stencil Test Function</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>3D_Compare_Function</td> </tr> </table> <p>This field specifies the comparison function used in the (front face) StencilTest function.</p>	Project:	All	Format:	3D_Compare_Function
Project:	All				
Format:	3D_Compare_Function				

3DSTATE_WM_DEPTH_STENCIL					
7:5	Depth Test Function				
	Project:	All			
	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			
		Project:	All		
		Format:	Enable		
		Enable doubled sided stencil operations.			
		Value	Name	Description	Project
		0h	False	Double Sided Stencil Disabled	All
1h		True	Double Sided Stencil Enabled	All	
Programming Notes					
<ul style="list-style-type: none"> • 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			
	Project:	All			
	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				
	Project:	All			
	Format:	Enable			
	Enables writes to the Stencil Buffer.				
	Programming Notes				
If this field is enabled, Stencil Test Enable must also be enabled.					

3DSTATE_WM_DEPTH_STENCIL		
1	Depth Test Enable	
	Project:	All
	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	
	Project:	All
	Format:	Enable
Enables writes to the Depth Buffer.		
Programming Notes		Project
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.		BDW
Workaround		Project
If Depth_Test_Enable = 1 AND Depth_Test_func = EQUAL, the Depth_Write_Enable must be set to 0		BDW
2	Stencil Test Mask	
	Project:	All
	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	
	Project:	All
	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.		



3DSTATE_WM_DEPTH_STENCIL					
15:8	<p>Backface Stencil Test Mask</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>This field specifies a bit mask applied to backface stencil test values. Both the stencil reference value and value read from the stencil buffer will be logically ANDed with this mask before the stencil comparison test is performed.</p>	Project:	All	Format:	U8
	Project:	All			
Format:	U8				
7:0	<p>Backface Stencil Write Mask</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>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.</p>	Project:	All	Format:	U8
	Project:	All			
Format:	U8				

3DSTATE_WM

		3DSTATE_WM	
Project:		BDW	
Source:		RenderCS	
Length Bias:		2	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		14h 3DSTATE_WM	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
	Total Length - 2		
1	31	Statistics Enable	
		Project:	All
		Format:	Enable
	<p>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.</p>		
	<p style="text-align: center;">Programming Notes</p> <p>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.</p>		

3DSTATE_WM					
30	<p>Legacy Depth Buffer Clear Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When set, the depth buffer is initialized as a side-effect of rendering pixels.</p> <div style="background-color: #e6f2ff; text-align: center; padding: 2px;">Programming Notes</div> <p>If this field is enabled,</p> <ol style="list-style-type: none"> 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. <ol style="list-style-type: none"> 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 <p>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.</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
29	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				
28	<p>Legacy Depth Buffer Resolve Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <div style="background-color: #e6f2ff; text-align: center; padding: 2px;">Programming Notes</div> <p>If this field is enabled,</p> <ol style="list-style-type: none"> 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. <p>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.</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				

3DSTATE_WM						
27	Legacy Hierarchical Depth Buffer Resolve Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>If this field is enabled,</p> <ol style="list-style-type: none"> 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. <p>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.</p>	Project:	All	Format:	Enable	Programming Notes
	Project:	All				
	Format:	Enable				
Programming Notes						
26	Legacy Diamond Line Rasterization <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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).</p>	Project:	All	Format:	Enable	
	Project:	All				
	Format:	Enable				
25:23	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					

3DSTATE_WM				
22:21	Early Depth/Stencil Control			
	Project:	BDW		
	Format:	U2 Enumerated Type		
	This field specifies the behavior of early depth/stencil test.			
	Value	Name	Description	Project
	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)	All
	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)	All
	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.	All
	3h	Reserved		All
	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			
	Project:	All		
	Value	Name	Description	Project
	0h	Normal	WM_INT::ThreadDispatchEnable is computed normally	All
	1h	ForceOff	Forces WM_INT::ThreadDispatchEnable Off	All
	2h	ForceON	Forces WM_INT::ThreadDispatchEnable On	All
	3h	Reserved		All
	Programming Notes			
	This must always be set to Normal.			

3DSTATE_WM																						
18:17	Position ZW Interpolation Mode																					
	Project: All																					
	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).																					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>INTERP_PIXEL</td> <td>Evaluate Z & W at the pixel center or UL corner (as specified by Pixel Location of 3DSTATE_MULTISAMPLE)</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Reserved</td> <td></td> <td>All</td> </tr> <tr> <td>2h</td> <td>INTERP_CENTROID</td> <td></td> <td>All</td> </tr> <tr> <td>3h</td> <td>INTERP_SAMPLE</td> <td></td> <td>All</td> </tr> </tbody> </table>		Value	Name	Description	Project	0h	INTERP_PIXEL	Evaluate Z & W at the pixel center or UL corner (as specified by Pixel Location of 3DSTATE_MULTISAMPLE)	All	1h	Reserved		All	2h	INTERP_CENTROID		All	3h	INTERP_SAMPLE		All
	Value	Name	Description	Project																		
	0h	INTERP_PIXEL	Evaluate Z & W at the pixel center or UL corner (as specified by Pixel Location of 3DSTATE_MULTISAMPLE)	All																		
	1h	Reserved		All																		
	2h	INTERP_CENTROID		All																		
	3h	INTERP_SAMPLE		All																		
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																					
	Project: All																					
	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																					
	<table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td colspan="2">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.</td> </tr> </tbody> </table>		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.																	
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																					
	Project: All																					
	Format: MBZ																					

3DSTATE_WM					
9:8	Line End Cap Antialiasing Region Width				
	Project:		All		
	Format:		U2		
	This field specifies the distances over which the coverage of anti-aliased line end caps are computed.				
	Value	Name	Description	Project	
	0h	0.5 pixels	0.5 pixels	All	
	1h	1.0 pixels	1.0 pixels	All	
	2h	2.0 pixels	2.0 pixels	All	
	3h	4.0 pixels	4.0 pixels	All	
	7:6	Line Antialiasing Region Width			
		Project:		All	
		Format:		U2	
		This field specifies the distance over which the anti-aliased line coverage is computed.			
		Value	Name	Description	Project
		0h	0.5 pixels	0.5 pixels	All
1h		1.0 pixels	1.0 pixels	All	
2h		2.0 pixels	2.0 pixels	All	
3h		4.0 pixels	4.0 pixels	All	
5		Reserved			
	Project:		All		
	Format:		MBZ		
4	Polygon Stipple Enable				
	Project:		All		
	Format:		Enable		
Enables the Polygon Stipple function.					
3	Line Stipple Enable				
	Project:		All		
	Format:		Enable		
Enables the Line Stipple function.					

3DSTATE_WM			
2	Point Rasterization Rule		
	Project:	All	
	Format:	3D_RasterizationRule	
	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		
	Project:	All	
	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		
	This must always be set to Normal.		



3DSTATE_WM_HZ_OP

3DSTATE_WM_HZ_OP			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
This command provides for clearing Z and/or stencil or resolving either HZ buffer or Z buffer.			
Programming Notes		Project	
3DSTATE_MULTISAMPLE packet must be used prior to this packet to change the Number of Multisamples. This packet must not be used to change Number of Multisamples in a rendering sequence. 3DSTATE_RASTER if used must be programmed prior to using this packet.		BDW	
This command does support predication from the use of the MI_PREDICATE register. To predicate depth clears and resolves on [BDW] you must fall back to using the 3D_PRIMITIVE or GPGPU_WALKER commands.		BDW	
As this command generates an implicit rectangle, SW must make sure any MMIO register writes following WM_HZ_OP must be preceded by PIPE_CONTROL with Command Streamer Stall Enable bit set.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	0h 3DSTATE_PIPELINED
		Format:	OpCode
	23:16	3D Command Sub Opcode	
		Default Value:	52h 3DSTATE_WM_HZ_OP
		Format:	OpCode
	15:8	Reserved	
		Project:	All
		Format:	MBZ

3DSTATE_WM_HZ_OP										
	7:0	<p>Dword Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>03h Excludes Dword (0,1)</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2</p>	Default Value:	03h Excludes Dword (0,1)	Project:	All	Format:	=n		
Default Value:	03h Excludes Dword (0,1)									
Project:	All									
Format:	=n									
1	31	<p>Stencil Buffer Clear Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When set, the stencil buffer is initialized.</p> <p style="text-align: center;">Programming Notes</p> <p>If this field is enabled,</p> <ol style="list-style-type: none"> the Depth Buffer Resolve Enable and Hierarchical Depth Buffer Resolve Enable fields must both be disabled. 3DSTATE_DEPTH_BUFFER::Stencil Write Enable must be set if 3DSTATE_STENCIL_BUFFER::Stencil buffer enable is set. 	Project:	All	Format:	Enable				
Project:	All									
Format:	Enable									
	30	<p>Depth Buffer Clear Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>When set, the depth buffer is initialized.</p> <p style="text-align: center;">Programming Notes</p> <p>If this field is enabled,</p> <ol style="list-style-type: none"> the Depth Buffer Resolve Enable and Hierarchical Depth Buffer Resolve Enable fields must both be disabled. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. 	Project:	All	Format:	Enable				
Project:	All									
Format:	Enable									
	29	<p>Scissor Rectangle Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Enables operation of Scissor Rectangle.</p> <p style="text-align: center;">Programming Notes</p> <p>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.</p> <table border="1"> <tr> <td style="text-align: center;">Workaround</td> <td style="text-align: center;">Project</td> </tr> <tr> <td>Workaround (BDW bug# 1911422) : Due to a Hardware issue this bit must not be set.</td> <td>BDW</td> </tr> </table>	Project:	All	Format:	Enable	Workaround	Project	Workaround (BDW bug# 1911422) : Due to a Hardware issue this bit must not be set.	BDW
Project:	All									
Format:	Enable									
Workaround	Project									
Workaround (BDW bug# 1911422) : Due to a Hardware issue this bit must not be set.	BDW									



3DSTATE_WM_HZ_OP						
	28	<p>Depth Buffer Resolve Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <p style="text-align: center;">Programming Notes</p> <p>If this field is enabled,</p> <ol style="list-style-type: none"> the Depth Buffer Clear and Hierarchical Depth Buffer Resolve Enable fields must both be disabled. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. <p>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.</p>	Project:	All	Format:	Enable
	Project:	All				
Format:	Enable					
27	<p>Hierarchical Depth Buffer Resolve Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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.</p> <p style="text-align: center;">Programming Notes</p> <p>If this field is enabled,</p> <ol style="list-style-type: none"> the Depth Buffer Clear and Depth Buffer Resolve Enable fields must both be disabled. 3DSTATE_DEPTH_BUFFER::Depth Write Enable must be set. <p>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.</p>	Project:	All	Format:	Enable	
Project:	All					
Format:	Enable					

3DSTATE_WM_HZ_OP					
	<p>26</p> <p>Pixel Position Offset Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1 Enumerated Type</td> </tr> </table> <p>Enables the device to offset pixel positions by 0.5 both in horizontal and vertical directions.</p> <p style="text-align: center;">Programming Notes</p> <p>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.</p>	Project:	All	Format:	U1 Enumerated Type
	Project:	All			
	Format:	U1 Enumerated Type			
	<p>25</p> <p>Full Surface Depth Clear</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <p>Setting this field to "1" along with "Depth buffer clear" will cause all the pixels/samples in an 8x4 block in the HIZ buffer to be cleared. If "Stc-buffer clear" is also set, then all pixels/samples in a 8x8 block of STC buffer will be cleared to the stc-ref value. 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 an 8x8 block in the STC buffer to get the stc-ref value.</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
<p>24</p> <p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All				
Format:	MBZ				
<p>23:16</p> <p>Stencil Clear Value</p> <table border="1"> <tr> <td>Format:</td> <td>U8.0</td> </tr> </table> <p>This field specifies the stencil clear value.</p>	Format:	U8.0			
Format:	U8.0				

3DSTATE_WM_HZ_OP																																			
	15:13	Number of Multisamples <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3 Enumerated Type</td> </tr> </table> <p>This field specifies how many samples/pixel exist in the Depth Buffer and Stencil buffers, as $\log_2(\#samples)$.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>1</td> <td>1 sample/pixel</td> <td>All</td> </tr> <tr> <td>1h</td> <td>2</td> <td>2 samples/pixel</td> <td>All</td> </tr> <tr> <td>2h</td> <td>4</td> <td>4 samples/pixel</td> <td>All</td> </tr> <tr> <td>3h</td> <td>8</td> <td>8 samples/pixel</td> <td>All</td> </tr> <tr> <td>4h</td> <td>Reserved</td> <td></td> <td>BDW</td> </tr> <tr> <td>5h-7h</td> <td>Reserved</td> <td></td> <td>All</td> </tr> </tbody> </table>		Project:	All	Format:	U3 Enumerated Type	Value	Name	Description	Project	0h	1	1 sample/pixel	All	1h	2	2 samples/pixel	All	2h	4	4 samples/pixel	All	3h	8	8 samples/pixel	All	4h	Reserved		BDW	5h-7h	Reserved		All
	Project:	All																																	
Format:	U3 Enumerated Type																																		
Value	Name	Description	Project																																
0h	1	1 sample/pixel	All																																
1h	2	2 samples/pixel	All																																
2h	4	4 samples/pixel	All																																
3h	8	8 samples/pixel	All																																
4h	Reserved		BDW																																
5h-7h	Reserved		All																																
	12:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> </table>		Project:	All																														
Project:	All																																		
<p style="text-align: center;">2</p> <p>Programming Notes: The clear rectangle x and y min and max 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).</p>	31:16	Clear Rectangle Y Min <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Depth Buffer origin (upper left corner)</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>(Device ignores bits 31:30)</td> <td>BDW</td> </tr> </tbody> </table>		Project:	All	Format:	U16 in Pixels from Depth Buffer origin (upper left corner)	Value	Name	Project	[0,16383]	(Device ignores bits 31:30)	BDW																						
	Project:	All																																	
Format:	U16 in Pixels from Depth Buffer origin (upper left corner)																																		
Value	Name	Project																																	
[0,16383]	(Device ignores bits 31:30)	BDW																																	
	15:0	Clear Rectangle X Min <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Depth Buffer origin (upper left corner)</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>(Device ignores bits 15:14)</td> <td>BDW</td> </tr> </tbody> </table>		Project:	All	Format:	U16 in Pixels from Depth Buffer origin (upper left corner)	Value	Name	Project	[0,16383]	(Device ignores bits 15:14)	BDW																						
Project:	All																																		
Format:	U16 in Pixels from Depth Buffer origin (upper left corner)																																		
Value	Name	Project																																	
[0,16383]	(Device ignores bits 15:14)	BDW																																	
<p style="text-align: center;">3</p> <p>Programming Notes: See the programming note in the previous DWORD for the Min values. The clear rectangle x and y min and max</p>	31:16	Clear Rectangle Y Max <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Depth Buffer origin (lower right corner)</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>(Device ignores bits 31:30)</td> <td>BDW</td> </tr> </tbody> </table>		Project:	All	Format:	U16 in Pixels from Depth Buffer origin (lower right corner)	Value	Name	Project	[0,16383]	(Device ignores bits 31:30)	BDW																						
	Project:	All																																	
Format:	U16 in Pixels from Depth Buffer origin (lower right corner)																																		
Value	Name	Project																																	
[0,16383]	(Device ignores bits 31:30)	BDW																																	
		<table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>(Device ignores bits 31:30)</td> <td>BDW</td> </tr> </tbody> </table>		Value	Name	Project	[0,16383]	(Device ignores bits 31:30)	BDW																										
Value	Name	Project																																	
[0,16383]	(Device ignores bits 31:30)	BDW																																	

3DSTATE_WM_HZ_OP												
<p>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). Hence the max values must be less than or equal to: (Surface Width » LOD) and (Surface Height » LOD) for X Max and Y Max respectively.</p>	15:0	<p>Clear Rectangle X Max</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16 in Pixels from Depth Buffer origin (lower right corner)</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td>(Device ignores bits 15:14)</td> <td>BDW</td> </tr> </tbody> </table>	Project:	All	Format:	U16 in Pixels from Depth Buffer origin (lower right corner)	Value	Name	Project	[0,16383]	(Device ignores bits 15:14)	BDW
	Project:	All										
Format:	U16 in Pixels from Depth Buffer origin (lower right corner)											
Value	Name	Project										
[0,16383]	(Device ignores bits 15:14)	BDW										
4	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All											
Format:	MBZ											
	15:0	<p>Sample Mask</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <p>Format: Right-justified bitmask (Bit 0 = Sample0). Number of bits that are used is determined by Num Multisamples (3DSTATE_WM_HZ_OP)</p> <p>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.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td>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.</td> </tr> </tbody> </table>	Project:	All	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.						
Project:	All											
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.												



A64 Byte Scattered Write MSD

MSD1W_A64_BS - A64 Byte Scattered Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Scattered R/W	
Group:	Byte Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 1Ah		
Project: All		
Format: Opcode		
A64 Scattered Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12	Reserved	
	Project: BDW	
	Format: Ignore	
Ignored		
11:10	Data Elements	
	Project: All	
	Format: MDC_A64_DS	
Specifies the number of data elements to be read or written		

MSD1W_A64_BS - A64 Byte Scattered Write MSD																
	<table border="1"> <tr> <td style="text-align: center; vertical-align: top;">9:8</td> <td colspan="2">A64 Scattered Message Subtype</td> </tr> <tr> <td></td> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td></td> <td>Project:</td> <td>All</td> </tr> <tr> <td></td> <td>Format:</td> <td>Opcode</td> </tr> <tr> <td></td> <td colspan="2">Byte Read/Write subtype</td> </tr> </table>	9:8	A64 Scattered Message Subtype			Default Value:	0h		Project:	All		Format:	Opcode		Byte Read/Write subtype	
	9:8	A64 Scattered Message Subtype														
	Default Value:	0h														
	Project:	All														
	Format:	Opcode														
	Byte Read/Write subtype															
	<table border="1"> <tr> <td style="text-align: center; vertical-align: top;">7:0</td> <td colspan="2">Binding Table Index</td> </tr> <tr> <td></td> <td>Project:</td> <td>All</td> </tr> <tr> <td></td> <td>Format:</td> <td>MDC_STATELESS</td> </tr> <tr> <td></td> <td colspan="2">Specifies the message is stateless</td> </tr> </table>	7:0	Binding Table Index			Project:	All		Format:	MDC_STATELESS		Specifies the message is stateless				
7:0	Binding Table Index															
	Project:	All														
	Format:	MDC_STATELESS														
	Specifies the message is stateless															

A64 Dword Scattered Read MSD

MSD1R_A64_DWS - A64 Dword Scattered Read MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Scattered R/W	
Group:	DW Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 10h		
Project: All		
Format: Opcode		
A64 Scattered Read message		
13	Invalidate After Read	
	Project: All	
	Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs		
12	Reserved	
	Project: BDW	
	Format: Ignore	
Ignored		
11:10	Data Elements	
	Project: All	
	Format: MDC_A64_DS	
Specifies the number of data elements to be read or written		

MSD1R_A64_DWS - A64 Dword Scattered Read MSD	
9:8	A64 Scattered Message Subtype
	Default Value: 1h
	Project: All
	Format: Opcode
Dword Read/Write subtype	
7:0	Binding Table Index
	Project: All
	Format: MDC_STATELESS
Specifies the message is stateless	



A64 Dword Scattered Write MSD

MSD1W_A64_DWS - A64 Dword Scattered Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Scattered R/W	
Group:	DW Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 1Ah		
Project: All		
Format: Opcode		
A64 Scattered Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12	Reserved	
	Project: BDW	
	Format: Ignore	
Ignored		
11:10	Data Elements	
	Project: All	
	Format: MDC_A64_DS	
Specifies the number of data elements to be read or written		

MSD1W_A64_DWS - A64 Dword Scattered Write MSD																
	<table border="1"> <tr> <td style="text-align: center; vertical-align: top;">9:8</td> <td colspan="2">A64 Scattered Message Subtype</td> </tr> <tr> <td></td> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td></td> <td>Project:</td> <td>All</td> </tr> <tr> <td></td> <td>Format:</td> <td>Opcode</td> </tr> <tr> <td colspan="3">Dword Read/Write subtype</td> </tr> </table>	9:8	A64 Scattered Message Subtype			Default Value:	1h		Project:	All		Format:	Opcode	Dword Read/Write subtype		
	9:8	A64 Scattered Message Subtype														
	Default Value:	1h														
	Project:	All														
	Format:	Opcode														
Dword Read/Write subtype																
<table border="1"> <tr> <td style="text-align: center; vertical-align: top;">7:0</td> <td colspan="2">Binding Table Index</td> </tr> <tr> <td></td> <td>Project:</td> <td>All</td> </tr> <tr> <td></td> <td>Format:</td> <td>MDC_STATELESS</td> </tr> <tr> <td colspan="3">Specifies the message is stateless</td> </tr> </table>	7:0	Binding Table Index			Project:	All		Format:	MDC_STATELESS	Specifies the message is stateless						
7:0	Binding Table Index															
	Project:	All														
	Format:	MDC_STATELESS														
Specifies the message is stateless																



A64 Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD

MSD1R_A64_DWAI2_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
	18:14	Message Type					
<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>		Default Value:	13h	Project:	All	Format:	Opcode
Default Value:		13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						

MSD1R_A64_DWAI2_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD

MSD1W_A64_DWAI2_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode
	Default Value:	13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						

MSD1W_A64_DWAI2_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD

MSD1R_A64_DWAI3_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Untyped Atomic Operation							
Group:	Dword Untyped Atomic Integer Trinary Operation							
DWord	Bit	Description						
0	19	Header Present						
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF		
	Project:	All						
	Format:	MDC_MHF						
18:14	Message Type							
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode	
	Default Value:	13h						
Project:	All							
Format:	Opcode							
Return Data Control								
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode		
Default Value:	1h							
Project:	All							
Format:	Opcode							
12	Data Width	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
		Default Value:	0h					
		Project:	All					
	Format:	Opcode						

MSD1R_A64_DWAI3_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD														
	<table border="1"> <tr> <td style="text-align: center; vertical-align: top;">11:8</td> <td> <p>Atomic Integer Operation</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3S</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p> <table border="1"> <tr> <td style="text-align: center;">Workaround</td> </tr> </table> <p>CMPWR_2W Operation is not supported in A64 SIMD4x2.</p> </td> </tr> <tr> <td style="text-align: center; vertical-align: top;">7:0</td> <td> <p>Binding Table Index</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p> </td> </tr> </table>	11:8	<p>Atomic Integer Operation</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3S</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p> <table border="1"> <tr> <td style="text-align: center;">Workaround</td> </tr> </table> <p>CMPWR_2W Operation is not supported in A64 SIMD4x2.</p>	Project:	BDW	Format:	MDC_AOP3S	Workaround	7:0	<p>Binding Table Index</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
	11:8	<p>Atomic Integer Operation</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3S</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p> <table border="1"> <tr> <td style="text-align: center;">Workaround</td> </tr> </table> <p>CMPWR_2W Operation is not supported in A64 SIMD4x2.</p>	Project:	BDW	Format:	MDC_AOP3S	Workaround							
Project:	BDW													
Format:	MDC_AOP3S													
Workaround														
7:0	<p>Binding Table Index</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS									
Project:	All													
Format:	MDC_STATELESS													



A64 Dword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD

MSD1W_A64_DWAI3_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Untyped Atomic Operation		
Group:	Dword Untyped Atomic Integer Trinary Operation		
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHF
		The message forbids a header	
	18:14	Message Type	
		Default Value:	13h
		Project:	All
		Format:	Opcode
			A64 Untyped Atomic Integer Operation SIMD4x2 message
	13	Return Data Control	
		Default Value:	0h
		Project:	All
Format:		Opcode	
		Specifies that no return data is sent back to the thread.	
12	Data Width		
	Default Value:	0h	
	Project:	All	
	Format:	Opcode	
		Operations are on 32-bit integers	

MSD1W_A64_DWAI3_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD

	11:8	Atomic Integer Operation	
		Project:	BDW
		Format:	MDC_AOP3S
		Specifies the atomic integer operation to be performed.	
		Workaround	
		CMPWR_2W is not supported by A64 SIMD4x2.	
	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD

MSD1R_A64_DWAI1_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode
	Default Value:	13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1			
Project:	All						
Format:	MDC_AOP1						
11:8							

MSD1R_A64_DWAI1_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD

MSD1W_A64_DWAI1_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode
	Default Value:	13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1			
Project:	All						
Format:	MDC_AOP1						

MSD1W_A64_DWAI1_4x2 - A64 Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Binary with Return Data Operation MSD

MSD1R_A64_DWAI2 - A64 Dword Untyped Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
	18:14	Message Type					
<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>		Default Value:	12h	Project:	All	Format:	Opcode
Default Value:		12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
	Project:	All					
Format:	MDC_AOP2						

MSD1R_A64_DWAI2 - A64 Dword Untyped Atomic Integer Binary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Binary Write Only Operation MSD

MSD1W_A64_DWAI2 - A64 Dword Untyped Atomic Integer Binary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						

MSD1W_A64_DWAI2 - A64 Dword Untyped Atomic Integer Binary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Trinary with Return Data Operation MSD

MSD1R_A64_DWAI3 - A64 Dword Untyped Atomic Integer Trinary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Trinary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP3		
	Project:	All					
Format:	MDC_AOP3						

MSD1R_A64_DWAI3 - A64 Dword Untyped Atomic Integer Trinary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Trinary Write Only Operation MSD

MSD1W_A64_DWAI3 - A64 Dword Untyped Atomic Integer Trinary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Untyped Atomic Integer Trinary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
		The message forbids a header
18:14	Message Type	Default Value: 12h
		Project: All
		Format: Opcode
		A64 Untyped Atomic Integer Operation message
13	Return Data Control	Default Value: 0h
		Project: All
		Format: Opcode
		Specifies that no return data is sent back to the thread.
12	Data Width	Default Value: 0h
		Project: All
		Format: Opcode
		Operations are on 32-bit integers
11:8	Atomic Integer Operation	Project: All
		Format: MDC_AOP3
		Specifies the atomic integer operation to be performed.

MSD1W_A64_DWAI3 - A64 Dword Untyped Atomic Integer Trinary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Unary with Return Data Operation MSD

MSD1R_A64_DWAI1 - A64 Dword Untyped Atomic Integer Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 32-bit integers</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1			
Project:	All						
Format:	MDC_AOP1						
11:8							

MSD1R_A64_DWAI1 - A64 Dword Untyped Atomic Integer Unary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Dword Untyped Atomic Integer Unary Write Only Operation MSD

MSD1W_A64_DWAI1 - A64 Dword Untyped Atomic Integer Unary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Untyped Atomic Integer Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	The message forbids a header	
	18:14	Message Type
Default Value: 12h		
Project: All		
Format: Opcode		
A64 Untyped Atomic Integer Operation message		
13	Return Data Control	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Specifies that no return data is sent back to the thread.		
12	Data Width	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Operations are on 32-bit integers		
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP1	
Specifies the atomic integer operation to be performed.		

MSD1W_A64_DWAI1 - A64 Dword Untyped Atomic Integer Unary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	

A64 Hword Block Read MSD

MSD1R_A64_HWB - A64 Hword Block Read MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Block R/W		
Group:	HW Block R/W		
DWord	Bit	Description	
0	19	Header Present	
		Project: All	
		Format: MDC_MHR	
		Indicates that the message requires a header.	
		Message Type	
18:14		Default Value: 14h	
		Project: All	
		Format: Opcode	
		A64 Oword Block Read message	
13		Invalidate After Read	
		Project: All	
		Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs			
12:11		A64 Block Message Subtype	
		Default Value: 3h	
		Project: All	
		Format: Opcode	
Hword Block Read/Write subtype			
10:8		Data Elements	
		Project: All	
		Format: MDC_A64_DB_HW	
Specifies the number of contiguous Hwords to be read or written			

MSD1R_A64_HWB - A64 Hword Block Read MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
Project:	All				
Format:	MDC_STATELESS				

A64 Hword Block Write MSD

MSD1W_A64_HWB - A64 Hword Block Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Block R/W	
Group:	HW Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18:14	Message Type
Default Value: 15h		
Project: All		
Format: Opcode		
A64 Hword Block Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12:11	A64 Block Message Subtype	
	Default Value: 3h	
	Project: All	
	Format: Opcode	
Hword Block Read/Write subtype		
10:8	Data Elements	
	Project: All	
	Format: MDC_A64_DB_HW	
Specifies the number of contiguous Hwords to be read or written		

MSD1W_A64_HWB - A64 Hword Block Write MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
Project:	All				
Format:	MDC_STATELESS				

A64 Oword Block Read MSD

MSD1R_A64_OWB - A64 Oword Block Read MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Block R/W		
Group:	OW Block R/W		
DWord	Bit	Description	
0	19	Header Present	
		Project: All	
		Format: MDC_MHR	
		Indicates that the message requires a header.	
		Message Type	
18:14		Default Value: 14h	
		Project: All	
		Format: Opcode	
		A64 Oword Block Read message	
13		Invalidate After Read	
		Project: All	
		Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs			
12:11		A64 Block Message Subtype	
		Default Value: 0h	
		Project: All	
		Format: Opcode	
Oword Block Read/Write subtype			
10:8		Data Elements	
		Project: All	
		Format: MDC_A64_DB_OW	
Specifies the number of contiguous Owords to be read or written			

MSD1R_A64_OWB - A64 Oword Block Read MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
Project:	All				
Format:	MDC_STATELESS				

A64 Oword Block Write MSD

MSD1W_A64_OWB - A64 Oword Block Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Block R/W	
Group:	OW Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18:14	Message Type
Default Value: 15h		
Project: All		
Format: Opcode		
A64 Oword Block Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12:11	A64 Block Message Subtype	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Oword Block Read/Write subtype		
10:8	Data Elements	
	Project: All	
	Format: MDC_A64_DB_OW	
Specifies the number of contiguous Owords to be read or written		

MSD1W_A64_OWB - A64 Oword Block Write MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
Project:	All				
Format:	MDC_STATELESS				

A64 Oword Dual Block Read MSD

MSD1R_A64_OWDB - A64 Oword Dual Block Read MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Block R/W		
Group:	OW Dual Block R/W		
DWord	Bit	Description	
0	19	Header Present	
		Project: All	
		Format: MDC_MHR	
		Indicates that the message requires a header.	
		Message Type	
18:14		Default Value: 14h	
		Project: All	
		Format: Opcode	
		A64 Oword Block Read message	
13		Invalidate After Read	
		Project: All	
		Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs			
12:11		A64 Block Message Subtype	
		Default Value: 2h	
		Project: All	
		Format: Opcode	
Oword Dual Block Read/Write subtype			
10:8		Data Elements	
		Project: All	
		Format: MDC_A64_DB_OWD	
Specifies the number of contiguous Owords to be read or written			

MSD1R_A64_OWDB - A64 Oword Dual Block Read MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	

A64 Oword Dual Block Write MSD

MSD1W_A64_OWDB - A64 Oword Dual Block Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Block R/W	
Group:	OW Dual Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18:14	Message Type
Default Value: 15h		
Project: All		
Format: Opcode		
A64 Oword Block Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12:11	A64 Block Message Subtype	
	Default Value: 2h	
	Project: All	
	Format: Opcode	
Oword Dual Block Read/Write subtype		
10:8	Data Elements	
	Project: All	
	Format: MDC_A64_DB_OWD	
Specifies the number of contiguous Owords to be read or written		

MSD1W_A64_OWDB - A64 Oword Dual Block Write MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_STATELESS</td> </tr> </table> <p>Specifies the message is stateless</p>	Project:	All	Format:	MDC_STATELESS
Project:	All				
Format:	MDC_STATELESS				



A64 Oword Unaligned Block Read MSD

MSD1R_A64_OWUB - A64 Oword Unaligned Block Read MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Block R/W	
Group:	OW Unaligned Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18:14	Message Type
Default Value: 14h		
Project: All		
Format: Opcode		
A64 Oword Block Read message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12:11	A64 Block Message Subtype	
	Default Value: 1h	
	Project: All	
	Format: Opcode	
Oword Unaligned Block Read subtype		
10:8	Data Elements	
	Project: All	
	Format: MDC_A64_DB_OW	
Specifies the number of contiguous Owords to be read		

MSD1R_A64_OWUB - A64 Oword Unaligned Block Read MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Scattered Write MSD

MSD1W_A64_QWS - A64 Qword Scattered Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Scattered R/W	
Group:	QW Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 1Ah		
Project: All		
Format: Opcode		
A64 Scattered Write message		
13	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
12	Reserved	
	Project: BDW	
	Format: Ignore	
Ignored		
11:10	Data Elements	
	Project: All	
	Format: MDC_A64_DS	
Specifies the number of data elements to be read or written		

MSD1W_A64_QWS - A64 Qword Scattered Write MSD			
	9:8	A64 Scattered Message Subtype	
		Default Value:	2h
		Project:	All
		Format:	Opcode
		Qword Read/Write subtype	
	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD

MSD1R_A64_QWAI2_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Qword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
	18:14	Message Type					
<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>		Default Value:	13h	Project:	All	Format:	Opcode
Default Value:		13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
	Project:	All					
Format:	MDC_AOP2						

MSD1R_A64_QWAI2_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD

MSD1W_A64_QWAI2_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Qword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode
	Default Value:	13h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						

MSD1W_A64_QWAI2_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Binary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD

MSD1R_A64_QWAI3_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Untyped Atomic Operation							
Group:	Qword Untyped Atomic Integer Trinary Operation							
DWord	Bit	Description						
0	19	Header Present						
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF		
	Project:	All						
	Format:	MDC_MHF						
18:14	Message Type							
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode	
	Default Value:	13h						
Project:	All							
Format:	Opcode							
Return Data Control								
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode		
Default Value:	1h							
Project:	All							
Format:	Opcode							
12	Data Width	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
		Default Value:	1h					
		Project:	All					
	Format:	Opcode						

MSD1R_A64_QWAI3_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Trinary with Return Data Operation MSD

	11:8	Atomic Integer Operation	
	Project:		BDW
Specifies the atomic integer operation to be performed.			
Workaround			
CMPWR_2W is not supported in A64 SIMD4x2			
	7:0	Binding Table Index	
	Project:		All
	Format:		MDC_STATELESS
Specifies the message is stateless			



A64 Qword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD

MSD1W_A64_QWAI3_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Untyped Atomic Operation							
Group:	Qword Untyped Atomic Integer Trinary Operation							
DWord	Bit	Description						
0	19	Header Present						
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF		
	Project:	All						
	Format:	MDC_MHF						
18:14	Message Type							
	<table border="1"> <tr> <td>Default Value:</td> <td>13h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	13h	Project:	All	Format:	Opcode	
	Default Value:	13h						
Project:	All							
Format:	Opcode							
Return Data Control								
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode		
Default Value:	0h							
Project:	All							
Format:	Opcode							
12	Data Width	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
		Default Value:	1h					
		Project:	All					
	Format:	Opcode						

MSD1W_A64_QWAI3_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD

	11:8	Atomic Integer Operation	
		Project:	BDW
		Format:	MDC_AOP3S
		Specifies the atomic integer operation to be performed.	
		Workaround	
		CMPWR_2W is not supported in A64 SIMD4x2.	
	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD

MSD1R_A64_QWAI1_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Untyped Atomic Operation		
Group:	Qword Untyped Atomic Integer Unary Operation		
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHF
		The message forbids a header	
	18:14	Message Type	
		Default Value:	13h
		Project:	All
		Format:	Opcode
			A64 Untyped Atomic Integer Operation SIMD4x2 message
	13	Return Data Control	
		Default Value:	1h
		Project:	All
		Format:	Opcode
			Specifies that return data is sent back to the thread.
	12	Data Width	
		Default Value:	1h
Project:		All	
Format:		Opcode	
		Operations are on 64-bit integers	
11:8	Atomic Integer Operation		
	Project:	All	
	Format:	MDC_AOP1	
	Specifies the atomic integer operation to be performed.		

MSD1R_A64_QWAI1_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD

MSD1W_A64_QWAI1_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Qword Untyped Atomic Integer Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
		The message forbids a header
18:14	Message Type	Default Value: 13h
		Project: All
		Format: Opcode
		A64 Untyped Atomic Integer Operation SIMD4x2 message
13	Return Data Control	Default Value: 0h
		Project: All
		Format: Opcode
		Specifies that no return data is sent back to the thread.
12	Data Width	Default Value: 1h
		Project: All
		Format: Opcode
		Operations are on 64-bit integers
11:8	Atomic Integer Operation	Project: All
		Format: MDC_AOP1
		Specifies the atomic integer operation to be performed.

MSD1W_A64_QWAI1_4x2 - A64 Qword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Untyped Atomic Integer Binary with Return Data Operation MSD

MSD1R_A64_QWAI2 - A64 Qword Untyped Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Qword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						
11:8							

MSD1R_A64_QWAI2 - A64 Qword Untyped Atomic Integer Binary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Untyped Atomic Integer Binary Write Only Operation MSD

MSD1W_A64_QWAI2 - A64 Qword Untyped Atomic Integer Binary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Qword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All						
Format:	MDC_AOP2						

MSD1W_A64_QWAI2 - A64 Qword Untyped Atomic Integer Binary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Untyped Atomic Integer Ternary with Return Data Operation MSD

MSD1R_A64_QWAI3 - A64 Qword Untyped Atomic Integer Ternary with Return Data Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Qword Untyped Atomic Integer Ternary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
		The message forbids a header
18:14	Message Type	Default Value: 12h
		Project: All
		Format: Opcode
		A64 Untyped Atomic Integer Operation message
13	Return Data Control	Default Value: 1h
		Project: All
		Format: Opcode
		Specifies that return data is sent back to the thread.
12	Data Width	Default Value: 1h
		Project: All
		Format: Opcode
		Operations are on 64-bit integers

MSD1R_A64_QWAI3 - A64 Qword Untyped Atomic Integer Trinary with Return Data Operation MSD

	11:8	Atomic Integer Operation	
		Project:	BDW
		Format:	MDC_AOP3S
		Specifies the atomic integer operation to be performed.	
		Workaround	
		CMPWR_2W is not supported in A64 QWord SIMD8.	
	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Untyped Atomic Integer Trinary Write Only Operation MSD

MSD1W_A64_QWAI3 - A64 Qword Untyped Atomic Integer Trinary Write Only Operation MSD

Project:	BDW			
Source:	DataPort 1			
Length Bias:	1			
Family:	Untyped Atomic Operation			
Group:	Qword Untyped Atomic Integer Trinary Operation			
DWord	Bit	Description		
0	19	Header Present		
		Project:	All	
		Format:	MDC_MHF	
		The message forbids a header		
	18:14	Message Type		
		Default Value:	12h	
		Project:	All	
		Format:	Opcode	
	A64 Untyped Atomic Integer Operation message			
	13	Return Data Control		
		Default Value:	0h	
		Project:	All	
Format:		Opcode		
Specifies that no return data is sent back to the thread.				
12	Data Width			
	Default Value:	1h		
	Project:	All		
	Format:	Opcode		
Operations are on 64-bit integers				

MSD1W_A64_QWAI3 - A64 Qword Untyped Atomic Integer Trinary Write Only Operation MSD

	11:8	Atomic Integer Operation	
		Project:	BDW
		Format:	MDC_AOP3S
		Specifies the atomic integer operation to be performed.	
		Workaround	
		CMPWR_2W is not supported in A64 QWord SIMD8.	
	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	



A64 Qword Untyped Atomic Integer Unary with Return Data Operation MSD

MSD1R_A64_QWAI1 - A64 Qword Untyped Atomic Integer Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Qword Untyped Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHF</td> </tr> </table> <p>The message forbids a header</p>	Project:	All	Format:	MDC_MHF	
	Project:	All					
	Format:	MDC_MHF					
18:14	Message Type						
	<table border="1"> <tr> <td>Default Value:</td> <td>12h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>A64 Untyped Atomic Integer Operation message</p>	Default Value:	12h	Project:	All	Format:	Opcode
	Default Value:	12h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Data Width						
	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Operations are on 64-bit integers</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
Project:	All						
Format:	Opcode						
Atomic Integer Operation							
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1		
	Project:	All					
Format:	MDC_AOP1						

MSD1R_A64_QWAI1 - A64 Qword Untyped Atomic Integer Unary with Return Data Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	

A64 Qword Untyped Atomic Integer Unary Write Only Operation MSD

MSD1W_A64_QWAI1 - A64 Qword Untyped Atomic Integer Unary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Qword Untyped Atomic Integer Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
		The message forbids a header
18:14	Message Type	Default Value: 12h
		Project: All
		Format: Opcode
		A64 Untyped Atomic Integer Operation message
13	Return Data Control	Default Value: 0h
		Project: All
		Format: Opcode
		Specifies that no return data is sent back to the thread.
12	Data Width	Default Value: 1h
		Project: All
		Format: Opcode
		Operations are on 64-bit integers
11:8	Atomic Integer Operation	Project: All
		Format: MDC_AOP1
		Specifies the atomic integer operation to be performed.

MSD1W_A64_QWAI1 - A64 Qword Untyped Atomic Integer Unary Write Only Operation MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_STATELESS
		Specifies the message is stateless	

A64 Untyped Surface Read MSD

MSD1R_A64_US - A64 Untyped Surface Read MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Surface R/W	
Group:	Scattered Untyped Surface R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 11h		
Project: All		
Format: Opcode		
A64 Untyped Surface Read message		
13:12	SIMD Mode	
	Project: All	
	Format: MDC_SM3	
Specifies the SIMD mode of the message (number of slots processed)		
11:8	Channel Mask	
	Project: All	
	Format: MDC_CMASK	
Specifies which RGBA channels are included in the message payload.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_STATELESS	
Specifies the message is stateless		

A64 Untyped Surface Write MSD

MSD1W_A64_US - A64 Untyped Surface Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Surface R/W	
Group:	Scattered Untyped Surface R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHF
	Indicates that the message forbids a header	
	18:14	Message Type
Default Value: 19h		
Project: All		
Format: Opcode		
A64 Untyped Surface Write message		
13:12	SIMD Mode	
	Project: All	
	Format: MDC_SM3	
Specifies the SIMD mode of the message (number of slots processed)		
11:8	Channel Mask	
	Project: All	
	Format: MDC_UW_CMASK	
Specifies which RGBA channels are included in the message payload.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_STATELESS	
Specifies the message is stateless		

Addition

add - Addition			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
The add instruction performs component-wise addition of src0 and src1 and stores the results in dst. Addition of two floating-point numbers follows rules in add (IEEE mode) or add (ALT mode).			
Format: [(pred)] add[.cmod] (exec_size) dst src0 src1			
Programming Notes			
Use a source modifier with add to implement subtraction.			
Syntax			
[(pred)] add[.cmod] (exec_size) reg reg reg [(pred)] add[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] + src1.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*B,*W,*D	F		
F	F		
DF	DF	BDW	
HF	HF	BDW	
*B,*W,*D	HF	BDW	
*W,*D,*Q	*W,*D,*Q	BDW	
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	

add - Addition			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER

Addition with Carry

addc - Addition with Carry			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The addc instruction performs component-wise addition of src0 and src1 and stores the results in dst; it also stores the carry into acc. If the operation produces a carry out, 0x00000001 is stored in acc, else 0x00000000 is stored in acc.</p>			
<p>Format: [(pred)] addc[.cmode] (exec_size) dst src0 src1</p>			
Restriction			
<p>AccWrEn is required. The accumulator is an implicit destination and thus cannot be an explicit destination operand.</p>			
Syntax			
<p>[(pred)] addc[.cmode] (exec_size) reg reg reg [(pred)] addc[.cmode] (exec_size) reg reg imm32</p>			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] + src1.chan[n]; acc.chan[n] = carry(src0.chan[n] + src1.chan[n]); } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	N
Src Types	Dst Types		
UD	UD		
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	

Arithmetic Shift Right

asr - Arithmetic Shift Right			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>Perform component-wise arithmetic right shift of the bits in src0 by the shift count indicated in src1, storing the results in dst. If src0 has a signed type, insert copies of src0's sign bit in the number of MSBs indicated by the shift count. Otherwise insert 0 bits. [Pre-DevBDW]: The shift count is taken from the low five bits of src1, regardless of the src1 type and treated as an unsigned integer in the range 0 to 31. [BDW]: In QWord mode, the shift count is taken from the low six bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 63. Otherwise the shift count is taken from the low five bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 31. The operation uses QWord mode if src0 or dst has the Q or UQ type but not if src1 is the only operand with the Q or UQ type. For positive values, this operation is $src0 / 2^{shiftCount}$ and for negative values, this operation is $src0 / 2^{shiftCount} - 1$.</p>			
Format: [(pred)] asr[.cmod] (exec_size) dst src0 src1			
Programming Notes			
If src0 is -1, the result is -1 regardless of the shift count.			
For unsigned src0 types, asr and shr produce the same result.			
Syntax			
[(pred)] asr[.cmod] (exec_size) reg reg [(pred)] asr[.cmod] (exec_size) reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.channel[n]) { [Pre-DevBDW]: shiftCnt = src1.chan[n] & 0x1F; // Always use low 5 bits for shift count.[BDW]: shiftCnt = src0 or dst has Q or UQ type ? src1.chan[n] & 0x3F : src1.chan[n] & 0x1F if (src0.chan[n] >= 0) { dst.chan[n] = src0.chan[n] » shiftCnt; } else { int maskLSB = pow(2, shiftCnt) - 1; if (maskLSB & src0.chan[n] == 0) { dst.chan[n] = sign(src0.chan[n]) * ((abs)src0.chan[n] » shiftCnt); } else { dst.chan[n] = sign(src0.chan[n]) * ((abs)src0.chan[n] » shiftCnt) - 1; } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	([RegSource][Src1.RegFile]!='IMM')
		Format:	EU_INSTRUCTION_SOURCES_REG_REG

asr - Arithmetic Shift Right		
	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

Average

avg - Average			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The avg instruction performs component-wise integer average of src0 and src1 and stores the results in dst. An integer average uses integer upward rounding. It is equivalent to increment one to the addition of src0 and src1 and then apply an arithmetic right shift to this intermediate value.</p>			
<p>Format: The avg instruction performs component-wise integer average of src0 and src1 and stores the results in dst. An integer average uses integer upward rounding. It is equivalent to increment one to the addition of src0 and src1 and then apply an arithmetic right shift to this intermediate value.</p>			
Syntax			
[[pred]] avg[.cmod] (exec_size) reg reg reg [[pred]] avg[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = (src0.chan[n] + src1.chan[n] + 1) » 1; // Use arithmetic shift right. } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
*B,*W,*D	*B,*W,*D		
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	



Bit Field Extract

bfe - Bit Field Extract			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>Component-wise extract a bit field from src2 using the bit field width from src0 and the bit field offset from src1. Store the extracted bit field value in the low bits of dst and sign extend (if D type) or zero extend (if UD type). The width and offset values are from the low five bits of src0 and src1 respectively, or src0 & 0x1f and src1 & 0x1f. If width is zero, the result is zero. If offset + width > 32 then the extracted bit field is bits offset to 31 of src2, extracting only 32 - offset bits, less than width as the bit field cannot extend past the MSB of the source value. Otherwise extract width bits extending from bit positions offset to offset + width - 1.</p>			
Format: [(pred)] bfe (exec_size) dst src0 src1 src2			
Restriction			Project
No accumulator access, implicit or explicit.			
All three-source instructions have certain restrictions, described in Instruction Formats [BDW].			BDW
Syntax			
[(pred)] bfe (exec_size) reg reg reg reg			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD width = src0.chan[n][4:0]; UD offset = src1.chan[n][4:0]; if (width == 0) { dst.chan[n] = 0x00000000; } else if ((width + offset) < 32) { dst.chan[n] = src2.chan[n] << (32 - width - offset); if (src2 is signed) { dst.chan[n] = dst.chan[n] >> (32 - width); // pad sign bit of dst.chan } else { dst.chan[n] = dst.chan[n] >> (32 - width); // pad 0 } } else { if (src2 is signed) { dst.chan[n] = src2.chan[n] >> offset; // pad sign bit } else { dst.chan[n] = src2.chan[n] >> offset; // pad 0 } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UD	UD		
D	D		
DWord	Bit	Description	
0..3	127:126	Reserved Format: MBZ	
	125:106	Source 2 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	105	Reserved Format: MBZ	

bfe - Bit Field Extract											
104:85	Source 1										
	Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC										
84	Reserved										
	Format: MBZ										
83:64	Source 0										
	Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC										
63:56	Destination Register Number										
	Format: DstRegNum										
55:53	Destination Subregister Number										
	Format: DstSubRegNum[2:0]										
52:49	Destination Channel Enable										
	Format: ChanEn[4]										
<p>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 <i>x</i>, <i>y</i>, <i>z</i>, and <i>w</i>, respectively, where <i>x</i> corresponds to Channel 0 in the group and <i>w</i> corresponds to channel 3 in the group</p>											
48	Reserved										
	Project: MBZ										
47	NibCtrl										
	Format: NibCtrl										
46	Reserved										
	Project: MBZ										
45:44	Destination Data Type										
	Project: This field contains the data type for the destination										
<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Single Precision Float</td> </tr> <tr> <td>01b</td> <td>DWord</td> </tr> <tr> <td>10b</td> <td>Unsigned DWord</td> </tr> <tr> <td>11b</td> <td>Double Precision Float</td> </tr> </tbody> </table>		Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
Value	Name										
00b	Single Precision Float										
01b	DWord										
10b	Unsigned DWord										
11b	Double Precision Float										

bfe - Bit Field Extract													
43:42	Source Data Type Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> This field contains the data type for all three sources <table border="1" style="width: 100%;"><thead><tr><th style="text-align: center;">Value</th><th style="text-align: center;">Name</th></tr></thead><tbody><tr><td>00b</td><td>Single Precision Float</td></tr><tr><td>01b</td><td>DWord</td></tr><tr><td>10b</td><td>Unsigned DWord</td></tr><tr><td>11b</td><td>Double Precision Float</td></tr></tbody></table>			Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
	Value	Name											
	00b	Single Precision Float											
	01b	DWord											
	10b	Unsigned DWord											
11b	Double Precision Float												
41:40	Source 2 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier] == 'true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier] == 'true')											
	/// ([Property[Source Modifier] == 'true')												
Source 1 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier] == 'true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier] == 'true')												
/// ([Property[Source Modifier] == 'true')													
37:36	Source 0 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier] == 'true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier] == 'true')											
	/// ([Property[Source Modifier] == 'true')												
Reserved Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>		MBZ											
	MBZ												
34	Flag Register Number Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> This field contains the flag register number for instructions with a non-zero Conditional Modifier.												
33	Flag Subregister Number This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.												
32	Reserved Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>				MBZ								
	MBZ												
Header Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>EU_INSTRUCTION_HEADER</td></tr></table>		EU_INSTRUCTION_HEADER											
	EU_INSTRUCTION_HEADER												

Bit Field Insert 1

bfi1 - Bit Field Insert 1			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The bfi1 instruction is the first instruction in a two-instruction macro for bfi (Bit Field Insert). The bfi1 instruction component-wise generates mask with control from src0 and src1 and stores the results in dst. The mask is used in the bfi2 instruction to generate the final result of bfi. Create a bit mask corresponding to the bit field width and offset in src0 and src1. Store the bit mask in dst. The mask has all bits in the bit field set to 1 and all other bits as 0. The width and offset values are from the low five bits of src0 and src1 respectively, or src0 & 0x1f and src1 & 0x1f. If width is zero, the result is zero. The bfi macro has four source operands: src0 - bit field width in low five bits, src1 - bit field offset/starting bit position in low five bits, src2 - bit field value to insert, using only the number of least significant bits given by width in src0, and src3 - overall value into which the bit field is inserted, providing all bits other than the inserted bits for the result value. bfi dst src0 src1 src2 src3 // Translates to these two instructions: bfi1 dst src0 src1 bfi2 dst dst src2 src3</p>			
Format: [(pred)] bfi1 (exec_size) dst src0 src1			
Programming Notes			
No accumulator access, implicit or explicit.			
Syntax			
[(pred)] bfi1 (exec_size) reg reg reg [(pred)] bfi1 (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD width = src0.chan[n][4:0]; UD offset = src1.chan[n][4:0]; dst = ((1 << width) - 1) << offset; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UD	UD		
D	D		
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



bfi1 - Bit Field Insert 1		
	63:32	Operand Controls Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

Bit Field Insert 2

bfi2 - Bit Field Insert 2			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The bfi2 instruction is the second instruction in a two-instruction macro for bfi (Bit Field Insert). The bfi2 instruction component-wise performs the bitfield insert operation on src1 and src2 based on the mask in src0. Use the mask in src0 to take a bit field value from the low bits of src1 and combine it with the value from src2 (so src2 provides all bits other than those masked out and replaced by the bit field value). Store the result in dst. The bfi macro has four source operands: src0 - bit field width in low five bits, src1 - bit field offset/starting bit position in low five bits, src2 - bit field value to insert, using only the number of least significant bits given by width in src0, and src3 - overall value into which the bit field is inserted, providing all bits other than the inserted bits for the result value. bfi dst src0 src1 src2 src3 // Translates to these two instructions: bfi1 dst src0 src1 bfi2 dst dst src2 src3</p>			
Format: [(pred)] bfi2 (exec_size) dst src0 src1 src2			
Restriction			Project
No accumulator access, implicit or explicit.			
All three-source instructions have certain restrictions, described in Instruction Formats [BDW].			BDW
Syntax			
[(pred)] bfi2 (exec_size) reg reg reg reg			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD offset = LZD(reverse(src0.chan[n]))-1; // offset is the number of LSB zero bits below the bit mask which has all 1s. // width (implied by the logic) is the number of 1 bits in the mask value, which should be all 1s. dst.chan[n] = ((src1.chan[n] << offset) & src0.chan[n]) (src2.chan[n] & ! src0.chan[n]); }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UD	UD		
D	D		
DWord	Bit	Description	
0..3	127:126	Reserved	
		Format:	MBZ
	125:106	Source 2	
		Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC

bfi2 - Bit Field Insert 2												
105	Reserved	Format: MBZ										
104:85	Source 1	Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC										
84	Reserved	Format: MBZ										
83:64	Source 0	Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC										
63:56	Destination Register Number	Format: DstRegNum										
55:53	Destination Subregister Number	Format: DstSubRegNum[2:0]										
52:49	Destination Channel Enable	Format: ChanEn[4] Four channel enables are defined for controlling which channels are written into the destination region. These channel mask bits are applied in a modulo-four manner to all ExecSize channels. There is 1-bit Channel Enable for each channel within the group of 4. If the bit is cleared, the write for the corresponding channel is disabled. If the bit is set, the write is enabled. Mnemonics for the bit being set for the group of 4 are <i>x</i> , <i>y</i> , <i>z</i> , and <i>w</i> , respectively, where <i>x</i> corresponds to Channel 0 in the group and <i>w</i> corresponds to channel 3 in the group										
48	Reserved	Project: Format: MBZ										
47	NibCtrl	Project: Format: NibCtrl										
46	Reserved	Project: Format: MBZ										
45:44	Destination Data Type	Project: This field contains the data type for the destination <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">00b</td> <td>Single Precision Float</td> </tr> <tr> <td style="text-align: center;">01b</td> <td>DWord</td> </tr> <tr> <td style="text-align: center;">10b</td> <td>Unsigned DWord</td> </tr> <tr> <td style="text-align: center;">11b</td> <td>Double Precision Float</td> </tr> </tbody> </table>	Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
Value	Name											
00b	Single Precision Float											
01b	DWord											
10b	Unsigned DWord											
11b	Double Precision Float											

bfi2 - Bit Field Insert 2													
43:42	Source Data Type Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> This field contains the data type for all three sources <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Single Precision Float</td> </tr> <tr> <td>01b</td> <td>DWord</td> </tr> <tr> <td>10b</td> <td>Unsigned DWord</td> </tr> <tr> <td>11b</td> <td>Double Precision Float</td> </tr> </tbody> </table>			Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
	Value	Name											
	00b	Single Precision Float											
	01b	DWord											
	10b	Unsigned DWord											
11b	Double Precision Float												
41:40	Source 2 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier]='true')											
	/// ([Property[Source Modifier]='true')												
Source 1 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier]='true')												
/// ([Property[Source Modifier]='true')													
37:36	Source 0 Modifier Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table> Format: SrcMod	/// ([Property[Source Modifier]='true')											
	/// ([Property[Source Modifier]='true')												
Reserved Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>		MBZ											
	MBZ												
34	Flag Register Number Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> This field contains the flag register number for instructions with a non-zero Conditional Modifier.												
33	Flag Subregister Number This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.												
32	Reserved Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>				MBZ								
	MBZ												
Header Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>EU_INSTRUCTION_HEADER</td></tr></table>		EU_INSTRUCTION_HEADER											
	EU_INSTRUCTION_HEADER												

Bit Field Reverse

bfrev - Bit Field Reverse			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
The bfrev instruction component-wise reverses all the bits in src0 and stores the results in dst.			
Format: [(pred)] bfrev (exec_size) dst src0			
Restriction			
No accumulator access, implicit or explicit.			
Syntax			
[(pred)] bfrev (exec_size) reg reg [(pred)] bfrev (exec_size) reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { for (idx = 0; idx < 32; idx++) { dst.chan[n][idx] = src0.chan[n][31-idx]; } } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UD	UD		
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	

Branch Converging

brc - Branch Converging							
Project:	BDW						
Source:	Eulsa						
Length Bias:	4						
Description			Project				
<p>The brc instruction redirects the execution forward or backward to the instruction pointed by (current IP + offset). The jump will occur if all channels are branched away. UIP should reference the instruction where all channels are expected to come together. JIP should reference the end of the innermost conditional block.</p>							
<p>In GEN binary, JIP and UIP use locations src1 and src0 respectively when immediate and location src0 when reg64, where reg64 is accessed as paired DWord (regioning being <2;2,1>). dst must be IP. When the offsets are immediate, src0 regfile must be immediate.</p>			BDW				
Format: [(pred)] brc (exec_size) JIP UIP							
Restriction			Project				
A brc instruction cannot use the Switch instruction option.			BDW				
Syntax			Project				
[(pred)] brc (exec_size) imm32 imm32 [(pred)] brc (exec_size) reg64			BDW				
Pseudocode							
<pre>Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn[n]) { PclP[n] = IP + UIP; } else { PclP[n] = IP + 1; } } if (all PclP != IP + 1) { // for all channels Jump(IP + JIP); }</pre>							
Errata	Description		Project				
	[DevBDW:A]: A brc instruction must not be followed by any instruction requiring register indirect access on source operands.		BDW				
Predication	Conditional Modifier	Saturation	Source Modifier				
Y	N	N	N				
Src Types							
D							
DWord	Bit	Description					
0..3	127:96	<p>JIP</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte-aligned jump distance if a jump is taken for the channel.</p>		Project:	BDW	Format:	S31
Project:	BDW						
Format:	S31						



brc - Branch Converging						
	95:64	<p>UIP</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte aligned jump distance if a jump is taken for the instruction.</p>	Project:	BDW	Format:	S31
	Project:	BDW				
	Format:	S31				
63:32	<p>Operand Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>EU_INSTRUCTION_OPERAND_CONTROLS</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_CONTROLS			
Format:	EU_INSTRUCTION_OPERAND_CONTROLS					
31:0	<p>Header</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>EU_INSTRUCTION_HEADER</td> </tr> </table>	Format:	EU_INSTRUCTION_HEADER			
Format:	EU_INSTRUCTION_HEADER					

Branch Diverging

brd - Branch Diverging			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
The brd instruction redirects the execution forward or backward to the instruction pointed by (current IP + offset). The jump will occur if any channels are branched away.			
In GEN binary, JIP is at location src1 when immediate and at location src0 when reg32, where reg32 is accessed as a scalar DWord. The ip register must be used (for example, by the assembler) as dst.			BDW
Format: [(pred)] brd (exec_size) JIP			
Restriction			Project
A brd instruction cannot use the Switch instruction option.			BDW
Syntax			Project
[(pred)] brd (exec_size) imm32 [(pred)] brd (exec_size) reg32			BDW
Pseudocode			
Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn[n]) { PclP[n] = IP + JIP; } else { PclP[n] = IP + 1; } } if (any PclP == ExIP + JIP) { // any channel Jump(ExIP + JIP); }			
Errata	Description		Project
	[DevBDW:A]: A brd instruction must not be followed by any instruction requiring register indirect access on source operands.		BDW
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types			
D			
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		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	
		Project:	BDW

brd - Branch Diverging				
94:91	Src1.SrcType			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>SrcType</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	SrcType			
90:89	Src1.RegFile			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>RegFile</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	RegFile			
88:64	Source 0			
	<table border="1"> <tr> <td>Exists If:</td> <td>(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align16')</td> </tr> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16</td> </tr> </table>	Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align16')	Format:
Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align16')			
Format:	EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16			
88:64	Source 0			
	<table border="1"> <tr> <td>Exists If:</td> <td>(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align1')</td> </tr> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1</td> </tr> </table>	Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align1')	Format:
Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align1')			
Format:	EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1			
63:32	Operand Control			
	<table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_CONTROLS</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	
Format:	EU_INSTRUCTION_OPERAND_CONTROLS			
31:0	Header			
	<table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_HEADER</td> </tr> </table>	Format:	EU_INSTRUCTION_HEADER	
Format:	EU_INSTRUCTION_HEADER			

Break

break - Break			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
<p>The break instruction is used to early-out from the inner most loop, or early out from the inner most switch block. When used in a loop, upon execution, the break instruction terminates the loop for all execution channels enabled. If all the enabled channels hit the break instruction, jump to the instruction referenced by JIP. JIP should be the offset to the end of the inner most conditional or loop block, UIP should be the offset to the while instruction of the loop block. If SPF is ON, the UIP must be used to update IP; JIP is not used in this case</p>			
<p>The following table describes the two 32-bit instruction pointer offsets. Both the JIP and UIP are signed 32-bit numbers, added to IP pre-increment. In GEN binary, JIP and UIP are at locations src0 and src1 and must be of type DW (signed DWord integer). When the offsets are immediate, src0 regfile must be immediate.</p>			BDW
Format: [(pred)] break (exec_size) JIP UIP			
Syntax			Project
[(pred)] break (exec_size) imm16 imm16			BDW
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.channel[n]) { PclIP[n] = IP + UIP; else { PclIP[n] = IP + 1; } } if (PclIP != (IP + 1)) { // all channels Jump(IP + JIP); }			
Errata	Description		Project
	[DevBDW:A]: A break instruction must not be followed by any instruction requiring register indirect access on source operands.		BDW
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
	Format:	S31	
	The byte-aligned jump distance if a jump is taken for the channel.		
0..3	95:64	UIP	
		Project:	BDW
	Format:	S31	
	The byte aligned jump distance if a jump is taken for the instruction.		



break - Break		
	63:32	Operand Control Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

Byte Scattered Read MSD

MSDOR_BS - Byte Scattered Read MSD								
Project:	BDW							
Source:	DataPort 0							
Length Bias:	1							
Family:	Scattered R/W							
Group:	Byte Scattered R/W							
DWord	Bit	Description						
0	19	<p>Header Present</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	Enable		
	Project:	All						
	Format:	Enable						
	18	<p>Legacy Message</p> <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Legacy Message</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h						
	Project:	All						
	Format:	Opcode						
	17:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>04h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Byte Scattered Read message</p>	Default Value:	04h	Project:	All	Format:	Opcode
	Default Value:	04h						
	Project:	All						
Format:	Opcode							
13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ			
Project:	BDW							
Format:	MBZ							
12	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ			
Project:	All							
Format:	MBZ							
11:10	<p>Data Elements</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DS</td> </tr> </table> <p>Specifies the number of Bytes to be read or written per Dword</p>	Project:	All	Format:	MDC_DS			
Project:	All							
Format:	MDC_DS							



MSDOR_BS - Byte Scattered Read MSD			
	9	Reserved	
		Project:	All
		Format:	MBZ
		Ignored	
	8	SIMD Mode	
		Project:	All
		Format:	MDC_SM2
		Specifies the SIMD mode of the message (number of slots processed)	
	7:0	Binding Table Index	
	Project:	All	
	Format:	MDC_BTS_SLM_A32	
	Specifies the Binding Table Index for the message		

Byte Scattered Write MSD

MSD0W_BS - Byte Scattered Write MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Scattered R/W	
Group:	Byte Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: Enable
	If set, indicates that the message includes the header.	
	18	Legacy Message
		Default Value: 0h
Project: All		
Format: Opcode		
Legacy Message		
17:14	Message Type	
	Default Value: 0Ch	
	Project: All	
	Format: Opcode	
Byte Scattered Write message		
13:12	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
11:10	Data Elements	
	Project: All	
	Format: MDC_DS	
Specifies the number of Bytes to be read or written per Dword		
9	Reserved	
	Project: All	
	Format: MBZ	
	Ignored	



MSD0W_BS - Byte Scattered Write MSD					
8	<p>SIMD Mode</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_SM2</td> </tr> </table> <p>Specifies the SIMD mode of the message (number of slots processed)</p>	Project:	All	Format:	MDC_SM2
	Project:	All			
Format:	MDC_SM2				
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_SLM_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS_SLM_A32
	Project:	All			
Format:	MDC_BTS_SLM_A32				

Call

call - Call			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The call instruction jumps to a subroutine. It can be predicated or non-predicated. If non-predicated, all enabled channels jump to the subroutine. If predicated, only the channels enabled by PMask jump to the subroutine; the rest of the channels move to the next instruction after the call instruction. If none of the channels jump into the subroutine, the call instruction is treated as a nop. In case of a jump, the call instruction stores the return IP onto the first DWord of the destination register and stores the CallMask in the second DWord of the destination register. When SPF is on, the predication control must be scalar.</p> <p>The following table describes JIP, the jump offset. JIP can be an immediate or register value. When a jump occurs, this value is added to IP pre-increment. For BDW, in GEN binary, JIP is at location src1 when immediate and at location src0 when in a register. The IP register must be put (for example, by the assembler) at dst location. When offsets are immediate, src0 must be null.</p> <p>Format: [(pred)] call (exec_size) dst JIP</p>			
Restriction			
The call instruction must have DWord source and destination type, and the destination must be QWord aligned.			
A call instruction must use a Switch			
A call instruction must be followed by an instruction that supports Switch. When call takes a jump, the first instruction must have a Switch.			
Syntax			
[(pred)] call (exec_size) reg imm32 [(pred)] call (exec_size) reg reg32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { PclP[n] = IP + JIP; CallMask[n] = 1; } else { PclP[n] = IP + 1; CallMask[n] = 0; } } if (PclP[n] != (IP + 1)) { // any channel jumped dst.chan[0] = IP + 1; dst.chan[1] = CallMask; Jump(IP + JIP); }</pre>			
Errata	Description		
	[DevBDW:A]: A call instruction must not be followed by any instruction requiring register indirect access on source operands.		
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Dst Types			
D, UD			

DWord	Bit	Description	
0..3	127:96	JIP	
		Project: BDW	
		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	
		Project: BDW	
	94:91	Src1.SrcType	
		Project: BDW	
		Format: SrcType	
	90:89	Src1.RegFile	
Project: BDW			
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		

Call Absolute

calla - Call Absolute			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The calla instruction jumps to a subroutine. It can be predicated or non-predicated. If non-predicated, all enabled channels jump to the subroutine. If predicated, only the channels enabled by PMask jump to the subroutine; the rest of the channels move to the next instruction after the calla instruction. If none of the channels jump into the subroutine, the calla instruction is treated as a nop. In case of a jump, the call instruction stores the return IP onto the first DWord of the destination register and stores the CallMask in the second DWord of the destination register. If SPF is ON, none of the PclP are updated. When SPF is on, the predication control must be scalar. The difference between calla and call is that calla uses JIP as the IP value rather than adding it to the IP value.</p>			
Format: [(pred)] calla (exec_size) dst JIP			
Restriction			
The calla instruction must have DWord source and destination type, and the destination must be QWord-aligned.			
The src0 regioning control must be <2;2,1>			
Syntax			Project
[(pred)] calla (exec_size) reg imm32			BDW
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.channel[n]) { PclP[n] = JIP; CallMask[n] = 1; } else { PclP[n] = IP + 1; CallMask[n] = 0; } } if (PclP[n] != (IP + 1)) { // any channel jumped dst.chan[0] = IP + 1; dst.chan[1] = CallMask; Jump(JIP); }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Dst Types			
D, UD			
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		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	
		Project:	BDW

calla - Call Absolute		
94:91	Src1.SrcType	
	Project:	BDW
	Format:	SrcType
90:89	Src1.RegFile	
	Project:	BDW
	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

Compare

cmp - Compare			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The cmp instruction performs component-wise comparison of src0 and src1 and stores the results in the selected flag register and in dst. It takes component-wise subtraction of src0 and src1, evaluating the conditional code (excluding NS signal) based on the conditional modifier, and storing the conditional bits in bit-packed form in the destination flag register and all bits of dst channels. If the dst is not null, for the enabled channels, then all bits of the destination channel will contain the flag value for the channel. When the instruction operates on packed word format, one general register may store up to 16 such comparison results. In DWord format, one general register may store up to 8 results. A conditional modifier must be specified; the conditional modifier field cannot be 0000b. The comparison does not use the NS (NaN source) signals, as described in the Creating Conditional Flags section. Accordingly the conditional modifier should not be .u (unordered). For each enabled channel 0b or 1b is assigned to the appropriate flag bit and 0/all zeros or all ones (e.g, byte 0xFF, word 0xFFFF, DWord 0xFFFFFFFF) is assigned to dst. When any source type is floating-point, the cmp instruction obeys the rules described in the tables in the Floating Point Modes section of the Data Types chapter.</p>			
Format: [(pred)] cmp[.cmod] (exec_size) dst src0 src1			
Restriction			
Accumulator cannot be destination, implicit or explicit. The destination must be a general register or the null register.			
Syntax			
[(pred)] cmp[.cmod] (exec_size) reg reg reg [(pred)] cmp[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { bitMask[n] = 0; if (WrEn.chan[n]) { results[n] = src0.chan[n] - src1.chan[n]; bitMask[n] = Condition(results[n]); dst.chan[n] = bitMask[n]; // All bits for dst channel } } flag# = bitMask;			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	Y
Src Types		Dst Types	
*B,*W,*D		*B,*W,*D	
*B,*W,*D		F	
F		F	
DF		DF	
HF		HF	
*B,*W,*D		HF	

cmp - Compare		
*W,*D,*Q		*W,*D,*Q
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	

Compare NaN

cmpn - Compare NaN			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The cmpn instruction performs component-wise special-NaN comparison of src0 and src1 and stores the results in the selected flag register and in dst. It takes component-wise subtraction of src0 and src1, evaluating the conditional signals including NS based on the conditional modifier, and storing the conditional flag bits in bit-packed form in the destination flag register and all bits of dst channels. If the dst is not null, for the enabled channels, then all bits of the destination channel will contain the flag value for the channel. When the instruction operates on packed word format, one general register may store up to 16 such comparison results. In DWord format, one general register may store up to 8 results. A conditional modifier must be specified; the conditional modifier field cannot be 0000b. More information about the conditional signals used is in the Creating Conditional Flags section. For each enabled channel 0b or 1b is assigned to the appropriate flag bit and 0/all zeros or all ones (e.g, byte 0xFF, word 0xFFFF, DWord 0xFFFFFFFF) is assigned to dst. Min/Max instructions use cmpn to select the destination from the input sources (see the Min Max of Floating Point Numbers section for details).</p>			
Format: [(pred)] cmpn[.cmod] (exec_size) dst src0 src1			
Restriction			Project
Accumulator cannot be destination, implicit or explicit. The destination must be a general register or the null register.			
.l and .ge are the only two conditional modifiers are supported for this instruction.			BDW
Syntax			
[(pred)] cmpn[.cmod] (exec_size) reg reg reg [(pred)] cmpn[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { bitMask[n] = 0; if (WrEn.chan[n]) { results[n] = src0.chan[n] - src1.chan[n]; bitMask[n] = ConditionNaN(results[n]); dst.chan[n][0] = bitMask[n]; // All bits for dst channel } } flag# = bitMask;</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*B,*W,*D	F		
F	F		
DF	DF	BDW	
HF	HF	BDW	



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

Conditional Select

csel - Conditional Select			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The csel instruction selectively moves components in src0 or src1 to the dst based on the result of the compare of src2 with zero. If the channel condition is true, data in src0 is moved into dst. Otherwise, data in src1 is moved into dst. The csel instruction provides the function of a cmp followed by sel. The instruction must not be used if cmpn is required. The instruction does not update the flag register. The comparison follows the same rule as cmp instruction for that data type.</p>			
Format: csel (exec_size) dst src0 src1 src2			
Syntax			
csel[.cmod] (exec_size) reg reg reg reg			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { bitMask[n] = 0; if (EMask.chan[n]) { result[n] = src2.chan[n] - 0; bitMask[n] = Condition(result[n]); if (bitMask[n] = 1) { dst.chan[n] = src0.chan[n]; } else { dst.chan[n] = src1.chan[n]; } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
N	Y	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:126	Reserved Format: MBZ	
	125:106	Source 2 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	105	Reserved Format: MBZ	
	104:85	Source 1 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	84	Reserved Format: MBZ	
	83:64	Source 0 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	

csel - Conditional Select		
63:56	Destination Register Number	
	Format: DstRegNum	
55:53	Destination Subregister Number	
	Format: DstSubRegNum[2:0]	
52:49	Destination Channel Enable	
	Format: ChanEn[4] Four channel enables are defined for controlling which channels are written into the destination region. These channel mask bits are applied in a modulo-four manner to all ExecSize channels. There is 1-bit Channel Enable for each channel within the group of 4. If the bit is cleared, the write for the corresponding channel is disabled. If the bit is set, the write is enabled. Mnemonics for the bit being set for the group of 4 are <i>x</i> , <i>y</i> , <i>z</i> , and <i>w</i> , respectively, where <i>x</i> corresponds to Channel 0 in the group and <i>w</i> corresponds to channel 3 in the group	
48	Reserved	
	Project: MBZ Format: MBZ	
47	NibCtrl	
	Project: NibCtrl Format: NibCtrl	
46	Reserved	
	Project: MBZ Format: MBZ	
45:44	Destination Data Type	
	Project: MBZ	
	This field contains the data type for the destination	
	Value	Name
	00b	Single Precision Float
	01b	DWord
	11b	Double Precision Float
43:42	Source Data Type	
	Project: MBZ	
	This field contains the data type for all three sources	
	Value	Name
	00b	Single Precision Float
	01b	DWord
	11b	Double Precision Float

csel - Conditional Select		
41:40	Source 2 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
39:38	Source 1 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
37:36	Source 0 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
35	Reserved	
	Format:	MBZ
34	Flag Register Number	
	Project:	
	This field contains the flag register number for instructions with a non-zero Conditional Modifier.	
33	Flag Subregister Number	
	This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.	
32	Reserved	
	Project:	
	Format:	MBZ
31:0	Header	
	Format:	EU_INSTRUCTION_HEADER

Conditional Send Message

sendc - Conditional Send Message			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The sendc instruction has the same behavior as the send instruction except the following. sendc first checks the dependent threads inside the Thread Dependency Register. There are up to 8 dependent threads in the TDR register. The sendc instruction executes only when all the dependent threads in the TDR register are retired. Wait for dependencies in the TDR Register to clear, then send a message stored in registers starting at src to a shared function identified by exdesc along with control from desc with a general register writeback location at dst.</p>			
Format: [(pred)] sendc (exec_size) dst src0 exdesc desc			
Restriction			
The sendc instruction has the same restrictions as the send instruction.			
Pseudocode			
<pre>if (TDR[7] ... TDR[2] TDR[1] TDR[0]) { wait; } Evaluate(WrEn); MsgChEnable = WrEn; SourceReg = src0.RegNum; MessageQueue(MsgChEnable, ResponseReg, SourceReg, desc, exdesc);</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
DWord	Bit	Description	
0..3	127:96	Message	
		Format:	EU_INSTRUCTION_OPERAND_SEND_MSG
	95:89	Flags	
		Format:	EU_INSTRUCTION_FLAGS
	88:64	Source 0	
		Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]='Align1')
		Format:	EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1
	88:64	Source 0	
	Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]='Align16')	
	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16	
63:32	Operand Control		
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	
31:28	Controls B		
	Format:	EU_INSTRUCTION_CONTROLS_B	
27:24	Shared Function ID (SFID)		
	Format:	SFID	

sendc - Conditional Send Message		
	23:8	Controls A Format: EU_INSTRUCTION_CONTROLS_A
	7	Reserved Format: MBZ
	6:0	Opcode Format: EU_OPCODE

Constant Cache Oword Block Read MSD

MSD_CC_OWB - Constant Cache Oword Block Read MSD		
Project:	BDW	
Source:	Read-Only DataPort	
Length Bias:	1	
Family:	Block R/W	
Group:	OW Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18	Reserved
		Project: All
Format: MBZ		
Ignored		
17:14	Message Type	
	Default Value: 00h	
	Project: All	
	Format: Opcode	
Oword Block Read Constant Cache message		
13	Invalidate After Read	
	Project: All	
	Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs		
12:11	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
10:8	Data Elements	
	Project: All	
	Format: MDC_DB_OW	
Specifies the number of contiguous Owords to be read or written		

MSD_CC_OWB - Constant Cache Oword Block Read MSD					
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS
Project:	All				
Format:	MDC_BTS				



Constant Cache Oword Unaligned Block Read MSD

MSD_CC_OWUB - Constant Cache Oword Unaligned Block Read MSD							
Project:	BDW						
Source:	Read-Only DataPort						
Length Bias:	1						
Family:	Block R/W						
Group:	OW Unaligned Block R/W						
DWord	Bit	Description					
0	19	Header Present <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> Indicates that the message requires a header.	Project:	All	Format:	MDC_MHR	
	Project:	All					
	Format:	MDC_MHR					
	18	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> Ignored	Project:	All	Format:	MBZ	
	Project:	All					
	Format:	MBZ					
17:14	Message Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>01h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Oword Unaligned Block Read Constant Cache message	Default Value:	01h	Project:	All	Format:	Opcode
Default Value:	01h						
Project:	All						
Format:	Opcode						
13:11	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> Ignored	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
10:8	Data Elements <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DB_OW</td> </tr> </table> Specifies the number of contiguous Owords to be read	Project:	All	Format:	MDC_DB_OW		
Project:	All						
Format:	MDC_DB_OW						
7:0	Binding Table Index <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> Specifies the Binding Table Index for the message	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						

Continue

cont - Continue							
Project:	BDW						
Source:	Eulsa						
Length Bias:	4						
Description							
<p>The cont instruction disables execution for the subset of channels for the remainder of the current loop iteration. Channels remain disabled until right before the while instruction or right before the condition check code block for the while instruction. If all enabled channels hit this instruction, jump to the instruction referenced by JIP where execution continues. UIP should always reference the loop's associated while instruction. JIP should point to the last instruction of the inner most conditional block if the cont instruction is inside a conditional block. In case of the break instruction directly under the loop, the JIP and the UIP are the same. If SPF is ON, the UIP must be used to update IP; JIP is not used in this case.</p> <p>The following table describes the two 32-bit instruction pointer offsets. Both the JIP and UIP are signed 32-bit numbers, added to IP pre-increment. In GEN binary, JIP and UIP are at locations src0 and src1 and must be of type DW (signed DWord integer). When the offsets are immediate, src0 regfile must be immediate.</p> <p>Format: [(pred)] cont (exec_size) JIP UIP</p>							
Restriction							
The execution size must be the same for the while, break, and cont instructions of the same code block.							
Syntax							
[(pred)] cont (exec_size) imm32 imm32							
Pseudocode							
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.channel[n]) { if (PMask[n]) { // PMask is for all channels enabled for the cont instruction. PclP[n] = IP + UIP; } else { PclP[n] = IP + 1; } } } for (n = exec_size; n < 32; n++) { PclP[n] = IP + 1; } if (PclP != (IP + 1)) { // all channels true Jump(IP + JIP); }</pre>							
Errata	Description						
	[DevBDW:A]: A cont instruction must not be followed by any instruction requiring register indirect access on source operands.						
Predication	Conditional Modifier	Saturation	Source Modifier				
Y	N	N	N				
DWord	Bit	Description					
0..3	127:96	JIP <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte-aligned jump distance if a jump is taken for the channel.</p>		Project:	BDW	Format:	S31
Project:	BDW						
Format:	S31						

cont - Continue						
	95:64	<p>UIP</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte aligned jump distance if a jump is taken for the instruction.</p>	Project:	BDW	Format:	S31
	Project:	BDW				
	Format:	S31				
63:32	<p>Operand Control</p> <table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_CONTROLS</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_CONTROLS			
Format:	EU_INSTRUCTION_OPERAND_CONTROLS					
31:0	<p>Header</p> <table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_HEADER</td> </tr> </table>	Format:	EU_INSTRUCTION_HEADER			
Format:	EU_INSTRUCTION_HEADER					

Count Bits Set

cbit - Count Bits Set			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
The cbit instruction counts component-wise the total bits set in src0 and stores the resulting counts in dst.			
Format: [(pred)] cbit (exec_size) dst src0			
Restriction			
No accumulator access, implicit or explicit.			
Syntax			
[(pred)] cbit (exec_size) reg reg [(pred)] cbit (exec_size) reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD cnt = 0; UD val = src0.chan[n]; while (val) { if (val & 1) { cnt ++; } val = val » 1; } dst.chan[n] = cnt; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UB, UW, UD	UD		
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	

Dot Product 2

dp2 - Dot Product 2			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The dp2 instruction performs a two-wide dot product on four-tuple vector basis and storing the same scalar result per four tuple to all four channels in dst. This instruction is similar to dp4 except that every third and fourth element of src0 (post-source-swizzle if present) are not involved in the computation. The dot product of two vectors of equal length is the sum of the products of each pair of corresponding elements. The dp4 instruction includes all four elements of each vector in the dot product. The dp3 instruction includes the first three elements of each vector in the dot product.</p>			
Format: [(pred)] dp2[.cmod] (exec_size) dst src0 src1			
Restriction			
Execution size cannot be less than 4.			
Horizontal strides must be 1.			
Source operands cannot be accumulators.			
Syntax			
[(pred)] dp2[.cmod] (exec_size) reg reg reg [(pred)] dp2[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n += 4) { fTmp = src0.chan[n] * src1.chan[n] + src0.chan[n+1] * src1.chan[n+1]; if (WrEn.chan[n]) dst.chan[n] = fTmp; if (WrEn.chan[n+1]) dst.chan[n+1] = fTmp; if (WrEn.chan[n+2]) dst.chan[n+2] = fTmp; if (WrEn.chan[n+3]) dst.chan[n+3] = fTmp; }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	([RegSource][Src1.RegFile]!='IMM')
	Format:	EU_INSTRUCTION_SOURCES_REG_REG	
127:64	127:64	ImmSource	
		Exists If:	([ImmSource][Src1.RegFile]='IMM')
Format:	EU_INSTRUCTION_SOURCES_REG_IMM		
63:32	63:32	Operand Controls	
		Format:	EU_INSTRUCTION_OPERAND_CONTROLS

dp2 - Dot Product 2

	31:0	Header
		Format: EU_INSTRUCTION_HEADER



Dot Product 3

dp3 - Dot Product 3			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The dp3 instruction performs a three-wide dot product on four-tuple vector basis and storing the same scalar result per four tuple to all four channels in dst. This instruction is similar to dp4 except that every fourth element of src0 (post-source-swizzle if present) is not involved in the computation. The dot product of two vectors of equal length is the sum of the products of each pair of corresponding elements. The dp4 instruction includes all four elements of each vector in the dot product. The dp2 instruction includes the first two elements of each vector in the dot product.</p>			
Format: [(pred)] dp3[.cmod] (exec_size) dst src0 src1			
Restriction			
Execution size cannot be less than 4.			
Horizontal strides must be 1.			
Source operands cannot be accumulators.			
Syntax			
[(pred)] dp3[.cmod] (exec_size) reg reg reg [(pred)] dp3[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n += 4) { fTmp = src0.chan[n] * src1.chan[n] + src0.chan[n+1] * src1.chan[n+1] + src0.chan[n+2] * src1.chan[n+2]; if (WrEn.chan[n]) dst.chan[n] = fTmp; if (WrEn.chan[n+1]) dst.chan[n+1] = fTmp; if (WrEn.chan[n+2]) dst.chan[n+2] = fTmp; if (WrEn.chan[n+3]) dst.chan[n+3] = fTmp; }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
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		

dp3 - Dot Product 3		
	63:32	Operand Controls Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

Dot Product 4

dp4 - Dot Product 4			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The dp4 instruction performs a four-wide dot product on four-tuple vector basis and storing the same scalar result per four tuple to all four channels in dst. The dot product of two vectors of equal length is the sum of the products of each pair of corresponding elements.</p>			
Format: [(pred)] dp4[.cmod] (exec_size) dst src0 src1			
Restriction			
Execution size cannot be less than 4.			
Horizontal strides must be 1.			
Source operands cannot be accumulators.			
Syntax			
[(pred)] dp4[.cmod] (exec_size) reg reg reg [(pred)] dp4[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n += 4) { fTmp = src0.chan[n] * src1.chan[n] + src0.chan[n+1] * src1.chan[n+1] + src0.chan[n+2] * src1.chan[n+2] + src0.chan[n+3] * src1.chan[n+3]; if (WrEn.chan[n]) dst.chan[n] = fTmp; if (WrEn.chan[n+1]) dst.chan[n+1] = fTmp; if (WrEn.chan[n+2]) dst.chan[n+2] = fTmp; if (WrEn.chan[n+3]) dst.chan[n+3] = fTmp; }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
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

dp4 - Dot Product 4			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER

Dot Product Homogeneous

dph - Dot Product Homogeneous			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The dph instruction performs a four-wide homogeneous dot product on four-tuple vector basis and storing the same scalar result per four tuple to all four channels in dst. This instruction is similar to dp4 except that every fourth element of src0 (post-source-swizzle if present) is forced to 1.0f. Use the dp4 instruction to do a four-wide dot product that includes all elements of src0 and src1.</p>			
Format: <code>[(pred)] dph[.cmod] (exec_size) dst src0 src1</code>			
Restriction			
Execution size cannot be less than 4.			
Horizontal strides must be 1.			
Source operands cannot be accumulators.			
Syntax			
<code>[(pred)] dph[.cmod] (exec_size) reg reg reg [(pred)] dph[.cmod] (exec_size) reg reg imm32</code>			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n += 4) { fTmp = src0.chan[n] * src1.chan[n] + src0.chan[n+1] * src1.chan[n+1] + src0.chan[n+2] * src1.chan[n+2] + src1.chan[n+3]; // Use 1.0f in place of src0.chan[n+3]. if (WrEn.chan[n]) dst.chan[n] = fTmp; if (WrEn.chan[n+1]) dst.chan[n+1] = fTmp; if (WrEn.chan[n+2]) dst.chan[n+2] = fTmp; if (WrEn.chan[n+3]) dst.chan[n+3] = fTmp; }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	<code>[(RegSource)[Src1.RegFile]!='IMM']</code>
	Format:	EU_INSTRUCTION_SOURCES_REG_REG	
	127:64	ImmSource	
		Exists If:	<code>[(ImmSource)[Src1.RegFile]='IMM']</code>
	Format:	EU_INSTRUCTION_SOURCES_REG_IMM	
63:32	Operand Controls		
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	

dph - Dot Product Homogeneous		
	31:0	Header
		Format: EU_INSTRUCTION_HEADER

Dword Atomic Counter Binary with Return Data Operation MSD

MSD1R_DWAC2 - Dword Atomic Counter Binary with Return Data Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Atomic Counter Binary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
		Indicates that the message requires a header
	18:14	Message Type
		Default Value: 0Bh
		Project: All
		Format: Opcode Atomic Counter Operation message
	13	Return Data Control
		Default Value: 1h
		Project: All
		Format: Opcode Specifies that return data is sent back to the thread.
	12	SIMD Mode
		Project: All
		Format: MDC_SM2RS Specifies the SIMD mode of the message (number of slots processed)
	11:8	Atomic Integer Operation
		Project: All
		Format: MDC_AOP2 Specifies the atomic integer operation to be performed.
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS Specifies the Binding Table Index for the message	

Dword Atomic Counter Binary Write Only Operation MSD

MSD1W_DWAC2 - Dword Atomic Counter Binary Write Only Operation MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Untyped Atomic Operation							
Group:	Dword Atomic Counter Binary Operation							
DWord	Bit	Description						
0	19	<p>Header Present</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> <p>Indicates that the message requires a header</p>	Project:	All	Format:	MDC_MHR		
	Project:	All						
	Format:	MDC_MHR						
	18:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0Bh</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Atomic Counter Operation message</p>	Default Value:	0Bh	Project:	All	Format:	Opcode
	Default Value:	0Bh						
	Project:	All						
Format:	Opcode							
13	<p>Return Data Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h							
Project:	All							
Format:	Opcode							
12	<p>SIMD Mode</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_SM2RS</td> </tr> </table> <p>Specifies the SIMD mode of the message (number of slots processed)</p>	Project:	All	Format:	MDC_SM2RS			
Project:	All							
Format:	MDC_SM2RS							
11:8	<p>Atomic Integer Operation</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2			
Project:	All							
Format:	MDC_AOP2							
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS			
Project:	All							
Format:	MDC_BTS							



Dword Atomic Counter Unary with Return Data Operation MSD

MSD1R_DWAC1 - Dword Atomic Counter Unary with Return Data Operation MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Untyped Atomic Operation							
Group:	Dword Atomic Counter Unary Operation							
DWord	Bit	Description						
0	19	Header Present <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> Indicates that the message requires a header	Project:	All	Format:	MDC_MHR		
		Project:	All					
		Format:	MDC_MHR					
	18:14	Message Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0Bh</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Atomic Counter Operation message	Default Value:	0Bh	Project:	All	Format:	Opcode
		Default Value:	0Bh					
		Project:	All					
Format:	Opcode							
Return Data Control <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Specifies that return data is sent back to the thread.	Default Value:	1h	Project:	All	Format:	Opcode		
Default Value:	1h							
Project:	All							
Format:	Opcode							
12	SIMD Mode <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_SM2RS</td> </tr> </table> Specifies the SIMD mode of the message (number of slots processed)	Project:	All	Format:	MDC_SM2RS			
	Project:	All						
Format:	MDC_SM2RS							
11:8	Atomic Integer Operation <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> Specifies the atomic integer operation to be performed.	Project:	All	Format:	MDC_AOP1			
Project:	All							
Format:	MDC_AOP1							
7:0	Binding Table Index <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> Specifies the Binding Table Index for the message	Project:	All	Format:	MDC_BTS			
	Project:	All						
	Format:	MDC_BTS						

Dword Atomic Counter Unary Write Only Operation MSD

MSD1W_DWAC1 - Dword Atomic Counter Unary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Atomic Counter Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
		Indicates that the message requires a header
	18:14	Message Type
		Default Value: 0Bh
		Project: All
		Format: Opcode
	Atomic Counter Operation message	
	13	Return Data Control
		Default Value: 0h
		Project: All
Format: Opcode		
Specifies that no return data is sent back to the thread.		
12	SIMD Mode	
	Project: All	
	Format: MDC_SM2RS	
	Specifies the SIMD mode of the message (number of slots processed)	
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP1	
	Specifies the atomic integer operation to be performed.	
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS	
	Specifies the Binding Table Index for the message	

Dword Scattered Write MSD

MSD0W_DWS - Dword Scattered Write MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Scattered R/W	
Group:	DW Scattered R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: Enable
		If set, indicates that the message includes the header.
	18	Legacy Message
		Default Value: 0h
		Project: All
		Format: Opcode
	Legacy Message	
	17:14	Message Type
		Default Value: 0Bh
		Project: All
		Format: Opcode
	Dword Scattered Write message	
	13:12	Reserved
		Project: All
		Format: MBZ
		Ignored
	11:10	Reserved
		Project: BDW
		Format: MBZ
		Ignored
	9	Legacy SIMD Mode
		Default Value: 1h
Project: All		
Format: Opcode		
Must be set for compatibility.		

MSD0W_DWS - Dword Scattered Write MSD						
	8	SIMD Mode <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_SM2</td> </tr> </table> <p>Specifies the SIMD mode of the message (number of slots processed)</p>	Project:	All	Format:	MDC_SM2
	Project:	All				
Format:	MDC_SM2					
	7:0	Binding Table Index <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	BDW	Format:	MDC_BTS_A32
Project:	BDW					
Format:	MDC_BTS_A32					



Dword SIMD4x2 Atomic Counter Binary with Return Data Operation MSD

MSD1R_DWAC2_4x2 - Dword SIMD4x2 Atomic Counter Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Atomic Counter Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> <p>Indicates that the message requires a header</p>	Project:	All	Format:	MDC_MHR	
	Project:	All					
	Format:	MDC_MHR					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Atomic Counter Operation SIMD4x2 message</p>	Default Value:	0Ch	Project:	All	Format:
Default Value:		0Ch					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
Project:	All						
Format:	MDC_AOP2						
7:0	Binding Table Index						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						

Dword SIMD4x2 Atomic Counter Binary Write Only Operation MSD

MSD1W_DWAC2_4x2 - Dword SIMD4x2 Atomic Counter Binary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Atomic Counter Binary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
		Indicates that the message requires a header
	18:14	Message Type
		Default Value: 0Ch
		Project: All
		Format: Opcode
	Atomic Counter Operation SIMD4x2 message	
	13	Return Data Control
		Default Value: 0h
		Project: All
		Format: Opcode
	Specifies that no return data is sent back to the thread.	
	12	Reserved
		Project: All
		Format: MBZ
		Ignored
	11:8	Atomic Integer Operation
		Project: All
		Format: MDC_AOP2
		Specifies the atomic integer operation to be performed.
	7:0	Binding Table Index
		Project: All
Format: MDC_BTS		
Specifies the Binding Table Index for the message		



Dword SIMD4x2 Atomic Counter Unary with Return Data Operation MSD

MSD1R_DWAC1_4x2 - Dword SIMD4x2 Atomic Counter Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Atomic Counter Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> <p>Indicates that the message requires a header</p>	Project:	All	Format:	MDC_MHR	
	Project:	All					
	Format:	MDC_MHR					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Atomic Counter Operation SIMD4x2 message</p>	Default Value:	0Ch	Project:	All	Format:
Default Value:		0Ch					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1		
Project:	All						
Format:	MDC_AOP1						
7:0	Binding Table Index						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						

Dword SIMD4x2 Atomic Counter Unary Write Only Operation MSD

MSD1W_DWAC1_4x2 - Dword SIMD4x2 Atomic Counter Unary Write Only Operation MSD

Project: BDW
 Source: DataPort 1
 Length Bias: 1
 Family: Untyped Atomic Operation
 Group: Dword Atomic Counter Unary Operation

DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header	
	18:14	Message Type
		Default Value: 0Ch
Project: All		
Format: Opcode		
Atomic Counter Operation SIMD4x2 message		
13	Return Data Control	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Specifies that no return data is sent back to the thread.		
12	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP1	
Specifies the atomic integer operation to be performed.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS	
Specifies the Binding Table Index for the message		



Dword SIMD4x2 Typed Atomic Integer Binary with Return Data Operation MSD

MSD1R_DWTAI2_4x2 - Dword SIMD4x2 Typed Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Typed Atomic Operation						
Group:	Dword Typed Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>07h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Atomic Integer Operation SIMD4x2 message</p>	Default Value:	07h	Project:	All	Format:
Default Value:		07h					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
	Default Value:	1h					
	Project:	All					
Format:	Opcode						
Reserved							
12	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
	Project:	All					
Format:	MBZ						
Atomic Integer Operation							
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
	Project:	All					
Format:	MDC_AOP2						
Binding Table Index							
7:0	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
	Project:	All					
Format:	MDC_BTS						

Dword SIMD4x2 Typed Atomic Integer Binary Write Only Operation MSD

MSD1W_DWTAI2_4x2 - Dword SIMD4x2 Typed Atomic Integer Binary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Typed Atomic Operation						
Group:	Dword Typed Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>07h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Atomic Integer Operation SIMD4x2 message</p>	Default Value:	07h	Project:	All	Format:
Default Value:		07h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
Project:	All						
Format:	MDC_AOP2						
7:0	Binding Table Index						
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						



Dword SIMD4x2 Typed Atomic Integer Trinary with Return Data Operation MSD

MSD1R_DWTAI3_4x2 - Dword SIMD4x2 Typed Atomic Integer Trinary with Return Data Operation MSD					
Project:	BDW				
Source:	DataPort 1				
Length Bias:	1				
Family:	Typed Atomic Operation				
Group:	Dword Typed Atomic Integer Trinary Operation				
DWord	Bit	Description			
0	19	Header Present			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:
	Project:	All			
	Format:	MDC_MHP			
	18:14	Message Type	Default Value:	07h	
			Project:	All	
Format:			Opcode		
Typed Atomic Integer Operation SIMD4x2 message					
13	Return Data Control	Default Value:	1h		
		Project:	All		
		Format:	Opcode		
Specifies that return data is sent back to the thread.					
12	Reserved	Project:	All		
		Format:	MBZ		
Ignored					
11:8	Atomic Integer Operation	Project:	All		
		Format:	MDC_AOP3S		
Specifies the atomic integer operation to be performed.					
7:0	Binding Table Index	Project:	All		
		Format:	MDC_BTS		
Specifies the Binding Table Index for the message					

Dword SIMD4x2 Typed Atomic Integer Trinary Write Only Operation MSD

MSD1W_DWTAI3_4x2 - Dword SIMD4x2 Typed Atomic Integer Trinary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Typed Atomic Operation						
Group:	Dword Typed Atomic Integer Trinary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>07h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Atomic Integer Operation SIMD4x2 message</p>	Default Value:	07h	Project:	All	Format:
Default Value:		07h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3S</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP3S		
Project:	All						
Format:	MDC_AOP3S						
7:0	Binding Table Index						
	<table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						



Dword SIMD4x2 Typed Atomic Integer Unary with Return Data Operation MSD

MSD1R_DWTAI1_4x2 - Dword SIMD4x2 Typed Atomic Integer Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Typed Atomic Operation						
Group:	Dword Typed Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>07h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Atomic Integer Operation SIMD4x2 message</p>	Default Value:	07h	Project:	All	Format:
Default Value:		07h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1		
Project:	All						
Format:	MDC_AOP1						
7:0	Binding Table Index						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						

Dword SIMD4x2 Typed Atomic Integer Unary Write Only Operation MSD

MSD1W_DWTAI1_4x2 - Dword SIMD4x2 Typed Atomic Integer Unary Write Only Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Typed Atomic Operation						
Group:	Dword Typed Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Default Value:</td> <td>07h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Atomic Integer Operation SIMD4x2 message</p>	Default Value:	07h	Project:	All	Format:
Default Value:		07h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that no return data is sent back to the thread.</p>	Default Value:	0h	Project:	All	Format:	Opcode	
Default Value:	0h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1		
Project:	All						
Format:	MDC_AOP1						
7:0	Binding Table Index						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						



Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD

MSD1R_DWAI2_4x2 - Dword SIMD4x2 Untyped Atomic Integer Binary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Binary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>03h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	03h	Project:	All	Format:
Default Value:		03h					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP2</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP2		
Project:	All						
Format:	MDC_AOP2						
7:0	Binding Table Index						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_SLM_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS_SLM_A32		
Project:	All						
Format:	MDC_BTS_SLM_A32						

Dword SIMD4x2 Untyped Atomic Integer Ternary with Return Data Operation MSD

MSD1R_DWAI3_4x2 - Dword SIMD4x2 Untyped Atomic Integer Ternary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Ternary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
		<table border="1"> <tr> <td>Default Value:</td> <td>03h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Untyped Atomic Integer Operation SIMD4x2 message</p>	Default Value:	03h	Project:	All	Format:
Default Value:		03h					
Project:	All						
Format:	Opcode						
Return Data Control							
<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode	
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						
11:8	Atomic Integer Operation						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP3</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP3		
Project:	All						
Format:	MDC_AOP3						
7:0	Binding Table Index						
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_SLM_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS_SLM_A32		
Project:	All						
Format:	MDC_BTS_SLM_A32						



Dword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD

MSD1W_DWAI3_4x2 - Dword SIMD4x2 Untyped Atomic Integer Trinary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Untyped Atomic Integer Trinary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	If set, indicates that the message includes the header.	
	18:14	Message Type
		Default Value: 03h
Project: All		
Format: Opcode		
Untyped Atomic Integer Operation SIMD4x2 message		
13	Return Data Control	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Specifies that no return data is sent back to the thread.		
12	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP3	
Specifies the atomic integer operation to be performed.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS_SLM_A32	
Specifies the Binding Table Index for the message		

Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD

MSD1R_DWAI1_4x2 - Dword SIMD4x2 Untyped Atomic Integer Unary with Return Data Operation MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Untyped Atomic Operation						
Group:	Dword Untyped Atomic Integer Unary Operation						
DWord	Bit	Description					
0	19	Header Present					
		<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP	
	Project:	All					
	Format:	MDC_MHP					
	18:14	Message Type					
<table border="1"> <tr> <td>Default Value:</td> <td>03h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Untyped Atomic Integer Operation SIMD4x2 message</p>		Default Value:	03h	Project:	All	Format:	Opcode
Default Value:		03h					
Project:	All						
Format:	Opcode						
Return Data Control							
13	<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Specifies that return data is sent back to the thread.</p>	Default Value:	1h	Project:	All	Format:	Opcode
Default Value:	1h						
Project:	All						
Format:	Opcode						
12	Reserved						
11:8	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ		
	Project:	All					
Format:	MBZ						
Atomic Integer Operation							
7:0	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_AOP1</td> </tr> </table> <p>Specifies the atomic integer operation to be performed.</p>	Project:	All	Format:	MDC_AOP1		
Project:	All						
Format:	MDC_AOP1						
Binding Table Index							
7:0	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_SLM_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS_SLM_A32		
Project:	All						
Format:	MDC_BTS_SLM_A32						



Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD

MSD1W_DWAI1_4x2 - Dword SIMD4x2 Untyped Atomic Integer Unary Write Only Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Atomic Operation	
Group:	Dword Untyped Atomic Integer Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	If set, indicates that the message includes the header.	
	18:14	Message Type
		Default Value: 03h
Project: All		
Format: Opcode		
Untyped Atomic Integer Operation SIMD4x2 message		
13	Return Data Control	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Specifies that no return data is sent back to the thread.		
12	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP1	
Specifies the atomic integer operation to be performed.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS_SLM_A32	
Specifies the Binding Table Index for the message		

Dword Typed Atomic Integer Binary with Return Data Operation MSD

MSD1R_DWTAI2 - Dword Typed Atomic Integer Binary with Return Data Operation MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Typed Atomic Operation		
Group:	Dword Typed Atomic Integer Binary Operation		
DWord	Bit	Description	
0	19	Header Present	
		Project: All	
		Format: MDC_MHP	
		If set, indicates that the message includes the header.	
	18:14	Message Type	
		Default Value: 06h	
		Project: All	
		Format: Opcode	
			Typed Atomic Integer Operation message
	13	Return Data Control	
		Default Value: 1h	
		Project: All	
		Format: Opcode	
			Specifies that return data is sent back to the thread.
	12	Slot Group	
		Project: All	
		Format: MDC_SG2	
		Specifies the Slot Group mode of the message (which slots are processed)	
	11:8	Atomic Integer Operation	
		Project: All	
		Format: MDC_AOP2	
		Specifies the atomic integer operation to be performed.	
	7:0	Binding Table Index	
		Project: All	
Format: MDC_BTS			
Specifies the Binding Table Index for the message			

Dword Typed Atomic Integer Binary Write Only Operation MSD

MSD1W_DWTAI2 - Dword Typed Atomic Integer Binary Write Only Operation MSD

Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Typed Atomic Operation	
Group:	Dword Typed Atomic Integer Binary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	If set, indicates that the message includes the header.	
	18:14	Message Type
		Default Value: 06h
Project: All		
Format: Opcode		
Typed Atomic Integer Operation message		
13	Return Data Control	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Specifies that no return data is sent back to the thread.		
12	Slot Group	
	Project: All	
	Format: MDC_SG2	
Specifies the Slot Group mode of the message (which slots are processed)		
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP2	
Specifies the atomic integer operation to be performed.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS	
Specifies the Binding Table Index for the message		

Dword Typed Atomic Integer Trinary with Return Data Operation MSD

MSD1R_DWTAI3 - Dword Typed Atomic Integer Trinary with Return Data Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Typed Atomic Operation	
Group:	Dword Typed Atomic Integer Trinary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
		If set, indicates that the message includes the header.
	18:14	Message Type
		Default Value: 06h
		Project: All
		Format: Opcode Typed Atomic Integer Operation message
	13	Return Data Control
		Default Value: 1h
Project: All		
Format: Opcode Specifies that return data is sent back to the thread.		
12	Slot Group	
	Project: All	
	Format: MDC_SG2 Specifies the Slot Group mode of the message (which slots are processed)	
11:8	Atomic Integer Operation	
	Project: All	
	Format: MDC_AOP3S Specifies the atomic integer operation to be performed.	
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS Specifies the Binding Table Index for the message	

Dword Typed Atomic Integer Trinary Write Only Operation MSD

MSD1W_DWTAI3 - Dword Typed Atomic Integer Trinary Write Only Operation MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Typed Atomic Operation		
Group:	Dword Typed Atomic Integer Trinary Operation		
DWord	Bit	Description	
0	19	Header Present	
		Project: All	
		Format: MDC_MHP	
		If set, indicates that the message includes the header.	
	18:14	Message Type	
		Default Value: 06h	
		Project: All	
		Format: Opcode	
			Typed Atomic Integer Operation message
	13	Return Data Control	
		Default Value: 0h	
		Project: All	
Format: Opcode			
		Specifies that no return data is sent back to the thread.	
12	Slot Group		
	Project: All		
	Format: MDC_SG2		
		Specifies the Slot Group mode of the message (which slots are processed)	
11:8	Atomic Integer Operation		
	Project: All		
	Format: MDC_AOP3S		
		Specifies the atomic integer operation to be performed.	
7:0	Binding Table Index		
	Project: All		
	Format: MDC_BTS		
		Specifies the Binding Table Index for the message	

Dword Typed Atomic Integer Unary with Return Data Operation MSD

MSD1R_DWTAI1 - Dword Typed Atomic Integer Unary with Return Data Operation MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Typed Atomic Operation	
Group:	Dword Typed Atomic Integer Unary Operation	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
		If set, indicates that the message includes the header.
	18:14	Message Type
		Default Value: 06h
		Project: All
		Format: Opcode
	Typed Atomic Integer Operation message	
	13	Return Data Control
		Default Value: 1h
		Project: All
		Format: Opcode
	Specifies that return data is sent back to the thread.	
	12	Slot Group
		Project: All
		Format: MDC_SG2
		Specifies the Slot Group mode of the message (which slots are processed)
	11:8	Atomic Integer Operation
		Project: All
		Format: MDC_AOP1
		Specifies the atomic integer operation to be performed.
	7:0	Binding Table Index
		Project: All
Format: MDC_BTS		
Specifies the Binding Table Index for the message		

Dword Typed Atomic Integer Unary Write Only Operation MSD

MSD1W_DWTAI1 - Dword Typed Atomic Integer Unary Write Only Operation MSD			
Project:	BDW		
Source:	DataPort 1		
Length Bias:	1		
Family:	Typed Atomic Operation		
Group:	Dword Typed Atomic Integer Unary Operation		
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHP
			If set, indicates that the message includes the header.
	18:14	Message Type	
		Default Value:	06h
		Project:	All
		Format:	Opcode
			Typed Atomic Integer Operation message
	13	Return Data Control	
Default Value:		0h	
Project:		All	
Format:		Opcode	
		Specifies that no return data is sent back to the thread.	
12	Slot Group		
	Project:	All	
	Format:	MDC_SG2	
		Specifies the Slot Group mode of the message (which slots are processed)	
11:8	Atomic Integer Operation		
	Project:	All	
	Format:	MDC_AOP1	
		Specifies the atomic integer operation to be performed.	
7:0	Binding Table Index		
	Project:	All	
	Format:	MDC_BTS	
		Specifies the Binding Table Index for the message	

Else

else - Else			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The else instruction is an optional statement within an if/else/endif block of code. It restricts execution within the else/endif portion to the opposite set of channels enabled under the if/else portion. Channels which were inactive prior to entering the if/endif block remain inactive throughout the entire block. All enabled channels upon arriving the else instruction will be redirected to the matching endif. If all channels are redirected (by else or before else), a relative jump is performed to the location specified by <JIP>. The jump target should be the matching endif instruction for that conditional block. The following table describes the 32-bit <JIP>. In GEN binary, <JIP> is at location <src1> and must be of type D (signed dword integer). <JIP> must be an immediate operand, it is a signed 32-bit number and is intended to be forward referencing. This value is added to IP pre-increment. If the <branch_ctrlt> bit is set, then the <JIP> points to the first join instruction within the else block and <UIP> points to the endif instruction. If the <branch_ctrl> bit is not set, <JIP> and <UIP>, both point to endif.</p>			
Format: else (<exec_size>) <JIP> <UIP>; <branch_ctrlt>			
Programming Notes			
If all channels are redirected (by else or before else), relative jump is performed to the location specified by <JIP> + 1.			
Restriction			
Predication is not allowed.			
The execution size must be the same for the if, else, and endif instructions of the same code block.			
Syntax			
else (<exec_size>) imm32 imm32 <branch_ctrlt>			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn.channel[n] == 1 <branch_ctrlt>) { PciP[n] = IP + <JIP>; } else { PciP[n] = IP + <UIP>; } } if (PciP != (IP+1)) { // for all channels Jump(IP + <JIP>;) }			
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		Format:	S31
The byte-aligned jump distance if a jump is taken for the channel.			



else - Else						
	95:64	<p>UIP</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte aligned jump distance if a jump is taken for the instruction.</p>	Project:	BDW	Format:	S31
	Project:	BDW				
	Format:	S31				
63:32	<p>Operand Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>EU_INSTRUCTION_OPERAND_CONTROLS</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_CONTROLS			
Format:	EU_INSTRUCTION_OPERAND_CONTROLS					
31:0	<p>Header</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>EU_INSTRUCTION_HEADER</td> </tr> </table>	Format:	EU_INSTRUCTION_HEADER			
Format:	EU_INSTRUCTION_HEADER					

End If

endif - End If			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The endif instruction terminates an if/else/endif block of code. It restores execution to the channels that were active prior to the if/else/endif block. The endif instruction is also used to hop out of nested conditionals by jumping to the end of the next outer conditional block when all channels are disabled.</p> <p>The following table describes the 32-bit JIP. In GEN binary, JIP is at location src1 and must be of type D (signed DWord integer). JIP must be an immediate operand, it is a signed 32-bit number. This value is added to IP pre-increment.</p>			
Format: endif JIP			
Restriction			
Predication is not allowed.			
The execution size must be the same for the if, else, and endif instructions of the same code block.			
Syntax			
endif (exec_size) imm32			
Pseudocode			
Evaluate(WrEn); if (WrEn == 0) { // all channels false Jump(IP + JIP); }			
Errata	Description		
	[DevBDW:A]: An endif instruction must not be followed by any instruction requiring register indirect access on source operands.		
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		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	
		Project:	BDW

endif - End If		
94:91	Src1.SrcType	
	Project:	BDW
	Format:	SrcType
90:89	Src1.RegFile	
	Project:	BDW
	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

Extended Math Function

math - Extended Math Function	
Project:	BDW
Source:	Eulsa
Length Bias:	4
<p>The math instruction performs extended math function on the components in src0, or src0 and src1, and write the output to the channels of dst. The type of extended math function are based on the FC[3:0] encoding in the table below.</p>	
<p>Format: [(pred)] math (exec_size) dst src0 src1 <FC></p>	
Restriction	
Accumulator access is allowed only for ieee macro functions.	
The math instruction does not support indirect addressing modes.	
The only supported rounding mode for math instruction is Round to Nearest Even.	
INT DIV function does not support SIMD16.	
INT DIV function does not support simultaneous write to two registers.	
INT DIV function does not support source modifiers.	
The FDIV function is not supported in ALT_MODE.	
The math instruction can use GRF or immediates as source. The source formats for immediates must be one of the source formats supported by math operation.	
DepCtrl must not be used.	
The math instruction must use GRF as destination.	
The supported regioning mode for math instruction is align1 and align16. The following restrictions apply for align1 mode: Scalar source is supported. Source and destination horizontal stride must be the same. Regioning must ensure Src.Vstride = Src.Width * Src.Hstride . Source and destination offset must be the same, except the case of scalar source.	
For one source math operations src1 must be NULL.	
Syntax	
[(pred)] math (exec_size) reg reg reg imm4	
Pseudocode	
<pre> Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.channel[n] == 1) { switch FC[3:0] { case 1h: dst.channel[n] = rcp(src0.channel[n]); case 2h: dst.channel[n] = log(src0.channel[n]); case 3h: dst.channel[n] = exp(src0.channel[n]); case 4h: dst.channel[n] = sqrt(src0.channel[n]); } } } </pre>	

math - Extended Math Function				
	6:0	Opcode		
		<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td style="width: 50%; text-align: center;">EU_OPCODE</td> </tr> </table>	Format:	EU_OPCODE
Format:	EU_OPCODE			

Find First Bit from LSB Side

fbl - Find First Bit from LSB Side			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
The fbl instruction counts component-wise the number of LSB 0 bits before the first 1 bit in src0, storing that number in dst.			
Format: [(pred)] fbl (exec_size) dst src0			
Programming Notes			
If src0 contains no 1 bits, store 0xFFFFFFFF in dst.			
Restriction			
No accumulator access, implicit or explicit.			
Syntax			
[(pred)] fbl (exec_size) reg reg [(pred)] fbl (exec_size) reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD cnt = 0; UD udScalar = src0.chan[n]; while ((udScalar & 1) == 0 && cnt != 32) { cnt ++; udScalar = udScalar » 1; } if (src0.chan[n] == 0x00000000) { dst.chan[n] = 0xFFFFFFFF; } else { dst.chan[n] = cnt; } } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
UD	UD		
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	

Find First Bit from MSB Side

fbh - Find First Bit from MSB Side			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>If src0 is unsigned, the fbh instruction counts component-wise the leading zeros from src0 and stores the resulting counts in dst. If src0 is signed and positive, the fbh instruction counts component-wise the leading zeros from src0 and stores the resulting counts in dst. If src0 is signed and negative, the fbh instruction counts component-wise the leading ones from src0 and stores the resulting counts in dst.</p>			
Format: [(pred)] fbh (exec_size) dst src0			
Programming Notes			
If src0 is zero, store 0xFFFFFFFF in dst.			
If src0 is signed and is -1 (0xFFFFFFFF), store 0xFFFFFFFF in dst.			
Restriction			
No accumulator access, implicit or explicit.			
Syntax			
[(pred)] fbh (exec_size) reg reg [(pred)] fbh (exec_size) reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD cnt = 0; if (src0 is unsigned) { UD udScalar = src0.chan[n]; while ((udScalar & (1 << 31)) == 0 && cnt != 32) { cnt ++; udScalar = udScalar << 1; } if (src0.chan[n] == 0x00000000) { dst.chan[n] = 0xFFFFFFFF; } else { dst.chan[n] = cnt; } } else { // src0 is signed. D dScalar = src0.chan[n]; bit cval = dScalar[31]; while ((dScalar & (1 << 31)) == cval && cnt != 32) { cnt ++; dScalar = dScalar << 1; } if ((src0.chan[n] == 0xFFFFFFFF) (src0.chan[n] == 0x00000000)) { dst.chan[n] = 0xFFFFFFFF; } else { dst.chan[n] = cnt; } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types		
D, UD	UD		
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		



fbh - Find First Bit from MSB Side		
	63:32	Operand Controls Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

Fraction

frc - Fraction			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The frc instruction computes, component-wise, the truncate-to-minus-infinity fractional values of src0 and stores the results in dst. The results, in the range of [0.0, 1.0], are the fractional portion of the source data. The result is in the range [0.0, 1.0] irrespective of the rounding mode. Floating-point fraction computation follows the rules in the following tables, based on the current floating-point mode.</p>			
Format: [(pred)] frc[.cmod] (exec_size) dst src0			
Syntax			
[(pred)] frc[.cmod] (exec_size) reg reg [(pred)] frc[.cmod] (exec_size) reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] - floor(src0.chan[n]); } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	Y
Src Types	Dst Types		
F	F		
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	



Goto

goto - Goto	
Project:	BDW
Source:	Eulsa
Length Bias:	4
<p>The goto instruction directs the instruction pointer to the offset specified by the UIP offset or to the next IP based on the BranchCtrl bit in the instruction. The active channels that are predicated on this instruction will take the IP + UIP path when BranchCtrl is set else the channels take IP + 1. The active channels that are not predicated on this instruction will be made inactive and waiting to be joined at the join IP. The join IP is IP + UIP when BranchCtrl is clear else it is the next IP.</p> <p>When there are no active channels the instruction pointer will move to IP + JIP.</p> <p>The goto instruction is used in conjunction with a join instruction. A goto deactivates some channels that are reactivated at some program-specified join instruction. See the join instruction for the activation rules.</p> <p>The goto and join instructions enable unstructured program control flow. These instructions must be used with additional care where dangling channels can result without proper compiler checks, meaning that it is expected that programs will navigate through these paths to reactivate the channels. Hardware does not provide native checks or reconvergence.</p> <p>The following table describes the two 32-bit instruction pointer offsets. Both the JIP and UIP are signed 32-bit numbers, added to IP pre-increment. In GEN binary, JIP and UIP are at locations src0 and src1 and must be of type DW (signed DWord integer).</p> <p>If SPF is ON, none of the PcIP are updated.</p>	
Format: [(pred)] goto (exec_size) JIP UIP BranchCtrl	
Restriction	
Cannot have a goto in the body and the corresponding join in the function or the subroutine. Similarly the brd and brc.	
Syntax	
[(pred)] goto (exec_size) imm32 imm32 BranchCtrl	
Pseudocode	
<pre> Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { // for the predicated active channels if (BranchCtrl) { PcIP[n] = IP + UIP; } else { PcIP[n] = IP + 1; } } else { // join IP, for the active non predicated channels if (BranchCtrl) { PcIP[n] = IP + 1; } else { PcIP[n] = IP + UIP; } } } if (BranchCtrl) { // if (PcIP != (IP + UIP)) { // for all channels if (PcIP != (IP + 1)) { // for all channels Jump(IP + JIP); } else { Jump(IP + 1); } } else { Jump(IP + UIP); } } else { // if (PcIP != (IP + 1)) { // for all channels Jump(IP + JIP); } else { Jump(IP + 1); } } } </pre>	
Errata	Description
	[DevBDW:A]: A goto instruction must not be followed by any instruction requiring register indirect access on source operands.

goto - Goto			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
	Format:	S31	
	The byte-aligned jump distance if a jump is taken for the channel.		
95:64	UIP		
	Project:	BDW	
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		



GPGPU_CSR_BASE_ADDRESS

GPGPU_CSR_BASE_ADDRESS			
Project:	BDW		
Length Bias:	2		
The GPGPU_CSR_BASE_ADDRESS command sets the base pointers for EU and L3 to Context Save and Restore EU State and SLM for GPGPU mid.			
Programming Notes			
Execution of this command causes a full pipeline flush, thus its use should be minimized for higher performance. State and instruction caches are flushed on completion of the flush.			
SW must always program PIPE_CONTROL with "CS Stall" and "Render Target Cache Flush Enable" set prior to programming GPGPU_CSR_BASE_ADDRESS command for GPGPU workloads i.e when pipeline select is GPGPU via PIPELINE_SELECT command. This is required to achieve better GPGPU preemption latencies for certain programming sequences. If programming PIPE_CONTROL has performance implications then preemption latencies can be trade off against performance by not implementing this programming note.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	Opcode
	28:27	Command SubType	
		Default Value:	0h GFXPIPE_COMMON
	26:24	3D Command Opcode	
		Default Value:	1h GFXPIPE_NONPIPELINED
	23:16	3D Command Sub Opcode	
		Default Value:	04h GPGPU_CSR_BASE_ADDRESS
	15:8	Reserved	
Format:		MBZ	
7:0	DWord Length		
	Format:	=n Total Length -2	
	Value	Name	Description
1h	[Default]	Excludes DWord(0,1)	Project BDW

GPGPU_CSR_BASE_ADDRESS					
1..2 Project: BDW	63:12	GPGPU CSR Base Address			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:12]</td> </tr> </table> <p>Specifies the 256K-byte aligned base address for GPGPU context GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	BDW	Format:
	Project:	BDW			
	Format:	GraphicsAddress[63:12]			
11:0	Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				

GPGPU_WALKER

GPGPU_WALKER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
Programming Notes			
If the threads spawned by this command are required to observe memory writes performed by threads spawned from a previous command, software must precede this command with a command that performs a memory flush (e.g., MI_FLUSH).			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h GPGPU_WALKER
		Format:	OpCode
	23:16	SubOpcode	
Default Value:		05h GPGPU_WALKER SubOp	
Format:		OpCode	
15:11	Reserved		
	Format:	MBZ	
10	Indirect Parameter Enable		
	Format:	Enable	
If set, the values in DW 7, 10, 12 are ignored and replaced by the current values of the corresponding GPGPU_xxx MMIO registers: <ul style="list-style-type: none"> • GPGPU_DISPATCHDIMX (instead of DW7) • GPGPU_DISPATCHDIMY (instead of DW10) • GPGPU_DISPATCHDIMZ (instead of DW12) 			
9	Reserved		
	Format:	MBZ	

GPGPU_WALKER										
	8	<p>Predicate Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>If set, this command is executed (or not) depending on the current value of the MI Predicate internal state bit. This command is ignored only if PredicateEnable is set and the Predicate state bit is 0.</p>	Format:	Enable						
	Format:	Enable								
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 40%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr> <td>0Dh</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	Description	0Dh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
Format:	=n Total Length - 2									
Value	Name	Description								
0Dh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)								
1	<p>31:8 Reserved</p> <p>7:6 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>5:0 Interface Descriptor Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U6</td> </tr> </table> <p>This field specifies the offset from the interface descriptor base pointer to the interface descriptor which will be applied to this object. It is specified in units of interface descriptors.</p>	Format:	MBZ	Format:	U6					
Format:	MBZ									
Format:	U6									
2	31:17	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
16:0	<p>Indirect Data Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U17 in bytes</td> </tr> </table> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. If Indirect Data is enabled, it is assumed that CURBE is not being used except for cross-thread constant data. This field must have the same alignment as the Indirect Object Data Start Address. It must be DQWord (32-byte) aligned. As the indirect data are sent directly to URB, the total size of Indirect data must be less than 63,488 (2048 URB lines - 64 lines for Interface Descriptors).</p> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 70%;">Workaround</th> <th style="width: 30%;">Project</th> </tr> </thead> <tbody> <tr> <td>The indirect payload is limited to 4032 bytes or less.</td> <td>BDW</td> </tr> </tbody> </table>	Format:	U17 in bytes	Workaround	Project	The indirect payload is limited to 4032 bytes or less.	BDW			
Format:	U17 in bytes									
Workaround	Project									
The indirect payload is limited to 4032 bytes or less.	BDW									
3	31:6	<p>Indirect Data Start Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>IndirectObjectOffset[31:6]</td> </tr> </table> <p>This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address. Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation. It is the 64-byte aligned address of the indirect data.</p> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0 - 512MB]</td> <td></td> <td>(Bits 31:29 MBZ)</td> </tr> </tbody> </table>	Format:	IndirectObjectOffset[31:6]	Value	Name	Description	[0 - 512MB]		(Bits 31:29 MBZ)
	Format:	IndirectObjectOffset[31:6]								
Value	Name	Description								
[0 - 512MB]		(Bits 31:29 MBZ)								
5:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									

GPGPU_WALKER														
4	31:30	<p>SIMD Size This field determines the size of the payload and the number of bits of the execution mask that are expected. The kernel pointed to by the interface descriptor should match the SIMD declared here.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>SIMD8</td> <td>8 LSBs of the execution mask are used</td> </tr> <tr> <td>1</td> <td>SIMD16</td> <td>16 LSBs used in execution mask</td> </tr> <tr> <td>2</td> <td>SIMD32</td> <td>32 bits of execution mask used</td> </tr> </tbody> </table>	Value	Name	Description	0	SIMD8	8 LSBs of the execution mask are used	1	SIMD16	16 LSBs used in execution mask	2	SIMD32	32 bits of execution mask used
	Value	Name	Description											
	0	SIMD8	8 LSBs of the execution mask are used											
	1	SIMD16	16 LSBs used in execution mask											
	2	SIMD32	32 bits of execution mask used											
	29:22	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
	21:16	<p>Thread Depth Counter Maximum The maximum value of the thread depth counter. Since the counter starts at 0, the depth is this value + 1. (Thread_Depth_Max+1)*(Thread_Height_Max+1)*(Thread_Width_Max+1) must equal Number of Threads in GPGPU Thread Group in the Interface Descriptor.</p>												
15:14	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
Format:	MBZ													
13:8	<p>Thread Height Counter Maximum</p> <table border="1"> <tr> <td>Format:</td> <td>U6-1</td> </tr> </table> <p>The maximum value of the thread height counter. The height is this value + 1.</p>	Format:	U6-1											
Format:	U6-1													
7:6	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
Format:	MBZ													
5:0	<p>Thread Width Counter Maximum</p> <table border="1"> <tr> <td>Format:</td> <td>U6-1</td> </tr> </table> <p>The maximum value of the thread width counter. The height is this value + 1.</p>	Format:	U6-1											
Format:	U6-1													
5	31:0	<p>Thread Group ID Starting X This is the initial value of the X component of the thread group. When X reaches the maximum value it rolls around to this value.</p>												
6	31:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ													
7	31:0	<p>Thread Group ID X Dimension The X dimension of the thread group (maximum X is dimension -1)</p>												
8	31:0	<p>Thread Group ID Starting Y This is the initial value of the Y component of the thread group. When Y reaches the maximum value it rolls around to this value.</p>												
9	31:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ													

GPGPU_WALKER		
10	31:0	Thread Group ID Y Dimension The Y dimension of the thread group (maximum Y is dimension -1)
11	31:0	Thread Group ID Starting/Resume Z This is the initial value of the Z component of the thread group.
12	31:0	Thread Group ID Z Dimension The Z dimension of the thread group (maximum Z is dimension -1)
13	31:0	Right Execution Mask The execution mask used for threads on the right (maximum X) of the thread group.
14	31:0	Bottom Execution Mask The execution mask used for threads on the bottom (maximum Y) of the thread group.

Halt

halt - Halt							
Project:	BDW						
Source:	Eulsa						
Length Bias:	4						
Description							
<p>The halt instruction temporarily suspends execution for all enabled compute channels. Upon execution, the enabled channels are sent to the instruction at (IP + UIP), if all channels are enabled at HALT, jump to the instruction at (IP + JIP). If the halt instruction is not inside any conditional code block, the values of JIP and UIP should be the same. If the halt instruction is inside a conditional code block, the UIP should be the end of the program and the JIP should be the end of the inner most conditional code block. The UIP must point to a HALT Instruction. If SPF is ON, the UIP must be used to update IP; JIP is not used in this case.</p> <p>The following table describes the two 32-bit instruction pointer offsets. Both the JIP and UIP are signed 32-bit numbers, added to IP pre-increment. In GEN binary, JIP and UIP are at locations src0 and src1 and must be of type DW (signed DWord integer). When the offsets are immediate, src0 regfile must be immediate.</p> <p>Format: [(pred)] halt (exec_size) JIP UIP</p>							
Restriction							
dst and src0 must be NULL.							
Syntax							
[(pred)] halt (exec_size) imm32 imm32							
Pseudocode							
<pre>Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn.channel[n]) { PclP[n] = IP + UIP; else { PclP[n] = IP + 1; } } if (PclP != (IP + 1)) { // for all channels Jump(IP + JIP); }</pre>							
Predication	Conditional Modifier	Saturation	Source Modifier				
Y	N	N	N				
DWord	Bit	Description					
0..3	127:96	JIP <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte-aligned jump distance if a jump is taken for the channel.</p>		Project:	BDW	Format:	S31
	Project:	BDW					
Format:	S31						
95:64	UIP <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S31</td> </tr> </table> <p>The byte aligned jump distance if a jump is taken for the instruction.</p>		Project:	BDW	Format:	S31	
Project:	BDW						
Format:	S31						

halt - Halt		
	63:32	Operand Control Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

HI8DS Render Target Write MSD

MSD_RTW_HI8DS - HI8DS Render Target Write MSD			
Project:	BDW		
Source:	Render Cache DataPort		
Length Bias:	1		
Family:	Other		
Group:	Render Target R/W		
DWord	Bit	Description	
0	31	Reserved	
		Project:	All
		Format:	MBZ
			Ignored
	30	Message Precision Subtype	
		Default Value:	0h
		Project:	All
		Format:	Opcode
			Full precision data message
	29	Reserved	
		Project:	All
		Format:	MBZ
		Ignored	
28:25	Message Length		
	Project:	All	
	Format:	U4	
		Specifies the number of 256-bit GRF registers sent as the message payload (including the header). Valid value ranges are 1 to 15.	
24:20	Response Length		
	Project:	All	
	Format:	U5	
		Specifies the number of 256-bit GRF registers expected as the message response payload. Valid value ranges are 0 to 16.	
19	Header Present		
	Project:	All	
	Format:	MDC_MHP	
		If set, indicates that the message includes the 2-register header.	

MSD_RTW_HI8DS - HI8DS Render Target Write MSD							
18	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
17:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Render Target Write message</p>	Default Value:	0Ch	Project:	All	Format:	Opcode
Default Value:	0Ch						
Project:	All						
Format:	Opcode						
13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
12	<p>Last Render Target Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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. In general, when threads are not launched by 3D FF, this bit must be zero.</p> <div style="border: 1px solid black; background-color: #e6f2ff; padding: 5px; text-align: center; margin: 10px 0;"> <p>Programming Notes</p> </div> <p>When a pixel shader has render target writes at finer granularity than the dispatch rate, last render target write to a null surface must be present at the dispatch rate with this bit set. In particular, if a kernel is dispatched at pixel rate and it only writes to render targets at sample-rate, it must include a pixel-rate render target write to a null surface with Last Render Target Select bit enabled.</p>	Project:	All	Format:	Enable		
Project:	All						
Format:	Enable						
11	<p>Slot Group Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_RT_SGS</td> </tr> </table> <p>This field selects whether slots 15:0 or slots 31:16 are used for bypassed data.</p>	Project:	All	Format:	MDC_RT_SGS		
Project:	All						
Format:	MDC_RT_SGS						
10:8	<p>Render Target Message Subtype</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Default Value:</td> <td>3h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>SIMD8 dual source message. Use slots [15:8] for pixel enables, X/Y addresses, and oMask.</p> <div style="border: 1px solid black; background-color: #e6f2ff; padding: 5px; text-align: center; margin: 10px 0;"> <p>Programming Notes</p> </div> <p>The above slots indicated are within the 16 slots selected by Slot Group Select. If SLOTGRP_HI is selected, slots [31:24] are referenced instead of [15:8].</p>	Default Value:	3h	Project:	All	Format:	Opcode
Default Value:	3h						
Project:	All						
Format:	Opcode						



MSD_RTW_HI8DS - HI8DS Render Target Write MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_BTS
		Specifies the Binding Table Index for the message	

If

if - If			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>An if instruction starts an if/endif or an if/else/endif block of code. It restricts execution within the conditional block to only those channels that were enabled via the predicate control. Each if instruction must have a matching endif instruction and may have up to one matching else instruction before the matching endif. If all channels are inactive (for the if/endif or if/else/endif block), a jump is performed to the instruction referenced by JIP. This jump must be to right after the matching else instruction when present, or otherwise to the matching endif instruction of the conditional block. If SPF is ON, the UIP must be used to update IP; JIP is not used in this case.</p> <p>The following table describes the 32-bit exit code <JIP> and <UIP>. If <branch_ctrl> is set, then the JIP points to the first join instruction within the if block. If <branch_ctrl> is not set, <JIP> should point to the instruction right after the matching else instruction if it exists, otherwise <JIP> should point to the endif instruction. <UIP> should always point to the endif instruction. When a jump occurs, this value is added to IP pre-increment. In GEN instruction binary, <JIP> and <UIP> are at location <src0> & <src1> and must be of type D (signed dword integer).</p> <p>Format: [(pred)] if (exec_size JIP UIP <branch_ctrl></p>			
Restriction			
The execution size must be the same for the if, else, and endif instructions of the same code block.			
Syntax			
[(pred)] if (exec_size) imm32 imm32 <branch_ctrl>			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn.channel[n] == 0) { PclP[n] = IP + JIP; } else { PclP[n] = IP + 1; } } if (PclP != (IP + 1)) { // for all channels Jump(IP + JIP); }			
Errata	Description		
	[DevBDW:A]: An if instruction must not be followed by any instruction requiring register indirect access on source operands.		
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		Format:	S31



if - If	
	The byte-aligned jump distance if a jump is taken for the channel.
95:64	UIP
	Project: BDW
	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

Illegal

illegal - Illegal			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The Illegal Opcode Exception Enable flag in cr0.1 is normally set so the normal processing of an illegal opcode is to transfer control to the System Routine. Instruction dispatch treats any unused 8-bit opcode (including bit 7 of the instruction, reserved for future opcode expansion) as if it is the illegal opcode. The illegal opcode is zero because that byte value is more likely than most to be read via a wayward instruction pointer. The illegal instruction is an instruction only in the same way that a NULL pointer in software is a pointer. Both are special values indicating invalid instances.</p>			
Format: illegal			
Restriction			
The illegal instruction takes no instruction options.			
Syntax			
illegal			
Pseudocode			
{ Set the Illegal Opcode Exception Status bit in cr0.1. if (Illegal Opcode Exception Enable is set in cr0.1) { Transfer control to the System Routine (return address to AIP, IP = SIP). }			
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
DWord	Bit	Description	
0..3	127:7	Reserved	
		Format:	MBZ
	6:0	Opcode	
		Format:	EU_OPCODE



Integer Subtraction with Borrow

subb - Integer Subtraction with Borrow			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The subb instruction performs component-wise subtraction of src0 and src1 and stores the results in dst, it also stores the borrow into acc. If the operation produces a borrow (src0 < src1), write 0x00000001 to acc, else write 0x00000000 to acc.</p>			
<p>Format: [(pred)] subb[.cmode] (exec_size) dst src0 src1</p>			
Restriction			
<p>AccWrEn is required. The accumulator is an implicit destination and thus cannot be an explicit destination operand.</p>			
Syntax			
<p>[(pred)] subb[.cmode] (exec_size) reg reg reg [(pred)] subb[.cmode] (exec_size) reg reg imm32</p>			
Pseudocode			
<p>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] - src1.chan[n]; acc.chan[n] = borrow(src.chan[n] - src1.chan[n]); } }</p>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	Y	N
Src Types	Dst Types		
UD	UD		
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	

Join

join - Join			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The join instruction makes the inactive channels active at the join IP if those channels are predicated. Any deactivated channels due to a goto instruction match the join IP are activated (qualified with predicates at join). If no IP is matched at this join, the program goes to the next IP with the active channels which followed the program path up to the join instruction. If no active channels are present after executing the join instruction, the program jumps to the offset specified by JIP instead of next IP. The join instruction is used in conjunction with a goto instruction. The join activates channels that are deactivated by the goto instruction. See the goto instruction for the deactivation rules. The goto and join instructions enable unstructured program control flow. These instructions must be used with additional care where dangling channels can result without proper compiler checks, meaning that it is expected that programs will navigate through these paths to reactivate the channels. Hardware does not provide native checks or reconvergence. The following table describes the 32-bit JIP. In GEN binary, JIP is at location src1 and must be of type D (signed DWord integer). JIP must be an immediate operand and is a signed 32-bit number. This value is added to IP pre-increment. If SPF is ON, none of the PciP are updated.</p>			
Format: [(pred)] join (exec_size) JIP			
Programming Notes			
An index of 0 does nothing, continuing execution with the next instruction.			
An index of -16 (if the jmpj instruction is in native format) or -8 (if the jmpj instruction is in compact format) is an infinite loop on the jmpj instruction.			
Restriction			
The {NoMask} instruction option must be specified.			
The index data type must be D (Signed DWord Integer).			
Syntax			
[(pred)] join (exec_size) imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { // for the predicated channels and the remaining channels PciP[n] = IP + 1; } } if (PciP != (IP + 1)) { // for all channels when no channels are activated and no other active channels Jump(IP + JIP); }			
Errata	Description		
	[DevBDW:A]: A join instruction must not be followed by any instruction requiring register indirect access on source operands.		
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N

join - Join			
DWord	Bit	Description	
0..3	127:96	JIP	
		Project: BDW	
		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	
		Project: BDW	
	94:91	Src1.SrcType	
		Project: BDW	
		Format: SrcType	
	90:89	Src1.RegFile	
Project: BDW			
Format: RegFile			
88:64	Source 0		
	Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align16')		
88:64	Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16		
	Source 0		
88:64	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		

Jump Indexed

jmp - Jump Indexed			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The jmp instruction redirects program execution to an index offset relative to the post-incremented instruction pointer. The index is a signed integer value, with positive or zero integers for forward jumps, and negative integers for backward jumps. Note: Unlike other flow control instructions, the offset used by jmp is relative to the incremented instruction pointer rather than the IP value for the instruction itself. In GEN binary, index is at location src1. The ip register must be put (for example, by the assembler) at the dst and src0 locations. Predication is allowed to provide conditional jump with a scalar condition. As the execution size is 1, the first channel of PMASK (flags post prediction control and negate) is used to determine whether the jump is taken or not. If the condition is false, the jump is not taken and execution continues with the next instruction.</p>			
Format: [(pred)] jmp (1) index {NoMask}			
Programming Notes			
An index of 0 does nothing, continuing execution with the next instruction.			
An index of -16 (if the jmp instruction is in native format) or -8 (if the jmp instruction is in compact format) is an infinite loop on the jmp instruction.			
Restriction			
The execution size must be 1.			
The {NoMask} instruction option must be specified.			
The index data type must be D (Signed DWord Integer).			
QtrCtrl must not be used for jmp instruction.			
Syntax			
[(pred)] jmp (1) reg32 {NoMask} [(pred)] jmp (1) imm32 {NoMask}			
Pseudocode			
Evaluate(WrEn); if (WrEn != 0) { Jump(IP + 1 + index); // IP + 1 is a pseudocode idiom for the IP of the following instruction. }			
Errata	Description		
	[DevBDW:A]: A jmp instruction must not be followed by any instruction requiring register indirect access on source operands.		
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N

jmpb - Jump Indexed			
Src Types			
D			
DWord	Bit	Description	
0..3	127:96	JIP	
		Project: BDW	
		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	
		Project: BDW	
	94:91	Src1.SrcType	
		Project: BDW	
	90:89	Src1.RegFile	
		Project: BDW	
88:64	Source 0		
	Exists If: (Structure[EI_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align16')		
88:64	Source 0		
	Exists If: (Structure[EI_INSTRUCTION_CONTROLS_A][AccessMode] == 'Align1')		
63:32	Operand Control		
	Format: EI_INSTRUCTION_OPERAND_CONTROLS		
31:0	Header		
	Format: EI_INSTRUCTION_HEADER		

Leading Zero Detection

lzd - Leading Zero Detection			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The lzd instruction counts component-wise the leading zeros from src0 and stores the resulting counts in dst. If src0 is zero, store 32 in dst.</p>			
<p>Format: [(pred)] lzd[.cmod] (exec_size) dst src0</p>			
Restriction			
Accumulator cannot be destination, implicit or explicit.			
Syntax			
[(pred)] lzd[.cmod] (exec_size) reg reg [(pred)] lzd[.cmod] (exec_size) reg reg			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { UD udScalar = src0.chan[n]; UD cnt = 0; while ((udScalar & (1 << 31)) == 0 && cnt != 32) { cnt ++; udScalar = udScalar << 1; } dst.chan[n] = cnt; } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
D, UD	UD		
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	



Line

line - Line			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The line instruction computes a component-wise line equation ($v = p * u + q$ where u, v are vectors and p, q are scalars) of <code>src0</code> and <code>src1</code> and stores the results in <code>dst</code>. <code>src1</code> is the input vector u. <code>src0</code> provides input scalars p and q, where p is the scalar value based on the region description of <code>src0</code> and q is the scalar value implied from <code>src0</code> region. Specifically, q is the fourth component of the 4-tuple (128-bit aligned) that p belongs to.</p>			
Format: <code>[(pred)] line[.cmod] (exec_size) dst src0 src1</code>			
Restriction			
This is a specialized instruction that only supports an execution size (ExecSize) of 8 or 16.			
The <code>src0</code> region must be a replicated scalar (with <code>HorzStride == VertStride == 0</code>).			
<code>src0</code> must specify <code>.0</code> or <code>.4</code> as the subregister number, corresponding to a subregister byte offset of 0 or 16.			
Source operands cannot be accumulators.			
Syntax			
<code>[(pred)] line[.cmod] (exec_size) reg reg reg [(pred)] line[.cmod] (exec_size) reg reg imm32</code>			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { dwP = src0.RegNum.SubRegNum[bits4:2]; // A DWord-aligned scalar. dwQ = src0.RegNum.(SubRegNum[bit4] 0x8); // Fourth component. if (WrEn.chan[n]) { dst.chan[n] = dwP * src1.chan[n] + dwQ; } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	<code>((RegSource)[Src1.RegFile] != 'IMM')</code>
		Format:	EU_INSTRUCTION_SOURCES_REG_REG
	127:64	ImmSource	
		Exists If:	<code>((ImmSource)[Src1.RegFile] == 'IMM')</code>
		Format:	EU_INSTRUCTION_SOURCES_REG_IMM
63:32	Operand Controls		
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	

line - Line			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER



Linear Interpolation

Irp - Linear Interpolation			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
The Irp instruction takes component-wise multiplication of src0 and src1, and adds the result to the component-wise multiplication of src2 and (1 - src0), and then stores the final results in dst.			
Format: [(pred)] Irp[.cmod] (exec_size) dst src0 src1 src2			
Restriction			
The vertical stride (VertStride) is overloaded to 4 in HW for 3-source instructions.			
The overflow conditional modifier (.o) is not allowed.			
No explicit accumulator access because this is a three-source instruction. AccWrEn is allowed for implicitly updating the accumulator.			
All three-source instructions have certain restrictions, described in Instruction Formats [BDW].			
Syntax			
[(pred)] Irp[.cmod] (exec_size) reg reg reg reg			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src1.chan[n] * src0.chan[n] + src2.chan[n] * (1.0 - src0.chan[n]); } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:126	Reserved Format: MBZ	
	125:106	Source 2 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	105	Reserved Format: MBZ	
	104:85	Source 1 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	84	Reserved Format: MBZ	

Irp - Linear Interpolation															
83:64	<p>Source 0</p> <table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC												
Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC														
63:56	<p>Destination Register Number</p> <table border="1"> <tr> <td>Format:</td> <td>DstRegNum</td> </tr> </table>	Format:	DstRegNum												
Format:	DstRegNum														
55:53	<p>Destination Subregister Number</p> <table border="1"> <tr> <td>Format:</td> <td>DstSubRegNum[2:0]</td> </tr> </table>	Format:	DstSubRegNum[2:0]												
Format:	DstSubRegNum[2:0]														
52:49	<p>Destination Channel Enable</p> <table border="1"> <tr> <td>Format:</td> <td>ChanEn[4]</td> </tr> </table> <p>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 <i>x</i>, <i>y</i>, <i>z</i>, and <i>w</i>, respectively, where <i>x</i> corresponds to Channel 0 in the group and <i>w</i> corresponds to channel 3 in the group</p>	Format:	ChanEn[4]												
Format:	ChanEn[4]														
48	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td></td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:		Format:	MBZ										
Project:															
Format:	MBZ														
47	<p>NibCtrl</p> <table border="1"> <tr> <td>Project:</td> <td></td> </tr> <tr> <td>Format:</td> <td>NibCtrl</td> </tr> </table>	Project:		Format:	NibCtrl										
Project:															
Format:	NibCtrl														
46	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td></td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:		Format:	MBZ										
Project:															
Format:	MBZ														
45:44	<p>Destination Data Type</p> <table border="1"> <tr> <td>Project:</td> <td></td> </tr> <tr> <td colspan="2">This field contains the data type for the destination</td> </tr> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> <tr> <td>00b</td> <td>Single Precision Float</td> </tr> <tr> <td>01b</td> <td>DWord</td> </tr> <tr> <td>10b</td> <td>Unsigned DWord</td> </tr> <tr> <td>11b</td> <td>Double Precision Float</td> </tr> </table>	Project:		This field contains the data type for the destination		Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
Project:															
This field contains the data type for the destination															
Value	Name														
00b	Single Precision Float														
01b	DWord														
10b	Unsigned DWord														
11b	Double Precision Float														

Irp - Linear Interpolation											
43:42	Source Data Type										
	Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table>										
	This field contains the data type for all three sources										
	<table border="1" style="width: 100%;"><thead><tr><th style="text-align: center;">Value</th><th style="text-align: center;">Name</th></tr></thead><tbody><tr><td>00b</td><td>Single Precision Float</td></tr><tr><td>01b</td><td>DWord</td></tr><tr><td>10b</td><td>Unsigned DWord</td></tr><tr><td>11b</td><td>Double Precision Float</td></tr></tbody></table>	Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
	Value	Name									
00b	Single Precision Float										
01b	DWord										
10b	Unsigned DWord										
11b	Double Precision Float										
00b	Single Precision Float										
01b	DWord										
10b	Unsigned DWord										
11b	Double Precision Float										
41:40	Source 2 Modifier										
	Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table>	/// ([Property[Source Modifier]='true')									
	/// ([Property[Source Modifier]='true')										
Format: SrcMod											
39:38	Source 1 Modifier										
	Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table>	/// ([Property[Source Modifier]='true')									
	/// ([Property[Source Modifier]='true')										
Format: SrcMod											
37:36	Source 0 Modifier										
	Exists If: <table border="1" style="width: 100%;"><tr><td>/// ([Property[Source Modifier]='true')</td></tr></table>	/// ([Property[Source Modifier]='true')									
	/// ([Property[Source Modifier]='true')										
Format: SrcMod											
35	Reserved										
	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>		MBZ								
	MBZ										
34	Flag Register Number										
	Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table> This field contains the flag register number for instructions with a non-zero Conditional Modifier.										
33	Flag Subregister Number										
	This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.										
32	Reserved										
	Project: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td></td></tr></table>										
Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td>MBZ</td></tr></table>		MBZ									
	MBZ										
31:0	Header										
	Format: EU_INSTRUCTION_HEADER										

LO8DS Render Target Write MSD

MSD_RTW_LO8DS - LO8DS Render Target Write MSD		
Project:	BDW	
Source:	Render Cache DataPort	
Length Bias:	1	
Family:	Other	
Group:	Render Target R/W	
DWord	Bit	Description
0	31	Reserved
		Project: All
		Format: MBZ
	Ignored	
	30	Message Precision Subtype
Default Value: 0h		
Project: All		
Format: Opcode		
Full precision data message		
29	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
28:25	Message Length	
	Project: All	
	Format: U4	
Specifies the number of 256-bit GRF registers sent as the message payload (including the header). Valid value ranges are 1 to 15.		
24:20	Response Length	
	Project: All	
	Format: U5	
Specifies the number of 256-bit GRF registers expected as the message response payload. Valid value ranges are 0 to 16.		
19	Header Present	
	Project: All	
	Format: MDC_MHP	
If set, indicates that the message includes the 2-register header.		

MSD_RTW_LO8DS - LO8DS Render Target Write MSD							
18	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
17:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Render Target Write message</p>	Default Value:	0Ch	Project:	All	Format:	Opcode
Default Value:	0Ch						
Project:	All						
Format:	Opcode						
13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
12	<p>Last Render Target Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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. In general, when threads are not launched by 3D FF, this bit must be zero.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>When a pixel shader has render target writes at finer granularity than the dispatch rate, last render target write to a null surface must be present at the dispatch rate with this bit set. In particular, if a kernel is dispatched at pixel rate and it only writes to render targets at sample-rate, it must include a pixel-rate render target write to a null surface with Last Render Target Select bit enabled.</p>	Project:	All	Format:	Enable	Programming Notes	
Project:	All						
Format:	Enable						
Programming Notes							
11	<p>Slot Group Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_RT_SGS</td> </tr> </table> <p>This field selects whether slots 15:0 or slots 31:16 are used for bypassed data.</p>	Project:	All	Format:	MDC_RT_SGS		
Project:	All						
Format:	MDC_RT_SGS						

MSD_RTW_LO8DS - LO8DS Render Target Write MSD	
10:8	Render Target Message Subtype
	Default Value: 2h
	Project: All
	Format: Opcode
	SIMD8 dual source message. Use slots [7:0] for pixel enables, X/Y addresses, and oMask.
Programming Notes	
The above slots indicated are within the 16 slots selected by Slot Group Select. If SLOTGRP_HI is selected, slots [23:16] are referenced instead of [7:0].	
7:0	Binding Table Index
	Project: All
	Format: MDC_BTS
Specifies the Binding Table Index for the message	

Logic And

and - Logic And			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The and instruction performs component-wise logic AND operation between src0 and src1 and stores the results in dst. Register source operands can use source modifiers: [Pre-DevBDW]: Any source modifier is numeric, optionally changing a source value s to -s, abs(s), or -abs(s) before the AND operation. [BDW]: Any source modifier is logical, optionally changing a source value s to ~s (inverting all source bits). This capability allows expressions like a AND (NOT b) to be calculated with one instruction. This operation does not produce sign or overflow conditions. Only the .e/.z or .ne/.nz conditional modifiers should be used.</p>			
Format: Source modifier is not allowed if source is an accumulator.			
Restriction			
Source modifier is not allowed if source is an accumulator.			
Syntax			
[(pred)] and[.cmod] (exec_size) reg reg reg [(pred)] and[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] & src1.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
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	

Logic Not

not - Logic Not			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
<p>The not instruction performs logical NOT operation (or one's complement) of src0 and storing the results in dst. This operation does not produce sign or overflow conditions. Only the .e/.z or .ne/.nz conditional modifiers should be used.</p>			
<p>A register source operand can use a source modifier: Any source modifier is logical, optionally changing a source value s to ~s (inverting all source bits). Such a source modifier is not particularly useful with the not instruction, as it changes the effect of not to just copying bits.</p>			BDW
Format: [(pred)] not[.cmod] (exec_size) dst src0			
Restriction			
Source modifier is not allowed if source is an accumulator.			
Syntax			
[(pred)] not[.cmod] (exec_size) reg reg [(pred)] not[.cmod] (exec_size) reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = ~ src0.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
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	



Logic Or

or - Logic Or			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The or instruction performs component-wise logic OR operation between src0 and src1 and stores the results in dst. This operation does not produce sign or overflow conditions. Only the .e/z or .ne/.nz conditional modifiers should be used.</p> <p>Register source operands can use source modifiers: Any source modifier is logical, optionally changing a source value s to ~s (inverting all source bits). This capability allows expressions like a OR (NOT b) to be calculated with one instruction.</p> <p>Format: <code>[(pred)] or[.cmod] (exec_size) dst src0 src1</code></p>			
Restriction			
Source modifier is not allowed if source is an accumulator.			
Syntax			
<code>[(pred)] or[.cmod] (exec_size) reg reg reg [(pred)] or[.cmod] (exec_size) reg reg imm32</code>			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] src1.chan[n]; } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	<code>((RegSource)[Src1.RegFile] != 'IMM')</code>
	Format:	EU_INSTRUCTION_SOURCES_REG_REG	
	127:64	ImmSource	
		Exists If:	<code>((ImmSource)[Src1.RegFile] == 'IMM')</code>
	Format:	EU_INSTRUCTION_SOURCES_REG_IMM	
63:32	Operand Controls		
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	

or - Logic Or			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER



Logic Xor

xor - Logic Xor			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The xor instruction performs component-wise logic XOR operation between src0 and src1 and stores the results in dst. This operation does not produce sign or overflow conditions. Only the .e/z or .ne/.nz conditional modifiers should be used.</p> <p>Register source operands can use source modifiers: Any source modifier is logical, optionally changing a source value s to ~s (inverting all source bits). This capability allows expressions like a XOR (NOT b) to be calculated with one instruction.</p> <p>Format: [(pred)] xor[.cmod] (exec_size) dst src0 src1</p>			
Restriction			
Source modifier is not allowed if source is an accumulator.			
Syntax			
[(pred)] xor[.cmod] (exec_size) reg reg reg [(pred)] xor[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] ^ src1.chan[n]; } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	N	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
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	

xor - Logic Xor			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER

MEDIA_CURBE_LOAD

MEDIA_CURBE_LOAD				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
Workaround : See "GPGPU Command Workarounds" section for additional programming constraints for this command.				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h GFXPIPE	
		Format:	OpCode	
	28:27	Pipeline		
		Default Value:	2h Media	
		Format:	OpCode	
	26:24	Media Command Opcode		
		Default Value:	0h MEDIA_CURBE_LOAD	
		Format:	OpCode	
	23:16	SubOpcode		
		Default Value:	1h MEDIA_CURBE_LOAD SubOp	
		Format:	OpCode	
	15:0	DWord Length	Project:	All
			Format:	=n Total Length - 2
Value		Name	Description	
2h		DWORD_COUNT_n [Default]	Excludes DWord (0,1)	
1	31:0	Reserved		
		Project:	All	
		Format:	MBZ	
2	31:17	Reserved		
		Project:	All	
		Format:	MBZ	

MEDIA_CURBE_LOAD		
	16:0	CURBE Total Data Length
		Project: All
		Format: U17 In Bytes
		Description
		This field provides the length in bytes of the CURBE data. This field must have the same alignment as the Curbe Object Data Start Address.As the CURBE data are sent directly to ROB, range is limited to CURBE Allocation Size. This field must be 64-byte aligned.
3	31:0	CURBE Data Start Address
		Project: All
		Format: DynamicStateOffset[31:0] CURBE
		Description
		Specifies the 64-byte aligned address of the CURBE data. This pointer is relative to the Dynamics Base Address .
		Value
		[0, FFFFFFFFh]
		Name



MEDIA_INTERFACE_DESCRIPTOR_LOAD

MEDIA_INTERFACE_DESCRIPTOR_LOAD								
Project:	BDW							
Source:	RenderCS							
Length Bias:	2							
A Media_State_Flush should be used before this command to ensure that the temporary Interface Descriptor storage is cleared.								
Workaround : See "GPGPU Command Workarounds" section for additional programming constraints for this command.								
DWord	Bit	Description						
0	31:29	Command Type						
		Default Value:	3h GFXPIPE					
		Format:	OpCode					
	28:27	Pipeline						
		Default Value:	2h Media					
		Format:	OpCode					
	26:24	Media Command Opcode						
		Default Value:	0h MEDIA_INTERFACE_DESCRIPTOR_LOAD					
		Format:	OpCode					
	23:16	SubOpcode						
Default Value:		2h MEDIA_INTERFACE_DESCRIPTOR_LOAD SubOp						
Format:		OpCode						
15:0	DWord Length	Format:	=n Total Length - 2					
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>2h</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>		Value	Name	Description	2h	DWORD_COUNT_n [Default]
	Value	Name	Description					
	2h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)					
1	31:0	Reserved						
		Format:	MBZ					
2	31:17	Reserved						
		Format:	MBZ					

MEDIA_INTERFACE_DESCRIPTOR_LOAD						
	16:0	Interface Descriptor Total Length				
		Format: U17 In bytes				
		This field provides the length in bytes of the Interface Descriptor data. This field must have the same alignment as the Interface Descriptor Data Start Address. It must be DQWord (32-byte) aligned. As the Interface Descriptor data are sent directly to ROB, range is limited to CURBE Allocation Size.				
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%; text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[32,2048]</td> <td style="text-align: center;">[1,64] interface descriptor entries</td> </tr> </tbody> </table>	Value	Name	[32,2048]	[1,64] interface descriptor entries
Value	Name					
[32,2048]	[1,64] interface descriptor entries					
3	31:0	Interface Descriptor Data Start Address				
		Format: DynamicStateOffset[31:0]INTERFACE_DESCRIPTOR_DATA				
		Description				
		This bit specifies the <u>64-byte</u> aligned address of the Interface Descriptor data. This pointer is relative to the Dynamics Base Address.				
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%; text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0, FFFFFFFFh]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0, FFFFFFFFh]	
Value	Name					
[0, FFFFFFFFh]						

MEDIA_OBJECT

MEDIA_OBJECT				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h GFXPIPE	
		Format:	OpCode	
	28:27	Media Command Pipeline	Default Value:	2h Media
			Format:	OpCode
	26:24	Media Command Opcode	Default Value:	1h MEDIA_OBJECT
			Format:	OpCode
	23:16	Media Command Sub-Opcode	Default Value:	0h MEDIA_OBJECT SubOp
Format:			OpCode	
15:0	DWord Length	Default Value:	4h DWORD_COUNT_n	
		Project:	BDW	
		Format:	=n Total Length - 2	
		Excludes DWords 0,1 Generic Mode: DWord Length = N+4, where N is in the range of [0,504]. The maximum is 504 DW (equivalent to 63 8-DW registers). When both inline and indirect data are fetched for this command, the total size in 8-DW registers must be less than 112 (with both inline data length N and indirect data length rounded up to 8-DW aligned individually). The minimal inline data length is 0.		
1	31:8	Reserved		
	7:6	Reserved		
		Format:	MBZ	
	5:0	Interface Descriptor Offset	Project:	BDW
			Format:	U6
			This field specifies the offset from the interface descriptor base pointer to the interface descriptor which will be applied to this object. It is specified in units of interface descriptors.	

MEDIA_OBJECT									
2	31	<p>Children Present</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">Enable</td> </tr> </table> <p>Indicates that the root thread may send spawn messages to spawn child threads and/or synchronized root threads. If Children Present is not set, TS signals VFE to dereference the URB handle immediately after it receives acknowledgement from TD that the thread is dispatched. If Children Present is set, the URB handle is forwarded to the root thread and serves as the return URB handle for the root thread. TS does not signal dereference at the time of dispatch. TS signals URB handle dereference only when it receives a resource dereference message from the thread. <i>In order avoid deadlock, such dereference must be issued once and only once for each URB handle.</i></p>	Format:	Enable					
	Format:	Enable							
	30:27	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
	26:25	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">BDW</td> </tr> </table>	Project:	BDW					
	Project:	BDW							
	24	<p>Thread Synchronization</p> <p>This field when set indicates that the dispatch of the thread originated from this command is based on the "spawn root thread" message.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>No thread synchronization</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Thread dispatch is synchronized by the 'spawn root thread' message</td> </tr> </tbody> </table>	Value	Name	0	No thread synchronization	1	Thread dispatch is synchronized by the 'spawn root thread' message	
	Value	Name							
0	No thread synchronization								
1	Thread dispatch is synchronized by the 'spawn root thread' message								
23	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								
22	<p>Force Destination</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">BDW</td> </tr> </table> <p>If set, bits 20:17 are used to determine the destination of this dispatch, if clear the destination will be chosen based on load.</p>	Project:	BDW						
Project:	BDW								
21	<p>Use Scoreboard</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">BDW</td> </tr> </table> <p>This field specifies whether the thread associated with this command uses hardware scoreboard. Only when this field is set, the scoreboard control fields in the VFE Dword are valid. If this field is cleared, the thread associated with this command bypasses hardware scoreboard.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Not using scoreboard</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Using scoreboard</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0	Not using scoreboard	1	Using scoreboard
Project:	BDW								
Value	Name								
0	Not using scoreboard								
1	Using scoreboard								

MEDIA_OBJECT			
20:19	Slice Destination Select		
	Project:	BDW	
	This bit along with the subslice destination select determines the slice that this thread must be sent to. Ignored if Force Destination = 0, or if product only has 1 slice.		
	Value	Name	Description
	00b	Slice 0	
	01b	Slice 1	Cannot be used in products without a Slice 1.
	10b	Slice 2	Cannot be used in products without a Slice 2.
	11b	Reserved	
	18:17	SubSlice Destination Select	
		Project:	BDW
		This field selects the SubSlice that this thread must be sent to. Ignored if Force Destination = 0	
		Value	Name
11b		Reserved	BDW
10b		SubSlice 2	
16:0	Indirect Data Length		
	Format:	U17 In bytes	
	This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address. It must be DQWord (32-byte) aligned. As the indirect data are sent directly to URB, range is limited to 496 DW. When both inline and indirect data are fetched for this command, the total size in 8-DW registers must be less than 112 (with both inline data length and indirect data length rounded up to 8-DW aligned).		

MEDIA_OBJECT			
3	31:0	Indirect Data Start Address Format: GraphicsAddress[31:0]	
		Description	
		This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address . Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation. This field specifies the 64-byte aligned address of the indirect data.	
		Value	
		Name	
		[0,512MB]	
Programming Notes			
Bits 31:29 MBZ			
4	31:25	Reserved Format: MBZ	
		Scoreboard Y Project: BDW Format: U9 This field provides the Y term of the scoreboard value of the current thread.	
	15:9	Reserved Format: MBZ	
		Scoreboard X Project: BDW Format: U9 This field provides the X term of the scoreboard value of the current thread.	
	5	31:20	Reserved Project: BDW Format: MBZ
			Scoreboard Color Project: BDW Format: U4 This field specifies which dependency color the current thread belongs to. It affects the dependency scoreboard control.
		19:16	Scoreboard Color Project: BDW Format: U4 This field specifies which dependency color the current thread belongs to. It affects the dependency scoreboard control.
			Scoreboard Color Project: BDW Format: U4 This field specifies which dependency color the current thread belongs to. It affects the dependency scoreboard control.



MEDIA_OBJECT		
	15:8	Reserved
		Format: MBZ
	7:0	Scoreboard Mask
		Project: BDW Format: Boolean
<p>Each bit indicates the corresponding dependency scoreboard is dependent on. This field is AND'd with the corresponding Scoreboard Mask field in the MEDIA_VFE_STATE command. Bit n (for n = 0...7): Scoreboard n is dependent, where bit 0 maps to n = 0.</p>		
6..n	31:0	Inline Data Generic Mode: The format of this data is specified by software. Hardware does not interpret this data; it merely passes it to the kernel for processing. The total size for the inline data and indirect data must not exceed 112 registers.

MEDIA_OBJECT_GRPID

MEDIA_OBJECT_GRPID				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
<p>The MEDIA_OBJECT_GRPID command is a variation of MEDIA_OBJECT which includes a group id which is used to allocate and track Barriers and Shared Local Memory. The Interface Descriptor is used to specify how much SLM is needed and how many threads will be reporting to the Barrier. All MEDIA_OBJECT_GRPIDs with the same group id should have the same interface descriptor and be dispatched to the same Tslice – the dispatcher will ensure this if Force Destination = 0, but software must ensure this if Force Destination = 1. Software should also ensure that all the threads needed for the Barrier will fit into a Tslice, or the Barrier will never be satisfied. Either SLM or a barrier must be used with MEDIA_OBJECT_GRPID, if neither is needed then a MEDIA_OBJECT must be used instead.</p> <p>MEDIA_OBJECT_GRPID supports the GPGPU version of payload delivery – either indirect or CURBE can be split between the threads in a group (per-thread payload), as well as a section which is sent to all threads (cross-thread payload). See the GPGPU payload section. For indirect, the same pointer must be sent with all the commands associated with the thread group for payload splitting to work properly. Inline data is not split, but the payload attached to each command is sent with that thread. Only one of inline, indirect, or CURBE is allowed, but at least one form of payload must be sent.</p> <p>MEDIA_STATE_FLUSH with the watermark bit must be placed between groups created by MEDIA_OBJECT_GRPID. The Interface Descriptor associated with the watermark must match the Interface Descriptor used for the following group.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h GFXPIPE	
		Format:	OpCode	
	28:27	28:27	Media Command Pipeline	
			Default Value:	2h Media
			Format:	OpCode
	26:24	26:24	Media Command Opcode	
			Default Value:	1h MEDIA_OBJECT_GRPID
			Format:	OpCode
	23:16	23:16	Media Command Sub-Opcode	
			Default Value:	6h MEDIA_OBJECT_GRPID SubOp
			Format:	OpCode

MEDIA_OBJECT_GRPID									
	15:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>5h DWORD_COUNT_n</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table> <p>Excludes DWords 0,1 Generic Mode: DWord Length = N+5, where N is in the range of [0,504]. The maximum is 504 DW (equivalent to 63 8-DW registers). When both inline and indirect data are fetched for this command, the total size in 8-DW registers must be less than 112 (with both inline data length N and indirect data length rounded up to 8-DW aligned individually). The minimal inline data length is 0.</p>	Default Value:	5h DWORD_COUNT_n	Format:	=n Total Length - 2			
	Default Value:	5h DWORD_COUNT_n							
Format:	=n Total Length - 2								
1	31:8	Reserved							
	7:6	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
5:0	<p>Interface Descriptor Offset</p> <table border="1"> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>This field specifies the offset from the interface descriptor base pointer to the interface descriptor which will be applied to this object. It is specified in units of interface descriptors.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,30]</td> <td></td> </tr> </tbody> </table>	Format:	U6	Value	Name	[0,30]			
Format:	U6								
Value	Name								
[0,30]									
2	31:25	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
	24	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW					
	Project:	BDW							
	23	<p>End of Thread Group</p> <p>This bit indicates that this dispatch is the last for the current thread group.</p>							
22	<p>Force Destination</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>If set, bits 20:17 are used to determine the destination of this dispatch, if clear the destination will be chosen based on load.</p>	Project:	BDW						
Project:	BDW								
21	<p>Use Scoreboard</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field specifies whether the thread associated with this command uses hardware scoreboard. Only when this field is set, the scoreboard control fields in the VFE Dword are valid. If this field is cleared, the thread associated with this command bypasses hardware scoreboard.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Not using scoreboard</td> </tr> <tr> <td>1</td> <td>Using scoreboard</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0	Not using scoreboard	1	Using scoreboard
Project:	BDW								
Value	Name								
0	Not using scoreboard								
1	Using scoreboard								

		MEDIA_OBJECT_GRPID															
	20:19	Slice Destination Select															
		Project:	BDW														
		This bit along with the SubSlice destination select determines the slice that this thread must be sent to. Ignored if Force Destination = 0, or if product only has 1 slice.															
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Slice 0</td> <td></td> </tr> <tr> <td>01b</td> <td>Slice 1</td> <td>Cannot be used in products without a Slice 1.</td> </tr> <tr> <td>10b</td> <td>Slice 2</td> <td>Cannot be used in products without a Slice 2.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	00b	Slice 0		01b	Slice 1	Cannot be used in products without a Slice 1.	10b	Slice 2	Cannot be used in products without a Slice 2.	11b	Reserved	
Value	Name	Description															
00b	Slice 0																
01b	Slice 1	Cannot be used in products without a Slice 1.															
10b	Slice 2	Cannot be used in products without a Slice 2.															
11b	Reserved																
	18:17	SubSlice Destination Select															
		Project:	BDW														
		This field selects the SubSlice that this thread must be sent to. Ignored if Force Destination = 0															
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>11b</td> <td>Reserved</td> <td>BDW</td> </tr> <tr> <td>10b</td> <td>SubSlice 2</td> <td></td> </tr> <tr> <td>01b</td> <td>SubSlice 1</td> <td></td> </tr> <tr> <td>00b</td> <td>SubSlice 0</td> <td></td> </tr> </tbody> </table>	Value	Name	Project	11b	Reserved	BDW	10b	SubSlice 2		01b	SubSlice 1		00b	SubSlice 0	
	Value	Name	Project														
11b	Reserved	BDW															
10b	SubSlice 2																
01b	SubSlice 1																
00b	SubSlice 0																
	16:0	Indirect Data Length															
		Format:	U17 In bytes														
		This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address. It must be DQWord (32-byte) aligned. As the indirect data are sent directly to URB, range is limited to 496 DW. When both inline and indirect data are fetched for this command, the total size in 8-DW registers must be less than 112 (with both inline data length and indirect data length rounded up to 8-DW aligned).															
3	31:0	Indirect Data Start Address															
		Format:	GraphicsAddress[31:0]														
			<table border="1"> <thead> <tr> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address. Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation.</td> <td></td> </tr> <tr> <td>It is the 64-byte aligned address of the indirect data.</td> <td>BDW</td> </tr> </tbody> </table>	Description	Project	This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address . Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation.		It is the 64-byte aligned address of the indirect data.	BDW								
		Description	Project														
		This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address . Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation.															
It is the 64-byte aligned address of the indirect data.	BDW																
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0-512MB]</td> <td></td> <td>Bits 31:29 MBZ</td> </tr> </tbody> </table>	Value	Name	Description	[0-512MB]		Bits 31:29 MBZ										
Value	Name	Description															
[0-512MB]		Bits 31:29 MBZ															
4	31:25	Reserved															
		Format:	MBZ														

MEDIA_OBJECT_GRPID						
	24:16	<p>Scoreboard Y</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> <p>This field provides the Y term of the scoreboard value of the current thread.</p>	Project:	BDW	Format:	U9
	Project:	BDW				
	Format:	U9				
15:9	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
	8:0	<p>Scoreboard X</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> <p>This field provides the X term of the scoreboard value of the current thread.</p>	Project:	BDW	Format:	U9
	Project:	BDW				
	Format:	U9				
5	31:20	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
	Project:	BDW				
	Format:	MBZ				
	19:16	<p>Scoreboard Color</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>This field specifies which dependency color the current thread belongs to. It affects the dependency scoreboard control.</p>	Project:	BDW	Format:	U4
	Project:	BDW				
Format:	U4					
15:8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
7:0	<p>Scoreboard Mask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Boolean</td> </tr> </table> <p>Each bit indicates the corresponding dependency scoreboard is dependent on. This field is AND'd with the corresponding Scoreboard Mask field in the MEDIA_VFE_STATE command. Bit n (for n = 0...7): Scoreboard n is dependent, where bit 0 maps to n = 0.</p>	Project:	BDW	Format:	Boolean	
Project:	BDW					
Format:	Boolean					
6	31:0	<p>GroupID</p> <p>A unique identifying number which describes the threads which share a barrier and/or SLM. Reuse of numbers is allowed as long as the old group is not currently running.</p>				
7..n	31:0	<p>Inline Data</p> <p>The format of this data is specified by software. Hardware does not interpret this data; it merely passes it to the kernel for processing. The total size for the inline data and indirect data must not exceed 112 registers.</p>				

MEDIA_OBJECT_PRT

MEDIA_OBJECT_PRT						
Project:	BDW					
Source:	RenderCS					
Length Bias:	2					
<p>command is for generating Persistent Root Thread for the media pipeline. It only supports loading of inline data but not indirect data. This command should be used for a root thread that might have to be present in the system for a period longer than the certain minimal context-switch interrupt latency. It has to honor the context interrupt signal to terminate upon request. It should also handle replay from the interrupted point upon context restore (the same thread being dispatched more than once). In contrary, if a thread is not a Persistent Root Thread, if dispatched, it must run to completion. The command can be used in all VFE modes, except VLD mode.</p> <p>For simplification, _PRT command has a fixed size of 16 DWORD</p>						
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value:	3h GFXPIPE			
		Format:	OpCode			
	28:27	Pipeline				
		Default Value:	2h Media			
		Format:	OpCode			
	26:24	Media Command Opcode				
		Default Value:	1h MEDIA_OBJECT_PRT			
		Format:	OpCode			
	23:16	SubOpcode				
Default Value:		2h MEDIA_OBJECT_PRT SubOp				
Format:		OpCode				
15:0	DWord Length					
	Project:	BDW				
	Format:	=n Total Length - 2				
	Note: Regardless of the mode, inline data must be present in this command. The command size must fit within 16 dwords.					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0Eh</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	0Eh	DWORD_COUNT_n [Default]
Value	Name	Description				
0Eh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)				
1	31:6	Reserved				
		Format:	MBZ			

MEDIA_OBJECT_PRT										
	5:0	Interface Descriptor Offset								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>This field specifies the offset from the interface descriptor base pointer to the interface descriptor which will be applied to this object. It is specified in units of interface descriptors.</p>	Project:	BDW	Format:	U6				
Project:	BDW									
Format:	U6									
2	31	Children Present								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>Indicates that the root thread may send spawn messages to spawn child threads and/or synchronized root threads. If Children Present is not set, TS signals VFE to dereference the URB handle immediately after it receives acknowledgement from TD that the thread is dispatched. If Children Present is set, the URB handle is forwarded to the root thread and serves as the return URB handle for the root thread. TS does not signal deference at the time of dispatch. TS signals URB handle deference only when it receives a resource dereference message from the thread. In order avoid deadlock, such de-reference must be issued once and only once for each URB handle.</p>	Format:	Enable						
	Format:	Enable								
	30:24	Reserved								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									
23	PRT_Fence Needed									
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>This field specifies that a PRT_Fence is generated after dispatching the thread associated with this MEDIA_OBJECT_PRT. The PRT_Fence prevents additional threads following this persistent root thread until a thread spawn message is sent. The PRT_Fence is generated on first dispatch of the persistent root, as well as on re-dispatches of the persistent root after context restore.</p>	Format:	Enable							
Format:	Enable									
22	PRT_FenceType									
	<p>This field specifies the type of fence the PRT thread uses. If this field is set to 0, the fence is set at the end of the root thread queue. It will block the dispatch of the next root thread, but allowed these root threads to be populated through VFE to the root thread queue in TS. If this field is set to 1, the fence is set at the entry of VFE, similar to the fence set by the MEDIA_STATE_FLUSH command. No more command can go into the media pipe until a thread spawn message is sent (by the PRT). This field is only valid when PRT_Fence Needed is set to 1. Otherwise, it is ignored by hardware.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 35%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Root thread queue</td> <td>Root thread queue fence</td> </tr> <tr> <td>1h</td> <td>VFE state flush</td> <td>VFE state flush fence</td> </tr> </tbody> </table>	Value	Name	Description	0h	Root thread queue	Root thread queue fence	1h	VFE state flush	VFE state flush fence
Value	Name	Description								
0h	Root thread queue	Root thread queue fence								
1h	VFE state flush	VFE state flush fence								
	21:0	Reserved								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									
3	31:0	Reserved								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									
4..15	31:0	Inline Data								
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table>	Format:	U32						
Format:	U32									

MEDIA_OBJECT_WALKER

MEDIA_OBJECT_WALKER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h MEDIA_OBJECT_WALKER
		Format:	OpCode
	23:16	SubOpcode	
Default Value:		03h MEDIA_OBJECT_WALKER SubOp	
Format:		OpCode	
15:0	DWord Length		
	Default Value:	0Fh DWORD_COUNT_n	
	Format:	=n Total Length - 2	
<p>Note: If this field is greater than 15, it indicates that inline data is present. If present, inline data is common for all threads generated from this command, If this field is 15, it indicates that inline data is not present. It should be noted that unlike other media object command, inline data is optional for this command.</p>			
1	31:8	Reserved	
	7:6	Reserved	
		Format:	Reserved
	5:0	Interface Descriptor Offset	
Project:		BDW	
Format:		U6	
<p>This field specifies the offset from the interface descriptor base pointer to the interface descriptor which will be applied to this object. It is specified in units of interface descriptors.</p>			

MEDIA_OBJECT_WALKER								
2	31	<p>Children Present</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>Boolean</td> </tr> </table> <p>Indicates that the root thread may send spawn messages to spawn child threads and/or synchronized root threads. If Children Present is not set, TS signals VFE to dereference the URB handle immediately after it receives acknowledgement from TD that the thread is dispatched. If Children Present is set, the URB handle is forwarded to the root thread and serves as the return URB handle for the root thread. TS does not signal dereference at the time of dispatch. TS signals URB handle dereference only when it receives a resource dereference message from the thread. <i>In order avoid deadlock, such dereference must be issued once and only once for each URB handle.</i></p>	Format:	Boolean				
	Format:	Boolean						
	30:25	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
	24	<p>Thread Synchronization</p> <p>This field when set indicates that the dispatch of the thread originated from this command is based on the "spawn root thread" message.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>No thread synchronization</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Thread dispatch is synchronized by the 'spawn root thread' message</td> </tr> </tbody> </table>	Value	Name	0	No thread synchronization	1	Thread dispatch is synchronized by the 'spawn root thread' message
	Value	Name						
	0	No thread synchronization						
	1	Thread dispatch is synchronized by the 'spawn root thread' message						
23:22	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
21	<p>Use Scoreboard</p> <p>This field specifies whether the thread associated with this command uses hardware scoreboard. Only when this field is set, the scoreboard control fields in the VFE Dword are valid. If this field is cleared, the thread associated with this command bypasses hardware scoreboard.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Not using scoreboard</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Using scoreboard</td> </tr> </tbody> </table>	Value	Name	0	Not using scoreboard	1	Using scoreboard	
Value	Name							
0	Not using scoreboard							
1	Using scoreboard							
20:17	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
16:0	<p>Indirect Data Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U17 in bytes</td> </tr> </table> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address. It must be DQWord (32-byte) aligned. As the indirect data are sent directly to URB, range is limited to 496 DW. When both inline and indirect data are fetched for this command, the total size in 8-DW registers must be less than or equal to 63 (with both inline data length and indirect data length rounded up to 8-DW aligned).</p>	Format:	U17 in bytes					
Format:	U17 in bytes							

MEDIA_OBJECT_WALKER																							
3	31:0	Indirect Data Start Address Format: IndirectObjectOffset[31:0]																					
		Description																					
		This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the Indirect Object Base Address . Hardware ignores this field if indirect data is not present. Alignment of this address depends on the mode of operation. It is the 64-byte aligned address of the indirect data																					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 33%;">Value</th> <th style="width: 33%;">Name</th> <th style="width: 33%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0 - 512MB]</td> <td></td> <td>(Bits 31:29 MBZ)</td> </tr> </tbody> </table>	Value	Name	Description	[0 - 512MB]		(Bits 31:29 MBZ)															
Value	Name	Description																					
[0 - 512MB]		(Bits 31:29 MBZ)																					
4	31:0	Reserved Format: MBZ																					
5	31:8	Group ID Loop Select Project: BDW																					
		This bit field chooses which of the nested loops of the walker are used to identify threads which share a group id and therefore a shared barrier and SLM. The programmer must ensure that each group will fit into a single subslice. When barriers are enabled every group must have the same number of threads matching the number specified in the Interface Descriptor.																					
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 80%;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td></td> <td>Groups are not created, barriers and SLM are not allocated</td> </tr> <tr> <td>1</td> <td></td> <td>Each complete iteration of the Color loop defines a group, the group id is the concatenation of the Outer global, Inner global, Outer local, Mid local and Inner local loop execution counts.</td> </tr> <tr> <td>2</td> <td></td> <td>Each complete iteration of the Inner local loop and Color loop defines a group, the group id is the concatenation of the Outer global loop to the Mid local loop execution counts.</td> </tr> <tr> <td>3</td> <td></td> <td>Each complete iteration of the Mid local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop to the Outer local loop execution counts.</td> </tr> <tr> <td>4</td> <td></td> <td>Each complete iteration of the Outer local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop and the Inner global loop execution counts.</td> </tr> <tr> <td>5</td> <td></td> <td>Each complete iteration of the Inner global loop and lower loops defines a group, the group id is the Outer global loop execution count.</td> </tr> </tbody> </table>	Value	Name	Description	0		Groups are not created, barriers and SLM are not allocated	1		Each complete iteration of the Color loop defines a group, the group id is the concatenation of the Outer global, Inner global, Outer local, Mid local and Inner local loop execution counts.	2		Each complete iteration of the Inner local loop and Color loop defines a group, the group id is the concatenation of the Outer global loop to the Mid local loop execution counts.	3		Each complete iteration of the Mid local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop to the Outer local loop execution counts.	4		Each complete iteration of the Outer local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop and the Inner global loop execution counts.	5		Each complete iteration of the Inner global loop and lower loops defines a group, the group id is the Outer global loop execution count.
		Value	Name	Description																			
		0		Groups are not created, barriers and SLM are not allocated																			
		1		Each complete iteration of the Color loop defines a group, the group id is the concatenation of the Outer global, Inner global, Outer local, Mid local and Inner local loop execution counts.																			
		2		Each complete iteration of the Inner local loop and Color loop defines a group, the group id is the concatenation of the Outer global loop to the Mid local loop execution counts.																			
3		Each complete iteration of the Mid local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop to the Outer local loop execution counts.																					
4		Each complete iteration of the Outer local loop and lower loops defines a group, the group id is the concatenation of the Outer global loop and the Inner global loop execution counts.																					
5		Each complete iteration of the Inner global loop and lower loops defines a group, the group id is the Outer global loop execution count.																					

MEDIA_OBJECT_WALKER				
6	7:0	<p>Scoreboard Mask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>Boolean</td> </tr> </table> <p>Each bit indicates the corresponding dependency scoreboard is dependent on. This field is AND'd with the corresponding Scoreboard Mask field in the MEDIA_VFE_STATE. All threads generated by this walker command share the same dynamic mask. Bit n (for n = 0...7): Scoreboard n is dependent, where bit 0 maps to n = 0.</p>	Format:	Boolean
	Format:	Boolean		
	31:29	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW
	Project:	BDW		
	28	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	27:24	<p>Color Count Minus One</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U4</td> </tr> </table> <p>This field specifies the number of repeat of the inner most loop of the walker. Each repeated walk position is assigned with an incremental Color number. The Color number together with the X and Y position of the thread is used for dependency scoreboard control. Usage Example: This allows multiple sets of dependency threads to be dispatched.</p>	Format:	U4
	Format:	U4		
	23:21	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	20:16	<p>Middle Loop Extra Steps</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U5</td> </tr> </table>	Format:	U5
	Format:	U5		
	15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
13:12	<p>Local Mid-Loop Unit Y</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>S1</td> </tr> </table>	Format:	S1	
Format:	S1			
11:10	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
9:8	<p>Mid-Loop Unit X</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>S1</td> </tr> </table>	Format:	S1	
Format:	S1			
7:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
7	31:26	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	25:16	<p>Global Loop Exec Count</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U10</td> </tr> </table>	Format:	U10
	Format:	U10		
15:10	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
9:0	<p>Local Loop Exec Count</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U10</td> </tr> </table>	Format:	U10	
Format:	U10			

MEDIA_OBJECT_WALKER		
8	31:25	Reserved Format: _____ MBZ
	24:16	Block Resolution Y Format: _____ U9 Vertical resolution of the local loop.
	15:9	Reserved Format: _____ MBZ
	8:0	Block Resolution X Format: _____ U9 Horizontal resolution of the local loop.
9	31:25	Reserved Format: _____ MBZ
	24:16	Local Start Y Format: _____ U9 Starting vertical position of the local loop.
	15:9	Reserved Format: _____ MBZ
	8:0	Local Start X Format: _____ U9 Starting horizontal position of the local loop.
10	31:25	Reserved Format: _____ MBZ
	24:16	Reserved Project: _____ BDW
		Format: _____ MBZ
	15:9	Reserved Format: _____ MBZ
8:0	Reserved Project: _____ BDW	
	Format: _____ MBZ	
11	31:26	Reserved Format: _____ MBZ
	25:16	Local Outer Loop Stride Y Format: _____ S9 Vertical stride of the local outer loop, in 2's complement.

MEDIA_OBJECT_WALKER		
	15:10	Reserved Format: _____ MBZ
	9:0	Local Outer Loop Stride X Format: _____ S9 Horizontal stride of the local outer loop, in 2's complement.
12	31:26	Reserved Format: _____ MBZ
	25:16	Local Inner Loop Unit Y Format: _____ S9 Vertical stride of the local inner loop, in 2's complement.
	15:10	Reserved Format: _____ MBZ
	9:0	Local Inner Loop Unit X Format: _____ S9 Horizontal stride of the local inner loop, in 2's complement.
13	31:25	Reserved Format: _____ MBZ
	24:16	Global Resolution Y Format: _____ U9 Vertical resolution of the global loop.
	15:9	Reserved Format: _____ MBZ
	8:0	Global Resolution X Format: _____ U9 Horizontal resolution of the global loop.
14	31:26	Reserved Format: _____ MBZ
	25:16	Global Start Y Format: _____ S9 Starting vertical location of the global loop, in 2's complement.
	15:10	Reserved Format: _____ MBZ
	9:0	Global Start X Format: _____ S9 Starting horizontal location of the global loop, in 2's complement.
15	31:26	Reserved Format: _____ MBZ

MEDIA_OBJECT_WALKER		
	25:16	Global Outer Loop Stride Y Format: S9 Vertical stride of the global outer loop, in 2's complement.
	15:10	Reserved Format: MBZ
	9:0	Global Outer Loop Stride X Format: S9 Horizontal stride of the global outer loop, in 2's complement.
16	31:26	Reserved Format: MBZ
	25:16	Global Inner Loop Unit Y Format: S9 Vertical stride of the global inner loop, in 2's complement.
	15:10	Reserved Format: MBZ
	9:0	Global Inner Loop Unit X Format: S9 Horizontal stride of the global inner loop, in 2's complement.
17..n	31:0	Inline Data

MEDIA_STATE_FLUSH

MEDIA_STATE_FLUSH			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>This command updates the Message Gateway state. In particular, it updates the state for a selected Interface Descriptor. This command can be considered same as a MI_Flush except that only media parser will get flushed instead of the entire 3D/media render pipeline. The command should be programmed prior to new Media state, curbe and/or interface descriptor commands when switching to a new context or programming new state for the same context. With this command, pipelined state change is allowed for the media pipe. It should be cautious when using this command when child_present flag in the media state is enabled. This is because that CURBE state as well as Interface Descriptor state are shared between root threads and child threads. Changing these states while child threads are generated on the fly may cause unexpected behavior. Combining with MI_ARB_ON/OFF command, it is possible to support interruptability with the following command sequence where interrupt may be allowed only when MI_ARB_ON_OFF is ON: MEDIA_STATE_FLUSH VFE_STATE // VFE will hold CS if watermark isn't met MI_ARB_OFF // There must be at least one VFE command before this one MEDIA_OBJECT MI_ARB_ON</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MEDIA_STATE_FLUSH
		Format:	OpCode
	23:16	SubOpcode	
		Default Value:	4h MEDIA_STATE_FLUSH SubOp
		Format:	OpCode
15:0	DWord Length		
	Project:	All	
	Format:	=n Total Length - 2	
	Value	Name	Description
	0h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
1	31:9	Reserved	
		Project:	All
		Format:	MBZ

MEDIA_STATE_FLUSH						
8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>N, BDW</td> </tr> </table>	Project:	N, BDW			
Project:	N, BDW					
7	<p>Flush to GO</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This bit indicates that the write data out of this thread group should be flushed to the point where it is visible to following commands.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Workaround</td> </tr> </table> <p>Use PIPE_CONTROL with CS stall and DC flush bits set, in place of MEDIA_STATE_FLUSH with Flush-to-GO set, to work around a preemption boundary condition.</p>	Project:	BDW	Format:	Enable	Workaround
Project:	BDW					
Format:	Enable					
Workaround						
6	<p>Watermark Required</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This is a single bit specifying if the MEDIA_STATE_FLUSH should stall further commands until there is enough room in a half-slice for the following thread group. The characteristics of the thread group are specified in the Interface Descriptor Offset. If set, the MEDIA_STATE_FLUSH stalls CS until there are enough threads in a half-slice, and enough SLM available in the same half-slice, and a free barrier if one is required. An Interface Descriptors can be updated after a Watermarked MEDIA_STATE_FLUSH only if it has not been used in the current context. Reusing an interface descriptor requires that this bit is clear to ensure the ID cache is reloaded. If clear, the MEDIA_STATE_FLUSH stalls CS until the TDL has dispatched the last thread, allowing the CURBE and Interface Descriptors to be updated by following commands.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>The Interface Descriptor Offset used for the flush must be the same as that used for the GPGPU_OBJECTs. GPGPU_WALKER automatically checks the Watermark conditions before starting a thread, so this bit should not be set before GPGPU_WALKER.</p> <p>If pre-emption is used, the WatermarkRequired bit must not be set.</p>	Project:	BDW	Programming Notes		
Project:	BDW					
Programming Notes						
5:0	<p>Interface Descriptor Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U6</td> </tr> </table> <p>This field specifies the offset from the interface descriptor base pointer to the interface descriptor which describes what resources are required to meet the watermark.</p>	Format:	U6			
Format:	U6					

MEDIA_VFE_STATE

MEDIA_VFE_STATE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>A stalling PIPE_CONTROL is required before MEDIA_VFE_STATE unless the only bits that are changed are scoreboard related: Scoreboard Enable, Scoreboard Type, Scoreboard Mask, Scoreboard * Delta. For these scoreboard related states, a MEDIA_STATE_FLUSH is sufficient.</p> <ul style="list-style-type: none"> MEDIA_STATE_FLUSH (optional, only if barrier dependency is needed) MEDIA_INTERFACE_DESCRIPTOR_LOAD (optional) 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MEDIA_VFE_STATE
		Format:	OpCode
	23:16	SubOpcode	
Default Value:		0h MEDIA_VFE_STATE SubOp	
Format:		OpCode	
15:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	Description
	07h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
1	31:10	Scratch Space Base Pointer	
		Format:	GeneralStateOffset[31:10]
	<p>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.</p>		
9:8	Reserved		
	Format:	MBZ	

MEDIA_VFE_STATE									
	7:4	<p>Stack Size</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td></td> <td>indicating [1KBytes, 2MBytes]</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Since the stack uses the upper portion of the scratch space, Stack Size = < Per Thread Scratch Space</p>	Value	Name	Description	[0,11]		indicating [1KBytes, 2MBytes]	
	Value	Name	Description						
[0,11]		indicating [1KBytes, 2MBytes]							
3:0	<p>Per Thread Scratch Space</p> <table border="1"> <tr> <td>Format:</td> <td>U4</td> </tr> </table> <p>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 threads in the device each get Per Thread Scratch Space size without exceeding the driver-allocated scratch space.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,11]</td> <td></td> <td>indicating [1k bytes, 2 Mbytes]: 0 -> 1k, 1->2k, 2->4k, 3->8k ... 11->2M</td> </tr> </tbody> </table>	Format:	U4	Value	Name	Description	[0,11]		indicating [1k bytes, 2 Mbytes]: 0 -> 1k, 1->2k, 2->4k, 3->8k ... 11->2M
Format:	U4								
Value	Name	Description							
[0,11]		indicating [1k bytes, 2 Mbytes]: 0 -> 1k, 1->2k, 2->4k, 3->8k ... 11->2M							
2	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
15:0	<p>Scratch Space Base Pointer High</p> <table border="1"> <tr> <td>Format:</td> <td>GeneralStateOffset[47:32]</td> </tr> </table> <p>This field specifies the high 16 bits of starting address of the Scratch Space Base Pointer</p>	Format:	GeneralStateOffset[47:32]						
Format:	GeneralStateOffset[47:32]								
3	31:16	<p>Maximum Number of Threads</p> <table border="1"> <tr> <td>Format:</td> <td>U16-1 representing thread count</td> </tr> </table> <p>Range: [0, n-1] where n = (# EUs) * (# threads/EU). See <i>Graphics Processing Engine</i> for listing of #EUs and #threads in each device.</p> <p>Specifies the maximum number of simultaneous root threads allowed to be active. Used to avoid potential deadlock. If child threads are not planning on being used then this field can be set to its maximum value and there will be no thread limit beyond what is currently available in the system; the maximum value can include threads in slices that have been shut down for power reasons. For GPGPU threads the maximum value must be used.</p> <p style="text-align: center;">Programming Notes</p> <p>MSB will be zero due to the range limit below.</p>	Format:	U16-1 representing thread count					
	Format:	U16-1 representing thread count							

MEDIA_VFE_STATE															
15:8	<p>Number of URB Entries</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U8</td> </tr> </table> <p>Specifies the number of URB entries that are used by the unit.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 30%;">Description</th> <th style="width: 40%;">Project</th> </tr> </thead> <tbody> <tr> <td>[1,64]</td> <td></td> <td>[1,64] Entries</td> <td>BDW:GT1, DevBDW:GT2</td> </tr> <tr> <td>[1,128]</td> <td></td> <td>[1,128] Entries</td> <td>BDW:GT3</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Please note that 0 is not allowed for this field.</p>	Format:	U8	Value	Name	Description	Project	[1,64]		[1,64] Entries	BDW:GT1, DevBDW:GT2	[1,128]		[1,128] Entries	BDW:GT3
	Format:	U8													
	Value	Name	Description	Project											
	[1,64]		[1,64] Entries	BDW:GT1, DevBDW:GT2											
	[1,128]		[1,128] Entries	BDW:GT3											
	7	<p>Reset Gateway Timer</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the reset of the timestamp counter maintained in Message Gateway.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Maintaining the existing timestamp state</td> </tr> <tr> <td>1h</td> <td>Resetting relative timer and latching the global timestamp</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0h	Maintaining the existing timestamp state	1h	Resetting relative timer and latching the global timestamp					
		Project:	BDW												
		Value	Name												
		0h	Maintaining the existing timestamp state												
	1h	Resetting relative timer and latching the global timestamp													
6	<p>Bypass Gateway Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field configures Gateway to use a simple message protocol.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Maintaining OpenGateway/ForwardMsg/CloseGateway protocol (legacy mode)</td> </tr> <tr> <td>1h</td> <td>Bypassing OpenGateway/CloseGateway protocol</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0h	Maintaining OpenGateway/ForwardMsg/CloseGateway protocol (legacy mode)	1h	Bypassing OpenGateway/CloseGateway protocol						
	Project:	BDW													
	Value	Name													
	0h	Maintaining OpenGateway/ForwardMsg/CloseGateway protocol (legacy mode)													
1h	Bypassing OpenGateway/CloseGateway protocol														
5:3	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
	Project:	BDW													
Format:	MBZ														
2	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
	Project:	BDW													
Format:	MBZ														
4	<p>Reserved</p>														
	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
	Format:	MBZ													
	3:2	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:		BDW													
Format:	MBZ														
	<p>Reserved</p>														

MEDIA_VFE_STATE																
1:0	<p>Slice Disable</p> <p>This field disables dispatch to slices and subslices for Media and GPGPU applications. It is used to limit the amount of scratch space that needs to be allocated for a context. If a particular configuration doesn't have slice or subslice then there is no impact to disabling it.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>All subslices are enabled.</td> </tr> <tr> <td>01b</td> <td></td> <td>Slice 2 and 1 are disabled, only Slice 0 with all subslices is enabled.</td> </tr> <tr> <td>10b</td> <td></td> <td>Reserved</td> </tr> <tr> <td>11b</td> <td></td> <td>Slice 2 and 1 are disabled, only Slice 0 with only subslice 0 enabled.</td> </tr> </tbody> </table>	Value	Name	Description	00b		All subslices are enabled.	01b		Slice 2 and 1 are disabled, only Slice 0 with all subslices is enabled.	10b		Reserved	11b		Slice 2 and 1 are disabled, only Slice 0 with only subslice 0 enabled.
Value	Name	Description														
00b		All subslices are enabled.														
01b		Slice 2 and 1 are disabled, only Slice 0 with all subslices is enabled.														
10b		Reserved														
11b		Slice 2 and 1 are disabled, only Slice 0 with only subslice 0 enabled.														
5	<p>31:16 URB Entry Allocation Size</p> <table border="1"> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p style="text-align: center;">Description</p> <p>Specifies the length of each URB entry used by the unit, in 256-bit register increments. ROB address for URB starts after CURBE Allocated region. (URB Entry Allocation Size * Number of URB Entries) + CURBE Allocation Size + Number of Interface Descriptors) must be less than (number of bytes allocated for the URB in L3CNTLREG / 32 bytes per entry). Note: Number of Interface Descriptors is 64.</p> <p>If SLM is enabled for GPGPU work then the number of available entries will be 1/2 the maximum URB entries.</p> <p style="text-align: center;">Programming Notes</p> <p>When Inline data is used with MEDIA_OBJECT or MEDIA_OBJECT_WALKER, then the URB entry allocation size must match the Inline data size. If Indirect data is being used with MEDIA_OBJECT or GPGPU_WALKER then the allocation size must be sufficient for the Indirect data. If both Inline and Indirect are being used, then the allocation size must match the sum of the Inline and Indirect.</p>	Format:	U16													
Format:	U16															
15:0	<p>CURBE Allocation Size</p> <table border="1"> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p style="text-align: center;">Description</p> <p>Specifies the total length allocated for CURBE, in 256-bit register increments. ROB address for CURBE starts at address 64. (URB Entry Allocation Size * Number of URB Entries) + CURBE Allocation Size + Interface Descriptor Entries) must be less than or equal to the number of entries in the URB as described in Configurations. Interface Descriptor Entries is 64</p> <p>If SLM is enabled for GPGPU work then the number of available entries will be ½ the maximum URB entries.</p> <p style="text-align: center;">Programming Notes</p> <p>CURBE Allocation Size should be 0 for GPGPU workloads that uses indirect instead of CURBE.</p>	Format:	U16													
Format:	U16															



MEDIA_VFE_STATE										
6	31	<p>Scoreboard Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field enables and disables the hardware scoreboard in the Media Pipeline. If this field is cleared, hardware ignores the following scoreboard state fields.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Scoreboard disabled</td> </tr> <tr> <td>1h</td> <td>Scoreboard enabled</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0h	Scoreboard disabled	1h	Scoreboard enabled
	Project:	BDW								
	Value	Name								
	0h	Scoreboard disabled								
	1h	Scoreboard enabled								
30	<p>Scoreboard Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field selects the type of scoreboard in use.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Stalling Scoreboard</td> </tr> <tr> <td>1h</td> <td>Non-Stalling Scoreboard</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	0h	Stalling Scoreboard	1h	Non-Stalling Scoreboard	
Project:	BDW									
Value	Name									
0h	Stalling Scoreboard									
1h	Non-Stalling Scoreboard									
29:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
15:8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
Project:	BDW									
Format:	MBZ									
7:0	<p>Scoreboard Mask</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable[8]</td> </tr> </table> <p>Each bit indicates the corresponding dependency scoreboard is enabled. The scoreboard is based on the relative (X, Y) distance from the current threads' (X, Y) position. Bit n (for n = 0...7): Score n is enabled.</p>	Project:	BDW	Format:	Enable[8]					
Project:	BDW									
Format:	Enable[8]									
7	31:28	<p>Scoreboard 3 Delta Y</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative vertical distance of the dependent instance assigned to scoreboard 3, in the form of 2's compliment.</p> <table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Project:	BDW	Format:	S3	Programming Notes	MBZ if scoreboard is disabled.		
	Project:	BDW								
	Format:	S3								
	Programming Notes									
MBZ if scoreboard is disabled.										
27:24	<p>Scoreboard 3 Delta X</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative horizontal distance of the dependent instance assigned to scoreboard 3, in the form of 2's compliment.</p> <table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Project:	BDW	Format:	S3	Programming Notes	MBZ if scoreboard is disabled.			
Project:	BDW									
Format:	S3									
Programming Notes										
MBZ if scoreboard is disabled.										

MEDIA_VFE_STATE	
23:20	Scoreboard 2 Delta Y
	Project: BDW
	Format: S3
	Relative vertical distance of the dependent instance assigned to scoreboard 2, in the form of 2's compliment.
	Programming Notes
MBZ if scoreboard is disabled.	
19:16	Scoreboard 2 Delta X
	Project: BDW
	Format: S3
	Relative horizontal distance of the dependent instance assigned to scoreboard 2, in the form of 2's compliment.
	Programming Notes
MBZ if scoreboard is disabled.	
15:12	Scoreboard 1 Delta Y
	Project: BDW
	Format: S3
	Relative vertical distance of the dependent instance assigned to scoreboard 1, in the form of 2's compliment.
	Programming Notes
MBZ if scoreboard is disabled.	
11:8	Scoreboard 1 Delta X
	Project: BDW
	Format: S3
	Relative horizontal distance of the dependent instance assigned to scoreboard 1, in the form of 2's compliment.
	Programming Notes
MBZ if scoreboard is disabled.	
7:4	Scoreboard 0 Delta Y
	Project: BDW
	Format: S3
	Relative vertical distance of the dependent instance assigned to scoreboard 0, in the form of 2's compliment.
	Programming Notes
MBZ if scoreboard is disabled.	

MEDIA_VFE_STATE							
	3:0	Scoreboard 0 Delta X <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative horizontal distance of the dependent instance assigned to scoreboard 0, in the form of 2's compliment.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>MBZ if scoreboard is disabled.</p>	Project:	BDW	Format:	S3	Programming Notes
	Project:	BDW					
	Format:	S3					
	Programming Notes						
	8	31:28	Scoreboard 7 Delta Y <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative vertical distance of the dependent instance assigned to scoreboard 7, in the form of 2's compliment.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>MBZ if scoreboard is disabled.</p>	Project:	BDW	Format:	S3
Project:		BDW					
Format:		S3					
Programming Notes							
27:24	Scoreboard 7 Delta X <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative horizontal distance of the dependent instance assigned to scoreboard 7, in the form of 2's compliment.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>MBZ if scoreboard is disabled.</p>	Project:	BDW	Format:	S3	Programming Notes	
Project:	BDW						
Format:	S3						
Programming Notes							
23:20	Scoreboard 6 Delta Y <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative vertical distance of the dependent instance assigned to scoreboard 6, in the form of 2's compliment.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>MBZ if scoreboard is disabled.</p>	Project:	BDW	Format:	S3	Programming Notes	
Project:	BDW						
Format:	S3						
Programming Notes							
19:16	Scoreboard 6 Delta X <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table> <p>Relative horizontal distance of the dependent instance assigned to scoreboard 6, in the form of 2's compliment.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>MBZ if scoreboard is disabled.</p>	Project:	BDW	Format:	S3	Programming Notes	
Project:	BDW						
Format:	S3						
Programming Notes							

MEDIA_VFE_STATE					
15:12	Scoreboard 5 Delta Y				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table>	Project:	BDW	Format:	S3
	Project:	BDW			
	Format:	S3			
Relative vertical distance of the dependent instance assigned to scoreboard 5, in the form of 2's compliment.					
<table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Programming Notes	MBZ if scoreboard is disabled.			
Programming Notes					
MBZ if scoreboard is disabled.					
11:8	Scoreboard 5 Delta X				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table>	Project:	BDW	Format:	S3
	Project:	BDW			
	Format:	S3			
Relative horizontal distance of the dependent instance assigned to scoreboard 5, in the form of 2's compliment.					
<table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Programming Notes	MBZ if scoreboard is disabled.			
Programming Notes					
MBZ if scoreboard is disabled.					
7:4	Scoreboard 4 Delta Y				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table>	Project:	BDW	Format:	S3
	Project:	BDW			
	Format:	S3			
Relative vertical distance of the dependent instance assigned to scoreboard 4, in the form of 2's compliment.					
<table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Programming Notes	MBZ if scoreboard is disabled.			
Programming Notes					
MBZ if scoreboard is disabled.					
3:0	Scoreboard 4 Delta X				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>S3</td> </tr> </table>	Project:	BDW	Format:	S3
	Project:	BDW			
	Format:	S3			
Relative horizontal distance of the dependent instance assigned to scoreboard 4, in the form of 2's compliment.					
<table border="1" style="width: 100%;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> <tr> <td>MBZ if scoreboard is disabled.</td> </tr> </table>	Programming Notes	MBZ if scoreboard is disabled.			
Programming Notes					
MBZ if scoreboard is disabled.					

Media Block Read MSD

MSD1R_MB - Media Block Read MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Other	
Group:	Media Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18:14	Message Type
Default Value: 04h		
Project: All		
Format: Opcode		
Media Block Read message		
13:11	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
10:8	Vertical Line Stride Override	
	Project: All	
	Format: MDC_VLSO	
If enabled, specifies the Vertical Line Stride and Vertical Line Stride Offset override fields.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS	
Specifies the Binding Table Index for the message		

Media Block Write MSD

MSD1W_MB - Media Block Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Other	
Group:	Media Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
		Indicates that the message requires a header.
	18:14	Message Type
		Default Value: 0Ah
		Project: All
		Format: Opcode
	Media Block Write message	
	13:11	Reserved
		Project: All
		Format: MBZ
		Ignored
	10:8	Vertical Line Stride Override
		Project: All
		Format: MDC_VLSO
		If enabled, specifies the Vertical Line Stride and Vertical Line Stride Offset override fields.
	7:0	Binding Table Index
		Project: All
		Format: MDC_BTS
Specifies the Binding Table Index for the message		



Media Transpose Read MSD

MSD1R_TT - Media Transpose Read MSD							
Project:	BDW						
Source:	DataPort 1						
Length Bias:	1						
Family:	Other						
Group:	Transpose Read						
DWord	Bit	Description					
0	19	Header Present <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> Indicates that the message requires a header.	Project:	All	Format:	MDC_MHR	
		Project:	All				
		Format:	MDC_MHR				
		18:14	Message Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>00h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Transpose Read message	Default Value:	00h	Project:	All
Default Value:	00h						
Project:	All						
Format:	Opcode						
13:8	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> Ignored	Project:	All	Format:	MBZ		
	Project:	All					
	Format:	MBZ					
7:0	Binding Table Index <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> Specifies the Binding Table Index for the message	Project:	All	Format:	MDC_BTS		
	Project:	All					
	Format:	MDC_BTS					

Memory Fence MSD

MSD_MEMFENCE - Memory Fence MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Other	
Group:	Memory Fence	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	Indicates that the message requires a header.	
	18	Legacy Message
Default Value: 0h		
Project: All		
Format: Opcode		
Legacy Message		
17:14	Message Type	
	Default Value: 07h	
	Project: All	
	Format: Opcode	
Memory Fence message		
13	Commit	
	Project: All	
	Format: Enable	
Specifies whether control is returned to the thread only after the fence has been honored.		
12:9	L3 Flush	
	The L3 Flush control is one of the following GSYNC signals.	
	Value	Name
	0h	Disabled [Default]
	The L3 caches are not flushed.	
Programming Notes		
If multiple caches need to be flushed, the commands need to be sent separately.		
8	Reserved	
	Project: BDW	
	Format: MBZ	
	Ignored	



MSD_MEMFENCE - Memory Fence MSD

	7:0	Reserved	
		Project:	BDW
		Format:	MBZ
		Ignored	

MFC_AVC_PAK_OBJECT

MFC_AVC_PAK_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFC_AVC_PAK_OBJECT command is the second primitive command for the AVC Encoding Pipeline. The same command is used for both CABAC and CAVLC modes. The MV Data portion of the bitstream is loaded as indirect data object. Before issuing a MFC_AVC_PAK_OBJECT command, all AVC MFX states need to be valid. Therefore the commands used to set these states need to have been issued prior to the issue of this command. MB record must be consecutive with no gaps, hence we do not need MB(x,y) in each MB command. Internal counter will keep track of the current MB address, starting from the Start_MB_In_Slice loaded at the beginning of each slice. MFC_AVC_PAK_OBJECT command follows the MbType definition like MFD. Many fields in this command are identical to that in VME output. This is intended to reduce software converting overhead from VME to PAK. Encoding statistical data such as the total size of the output bitstream are provided through MMIO registers. Software may access these registers through MI_STORE_REGISTER_MEM command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFC_AVC_PAK_OBJECT
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h AVC_ENC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	2h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	9h
Format:		OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length	Format:	=n Length -2
	Value	Name	Project
	000Ah	DWORD_COUNT_n [Default]	BDW

MFC_AVC_PAK_OBJECT						
1	31:10	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ		
	Format:	MBZ				
9:0	<p>Indirect PAK-MV Data Length</p> <p>This field provides the length in bytes of the indirect data, which contains all the MVs for the current MB (in any partitioning and subpartitioning form). A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect PAK-MV Data Start Address field is ignored. This field must have the same alignment as the Indirect PAK-MV Data Start Address. This field must be DW aligned (since each MV is 4 bytes in size). Driver has to derived this field from MVsize (MVquantity in DXVA, exact size) *4 bytes per MV.</p>					
2	31:29	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ		
	Format:	MBZ				
28:0	<p>Indirect PAK-MV Data Start Address Offset</p> <p>This field specifies the memory starting address (offset) of the MV data to be fetched into PAK Subsystem for processing. This pointer is relative to the MFC Indirect PAK-MV Object Base Address. Hardware ignores this field if indirect data is not present, i.e. the Indirect PAK-MV Data Length is set to 0. It is a Dword aligned address in all AVC encoding configuration, since each MV is 4 bytes in size.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 60%; text-align: center;">Value</th> <th style="width: 40%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,512MB)</td> <td></td> </tr> </tbody> </table>		Value	Name	[0,512MB)	
Value	Name					
[0,512MB)						
3..10	31:0	<p>Inline Data</p> <p>All the required MB level controls and parameters for encoding are captured as inline data of the MFC_AVC_PAK_OBJECT command. It has a fixed size of 8 DWs. Its definition is described in the next section.</p>				

MFC_MPEG2_PAK_OBJECT

MFC_MPEG2_PAK_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFC_MPEG2_PAK_OBJECT command is the second primitive command for the MPEG-2 Encoding Pipeline. Different from AVC, the MV Data portion of the bitstream is loaded as part of MB control data. Before issuing a MFC_MPEG2_PAK_OBJECT command, all MPEG2_MFX states need to be valid. Therefore the commands used to set these states need to have been issued prior to the issue of this command. MB record must be consecutive with no gaps, hence we do not need MB(x,y) in each MB command. Internal counter will keep track of the current MB address, starting from the Start_MB_In_Slice loaded at the beginning of each slice.</p> <p>MFC_MPEG2_PAK_OBJECT command follows the MbType definition like MFD. Many fields in this command are identical to that in VME output. This is intended to reduce software converting overhead from VME to PAK. Encoding statistical data such as the total size of the output bitstream are provided through MMIO registers. Software may access these registers through MI_STORE_REGISTER_MEM command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFC_AVC_PAK_INSERT_OBJECT
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	3h MPEG2
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	2h ENC
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	9h MEDIA_
		Format:	OpCode
15:12	Reserved		
	Project:	All	
	Format:	MBZ	



MFC_MPEG2_PAK_OBJECT								
	11:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>0007h Excludes DWord (0,1)</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0007h Excludes DWord (0,1)	Project:	All	Format:	=n Total Length - 2
Default Value:	0007h Excludes DWord (0,1)							
Project:	All							
Format:	=n Total Length - 2							
1..8	31:0	<p>Inline Data</p> <p>All the required MB level controls and parameters for encoding are captured as inline data of the MFC_MPEG2_PAK_OBJECT command. It has a fixed size of 8 DWs. Its definition is described in the next section</p>						

MFC_MPEG2_SLICEGROUP_STATE

MFC_MPEG2_SLICEGROUP_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a slice group level command and can be issued multiple times within a picture that is comprised of multiple slice groups. The same command is used for AVC encoder (PAK mode) and decoder (VLD and IT modes).</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MPEG2_SLICEGROUP_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	3h MPEG2
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	2h MEDIA_
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	3h MEDIA_
		Format:	OpCode
	15:12	Reserved	
		Project:	All
		Format:	MBZ
	11:0	DWord Length	
		Default Value:	6h Excludes DWord (0,1)
		Project:	All
Format:		=n Total Length - 2	

MFC_MPEG2_SLICEGROUP_STATE																	
1	31	<p>MbRateCtrlFlag- RateControlCounterEnable (Encoder-only) To enable the accumulation of bit allocation for rate control. This field enables hardware Rate Control logic. The rest of the RC control fields are only valid when this field is set to 1. Otherwise, hardware ignores these fields. Note: To reset MB level rate control (QRC), we need to set both bits MbRateCtrlFlag and MbRateCtrlReset to 1 in the new slice</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> </tr> <tr> <td>1h</td> <td>Enable</td> </tr> </tbody> </table>	Value	Name	0h	Disable	1h	Enable									
	Value	Name															
	0h	Disable															
	1h	Enable															
	30	<p>MbRateCtrlReset- ResetRateControlCounter (Encoder-only) To reset the bit allocation accumulation counter to 0 to restart the rate control.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 33%;">Value</th> <th style="width: 33%;">Name</th> <th style="width: 34%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Not reset</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>reset</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	Not reset	1h	Enable	reset						
	Value	Name	Description														
	0h	Disable	Not reset														
	1h	Enable	reset														
	29:28	<p>MbRateCtrlMode- RC Triggler Mode (Encoder-only)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>Always Rate Control, whereas RC becomes active if sum_act > sum_target or sum_act < sum_target</td> </tr> <tr> <td>01b</td> <td></td> <td>Gentle Rate Control, whereas RC becomes active if sum_act > upper_midpt or sum_act < lower_midpt</td> </tr> <tr> <td>10b</td> <td></td> <td>Loose Rate Control, whereas RC becomes active if sum_act > sum_max or sum_act < sum_min</td> </tr> <tr> <td>11b</td> <td></td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	Description	00b		Always Rate Control, whereas RC becomes active if sum_act > sum_target or sum_act < sum_target	01b		Gentle Rate Control, whereas RC becomes active if sum_act > upper_midpt or sum_act < lower_midpt	10b		Loose Rate Control, whereas RC becomes active if sum_act > sum_max or sum_act < sum_min	11b		Reserved
	Value	Name	Description														
00b		Always Rate Control, whereas RC becomes active if sum_act > sum_target or sum_act < sum_target															
01b		Gentle Rate Control, whereas RC becomes active if sum_act > upper_midpt or sum_act < lower_midpt															
10b		Loose Rate Control, whereas RC becomes active if sum_act > sum_max or sum_act < sum_min															
11b		Reserved															
27:24	<p>MbRateCtrlParam- RC Stable Tolerance (Encoder-only)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> <tr> <td colspan="2">This field specifies the tolerance required to deactivate RC once it has been triggered.</td> </tr> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> <tr> <td>[0, 15]</td> <td></td> </tr> </table>	Format:	U4	This field specifies the tolerance required to deactivate RC once it has been triggered.		Value	Name	[0, 15]									
Format:	U4																
This field specifies the tolerance required to deactivate RC once it has been triggered.																	
Value	Name																
[0, 15]																	
23	<p>RateCtrlPanicFlag - RC Panic Enable (Encoder-only) If this field is set to 1, RC enters panic mode when sum_act > sum_max. RC Panic Type field controls what type of panic behavior is invoked.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable</td> </tr> <tr> <td>1</td> <td>Enable</td> </tr> </tbody> </table>	Value	Name	0	Disable	1	Enable										
Value	Name																
0	Disable																
1	Enable																

MFC_MPEG2_SLICEGROUP_STATE

22	RateCtrlPanicType - RC Panic Type (Encoder-only) This field selects between two RC Panic methods. If it is set to 0, in panic mode, the macroblock QP is maxed out, setting to requested QP + QP_max_pos_mod. If it is set to 1, for an intra macroblock, AC CBPs are set to zero (note that DC CBPs are not modified). For inter macroblocks, AC and DC CBPs are forced to zero.						
	Value	Name	Description				
	0h		QP Panic				
	1h		CBP Panic				
21	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>			Project:	All	Format:	MBZ
Project:	All						
Format:	MBZ						
20	SkipConvDisabled - MB Type Skip Conversion Disable (Encoder-only) This field is only valid for a P or B slice. It must be zero for other slice types. Rules are provided in Section 2.3.3.1.6						
	Value	Name	Description				
	0h	Enable	Enable skip type conversion				
	1h	Disable	Disable skip type conversion				
19	IsLastSliceGrp IsLastSliceGrp = 1 if the current slice group is the last slice group of a picture; 0 otherwise. It is used by the zero filling in the Minimum Frame Size test.						
18	BitstreamOutputFlag - Compressed BitStream Output Disable Flag (Encoder-only)						
	Value	Name	Description				
	0h	Enable	enable the writing of the output compressed bitstream				
	1h	Disable	disable the writing of the output compressed bitstream				
17	HeaderPresentFlag - Header Insertion Present in Bitstream (Encoder-only)						
	Value	Name	Description				
	0h	Disable	no header insertion into the output bitstream buffer, in front of the current slice encoded bits				
	1h	Enable	header insertion into the output bitstream buffer is present, and is in front of the current slice encoded bits.				
16	SliceData PresentFlag - SliceData Insertion Present in Bitstream (Encoder-only)						
	Value	Name	Description				
	0h	Disable	no Slice Data insertion into the output bitstream buffer				
	1h	Enable	Slice Data insertion into the output bitstream buffer is present.				

		MFC_MPEG2_SLICEGROUP_STATE		
	15	TailPresentFlag - Tail Insertion Present in bitstream (Encoder-only)		
		Value	Name Description	
		0h		no tail insertion into the output bitstream buffer, after the current slice encoded bits
		1h		tail insertion into the output bitstream buffer is present, and is after the current slice encoded bits.
	14	FirstSliceHdrDisabled when this is on, the first slice header of the slice group is expected to be provided by the user via insertion command. PAK HW will skip it.		
	13	IntraSlice intra slice value included in slice headers, when IntraSliceFlag = 1.		
	12	IntraSliceFlag intra slice flag included in slice headers		
	11:8	Reserved		
		Format:	MBZ for SliceID extension	
	7:4	SliceID[3:0] (Encoder-only) To identify the output data (coding information record) returned for rate control from PAK to ENC and VPP		
3:2	Reserved			
	Format:	MBZ for StreamID extension		
1:0	StreamID[1:0] (Encoder-only) To identify the output data (coding information record) returned for rate control from PAK to ENC and VPP			
2	31:24	NextSgMbYcnt - also NextStartVertPos Vertical count of the first MB in the next slice group (Encoder-only)Note: This field restricts total number of MB in the Y direction to 255 or less.		
	23:16	NextSgMbXcnt - also NextStartHorzPos BitFieldDesc		
	15:8	FirstMbYcnt - also CurrStartVertPos		
		Project:	All	
	Format:	U8		
	also CurrStartVertPos, Vertical count of the first MB in the current slice group (Encoder-only)			
7:0	FirstMbXcnt - also CurrStartHorzPos			
	Project:	All		
	Format:	U8		
	Horizontal count of the first MB in the current slice group (Encoder-only)			
3	31:9	Reserved		
		Format:	MBZ	

MFC_MPEG2_SLICEGROUP_STATE								
	8	SliceGroupSkip <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>All macroblocks are skipped</p>	Project:	All	Exists If:	//Encoder Only	Format:	U1
	Project:	All						
	Exists If:	//Encoder Only						
Format:	U1							
7:6	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
	5:0	SliceGroupQp <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>Initial slice quality parameter</p>	Project:	All	Exists If:	//Encoder Only	Format:	U6
	Project:	All						
	Exists If:	//Encoder Only						
Format:	U6							
4	31:29	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
28:0	BitstreamOffset - Indirect PAK-BSE Data Start Address (Write) <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> </table> <p>This field specifies the memory starting address (offset) to write out the compressed bitstream data from the BSE processing. This pointer is relative to the MFC Indirect PAK-BSE Object Base Address. It is a byte-aligned address for the AVC bitstream data in both CABAC/CAVLC Modes. For Write, there is no need to have a data length field. It is assumed the global memory bound check specified in the IND_OBJ_BASE_ADDRESS command (Indirect PAK-BSE Object Access Upper Bound) will take care of any illegal write access. This field is only valid for AVC encode mode.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,512MB)</td> <td></td> </tr> </tbody> </table>	Exists If:	//Encoder Only	Value	Name	[0,512MB)		
Exists If:	//Encoder Only							
Value	Name							
[0,512MB)								
5	31:24	MaxQpNegModifier - Magnitude of QP Max Negative Modifier (Encoder-only) <table border="1"> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>This field specifies the lower limit of the QP modifier.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 51]</td> <td></td> </tr> </tbody> </table>	Format:	U8	Value	Name	[0, 51]	
	Format:	U8						
Value	Name							
[0, 51]								
23:16	MaxQpPosModifier - Magnitude of QP Max Positive Modifier (Encoder-only) <table border="1"> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>This field specifies the upper limit of the QP modifier.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 51]</td> <td></td> </tr> </tbody> </table>	Format:	U8	Value	Name	[0, 51]		
Format:	U8							
Value	Name							
[0, 51]								

MFC_MPEG2_SLICEGROUP_STATE		
6	15:12	ShrinkParam - Shrink Resistance (Encoder-only)
		Format: U4
		This field specifies the additional points added each time decreased correction is invoked.
		Value Name
		[0, 15]
	11:8	ShrinkParam - Shrink Init (Encoder-only)
		Format: U4
		This field specifies the initial points required to trip decreased control.
		Value Name
		[0, 15]
	7:4	GrowParam - Grow Resistance (Encoder-only)
		Format: U4
	This field specifies the additional points added each time increased correction is invoked.	
	Value Name	
	[0, 15]	
3:0	GrowParam - Grow Init (Encoder-only)	
	Format: U4	
	This field specifies the initial points required to trip increased control.	
	Value Name	
	[0, 15]	
31:24	Reserved	
	Format: MBZ	
23:20	CorrectPoints - Correct 6 (Encoder-only)	
	Format: U4	
	This field specifies the points used in the lowermost RC region when sum_act <= sum_min.	
	Value Name	
	[0, 15]	
19:16	CorrectPoints - Correct 5 (Encoder-only)	
	Format: U4	
	This field specifies the points used in the fifth RC region when sum_act > sum_min but <= lower_midpt.	
	Value Name	
	[0, 15]	

MFC_MPEG2_SLICEGROUP_STATE				
	15:12 CorrectPoints - Correct 4 (Encoder-only)			
	Format: U4			
	This field specifies the points used in the fourth RC region when sum_act > lower_midpt but <= sum_target.			
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 15]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0, 15]
Value	Name			
[0, 15]				
	11:8 CorrectPoints - Correct 3 (Encoder-only)			
	Format: U4			
	This field specifies the points used in the third RC region when sum_act > sum_target but <= upper_midpt.			
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 15]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0, 15]
Value	Name			
[0, 15]				
	7:4 CorrectPoints - Correct 2 (Encoder-only)			
	Format: U4			
	This field specifies the points used in the second RC region when sum_act > upper_midpt but <= sum_max.			
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 15]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0, 15]
Value	Name			
[0, 15]				
	3:0 CorrectPoints - Correct 1 (Encoder-only)			
	Format: U4			
	This field specifies the points used in the topmost RC region when sum_act > sum_max			
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 15]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0, 15]
Value	Name			
[0, 15]				
7	31:28 CV7 - Clamp Value 7 (Encoder-only)			
	Exists If: //Encoder Only			
	27:24 CV6 - Clamp Value 6 (Encoder-only)			
	Project: All			
	Exists If: //Encoder Only			
	Format: U4			
	23:20 CV5 - Clamp Value 5 (Encoder-only)			
	Project: All			
	Exists If: //Encoder Only			
	Format: U4			
	19:16 CV4 - Clamp Value 4 (Encoder-only)			
	Project: All			
	Exists If: //Encoder Only			
	Format: U4			

MFC_MPEG2_SLICEGROUP_STATE

	15:12	<p>CV3 - Clamp Value 3 (Encoder-only)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">Project:</td> <td>All</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table>	Project:	All	Exists If:	//Encoder Only	Format:	U4																																																																																																																										
Project:	All																																																																																																																																	
Exists If:	//Encoder Only																																																																																																																																	
Format:	U4																																																																																																																																	
	11:8	<p>CV2 - Clamp Value 2 (Encoder-only)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">Project:</td> <td>All</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table>	Project:	All	Exists If:	//Encoder Only	Format:	U4																																																																																																																										
Project:	All																																																																																																																																	
Exists If:	//Encoder Only																																																																																																																																	
Format:	U4																																																																																																																																	
	7:4	<p>CV1 - Clamp Value 1 (Encoder-only)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">Project:</td> <td>All</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>U4</td> </tr> </table>	Project:	All	Exists If:	//Encoder Only	Format:	U4																																																																																																																										
Project:	All																																																																																																																																	
Exists If:	//Encoder Only																																																																																																																																	
Format:	U4																																																																																																																																	
	3:0	<p>CV0 - Clamp Value 0 (Encoder-only)</p> <p>If the magnitude of coefficients at locations assigned with CV0 (mapping shown below) exceeds 2CV0-1, they are replaced with 2CV0-1. For coefficients at locations marked as 'none', no clamping is performed. The following mappings are only applied to luma and chroma blocks\subblocks containing AC coefficients (blocks\subblocks with only DC coeffs will not be clamped).</p> <p>For 8x8 frame block, each coefficient is mapped to one of the eight CV values as following:</p> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <tr><td>none</td><td>none</td><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td></tr> <tr><td>none</td><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td></tr> <tr><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td><td>CV2</td></tr> <tr><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td></tr> <tr><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td><td>CV1</td></tr> <tr><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td><td>CV1</td><td>CV0</td></tr> <tr><td>CV3</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td><td>CV1</td><td>CV0</td><td>CV0</td></tr> <tr><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td><td>CV1</td><td>CV0</td><td>CV0</td><td>CV0</td></tr> </table> <p>For 8x8 field block, each coefficient is mapped to one of the eight CV values as following:</p> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <tr><td>none</td><td>none</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV1</td></tr> <tr><td>none</td><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV1</td></tr> <tr><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV3</td><td>CV2</td><td>CV1</td></tr> <tr><td>CV7</td><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td></tr> <tr><td>CV6</td><td>CV5</td><td>CV4</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV1</td><td>CV0</td></tr> <tr><td>CV6</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV2</td><td>CV1</td><td>CV0</td></tr> <tr><td>CV5</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV1</td><td>CV1</td><td>CV0</td></tr> <tr><td>CV5</td><td>CV5</td><td>CV4</td><td>CV3</td><td>CV2</td><td>CV1</td><td>CV1</td><td>CV0</td></tr> </table>	none	none	CV7	CV6	CV5	CV4	CV3	CV3	none	CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV2	CV6	CV5	CV4	CV3	CV3	CV2	CV2	CV1	CV5	CV4	CV3	CV3	CV2	CV2	CV1	CV1	CV4	CV3	CV3	CV2	CV2	CV1	CV1	CV0	CV3	CV3	CV2	CV2	CV1	CV1	CV0	CV0	CV3	CV2	CV2	CV1	CV1	CV0	CV0	CV0	none	none	CV6	CV5	CV4	CV3	CV2	CV1	none	CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV1	CV7	CV6	CV5	CV4	CV3	CV2	CV2	CV1	CV6	CV5	CV4	CV4	CV3	CV2	CV1	CV0	CV6	CV5	CV4	CV3	CV2	CV2	CV1	CV0	CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0	CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0
none	none	CV7	CV6	CV5	CV4	CV3	CV3																																																																																																																											
none	CV7	CV6	CV5	CV4	CV3	CV3	CV2																																																																																																																											
CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV2																																																																																																																											
CV6	CV5	CV4	CV3	CV3	CV2	CV2	CV1																																																																																																																											
CV5	CV4	CV3	CV3	CV2	CV2	CV1	CV1																																																																																																																											
CV4	CV3	CV3	CV2	CV2	CV1	CV1	CV0																																																																																																																											
CV3	CV3	CV2	CV2	CV1	CV1	CV0	CV0																																																																																																																											
CV3	CV2	CV2	CV1	CV1	CV0	CV0	CV0																																																																																																																											
none	none	CV6	CV5	CV4	CV3	CV2	CV1																																																																																																																											
none	CV7	CV6	CV5	CV4	CV3	CV2	CV1																																																																																																																											
CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV1																																																																																																																											
CV7	CV6	CV5	CV4	CV3	CV2	CV2	CV1																																																																																																																											
CV6	CV5	CV4	CV4	CV3	CV2	CV1	CV0																																																																																																																											
CV6	CV5	CV4	CV3	CV2	CV2	CV1	CV0																																																																																																																											
CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0																																																																																																																											
CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0																																																																																																																											

MFD_AVC_BSD_OBJECT

MFD_AVC_BSD_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
Description			
<p>The MFD_AVC_BSD_OBJECT command is the only primitive command for the AVC Decoding Pipeline. The same command is used for both CABAC and CAVLD modes. The Slice Data portion of the bitstream is loaded as indirect data object. Before issuing a MFD_AVC_BSD_OBJECT command, all AVC states of the MFD Engine need to be valid. Therefore the commands used to set these states need to have been issued prior to the issue of a MFD_AVC_BSD_OBJECT command.</p>			
Context switch interrupt is not supported by this command.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFD_AVC_BSD_OBJECT
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h AVC_DEC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	1h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	8h
Format:		OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	
	4h	Excludes DWord (0,1) = 0004 [Default]	

MFD_AVC_BSD_OBJECT									
1	31:0	<p>Indirect BSD Data Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">U32</td> </tr> </table> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address. AVC Short Format : It is the length in bytes of the bitstream data for the current slice, including Slice Header + Slice Data + Emulation Prevention Bytes + any filling trailing zeros after the last MB. Hardware ignores the contents after the last non-zero byte. Trailing zero is allowed and handled correctly in both CABAC and CAVLC modes.</p>	Format:	U32					
Format:	U32								
2	31:29	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
28:0	<p>Indirect BSD Data Start Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>U29</td> </tr> </table> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This pointer is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the AVC bitstream data in both CABAC/CAVLD Modes. In implementing a phantom slice at the end of a picture for automatic error concealment, this field should set to 0. It includes the NAL Header (the NAL Header does not need to perform EMU detection). For AVC Base Layer, it is a single byte. But for MVC, the NAL Header is 4 Bytes long. These NAL Header Unit must be passed to HW in the compressed bitstream buffer.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%; text-align: center;">Value</th> <th style="width: 40%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,512MB)</td> <td></td> </tr> </tbody> </table>	Project:	BDW	Format:	U29	Value	Name	[0,512MB)	
Project:	BDW								
Format:	U29								
Value	Name								
[0,512MB)									
3..5	31:0	<p>Inline Data</p> <p>All the required Slice Header parameters and error handling settings are captured as InLine Data of the AVC_BSD_OBJECT command. It has a fixed size of 4 DWs. Its definition is described in the following section: Inline Data Description [BDW].</p>							

MFD_AVC_DPB_STATE

MFD_AVC_DPB_STATE							
Project:	BDW						
Source:	VideoCS						
Length Bias:	2						
<p>This is a frame level state command used only in DXVA2 AVC Short Slice Bitstream Format VLD mode. RefFrameList[16] of DXVA2 interface is replaced with intel Reference Picture Addresses[16] of MFX_PIPE_BUF_ADDR_STATE command. The LongTerm Picture flag indicator of all reference pictures are collected into LongTermPic_Flag[16]. FieldOrderCntList[16][2] and CurrFieldOrderCnt[2] of DXVA2 interface are replaced with intel POCList[34] of MFX_AVC_DIRECTMODE_STATE command.</p>							
DWord	Bit	Description					
0	31:29	Command Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>3h PARALLEL_VIDEO_PIPE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	3h PARALLEL_VIDEO_PIPE	Format:	OpCode	
	Default Value:	3h PARALLEL_VIDEO_PIPE					
	Format:	OpCode					
	28:27	Pipeline <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>2h MFX_MULTI_DW</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	2h MFX_MULTI_DW	Format:	OpCode	
	Default Value:	2h MFX_MULTI_DW					
	Format:	OpCode					
	26:24	Media Command Opcode <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>1h AVC_DEC</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	1h AVC_DEC	Format:	OpCode	
	Default Value:	1h AVC_DEC					
Format:	OpCode						
23:21	SubOpcode A <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>1h</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	1h	Format:	OpCode		
Default Value:	1h						
Format:	OpCode						
20:16	SubOpcode B <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>6h</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	6h	Format:	OpCode		
Default Value:	6h						
Format:	OpCode						
15:12	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
Format:	MBZ						
11:0	DWord Length <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr style="background-color: #e6f2ff;"> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>9h</td> <td>Excludes DWord (0,1) [Default]</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	9h	Excludes DWord (0,1) [Default]
Format:	=n Total Length - 2						
Value	Name						
9h	Excludes DWord (0,1) [Default]						

MFD_AVC_DPB_STATE																	
1	31:16	<p>LongTermFrame_Flag[16][1 bit] One-to-one correspondence with the entries of the Intel RefFrameList[16]. 1 bit per reference frame.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 55%;"></th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td></td> <td>the picture is a long term reference picture</td> </tr> <tr> <td style="text-align: center;">0</td> <td></td> <td>the picture is a short term reference picture</td> </tr> </tbody> </table>	Value	Name		1		the picture is a long term reference picture	0		the picture is a short term reference picture						
	Value	Name															
1		the picture is a long term reference picture															
0		the picture is a short term reference picture															
	15:0	<p>Non-ExistingFrame_Flag[16][1 bit] One-to-one correspondence with the entries of the Intel RefFrameList[16]. 1 bit per reference frame.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>INVALID</td> <td>the reference picture in that entry of RefFrameList[] does not exist anymore.</td> </tr> <tr> <td style="text-align: center;">0</td> <td>VALID</td> <td>the reference picture in that entry of RefFrameList[] is a valid reference</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center;">Programming Notes</p> <p>When an element of the list of frames is not relevant (e.g., due to the corresponding reference entry being empty or being marked as "not used for reference"), the value of the corresponding bit of NonExistingFrameFlags shall be set to 0.</p> </div>	Value	Name	Description	1	INVALID	the reference picture in that entry of RefFrameList[] does not exist anymore.	0	VALID	the reference picture in that entry of RefFrameList[] is a valid reference						
Value	Name	Description															
1	INVALID	the reference picture in that entry of RefFrameList[] does not exist anymore.															
0	VALID	the reference picture in that entry of RefFrameList[] is a valid reference															
2	31:0	<p>UsedForReference_Flag[16][2 bits] One-to-one correspondence with the entries of the Intel RefFrameList[16]. 2 bits per reference frame.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>NOT_REFERENCE</td> <td>indicates a frame is "not used for reference".</td> </tr> <tr> <td style="text-align: center;">1</td> <td>TOP_FIELD</td> <td>bit[0] indicates that the top field of a frame is marked as "used for reference".</td> </tr> <tr> <td style="text-align: center;">2</td> <td>BOTTOM_FIELD</td> <td>bit[1] indicates that the bottom field of a frame is marked as "used for reference".</td> </tr> <tr> <td style="text-align: center;">3</td> <td>FRAME</td> <td>bit[1:0] indicates that a frame (or field pair) is marked as "used for reference".</td> </tr> </tbody> </table>	Value	Name	Description	0	NOT_REFERENCE	indicates a frame is "not used for reference".	1	TOP_FIELD	bit[0] indicates that the top field of a frame is marked as "used for reference".	2	BOTTOM_FIELD	bit[1] indicates that the bottom field of a frame is marked as "used for reference".	3	FRAME	bit[1:0] indicates that a frame (or field pair) is marked as "used for reference".
Value	Name	Description															
0	NOT_REFERENCE	indicates a frame is "not used for reference".															
1	TOP_FIELD	bit[0] indicates that the top field of a frame is marked as "used for reference".															
2	BOTTOM_FIELD	bit[1] indicates that the bottom field of a frame is marked as "used for reference".															
3	FRAME	bit[1:0] indicates that a frame (or field pair) is marked as "used for reference".															
3..10	31:0	<p>LTSTFrameNumList[16][16 bits] One-to-one correspondence with the entries of the Intel RefFrameList[16]. 16 bits per reference frame. Depending on the corresponding LongTermFrame_Flag[], the content of this field is interpreted differently.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>LongTermFrame_Flag[i]</td> <td>LTSTFrameNumList[i] represent LongTermFrameIdx.</td> </tr> <tr> <td style="text-align: center;">0</td> <td>ShortTermFrame_Flag[i]</td> <td>LTSTFrameNumList[i] represent Short Term Picture FrameNum.</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center;">Programming Notes</p> <p>When an element of the list of frames is not relevant (e.g., due to the corresponding reference entry being empty or being marked as "not used for reference"), the value of the LTSTFrameNumList entry shall be set to 0.</p> </div>	Value	Name	Description	1	LongTermFrame_Flag[i]	LTSTFrameNumList[i] represent LongTermFrameIdx.	0	ShortTermFrame_Flag[i]	LTSTFrameNumList[i] represent Short Term Picture FrameNum.						
Value	Name	Description															
1	LongTermFrame_Flag[i]	LTSTFrameNumList[i] represent LongTermFrameIdx.															
0	ShortTermFrame_Flag[i]	LTSTFrameNumList[i] represent Short Term Picture FrameNum.															

		MFD_AVC_DPB_STATE		
11..18 Project: BDW	31:0	<p>ViewIDList[16][16 bits]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>One-to-one correspondence with the entries of the Intel RefFrameList[16]. 16 bits per reference frame. The view ids are 10-bits, the upper 6 bits are ignored."000000" & ViewId1[9:0] & "000000" & ViewId0[9:0]</p> <p style="text-align: center;">Programming Notes</p> <p>When an Intel RefFrameList[i] is not an valid entries, Viewid should be set to 0x00</p>	Project:	BDW
Project:	BDW			
19..22 Project: BDW	31:0	<p>ViewOrderListL0[16][8 bits]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>One-to-one correspondence with the entries of the Intel RefFrameList[16]. 8 bits per reference frame. The view order need 4-bits, the upper 4 bits are ignored. 0000 & ViewOrder3[3:0] & 0000 & ViewOrder2[3:0] & 0000 & ViewOrder1[3:0] & 0000 & ViewOrder0[3:0]</p> <p style="text-align: center;">Programming Notes</p> <p>When the ViewOrderListL0[i] is not an valid inter-view reference, its corresponding ViewOrder should be set to 0xF</p> <p>Since only interview with the same polarity will be used, there is no need to have field bit in this list. Hardware is going to append correct polarity bit as needed.</p>	Project:	BDW
Project:	BDW			
23..26 Project: BDW	31:0	<p>ViewOrderListL1[16][8 bits]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>One-to-one correspondence with the entries of the Intel RefFrameList[16]. 8 bits per reference frame. The view order need 4-bits, the upper 4 bits are ignored. 0000 & ViewOrder3[3:0] & 0000 & ViewOrder2[3:0] & 0000 & ViewOrder1[3:0] & 0000 & ViewOrder0[3:0]</p> <p style="text-align: center;">Programming Notes</p> <p>When the ViewOrderListL1[i] is not an valid inter-view reference, its corresponding ViewOrder should be set to 0xF</p> <p>Since only interview with the same polarity will be used, there is no need to have field bit in this list. Hardware is going to append correct polarity bit as needed.</p>	Project:	BDW
Project:	BDW			

MFD_AVC_PICID_STATE

MFD_AVC_PICID_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a frame level state command used for both AVC Long and Short Format in VLD mode. PictureID[16] contains the pictureID of each reference picture (16 maximum) so hardware can uniquely identify the reference picture across frames (this will be used for DMV operation). This command will be needed for both short and long format.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MULTI_DW
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h MFD_AVC_DPB_STATE
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		1h DEC	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	5h MEDIA_	
	Format:	OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0008h Excludes DWord (0,1)	
	Project:	BDW	
	Format:	=n Total Length - 2	
1	31:1	Reserved	
		Project:	All
		Format:	MBZ

MFD_AVC_PICID_STATE			
	0	PictureID Remapping Disable	
	Project: BDW		
	Value	Name	Description
	0h	AVC decoder will use 16 bits Picture ID to handle DMV and identify the reference picture	Desc
	1h	AVC decoder will use 4 bits FrameStoreID (index to RefFrameList) to handle DMV and identify the reference picture.	Desc
Programming Notes			
If Picture ID Remapping Disable is "1", PictureIDList will not be used.			
2..9	31:0	PictureIDList[16][16 bits]	
Project: BDW			
<p>One-to-one correspondence with the entries of the Intel RefFrameList[16]. 16 bits per reference frame. PictureID of each Frame uniquely identifies the reference picture across frames. The same number cannot be reused until the reference picture is completely retired (no longer used for reference). When an element of the list of frames is not relevant (e.g., due to the corresponding reference entry being empty or being marked as "not used for reference"), the value of the LTSTFrameNumList entry shall be set to 0.</p>			

MFD_AVC_SLICEADDR

MFD_AVC_SLICEADDR	
Project:	BDW
Source:	VideoCS
Length Bias:	2

This is a Slice level command used only for DXVA2 AVC Short Slice Bitstream Format VLD mode. When decoding a slice, H/W needs to know the last MB of the slice has reached in order to start decoding the next slice. It also needs to know if a slice is terminated but the last MB has not reached, error concealment should be invoked to generate those missing MBs. For AVC DXVA2 Short Format, the only way to know the last MB position of the current slice, H/W needs to snoop into the next slice's start MB address (a linear address encoded in the Slice Header). Since each BSD Object command can have only one indirect bitstream buffer address, this command is added to help H/W to snoop into the next slice's slice header and retrieve its Start MB Address. This command will take the next slice's bitstream buffer address as input (exactly the same way as a BSD Object command), and parse only the first_mb_in_slice syntax element. The result will be stored inside the H/W, and will be used to decode the current slice specified in the BSD Object command. Only the very first few bytes (max 5 bytes for a max 4K picture) of the Slice Header will be decoded, the rest of the bitstream are don't care. This is because the first_mb_in_slice is encoded in Exponential Golomb, and will take 33 bits to represent the max 256 x 256 = 64K-1 value. The indirect data of MFD_AVC_SLICEADDR is a valid BSD object and is decoded as in BSD OBJECT command. The next Slice Start MB Address is also exposed to the MMIO interface. The Slice Start MB Address (first_mb_in_slice) is a linear MB address count; but it is translated into the corresponding 2D MB X and Y raster position, and are stored internally as NextSliceMbY and NextSliceMbX.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFD_AVC_SLICEADDR
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h AVC_DEC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	1h
		Format:	OpCode
20:16	SubOpcode B		
	Default Value:	7h	
	Format:	OpCode	
15:12	Reserved		
	Format:	MBZ	

MFD_AVC_SLICEADDR		
	11:0	DWord Length
	Format: =n Total Length - 2	
	Value	Name
	1h	Excludes DWord (0,1) [Default]
1	31:0	Indirect BSD Data Length
		Project: BDW
		Format: U32
<p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. Driver always programs this up to 5 bytes; for bitstream less than 5 bytes, driver program the lesser value. (Emulation Prevention Byte should never happen for the first 5 bytes when the max picture size can only be 4Kx4K)It is the length in bytes of the bitstream data for the current slice, including Slice Header + Slice Data + Emulation Prevention Bytes + any filling trailing zeros after the last MB. Hardware ignores the contents after the last non-zero byte. Trailing zero is allowed and handled correctly in both CABAC and CAVLC modes.</p>		
2	31:29	Reserved
		Format: MBZ
	28:0	Indirect BSD Data Start Address
<p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This pointer is relative to the MFD Indirect Object Base Address.Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the AVC bitstream data in both CABAC/CAVLD Modes.In implementing a phantom slice at the end of a picture for automatic error concealment, this field should set to 0.It includes the NAL Header Byte. (but does not perform EMU detection).Must provide a valid MB address, even if error. MB must be clamped to within a pic boundary.</p>		
Value		Name
[0,512MB)		

MFD_IT_OBJECT

MFD_IT_OBJECT		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
All weight mode (default and implicit) are mapped to explicit mode. But the weights come in either as explicit or implicit.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h PARALLEL_VIDEO_PIPE
		Format: OpCode
	28:27	Pipeline
		Default Value: 2h MFD_IT_OBJECT
		Format: OpCode
	26:24	Media Command Opcode
		Default Value: 0h MFX_COMMON_DEC
		Format: OpCode
	23:21	SubOpcode A
Default Value: 1h		
Format: OpCode		
20:16	SubOpcode B	
	Default Value: 9h	
	Format: OpCode	
15:12	Reserved	
	Format: MBZ	
11:0	DWord Length	
	Default Value: 06h Excludes DWord (0,1) For AVC = Ch	
	Format: =n Total Length - 2 Note: Regardless of the mode, inline data must be present in this command.	
1	31:10	Reserved
		Format: MBZ

MFD_IT_OBJECT							
	9:0	<p>Indirect IT-MV Data Length</p> <table border="1"> <tr> <td>Format:</td> <td>U10 FormatDesc: In bytes</td> </tr> </table> <p>This field provides the length in bytes of the indirect data, which contains all the MVs for the current MB (in any partitioning and subpartitioning form). A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect IT-MV Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address.AVC-IT Mode: It must be DWord aligned (since each MV is 4bytes in size)Driver has to derived this field from MVsize (MVquantity in DXVA, exact size) *4 bytes per MV.This field is only valid in AVC decoder IT mode (VC1 and MPEG uses inline MV data).</p>	Format:	U10 FormatDesc: In bytes			
	Format:	U10 FormatDesc: In bytes					
2	<p>31:29 Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>28:0 Indirect IT-MV Data Start Address Offset</p> <p>This field specifies the memory starting address (offset) of the MV data to be fetched into the IT pipeline for processing. This pointer is relative to the Indirect IT-MV Object Base Address.Hardware ignores this field if indirect data is not present, i.e. the Indirect MV Data Length is set to 0. Alignment of this address depends on the mode of operation.AVC-IT Mode: It must be DWord aligned (since each MV is 4 bytes in size). This field is only valid in AVC decoder IT mode (VC1 and MPEG uses inline MV data).</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,512MB)</td> <td></td> </tr> </tbody> </table>	Format:	MBZ	Value	Name	[0,512MB)	
Format:	MBZ						
Value	Name						
[0,512MB)							
3	31:12	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
	Format:	MBZ					
11:0	<p>Indirect IT-COEFF Data Length</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <p>This field provides the length in bytes of the indirect data, which contains all the non-zero coefficients for the current MB. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect IT-COEFF Data Start Address field is ignored. Since each IT-COEFF data is 1 DW in size, with 12 bits, this field can be extended to support up to 4:4:4 format.(256 pixel * 3 byte pixel components * 4 bytes per coeff).This field must be integer multiple of 16-bytes for AVC (since each coefficient is 4 bytes in size).This field is only valid in AVC, VC1, MPEG2 decoder IT mode.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,3072]</td> <td>In bytes [0, 256*3*4]</td> </tr> </tbody> </table>	Project:	All	Value	Name	[0,3072]	In bytes [0, 256*3*4]
Project:	All						
Value	Name						
[0,3072]	In bytes [0, 256*3*4]						
4	31:29	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ						

MFD_IT_OBJECT								
	28:0	<p>Indirect IT-COEFF Data Start Address Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> </table> <p>This field specifies the memory starting address (offset) of the coeff data to be loaded into the IT pipeline for processing. This pointer is relative to the Indirect IT-COEFF Object Base Address. Hardware ignores this field if indirect IT-COEFF data is not present, i.e. the Indirect IT-COEFF Data Length is set to 0. This field must be DW aligned, since each coefficient is 4 bytes in size. Driver will determine the Num of EOB 4x4/8x8 must match the block cbp flags, if not match, hardware cannot hang - add error handling. This field is only valid in AVC, VC1, MPEG2 decoder IT mode.</p> <table border="1" style="width: 100%; text-align: center;"> <tr> <th style="width: 60%;">Value</th> <th>Name</th> </tr> <tr> <td>[0,512MB)</td> <td></td> </tr> </table>	Project:	All	Value	Name	[0,512MB)	
	Project:	All						
Value	Name							
[0,512MB)								
5	31:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
5:0	<p>Indirect IT-DBLK Control Data Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U6</td> </tr> </table> <p>This field provides the length in bytes of the indirect data, which contains all the deblocker control information for the current MB (in 4x4 sub-block partitioning). A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect IT-DBLK Data Start Address field is ignored. This field must have the same alignment as the Indirect IT-DBLK Data Start Address. It must be DWord aligned. Each Deblock Control Data record is 48 bytes or 12 DWords in size. This field is only valid in AVC decoder IT mode.</p>	Project:	All	Format:	U6			
Project:	All							
Format:	U6							
6	31:29	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
28:0	<p>Indirect IT-DBLK Control Data Start Address Offset</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>IndirectObjectBaseAddress[28:0]</td> </tr> </table> <p>This field specifies the memory starting address (offset) of the Deblocker control data to be fetched into the IT Pipeline for processing. This pointer is relative to the Indirect IT-DBLK Object Base Address. Hardware ignores this field if indirect data is not present, ie. The indirect IT-DBLK Control Data Length is set to 0. It must be DWord aligned. Each Deblock Control Data record is 48 bytes or 12 DWords in size. This field is only valid in AVC decoder IT mode.</p> <table border="1" style="width: 100%; text-align: center;"> <tr> <th style="width: 60%;">Value</th> <th>Name</th> </tr> <tr> <td>[0,512MB)</td> <td></td> </tr> </table>	Format:	IndirectObjectBaseAddress[28:0]	Value	Name	[0,512MB)		
Format:	IndirectObjectBaseAddress[28:0]							
Value	Name							
[0,512MB)								
7..n	31:0	<p>Inline Data</p> <p>Union for all 3 codecs Includes IT, MC, IntraPred inline data as well as Deblocker control information AVC-IT Modes: Hardware interprets this data in the specified format. VC1-IT Modes: Hardware interprets this data in the specified format. MV inline MPEG2-IT Modes: Hardware interprets this data in the specified format. (IS mode) MV inline For AVC there 7 DWords of inline data, hence N is equal to 13.</p>						

MFD_JPEG_BSD_OBJECT

MFD_JPEG_BSD_OBJECT		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
Exists If:	//Decoder	
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h PARALLEL_VIDEO_PIPE Format: OpCode
	28:27	Pipeline
		Default Value: 2h MFD_JPEG_BSD_OBJECT Format: OpCode
	26:24	Media Command Opcode
		Default Value: 7h JPEG_DEC Format: OpCode
	23:21	SubOpcode A
		Default Value: 1h Format: OpCode
	20:16	SubOpcode B
		Default Value: 8h Format: OpCode
	15:12	Reserved
		Project: All Format: MBZ
11:0	DWord Length	
	Default Value: 004h Excludes DWord (0,1) Project: All Format: =n Total Length - 2	
1	31:0	Indirect Data Length Project: All . It is the length in bytes of the bitstream data for the current Scan. It includes the first byte of the first MCU and the last non-zero byte of the last MCU in the Scan. Specifically, the zero-padding bytes (if present) are excluded. Hardware ignores the contents after the last non-zero byte.

MFD_JPEG_BSD_OBJECT									
2	31:29	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ			
Project:	All								
Format:	MBZ								
	28:0	Indirect Data Start Address							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>IndirectObjectOffset[28:0]</td> </tr> </table> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This pointer is relative to the BSD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the JPEG bitstream data</p>	Project:	All	Format:	IndirectObjectOffset[28:0]			
Project:	All								
Format:	IndirectObjectOffset[28:0]								
3	31:29	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ			
Project:	All								
Format:	MBZ								
	28:16	Scan Horizontal Position							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U13 bits in blocks</td> </tr> </table> <p>This field indicates the horizontal position (in block units) of the first MCU in the Scan.</p>	Project:	All	Format:	U13 bits in blocks			
Project:	All								
Format:	U13 bits in blocks								
	15:13	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ			
Project:	All								
Format:	MBZ								
	12:0	Scan Vertical Position							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U13 bits in blocks</td> </tr> </table> <p>This field indicates the vertical position (in block units) of the first MCU in the Scan.</p>	Project:	All	Format:	U13 bits in blocks			
Project:	All								
Format:	U13 bits in blocks								
4	31	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ								
	30	Interleaved							
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 35%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Non-Interleaved</td> <td>one component in the Scan</td> </tr> <tr> <td>1</td> <td>Interleaved</td> <td>multiple components in the Scan</td> </tr> </tbody> </table>	Value	Name	Description	0	Non-Interleaved	one component in the Scan	1
Value	Name	Description							
0	Non-Interleaved	one component in the Scan							
1	Interleaved	multiple components in the Scan							
	29:27	Scan Components Bit0: Y Bit1: U Bit2: V For example, if non-interleaved Y, then it will be set to 001b. If interleaved Y, U, and V, it will be set to 111b.							
	26	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ								
	25:0	MCU Count							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U26</td> </tr> </table> <p>This field indicates the number of MCUs in the Scan.</p>	Project:	All	Format:	U26			
Project:	All								
Format:	U26								

MFD_JPEG_BSD_OBJECT		
5	31:16	Reserved
		Project: All
		Format: MBZ
	15:0	RestartInterval(16 bit)
		Project: All
		Format: U16
Specifies the number of MCU in restart interval. Valid values are 1->0xFFFF Value of 0 implies that all the SCAN have only one ECS.		

MFD_MPEG2_BSD_OBJECT

MFD_MPEG2_BSD_OBJECT				
Project:	BDW			
Source:	VideoCS			
Length Bias:	2			
<p>Different from AVC and VC1, MFD_MPEG2_BSD_OBJECT command is pipelinable. This is for performance purpose as in MPEG2 a slice is defined as a group of MBs of any size that must be within a macroblock row. Slice header parameters are passed in as inline data and the bitstream data for the slice is passed in as indirect data. Of the inline data, slice_horizontal_position and slice_vertical_position determines the location within the destination picture of the first macroblock in the slice. The content in this command is identical to that in the MEDIA_OBJECT command in VLD mode described in the Media Chapter.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h PARALLEL_VIDEO_PIPE	
		Format:	OpCode	
	28:27	28:27	Pipeline	
			Default Value:	2h MFD_MPEG2_BSD_OBJECT
			Format:	OpCode
	26:24	26:24	Media Command Opcode	
			Default Value:	3h MPEG2_DEC
			Format:	OpCode
	23:21	23:21	SubOpcode A	
			Default Value:	1h
			Format:	OpCode
	20:16	20:16	SubOpcode B	
			Default Value:	8h
Format:			OpCode	
15:12	15:12	Reserved		
		Project:	All	
		Format:	MBZ	
11:0	11:0	DWord Length		
		Default Value:	0003h Excludes DWord (0,1)	
		Project:	All	
		Format:	=n Total Length - 2	

MFD_MPEG2_BSD_OBJECT									
1	31:0	Indirect BSD Data Length <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>It is the length in bytes of the bitstream data for the current slice. It includes the first byte of the first macroblock and the last non-zero byte of the last macroblock in the slice. Specifically, the zero-padding bytes (if present) and the next start-code are excluded. This field is sized to support beyond MPEG-2 MP@HL bitstream (<4K). According to Table 8-6 of ISO/IEC 13818-2, the maximum number of bits per macroblock for 4:2:0 is 4608. So the maximum slice size for 4K x 4K is $4608 * 256 / 8 = 147,456$ bytes (0x24000), which requires 18 bits.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th style="text-align: center; background-color: #e1eef6;">Programming Notes</th> </tr> <tr> <td>As MPEG-2 spec does not post any limitation of the size of zero-padding bytes, it is possible to have a slice data with large length (including zero-padding bytes). As the data beyond 0x10E00 would only be zero bytes for a valid slice data</td> </tr> <tr> <td>zero-padding restriction is removed</td> </tr> </table>	Project:	All	Format:	U32	Programming Notes	As MPEG-2 spec does not post any limitation of the size of zero-padding bytes, it is possible to have a slice data with large length (including zero-padding bytes). As the data beyond 0x10E00 would only be zero bytes for a valid slice data	zero-padding restriction is removed
		Project:	All						
		Format:	U32						
		Programming Notes							
As MPEG-2 spec does not post any limitation of the size of zero-padding bytes, it is possible to have a slice data with large length (including zero-padding bytes). As the data beyond 0x10E00 would only be zero bytes for a valid slice data									
zero-padding restriction is removed									
2	31:29	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ			
		Project:	All						
Format:	MBZ								
28:0	Indirect Data Start Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>IndirectObjectOffset[28:0]</td> </tr> </table> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This pointer is relative to the BSD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the MPEG2 VLD bitstream data This address points to the first byte of the MB layer data, i.e. not including slice header.</p>	Format:	IndirectObjectOffset[28:0]						
Format:	IndirectObjectOffset[28:0]								
3..4	31:0	Inline Data <p>All the required Slice Header parameters and error handling settings are captured as MPEG2_BSD_OBJECT Inline Data Descriptor structures. It has a fixed size of 2 DWs. Its definition is described in the next section.</p>							

MFD_VC1_BSD_OBJECT

MFD_VC1_BSD_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFD_VC1_BSD_OBJECT command is the only primitive command for the VC1 Decoding Pipeline. The macroblock data portion of the bitstream is loaded as indirect data object. Before issuing a MFD_VC1_BSD_OBJECT command, all VC1 states of the MFD Engine need to be valid. Therefore the commands used to set these states need to have been issued prior to the issue of a MFD_VC1_BSD_OBJECT command. VC1 deblock filter kernel cross the slice boundary if in the last MB row of a slice, so need to know the last MB row of a slice to disable the edge mask. There is why VC1 BSD hardware need to know the end of MB address for the current slice. As such no more phantom slice is needed for VC1, as long as the driver will program both start MB address in the current slice and the start MB address of the next slice. As a result, we can also support multiple picture state commands in between slices.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MULTI_DW
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	2h VC1_DEC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	1h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	8h
		Format:	OpCode
	15:12	Reserved	
		Project:	All
		Format:	MBZ
	11:0	DWord Length	
		Default Value:	0003h Excludes DWord (0,1)
		Project:	All
		Format:	=n Total Length - 2

MFD_VC1_BSD_OBJECT								
1	31:24	Reserved						
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
	Project:	All						
	Format:	MBZ						
23:0	Indirect BSD Data Length							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U24</td> </tr> </table> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address. GEN6 Long Format : It is the length in bytes of the bitstream data for the current slice/picture. It includes the first byte of the first macroblock and the last byte of the last macroblock in the slice/picture. Specifically, the zero-padding bytes (if present) and the next start-code are excluded. Hardware ignores the contents after the last non-zero byte (trailing zeros). This field is sized to support VC1 AP@L4 Level bitstream. It includes the byte that contains the First MB Bit Offset. GEN7 Short Format : It is the length in bytes of the bitstream data for the current slice, including Picture/Slice Header + Emulation Prevention Bytes + any filling trailing zeros after the last MB. Hardware ignores the contents after the last non-zero byte. Trailing zero is allowed and handled correctly.</p>	Project:	All	Format:	U24			
Project:	All							
Format:	U24							
2	31:29	Reserved						
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
	Project:	All						
	Format:	MBZ						
28:0	Indirect Data Start Address							
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>IndirectObjectOffset[28:0]</td> </tr> </table> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This pointer is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VC1 bitstream data.</p> <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="width: 60%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>[0,512MB)</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	IndirectObjectOffset[28:0]	Value	Name	[0,512MB)
Project:	All							
Format:	IndirectObjectOffset[28:0]							
Value	Name							
[0,512MB)								
3	31:24	Reserved						
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
	23:16	Slice Start Vertical Position						
15:9	Reserved							
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>This field specifies the position in y-direction of the first macroblock in the Slice in unit of macroblocks. For SecondField this value is reset to zero as opposed to the VC1 spec Ref: 9.1.2 Slice Layer. This field is for both Long and Short VC1 Interface Format.</p>	Project:	All	Format:	MBZ		
Project:	All							
Format:	MBZ							



MFD_VC1_BSD_OBJECT													
	8:0	<p>Next Slice Vertical Position</p> <p>This field specifies the position in y-direction of the first macroblock in the next Slice in unit of macroblocks. 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 picture (since y-direction is zero-based numbering) This field is maintained and provided by the driver for both Long and Short VC1 Interface Format.</p>											
4	31:16	<p>First_MB_Byte_Offset of Slice Data or Slice Header</p> <p>For DXVA2 VC1 Short Format only It gives the byte offset to locate the first MB data in the bitstream for a slice, relative to the Indirect BSD Data Start Address.</p>											
	15:5	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ							
	Project:	All											
	Format:	MBZ											
	4	<p>Emulation Prevention Byte Present</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 60%;">Description</th> <th style="width: 10%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>H/W needs to perform Emulation Byte Removal</td> <td>All</td> </tr> <tr> <td>1h</td> <td></td> <td>H/W does not need to perform Emulation Byte Removal</td> <td>All</td> </tr> </tbody> </table>	Value	Name	Description	Project	0h		H/W needs to perform Emulation Byte Removal	All	1h		H/W does not need to perform Emulation Byte Removal
Value	Name	Description	Project										
0h		H/W needs to perform Emulation Byte Removal	All										
1h		H/W does not need to perform Emulation Byte Removal	All										
3	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ								
Project:	All												
Format:	MBZ												
2:0	<p>FirstMbBitOffset (First Macroblock Bit Offset)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>U3</td> </tr> </table> <p>This field provides the bit offset of the first macroblock of the Slice in the first byte of the input compressed bitstream. It is used with First_MB_Byte_Offset for non-byte aligned position.</p>	Format:	U3										
Format:	U3												

MFD_VC1_LONG_PIC_STATE

MFD_VC1_LONG_PIC_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>MFX_VC1_LONG_PIC_STATE command encapsulates the decoding parameters that are read or derived from bitstream syntax elements above (inclusive) picture header layer. These parameters are static for a picture and when slice structure is present, these parameters are not changed from slice to slice of the same picture. Hence, this command is only issued at the beginning of processing a new picture and prior to the VC1_*_OBJECT command. The values set for these state variables are retained internally across slices. Only the parameters needed by hardware (BSD unit) to decode bit sequence for the macroblocks in a picture layer or a slice layer are presented in this command. Other parameters such as the ones used for inverse transform or motion compensation are provided in MFX_VC1_PRED_PIPE_STATE command. This Long interface format is intel proprietary interface. Driver will need to perform addition operations to generate all the fields in this command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFD_VC1_LONG_PIC_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	2h VC1_DEC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	1h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	1h
Format:		OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0004h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	

MFD_VC1_LONG_PIC_STATE												
1	31:24	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
	Format:	MBZ										
	23:16	PictureHeightInMBsMinus1 (Picture Height Minus 1 in Macroblocks) <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>This field indicates the height of the picture in unit of macroblocks. For example, for a 1920x1080 frame picture, PictureHeightInMBs equals 68 (1080 divided by 16, and rounded up, i.e. effectively specified as 1088 instead). This field is used in VLD and IT modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,254]</td> <td>Value_0_to_254</td> <td>a valid range of [0,254] [1, 255] MB</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Note: Even though the Advanced Profile allows frame dimensions (width, height) to not be aligned to macroblock boundary, it doesn't affect the bitstream decoding. And it is preferable to use 'intermediate buffer' that is macroblock aligned for decoding. In order to simplify the out-of-bound reference pixel access, the out-of-bound extrapolation rule in VC1 spec can be used to expand the expected decoded frame to the intermediate buffer dimension.</p>	Project:	BDW	Format:	U8	Value	Name	Description	[0,254]	Value_0_to_254	a valid range of [0,254] [1, 255] MB
	Project:	BDW										
	Format:	U8										
	Value	Name	Description									
	[0,254]	Value_0_to_254	a valid range of [0,254] [1, 255] MB									
	15:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
Format:	MBZ											
7:0	PictureWidthInMBsMinus1 (Picture Width Minus 1 in Macroblocks) <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U8-1</td> </tr> </table> <p>This field indicates the width of the picture in unit of macroblocks. For example, for a 1920x1080 frame picture, PictureWidthInMBs equals 120 (1920 divided by 16). This field is used in VLD and IT modes</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,254]</td> <td>Value_0_to_254</td> <td>[1,255] MB</td> </tr> </tbody> </table>	Project:	BDW	Format:	U8-1	Value	Name	Description	[0,254]	Value_0_to_254	[1,255] MB	
Project:	BDW											
Format:	U8-1											
Value	Name	Description										
[0,254]	Value_0_to_254	[1,255] MB										

MFD_VC1_LONG_PIC_STATE											
2	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center; vertical-align: top;">31:24</td> <td> <p>Bitplane Buffer Pitch Minus 1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U7-1 Pitch in (Bytes - 1).</td> </tr> </table> <p>Specifies the bitplane buffer pitch in (#Bytes - 1). Bitplane buffer is a linear buffer. It is needed only when the bitplane is not encoded as raw, and therefore is present in the header explicitly. In VC1 Long Format (Gen6 and Gen7), it is written by an application and later read by the HW. But in VC1 Short Format (Gen7 only), it is written and read by H/W only. This field is specified for better performance</p> <table border="1" style="width: 100%; text-align: center;"> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> <tr> <td>[0, FFFFFFFFh]</td> <td></td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <p>For Gen6 : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Long Format : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Short Format : If Pic Width is less than or equal to 2K pixels, bitplane pitch is set to 64 (one cacheline; programmed as 63) bytes per MB row. If Pic Width is greater than 2K pixels, bitplane pitch is set to 128 (two cachelines; programmed as 127) bytes per MB row. This field is not used in IT mode, used in VLD mode only. For VC1 DXVA2 Short Format, the bitplane specification is between H/W and Driver only. For Long Format, application is responsible for allocation with the driver.</p> </td> </tr> </table>	31:24	<p>Bitplane Buffer Pitch Minus 1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U7-1 Pitch in (Bytes - 1).</td> </tr> </table> <p>Specifies the bitplane buffer pitch in (#Bytes - 1). Bitplane buffer is a linear buffer. It is needed only when the bitplane is not encoded as raw, and therefore is present in the header explicitly. In VC1 Long Format (Gen6 and Gen7), it is written by an application and later read by the HW. But in VC1 Short Format (Gen7 only), it is written and read by H/W only. This field is specified for better performance</p> <table border="1" style="width: 100%; text-align: center;"> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> <tr> <td>[0, FFFFFFFFh]</td> <td></td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <p>For Gen6 : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Long Format : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Short Format : If Pic Width is less than or equal to 2K pixels, bitplane pitch is set to 64 (one cacheline; programmed as 63) bytes per MB row. If Pic Width is greater than 2K pixels, bitplane pitch is set to 128 (two cachelines; programmed as 127) bytes per MB row. This field is not used in IT mode, used in VLD mode only. For VC1 DXVA2 Short Format, the bitplane specification is between H/W and Driver only. For Long Format, application is responsible for allocation with the driver.</p>	Project:	BDW	Format:	U7-1 Pitch in (Bytes - 1).	Value	Name	[0, FFFFFFFFh]	
31:24	<p>Bitplane Buffer Pitch Minus 1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U7-1 Pitch in (Bytes - 1).</td> </tr> </table> <p>Specifies the bitplane buffer pitch in (#Bytes - 1). Bitplane buffer is a linear buffer. It is needed only when the bitplane is not encoded as raw, and therefore is present in the header explicitly. In VC1 Long Format (Gen6 and Gen7), it is written by an application and later read by the HW. But in VC1 Short Format (Gen7 only), it is written and read by H/W only. This field is specified for better performance</p> <table border="1" style="width: 100%; text-align: center;"> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> <tr> <td>[0, FFFFFFFFh]</td> <td></td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <p>For Gen6 : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Long Format : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Short Format : If Pic Width is less than or equal to 2K pixels, bitplane pitch is set to 64 (one cacheline; programmed as 63) bytes per MB row. If Pic Width is greater than 2K pixels, bitplane pitch is set to 128 (two cachelines; programmed as 127) bytes per MB row. This field is not used in IT mode, used in VLD mode only. For VC1 DXVA2 Short Format, the bitplane specification is between H/W and Driver only. For Long Format, application is responsible for allocation with the driver.</p>	Project:	BDW	Format:	U7-1 Pitch in (Bytes - 1).	Value	Name	[0, FFFFFFFFh]			
Project:	BDW										
Format:	U7-1 Pitch in (Bytes - 1).										
Value	Name										
[0, FFFFFFFFh]											
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center; vertical-align: top;">23:16</td> <td> <p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> </td> </tr> </table>	23:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
23:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
Project:	BDW										
Format:	MBZ										
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center; vertical-align: top;">15</td> <td> <p>DmvSurfaceValid</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Indicated when the DMV read surface is valid. This surface stored the direct motion vectors and Mb type. This field is set for B pictures that can refer to a previous P picture for DMV. If there is an I-picture before a B (in decoding order) then this field is not set (as a result, zero's DMV's will be assumed while decoding the B picture. That is, there is no explicit DMV buffer for an I-picture). When the current picture being decoded is an I, P or BI, this bit is set to 0, since there is no DMV read in these picture decoding process. This field is not used in IT mode, used in VLD mode only.</p> </td> </tr> </table>	15	<p>DmvSurfaceValid</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Indicated when the DMV read surface is valid. This surface stored the direct motion vectors and Mb type. This field is set for B pictures that can refer to a previous P picture for DMV. If there is an I-picture before a B (in decoding order) then this field is not set (as a result, zero's DMV's will be assumed while decoding the B picture. That is, there is no explicit DMV buffer for an I-picture). When the current picture being decoded is an I, P or BI, this bit is set to 0, since there is no DMV read in these picture decoding process. This field is not used in IT mode, used in VLD mode only.</p>	Project:	BDW						
15	<p>DmvSurfaceValid</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Indicated when the DMV read surface is valid. This surface stored the direct motion vectors and Mb type. This field is set for B pictures that can refer to a previous P picture for DMV. If there is an I-picture before a B (in decoding order) then this field is not set (as a result, zero's DMV's will be assumed while decoding the B picture. That is, there is no explicit DMV buffer for an I-picture). When the current picture being decoded is an I, P or BI, this bit is set to 0, since there is no DMV read in these picture decoding process. This field is not used in IT mode, used in VLD mode only.</p>	Project:	BDW								
Project:	BDW										
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center; vertical-align: top;">14</td> <td> <p>ImplicitQuantizer</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Derived by driver from QUANTIZER. This field is used in intel VC1 VLD Long Format only, not used in IT and DXVA2 VC1. This bit is set to 1 when syntax element QUANTIZER=0, else its set to 0</p> </td> </tr> </table>	14	<p>ImplicitQuantizer</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Derived by driver from QUANTIZER. This field is used in intel VC1 VLD Long Format only, not used in IT and DXVA2 VC1. This bit is set to 1 when syntax element QUANTIZER=0, else its set to 0</p>	Project:	BDW						
14	<p>ImplicitQuantizer</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <p>Derived by driver from QUANTIZER. This field is used in intel VC1 VLD Long Format only, not used in IT and DXVA2 VC1. This bit is set to 1 when syntax element QUANTIZER=0, else its set to 0</p>	Project:	BDW								
Project:	BDW										

MFD_VC1_LONG_PIC_STATE			
13	Interpolation Rounder Contro		
	Project:	BDW	
	Used only in MC operation. This field specifies the rounding control value used in interpolation operation of motion prediction process.This field is used in VLD and IT modes.		
Programming Notes			
This bit field is taken from bRcontrol in DXVA_PictureParameters data structure			
12	SyncMarker		
	Project:	BDW	
	Indicates whether sync markers are enabled/disabled. If enable, sync markers "may be" present in the current video sequence being decoded. It is a sequence level syntax element and is valid only for Simple and Main Profiles.		
	Value	Name	Description
	0h	Not Present	Sync Marker is not present in the bitstream
1h	Maybe present	Sync Marker maybe present in the bitstream	
Programming Notes			
This field is only valid in VLD mode.For Simple Profile, SyncMarker must set to 0.For Main Profile, SyncMarker can be set to 0 or 1.This field is used in both intel and MS DXVA2 VLD interface, but not used in IT mode.			
11:8	Motion Vector Mode		
	Project:	BDW	
	This field indicates one of the following motion compensation interpolation modes for P and B pictures. The MC interpolation modes apply to prediction values of luminance blocks and are always in quarter-sample. For chrominance blocks, it always performs bilinear interpolation with either half-pel or quarter-pel precision.Before the polarity of Chroma Half-pel or Q-pel is reversed from DXVA2 Spec, now I have fixed it to match with DXVA2 VC1 Spec.		
	Value	Name	Description
	0XX0b		Chroma Quarter -pel + Luma bicubic. (can only be 1MV)
	0XX1b		Chroma Half-pel + Luma bicubic. (can be 1MV or 4MV)
1XX0b		Chroma Quarter -pel + Luma bilinear. (can only be 1MV)	
1XX1b		Chroma Half-pel + Luma bilinear	
Programming Notes			
Bits 11:8 are taken from bMVprecisionAndChromaRelation in DXVA_PictureParameters data structure.Bit 11 of Motion Vector Mode = 1 for Luma Bilinear MC; = 0 for Luma Bicubic MCBit 8 of Motion Vector Mode = 1 for half-sample Chroma motion = 0 for quarter-sample Chroma motion.This field is used in both VLD and IT modes.			

MFD_VC1_LONG_PIC_STATE

7 RangeReductionScale

Project:	BDW
----------	-----

This field specifies whether the reference picture pixel values should be scaled up or scaled down on-the-fly, if RangeReduction is Enabled.

Value	Name	Description
0h		Scale down reference picture by factor of 2
1h		Scale up reference picture by factor of 2

Programming Notes

This bit is derived by driver for Main Profile only. Ignored in Simple and Advanced Profiles. This field is used in both VLD and IT modes. This is derived by driver from the history of RANGERED and RANGEREDFRM syntax elements (i.e. of forward/preceding reference picture) and those of the current picture. RANGERED is the same as (bPicOverflowBlocks » 3) & 1. RANGEREDFRM is the same as (bPicDeblocked » 5) & 1. For the current picture is a B picture, this field represents the state of the forward/preceding reference picture only. Driver is responsible to keep RangeReductionScale, RangeReduction Enable and RANGERED Present Flag of current picture coherent.

6 RangeReduction Enable

Project:	BDW
----------	-----

This field specifies whether on-the-fly pixel value range reduction should be performed for the preceding (or forward) reference picture. Along with RangeReductionScale to specify whether scale up or down should be performed. It is not the same value as RANGEREDFRM Syntax Element (DXVA_PictureParameters bPicDeblocked bit 5) in the Picture Header.

Value	Name	Description
0h	Disable	Range reduction is not performed
1h	Enable	Range reduction is performed

Programming Notes

This field is for Main Profile only. Simple Profile is always disable, and not applicable to Advanced Profile. This field is used in both VLD and IT modes. This is derived by driver from the history of RANGERED and RANGEREDFRM syntax elements (i.e. of forward/preceding reference picture) and those of the current picture. RANGERED is the same as (bPicOverflowBlocks » 3) & 1. RANGEREDFRM is the same as (bPicDeblocked » 5) & 1. For the current picture is a B picture, this field represents the state of the forward/preceding reference picture only. Driver is responsible to keep RangeReductionScale, RangeReduction Enable and RANGERED Present Flag of current picture coherent.

MFD_VC1_LONG_PIC_STATE											
3	5	<p>LOOPFILTER Enable Flag This field is the decoded syntax element LOOPFILTER in bitstream. It indicates if In-loop Deblocking is ON according to picture level bitstream syntax control. This bit affects BSD unit and also the loop filter unit. When this bit is set to 1, PostDeblockOutEnable field in MFX_PIPE_MODE_SELECT command must also be set to 1. In this case, in-loop deblocking operation follows the VC1 standard - deblocking doesn't cross slice boundary. When this bit is set to 0, but PostDeblockOutEnable field in MFX_PIPE_MODE_SELECT command is set to 1. This field is used in VLD mode only, not in IT mode.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Disables loop filter</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Enables loop filter</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	Disables loop filter	1h	Enable	Enables loop filter
	Value	Name	Description								
	0h	Disable	Disables loop filter								
	1h	Enable	Enables loop filter								
	4	<p>Overlap Smoothing Enable Flag This field is the decoded syntax element OVERLAP in bitstream. Indicates if Overlap smoothing is ON at the picture level. This field is used in both VLD and IT modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>to disable overlap smoothing filter</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>to enable overlap smoothing filter</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	to disable overlap smoothing filter	1h	Enable	to enable overlap smoothing filter
	Value	Name	Description								
	0h	Disable	to disable overlap smoothing filter								
	1h	Enable	to enable overlap smoothing filter								
	3	<p>Secondfield This flag is set for the second field in field pictures. This field is used in both VLD and IT modes.</p>									
	2:1	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ					
Project:	All										
Format:	MBZ										
0	<p>VC1 Profile specifies the bitstream profile. This field is used in both VLD and IT modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 80%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>current picture is in Advanced Profile</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This is required because 128 is added for intra blocks post inverse transform in advanced profile and also to find out if Motion vectors are adjusted or not.</p>	Value	Name	Description	0h	Disable	current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)	1h	Enable	current picture is in Advanced Profile	
Value	Name	Description									
0h	Disable	current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)									
1h	Enable	current picture is in Advanced Profile									
31	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										

MFD_VC1_LONG_PIC_STATE

30:29	CondOver	<p>This field is the decoded syntax element CONDOVER in a bitstream of advanced profile. It controls the overlap smoothing filter operation for an I frame or an BI frame when the picture level qualization step size PQUANT is 8 or lower. This field is used in intel VC1 VLD mode only, not in DXVA2 VC1 and IT modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>No overlap smoothing</td> </tr> <tr> <td>01b</td> <td></td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td></td> <td>Always perform overlap smoothing filter</td> </tr> <tr> <td>11b</td> <td></td> <td>Overlap smoothing on a per macroblock basis based on OVERFLAGS</td> </tr> </tbody> </table>	Value	Name	Description	00b		No overlap smoothing	01b		Reserved	10b		Always perform overlap smoothing filter	11b		Overlap smoothing on a per macroblock basis based on OVERFLAGS
Value	Name	Description															
00b		No overlap smoothing															
01b		Reserved															
10b		Always perform overlap smoothing filter															
11b		Overlap smoothing on a per macroblock basis based on OVERFLAGS															
28:26	PicType (Picture Type)	<p>This field specifies the coding type of the picture according to the Frame Coding Mode. When FCM = 00 01 (a Progressive or Interlaced Frame Picture): 000 = I001 = P010 = B011 = BI100 = Skipped Other encodings are reserved When FCM = 10 11 (a Field Picture) 000 = I/I001 = I/P010 = P/I011 = P/P100 = B/B101 = B/B110 = BI/B111 = BI/BI Although, for a field picture, it is set for a field-pair, but HW will only look at one field state only, and the other field state is don't care. This field is read and qualified with the SecondField flag internally. This field is unique to intel VC1 VLD Long format, and is used in IT mode as well. For DXVA2 VC1 IT mode, driver needs to convert the DXVA2 interface to intel HW VLD Long Format interface.</p>															
25:24	FCM (Frame Coding Mode)	<p>This is the same as the variable FCM defined in VC1. This field must be set to 0 for Simple and Main Profiles. This field is unique to intel VC1 VLD Long format, and is used in IT mode as well. For DXVA2 VC1 IT mode, driver needs to convert the DXVA2 interface to intel HW VLD Long Format interface.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Disable</td> <td>Progressive Frame Picture</td> </tr> <tr> <td>01b</td> <td>Enable</td> <td>Interlaced Frame Picture</td> </tr> <tr> <td>10b</td> <td></td> <td>Field Picture with Top Field First</td> </tr> <tr> <td>11b</td> <td></td> <td>Field Picture with Bottom Field First</td> </tr> </tbody> </table>	Value	Name	Description	00b	Disable	Progressive Frame Picture	01b	Enable	Interlaced Frame Picture	10b		Field Picture with Top Field First	11b		Field Picture with Bottom Field First
Value	Name	Description															
00b	Disable	Progressive Frame Picture															
01b	Enable	Interlaced Frame Picture															
10b		Field Picture with Top Field First															
11b		Field Picture with Bottom Field First															
23:21	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ											
Project:	All																
Format:	MBZ																
20:16	AltPQuant (Alternative Picture Quantization Value)	<p>This field is identical to the variable ALTPQUANT which is derived from VOPDQUANT configuration in the VC1 standard. This field must be set to 0 for Simple/Main I and BI pictures as VOPDQUANT is not present. This field is used in intel VC1 VLD Long Format mode only, not used in DXVA2 VC1 VLD and IT modes.</p>															
15:13	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ											
Project:	All																
Format:	MBZ																

MFD_VC1_LONG_PIC_STATE																	
	12:8	<p>PQuant (Picture Quantization Value)</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U5</td> </tr> </table> <p>This is the same as the calculated variable PQUANT in VC1 standard where PQuant = PQINDEX, except when QUANTIZER = 0 and PQINDEX > 8, it is given as PQuant = (PQINDEX < 29) ? PQINDEX - 3 : PQINDEX*2 - 31. This field is used in all picture types (I, P, B and BI) and all operating modes (IT mode and intel and DXVA2 VLD modes).</p>	Project:	All	Format:	U5											
	Project:	All															
	Format:	U5															
7:0	<p>BScaleFactor</p> <p>BScaleFactor This field is the scale factor for computing Direct-mode motion vectors. It is derived from the variable BFRACTION in the VC1 standard, section 8.4.5.4. There are only 21 valid values corresponding to the 21 encodings of BFRACTION as shown in the table here. Other values are reserved. MSB of this field can be used to determine if BFRACTION is greater than or equal to 1/2, which is used to determine Motion Prediction Type for B pictures. Effectively, condition "BFRACTION >= 1/2" is equivalent to condition "BScaleFactor >= 128". This field is only valid for B pictures. This field is used only in intel VC1 VLD Long format mode, it is not used in DXVA2 VC1 VLD and IT modes.</p> <p>BFRACTION</p> <p>VLCBFRACTION BScaleFactor 0001/21280011/3850102/31700111/4641003/41921011/5511102/510211100003/515311100014/520411100101/64311100115/621511101001/73711101012/77411101103/711111101114/714811110005/718511110016/722211110101/83211110113/89611111005/816011111017/8224</p>																
4	<p>31:30 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
Format:	MBZ																
	29:28	<p>UnifiedMvMode (Unified Motion Vector Mode)</p> <p>This field is a combination of the variables MVMODE and MVMODE2 in the VC1 standard, for parsing Luma MVD from the bitstream. This field is used to signal 1MV vs 4MV allowed (Mixed Mode). This field is also used to signal Q-pel or Half-pel MVD read from the bitstream. The bicubic or bilinear Luma MC interpolation mode is duplicate information from Motion Vector Mode field, and is ignored here. This field is used in intel VC1 VLD Long Format mode only, it is not used in DXVA2 VC1 VLD and IT modes.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>Mixed MV, Q-pel bicubic</td> </tr> <tr> <td>01b</td> <td></td> <td>1-MV, Q-pel bicubic</td> </tr> <tr> <td>10b</td> <td></td> <td>1-MV half-pel bicubic</td> </tr> <tr> <td>11b</td> <td></td> <td>1-MV half-pel bilinear</td> </tr> </tbody> </table>	Value	Name	Description	00b		Mixed MV, Q-pel bicubic	01b		1-MV, Q-pel bicubic	10b		1-MV half-pel bicubic	11b		1-MV half-pel bilinear
	Value	Name	Description														
	00b		Mixed MV, Q-pel bicubic														
01b		1-MV, Q-pel bicubic															
10b		1-MV half-pel bicubic															
11b		1-MV half-pel bilinear															
27	<p>FourMvSwitch (Four Motion Vector Switch)</p> <p>This field indicates if 4-MV is present for an interlaced frame P picture. It is identical to the variable 4MVSWITCH (4 Motion Vector Switch) in VC1 standard. This field is used in intel VC1 VLD Long Format mode only, it is not used in DXVA2 VC1 VLD and IT modes.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>only 1-MV</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>1, 2, or 4 MVs</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	only 1-MV	1h	Enable	1, 2, or 4 MVs							
Value	Name	Description															
0h	Disable	only 1-MV															
1h	Enable	1, 2, or 4 MVs															

MFD_VC1_LONG_PIC_STATE

26	FastUVMCFlag (Fast UV Motion Compensation Flag)	<p>This field specifies whether the motion vectors for UV is rounded to half or full pel position. It is identical to the variable FASTUVMC in VC1 standard. This field is used in both VLD and IT modes. It is derived from $FASTUVMC = (bPicSpatialResid8 \gg 4) \& 1$ in both VLD and IT modes, and should have the same value as Motion Vector Mode LSBit.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>no rounding</td> </tr> <tr> <td>1h</td> <td></td> <td>quarter-pel offsets to half/full pel positions</td> </tr> </tbody> </table>	Value	Name	Description	0h		no rounding	1h		quarter-pel offsets to half/full pel positions
Value	Name	Description									
0h		no rounding									
1h		quarter-pel offsets to half/full pel positions									
25	ReffFieldPicPolarity (Reference Field Picture Polarity)	<p>This field specifies the polarity of the one reference field picture used for a field P picture. It is derived from the variable REFFIELD defined in VC1 standard and is only valid when one field is referenced ($NUMREF = 0$) for a field P picture. When $NUMREF = 0$ and $REFFIELD = 0$, this field is the polarity of the reference I/P field that is temporally closest; When $NUMREF = 0$ and $REFFIELD = 1$, this field is the polarity of the reference I/P field that is the second most temporally closest. The distance is measured based on display order but ignoring the repeated field if present (due to $RFF = 1$). This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Top (even) field</td> </tr> <tr> <td>1h</td> <td></td> <td>Bottom (odd) field</td> </tr> </tbody> </table>	Value	Name	Description	0h		Top (even) field	1h		Bottom (odd) field
Value	Name	Description									
0h		Top (even) field									
1h		Bottom (odd) field									
24	NumRef (Number of References)	<p>This field indicates how many reference fields are referenced by the current (field) picture. It is identical to the variable NUMREF in the VC1 standard. This field is only valid for field P picture ($FCM = 10 11$). This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>One field referenced</td> </tr> <tr> <td>1h</td> <td></td> <td>Two fields referenced</td> </tr> </tbody> </table>	Value	Name	Description	0h		One field referenced	1h		Two fields referenced
Value	Name	Description									
0h		One field referenced									
1h		Two fields referenced									
23:20	BwdRefDist (Reference Distance)	<p>This field is valid only in B field pictures giving the value of BRFD. The field is ignored in P Picture. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>									
19:16	FwdRefDist (Reference Distance)	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>This field is the number of frames between the current frame and its reference frame. It is derived from the syntax element REFDIST (P Reference Distance) in the VC1 standard. 0 means that the previous frame is the reference frame. It has the same value as of FRFD for both P and B field pictures. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0, 15]</td> <td></td> </tr> </tbody> </table>	Format:	U4	Value	Name	[0, 15]				
Format:	U4										
Value	Name										
[0, 15]											

MFD_VC1_LONG_PIC_STATE																	
15:12	Reserved	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ													
Format:	MBZ																
11:10	ExtendedDMVRange (Extended Differential Motion Vector Range Flag) This field specifies the differential motion vector range in interlaced pictures. It is equivalent to the variable DMVRANGE in the VC1 standard. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>No extended range</td> </tr> <tr> <td>01b</td> <td></td> <td>Extended horizontally</td> </tr> <tr> <td>10b</td> <td></td> <td>Extended vertically</td> </tr> <tr> <td>11b</td> <td></td> <td>Extended in both directions</td> </tr> </tbody> </table>	Value	Name	Description	00b		No extended range	01b		Extended horizontally	10b		Extended vertically	11b		Extended in both directions
Value	Name	Description															
00b		No extended range															
01b		Extended horizontally															
10b		Extended vertically															
11b		Extended in both directions															
9:8	ExtendedMVRRange (Extended Motion Vector Range Flag) This field specifies the motion vector range in quarter-pel or half-pel modes. It is equivalent to the variable MVRANGE in the VC1 standard. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>[-256, 255] x [-128, 127]</td> </tr> <tr> <td>01b</td> <td></td> <td>512, 511] x [-256, 255]</td> </tr> <tr> <td>10b</td> <td></td> <td>[-2048, 2047] x [-1024, 1023]</td> </tr> <tr> <td>11b</td> <td></td> <td>[-4096, 4095] x [-2048, 2047]</td> </tr> </tbody> </table>	Value	Name	Description	00b		[-256, 255] x [-128, 127]	01b		512, 511] x [-256, 255]	10b		[-2048, 2047] x [-1024, 1023]	11b		[-4096, 4095] x [-2048, 2047]
Value	Name	Description															
00b		[-256, 255] x [-128, 127]															
01b		512, 511] x [-256, 255]															
10b		[-2048, 2047] x [-1024, 1023]															
11b		[-4096, 4095] x [-2048, 2047]															
7:4	AltPQuantEdgeMask (Alternative Picture Quantization Edge Mask) This field is a bit mask for the four edges in clock-wise order, indicating whether AltPQuant is used for the edge macroblocks. It is derived based on the following variables DQUANT, DQUANTFRM, DQPROFILE, DQSBEDGE, DQDBEDGE, and DQBILEVEL defined in the VC1 standard, as shown in Error! Reference source not found.. This field is valid only if AltPQuantConfig is 01. Bit 0: Left picture edge macroblocks Bit 1: Top picture edge macroblocks Bit 2: Right picture edge macroblocks Bit 3: Bottom picture edge macroblocks This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.																
3:2	AltPQuantConfig (Alternative Picture Quantization Configuration) This field specifies the way AltPQuant is used in the picture. It determines how to compute the macroblock quantizer step size, MQANT. It is derived based on the following variables DQUANT, DQUANTFRM, DQPROFILE, DQSBEDGE, DQDBEDGE, and DQBILEVEL defined in the VC1 standard, as shown in Error! Reference source not found.. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>AltPQuant not used</td> </tr> <tr> <td>01b</td> <td></td> <td>AltPQuant is used and applied to edge macroblocks only</td> </tr> <tr> <td>10b</td> <td></td> <td>MQANT is encoded in macroblock layer</td> </tr> <tr> <td>11b</td> <td></td> <td>AltPQuant and PQuant are selected on macroblock basis</td> </tr> </tbody> </table>	Value	Name	Description	00b		AltPQuant not used	01b		AltPQuant is used and applied to edge macroblocks only	10b		MQANT is encoded in macroblock layer	11b		AltPQuant and PQuant are selected on macroblock basis
Value	Name	Description															
00b		AltPQuant not used															
01b		AltPQuant is used and applied to edge macroblocks only															
10b		MQANT is encoded in macroblock layer															
11b		AltPQuant and PQuant are selected on macroblock basis															

MFD_VC1_LONG_PIC_STATE													
	1	<p>HalfQP</p> <p>This field is used for inverse quantization of AC coefficients. It is valid only when PQuant is used. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>											
	0	<p>PQuantUniform</p> <p>Indicating if uniform quantization applies to the picture. It is used for inverse quantization of the AC coefficients. QUANTIZER 001123PQUANTIZER - -01--PQINDEX>=9<=8----</p> <p>PQuantUniform010201ImplicitQuantizer = 0, and PQuantUniform = 0 is used to represent 2 cases : 1) QUANTIZER=01 and PQUANTIZER=0; and 2) QUANTIZER = 10b.ImplicitQuantizer = 0, and PQuantUniform = 1 is used to represent 2 cases : 1) QUANTIZER=01 and PQUANTIZER=1; and 2) QUANTIZER = 11bThis field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Non-uniform</td> <td>All</td> </tr> <tr> <td>1h</td> <td></td> <td>Uniform</td> <td>All</td> </tr> </tbody> </table>	Value	Name	Description	Project	0h		Non-uniform	All	1h		Uniform
Value	Name	Description	Project										
0h		Non-uniform	All										
1h		Uniform	All										
5	31	<p>BitplanePresentFlag (Bitplane Buffer Present Flag)</p> <p>This field indicates whether the bitplane buffer is present for the picture. If set, at least one of the fields listed in bits 22:16 is coded in non-raw mode, and Bitplane Buffer Base Address field in the VC1_BSD_BUF_BASE_STATE command points to the bitplane buffer. Otherwise, all the fields that are applicable for the current picture in bits 22:16 must be coded in raw mode. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>bitplane buffer is not present</td> </tr> <tr> <td>1h</td> <td></td> <td>bitplane buffer is present</td> </tr> </tbody> </table>	Value	Name	Description	0h		bitplane buffer is not present	1h		bitplane buffer is present		
	Value	Name	Description										
	0h		bitplane buffer is not present										
1h		bitplane buffer is present											
30	<p>ForwardMbRaw</p> <p>This field indicates whether the FORWARDMB field is coded in raw or non-raw mode. This field is only valid when PictureType is B. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>non-raw mode</td> </tr> <tr> <td>1h</td> <td></td> <td>raw mode</td> </tr> </tbody> </table>	Value	Name	Description	0h		non-raw mode	1h		raw mode			
Value	Name	Description											
0h		non-raw mode											
1h		raw mode											
29	<p>MvTypeMbRaw</p> <p>This field indicates whether the MVTYPREMB field is coded in raw or non-raw mode. This field is only valid when PictureType is P. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td></td> <td>Raw Mode</td> </tr> </tbody> </table>	Value	Name	Description	0h		Non-Raw Mode	1h		Raw Mode			
Value	Name	Description											
0h		Non-Raw Mode											
1h		Raw Mode											

MFD_VC1_LONG_PIC_STATE											
28	<p>SkipMbRaw This field indicates whether the SKIPMB field is coded in raw or non-raw mode. This field is only valid when PictureType is P or B. 0 = non-raw mode 1 = raw mode. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Raw Mode</td> </tr> </tbody> </table>		Value	Name	Description	0h	Disable	Non-Raw Mode	1h	Enable	Raw Mode
Value	Name	Description									
0h	Disable	Non-Raw Mode									
1h	Enable	Raw Mode									
27	<p>DirectMbRaw This field indicates whether the DIRECTMB field is coded in raw or non-raw mode. This field is only valid when PictureType is P or B. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td></td> <td>Raw Mode</td> </tr> </tbody> </table>		Value	Name	Description	0h		Non-Raw Mode	1h		Raw Mode
Value	Name	Description									
0h		Non-Raw Mode									
1h		Raw Mode									
26	<p>OverflagsRaw This field indicates whether the OVERFLAGS field is coded in raw or non-raw mode. This field is only valid when PictureType is I or BI. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td></td> <td>Raw Mode</td> </tr> </tbody> </table>		Value	Name	Description	0h		Non-Raw Mode	1h		Raw Mode
Value	Name	Description									
0h		Non-Raw Mode									
1h		Raw Mode									
25	<p>AcPredRaw This field indicates whether the ACPRED field is coded in raw or non-raw mode. This field is only valid when PictureType is I or BI. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Raw Mode</td> </tr> </tbody> </table>		Value	Name	Description	0h	Disable	Non-Raw Mode	1h	Enable	Raw Mode
Value	Name	Description									
0h	Disable	Non-Raw Mode									
1h	Enable	Raw Mode									
24	<p>FieldTxRaw This field indicates whether the FIELDTX field is coded in raw or non-raw mode. This field is only valid when PictureType is I or BI. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Non-Raw Mode</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Raw Mode</td> </tr> </tbody> </table>		Value	Name	Description	0h	Disable	Non-Raw Mode	1h	Enable	Raw Mode
Value	Name	Description									
0h	Disable	Non-Raw Mode									
1h	Enable	Raw Mode									
23	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </tbody> </table>		Project:	All	Format:	MBZ					
Project:	All										
Format:	MBZ										

MFD_VC1_LONG_PIC_STATE

22:20	<p>MvTab (Motion Vector Table)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> </table> <p>This field specifies which motion vector table(s) is (are) used for motion vector (differential) decoding in a P or B picture. This field is the combination of the variables MVTAB and IMVTAB in the VC1 standard. Two bits are defined for progressive frame pictures; And two or three bits are defined for interlaced field/frame pictures depending on NUMREF and P or B picture types. This field is valid for P and B pictures. It is not valid for I pictures. For P or B progressive frame pictures 0 = Motion Vector Differential VLD Table 01 = Motion Vector Differential VLD Table 12 = Motion Vector Differential VLD Table 23 = Motion Vector Differential VLD Table 3 The other encodings are reserved For P interlace field pictures with NUMREF = 0 or P/B interlace frame pictures 0 = 1-Reference Table 01 = 1-Reference Table 12 = 1-Reference Table 23 = 1-Reference Table 3 The other encodings are reserved For P interlace field picture with NUMREF = 1 or B interlaced field pictures 0 = 2-Reference Table 01 = 2-Reference Table 12 = 2-Reference Table 23 = 2-Reference Table 34 = 2-Reference Table 45 = 2-Reference Table 56 = 2-Reference Table 67 = 2-Reference Table 7 The other encodings are reserved This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>	Project:	All	Format:	U3
Project:	All				
Format:	U3				
19:18	<p>FourMvBpTab (4-MV Block Pattern Table)</p> <p>This field specifies which table is used to decode the 4-MV block pattern (4MVBp) syntax element in 4-MV macroblocks. It is identical to the variables 4MVBPTAB in the VC1 standard, section 9.1.1.37. This field is valid only in interlace frame P, B pictures, or interlace field P, B pictures. It is not valid for I picture. For interlace field P and B pictures, it is only valid if UnifiedMvMode is equal to Mixed-MV Type. For interlace frame P picture, it is only valid if FourMvSwitch is 1. For interlace frame B picture, it is always valid. 0 = 4MVBp Table 01 = 4MVBp Table 12 = 4MVBp Table 23 = 4MVBp Table 3 This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>				
17:16	<p>TwoMvBpTab (2MV Block Pattern Table)</p> <p>This field specifies which table is used to decode the 2MV block pattern (2MVBp) syntax element in 2MV field macroblocks. It is identical to the variables 2MVBPTAB in the VC1 standard, section 9.1.1.36. This field is valid only in interlace frame P/B pictures. It is not valid for I picture, nor for interlace field P or B pictures. 0 = 2MVBp Table 01 = 2MVBp Table 12 = 2MVBp Table 23 = 2MVBp Table 3 This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>				
15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				

MFD_VC1_LONG_PIC_STATE										
13:12	<p>TransType (Picture-level Transform Type)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field specifies the Transform Type at picture level. It is identical to the variable TFRM in the VC1 standard, section 7.1.1.41. This field is only valid when TransTypeMbFlag is 1. Otherwise, it is reserved and MBZ. This field is set to 00 when VSTRANSFORM is 0 in the entry point layer. 00 = 8x8 Transform 01 = 8x4 Transform 10 = 4x8 Transform 11 = 4x4 Transform This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>	Project:	All	Format:	U2					
Project:	All									
Format:	U2									
11	<p>TransTypeMbFlag (Macroblock Transform Type Flag)</p> <p>This field indicates whether Transform Type is fixed at picture level or variable at macroblock level. It is identical to the variable TTMBF in the VC1 standard, section 7.1.1.40. This field is set to 1 when VSTRANSFORM is 0 in the entry point layer. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>variable transform type in macroblock layer</td> </tr> <tr> <td>1h</td> <td></td> <td>use picture level transform type TransType</td> </tr> </tbody> </table>	Value	Name	Description	0h		variable transform type in macroblock layer	1h		use picture level transform type TransType
Value	Name	Description								
0h		variable transform type in macroblock layer								
1h		use picture level transform type TransType								
10:8	<p>MbModeTab (Macroblock Mode Table)</p> <p>This field signals which code table is used to decode the macroblock mode syntax element (MBMODETAB) in the macroblock layer in a P or B picture. This field is identical to the variables MBMODETAB in the VC1 standard, section 9.1.1.33. This field is valid for interlace frame P, B picture and interlace field P, B picture. It is not valid for I picture, nor progressive frame P, B pictures. Two bits are defined for interlace frame P, B pictures; And three bits are defined for interlaced field P, B pictures. Two bits are defined for interlace frame P, B pictures. There are two set of code tables selected based on if UnifiedMvMode is equal to 4-MV Type or not. 0 = Code Table 01 = Code Table 12 = Code Table 23 = Code Table 30 Other encodings are invalid Three bits are defined for interlace field P, B pictures. There are two set of code tables selected based on if UnifiedMvMode is equal to Mixed-MV Type or not. 0 = Code Table 01 = Code Table 12 = Code Table 23 = Code Table 34 = Code Table 45 = Code Table 56 = Code Table 67 = Code Table 7 This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>									
7:6	<p>TransAcY (Picture-level Transform Luma AC Coding Set Index, TRANSACTABLE2)</p> <p>BitFieldDesc</p>									
5:4	<p>TransAcUV (Picture-level Transform Chroma AC Coding Set Index, TRANSACTABLE)</p> <p>This field, together with PQINDEX, specifies which intra AC coding set to be used for decoding the non-zero AC coefficients in a coded luma (Y) block. This field is the combination of the variables TRANSACFRM and TRANSACFRM2 in the VC1 standard. For I pictures, TransAcY is the same as TRANSACFRM2. For other pictures, it is the same as TRANSACFRM, and therefore must be programmed to be the same as TransAcUV. This field is valid for all picture types. 0 = Coding set index 01 = Coding set index 12 = Coding set index 23 is invalid This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>									

MFD_VC1_LONG_PIC_STATE

3	<p>TransDcTab (Intra Transform DC Table)</p> <p>This field specifies whether the low motion tables or the high motion tables are used to decode the Transform DC coefficients in intra-coded blocks. This field is identical to the variable TRANSDCTAB in the VC1 standard, section 8.1.1.2. This field is valid for all picture types. This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td></td> <td>The high motion tables</td> <td style="text-align: center;">All</td> </tr> <tr> <td style="text-align: center;">1h</td> <td></td> <td>The low motion tables</td> <td style="text-align: center;">All</td> </tr> </tbody> </table>	Value	Name	Description	Project	0h		The high motion tables	All	1h		The low motion tables	All
Value	Name	Description	Project										
0h		The high motion tables	All										
1h		The low motion tables	All										
2:0	<p>CbpTab (Coded Block Pattern Table)</p> <p>This field specifies the table used to decode the CBPCY syntax element for each coded macroblock in P and B pictures. This field is combination of the variable CBPTAB for P and B frame pictures and the variable ICBPTAB in interlace field P, B pictures and interlace frame P, B pictures in the VC1 standard (Table 52 and Table 102). This field is reserved and MBZ for I or BI pictures as I only has a fixed table. 000 = Table 0 (Table 169 for P, B frames or Table 124 otherwise) 001 = Table 1 (Table 170 for P, B frames or Table 125 otherwise) 010 = Table 2 (Table 171 for P, B frames or Table 126 otherwise) 011 = Table 3 (Table 172 for P, B frames or Table 127 otherwise) 100 = Table 4 (Table 128 for interlace field/frame P, B pictures) 101 = Table 5 (Table 129 for interlace field/frame P, B pictures) 110 = Table 6 (Table 130 for interlace field/frame P, B pictures) 111 = Table 7 (Table 131 for interlace field/frame P, B pictures) This field is unique to intel VC1 VLD Long format mode, and is not used in IT and DXVA2 VC1 modes.</p>												

MFD_VC1_SHORT_PIC_STATE

MFD_VC1_SHORT_PIC_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFD_VC1_SHORT_PIC_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	2h VC1_DEC
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		1h	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	0h	
	Format:	OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0003h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:24	Reserved	
		Project:	All
		Format:	MBZ

MFD_VC1_SHORT_PIC_STATE										
	23:16	<p>Picture Height</p> <table border="1"> <tr> <td>Format:</td> <td>U8-1 Picture Height in Macroblocks</td> </tr> </table> <p>This field indicates the height of the picture in unit of macroblocks. For example, for a 1920x1080 frame picture, PictureHeightInMBs equals 68 (1080 divided by 16, and rounded up, i.e. effectively specified as 1088 instead). This field is used in VLD and IT modes. Note: Even though the Advanced Profile allows frame dimensions (width, height) to not be aligned to macroblock boundary, it doesn't affect the bitstream decoding. And it is preferable to use 'intermediate buffer' that is macroblock aligned for decoding. In order to simplify the out-of-bound reference pixel access, the out-of-bound extrapolation rule in VC1 spec can be used to expand the expected decoded frame to the intermediate buffer dimension.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,127]</td> <td>Value_0_to_127</td> <td>[1, 128] MB</td> </tr> </tbody> </table>	Format:	U8-1 Picture Height in Macroblocks	Value	Name	Description	[0,127]	Value_0_to_127	[1, 128] MB
	Format:	U8-1 Picture Height in Macroblocks								
	Value	Name	Description							
[0,127]	Value_0_to_127	[1, 128] MB								
15:8	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ					
Project:	All									
Format:	MBZ									
7:0	<p>Picture Width</p> <table border="1"> <tr> <td>Format:</td> <td>U8-1 Picture Width in Macroblocks</td> </tr> </table> <p>This field indicates the width of the picture in unit of macroblocks. For example, for a 1920x1080 frame picture, PictureWidthInMBs equals 120 (1920 divided by 16). This field is used in VLD and IT modes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,127]</td> <td>Value_0_to_127</td> <td>[1, 128] MB</td> </tr> </tbody> </table>	Format:	U8-1 Picture Width in Macroblocks	Value	Name	Description	[0,127]	Value_0_to_127	[1, 128] MB	
Format:	U8-1 Picture Width in Macroblocks									
Value	Name	Description								
[0,127]	Value_0_to_127	[1, 128] MB								
2	31:24	<p>Bitplane Buffer Pitch Minus 1</p> <table border="1"> <tr> <td>Format:</td> <td>U7-1 Pitch in Bytes</td> </tr> </table> <p>Specifies the bitplane buffer pitch in (#Bytes - 1). Bitplane buffer is a linear buffer. It is needed only when the bitplane is not encoded as raw, and therefore is present in the header explicitly. In VC1 Long Format (Gen6 and Gen7), it is written by an application and later read by the HW. In VC1 Long Format (Gen6 and Gen7), it is written by an application, and later read by the HW. But in VC1 Short Format (Gen7 only), it is written and read by H/W only. This field is specified for better performance. For Gen6 : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Long Format : The pitch must be equal to PictureWidthInMBs/2. For Gen7 VC1 Short Format : If Pic Width is less than or equal to 2K pixels, bitplane pitch is set to 64 (one cacheline; programmed as 63) bytes per MB row. If Pic Width is greater than 2K pixels, bitplane pitch is set to 128 (two cachelines; programmed as 127) bytes per MB row. This field is not used in IT mode, used in VLD mode only. For VC1 DXVA2 Short Format, the bitplane specification is between H/W and Driver only. For Long Format, application is responsible for allocation with the driver.</p>	Format:	U7-1 Pitch in Bytes						
	Format:	U7-1 Pitch in Bytes								
23	<p>Interpolation Rounder Control</p> <p>Used only in MC operation. This field specifies the rounding control value used in interpolation operation of motion prediction process. Note: This bit field is taken from bRcontrol in DXVA_PictureParameters data structure. This field is used in VLD and IT modes.</p>									

MFD_VC1_SHORT_PIC_STATE											
22:20	Reserved										
	Project:	All									
	Format:	MBZ									
19:16	Motion Vector Mode This field indicates one of the following motion compensation interpolation modes for P and B pictures. The MC interpolation modes apply to prediction values of luminance blocks and are always in quarter-sample. For chrominance blocks, it always performs bilinear interpolation with either half-pel or quarter-pel precision. 0XX0 = Chroma Quarter -pel + Luma bicubic. (can only be 1MV) 0XX1 = Chroma Half-pel + Luma bicubic. (can be 1MV or 4MV) 1XX0 = Chroma Quarter -pel + Luma bilinear. (can only be 1MV) 1XX1 = Chroma Half-pel + Luma bilinear Note: Bits 19:16 are taken from bMVprecisionAndChromaRelation in DXVA_PictureParameters data structure. Bit 19 of Motion Vector Mode = 1 for Luma Bilinear MC; = 0 for Luma Bicubic MC Bit 16 of Motion Vector Mode = 1 for half-sample Chroma motion = 0 for quarter-sample Chroma motion. This field is used in both VLD and IT modes. Before the polarity of Chroma Half-pel or Q-pel is reversed from DXVA2 Spec, now I have fixed it to match with DXVA2 VC1 Spec. ???										
15	DmvSurfaceValid Indicated when the DMV read surface is valid. This surface stored the direct motion vectors. This field is set fo B pictures that can refer to a previous P picture for DMV. If there is an I-picture before a B (in decoding order) then this field is not set (as a result, zero's DMV's will be assumed while decoding the B picture. That is, there is no explicit DMV buffer for an I-picture). This field is not used in IT mode, used in VLD mode only.										
14:12	Reserved										
	Project:	All									
	Format:	MBZ									
11	VC1 Profile Project: All specifies the bitstream profile. Note: This is required because 128 is added for intra blocks post inverse transform in advanced profile and also to find out if Motion vectors are adjusted or not. This field is used in both VLD and IT modes.										
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Value</th> <th style="text-align: left;">Name</th> <th style="text-align: left;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Default]</td> <td>current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)</td> </tr> <tr> <td>1h</td> <td></td> <td>current picture is in Advanced Profile</td> </tr> </tbody> </table>	Value	Name	Description	0h	Default]	current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)	1h		current picture is in Advanced Profile	
Value	Name	Description									
0h	Default]	current picture is in Simple or Main Profile (No need to distinguish Simple and Main Profile)									
1h		current picture is in Advanced Profile									
10:6	Reserved										
	Project:	All									
	Format:	MBZ									

MFD_VC1_SHORT_PIC_STATE				
5	Backward Prediction Present Flag Note : a B picture that only uses forward prediction may have this flag set to 1 as well. Driver may still need to provide a valid reference picture index. This field is used in both DXVA2 VC1 VLD mode and IT mode. It is the same parameter as bPicBackwardPrediction in DXVA2 VC1 spec. The Intra Picture Flag, Backward Prediction Present Flag and RefPicFlag are used to derive the picture type, as specified in PTYPE for a frame, and in FPTYPE for a field, in DXVA2 VC1 VLD and IT mode.			
	4	Intra Picture Flag This field is used in both DXVA2 VC1 VLD mode and IT mode. It is the same parameter as bPicIntra in DXVA2 VC1 spec. The Intra Picture Flag, Backward Prediction Present Flag and RefPicFlag are used to derive the picture type, as specified in PTYPE for a frame, and in FPTYPE for a field, in DXVA2 VC1 VLD and IT mode.		
		Value	Name	Description
		0h		entire picture can have a mixture of intra and inter MB type or just inter MB type.
		1h		entire picture is coded in intra MB type
	3	SecondField This flag is set for the second field in field pictures. This field is used in both VLD and IT modes.		
	2	Reserved		
		Project:	All	
		Format:	MBZ	
	1:0	Picture Structure This field is used in both DXVA2 VC1 VLD mode and IT mode. It is the same parameter as bPicStructure in DXVA2 VC1 spec. The Picture Structure and Progressive Pic Type are used to derive the picture structure as specified in FCM, in DXVA2 VC1 VLD and IT mode.		
Value		Name	Description	
01b			top field (bit 0)	
10b			bottom field (bit 1)	
11b			frame (both fields are present)	
00b			illegal	
3	31	Reserved		
		Project:	All	
		Format:	MBZ	
	30	Overlap Smoothing Enable Flag This field is the decoded syntax element OVERLAP in bitstream. Indicates if Overlap smoothing is ON at the picture level. This field is used in both VLD and IT modes.		
		Value	Name	Description
		0h	Disable	to disable overlap smoothing filter
		1h	Enable	to enable overlap smoothing filter

MFD_VC1_SHORT_PIC_STATE

29	Range Reduction Scale	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">All</td> </tr> <tr> <td>Access:</td> <td>None</td> </tr> </table> <p>This field specifies whether the reference picture pixel values should be scaled up or scaled down on-the-fly, if RangeReduction is Enabled. NOTE: This bit is derived by driver for Main Profile only. Ignored in Simple and Advanced Profiles. This field is used in both VLD and IT modes. This is derived by driver from the history of RANGERED and RANGEREDFRM syntax elements (i.e. of forward/preceding reference picture) and those of the current picture. RANGERED is the same as (bPicOverflowBlocks » 3) & 1. RANGEREDFRM is the same as (bPicDeblocked » 5) & 1. For the current picture is a B picture, this field represents the state of the forward/preceding reference picture only. Driver is responsible to keep RangeReductionScale, RangeReduction Enable and RANGERED Present Flag of current picture coherent.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 55%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable [Default]</td> <td>Scale down reference picture by factor of 2</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Scale up reference picture by factor of 2</td> </tr> </tbody> </table>	Project:	All	Access:	None	Value	Name	Description	0h	Disable [Default]	Scale down reference picture by factor of 2	1h	Enable	Scale up reference picture by factor of 2
Project:	All														
Access:	None														
Value	Name	Description													
0h	Disable [Default]	Scale down reference picture by factor of 2													
1h	Enable	Scale up reference picture by factor of 2													
28	Range Reduction Enable	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">All</td> </tr> </table> <p>This field specifies whether on-the-fly pixel value range reduction should be performed for the preceding (or forward) reference picture. Along with RangeReductionScale to specify whether scale up or down should be performed. It is not the same value as RANGEREDFRM Syntax Element (DXVA_PictureParameters bPicDeblocked bit 5) in the Picture Header. This field is for Main Profile only. Simple Profile is always disable, and not applicable to Advanced Profile. This field is used in both VLD and IT modes. This is derived by driver from the history of RANGERED and RANGEREDFRM syntax elements (i.e. of forward/preceding reference picture) and those of the current picture. RANGERED is the same as (bPicOverflowBlocks » 3) & 1. RANGEREDFRM is the same as (bPicDeblocked » 5) & 1. For the current picture is a B picture, this field represents the state of the forward/preceding reference picture only. Driver is responsible to keep RangeReductionScale, RangeReduction Enable and RANGERED Present Flag of current picture coherent.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 55%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable [Default]</td> <td>Range reduction is not performed</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Range reduction is performed</td> </tr> </tbody> </table>	Project:	All	Value	Name	Description	0h	Disable [Default]	Range reduction is not performed	1h	Enable	Range reduction is performed		
Project:	All														
Value	Name	Description													
0h	Disable [Default]	Range reduction is not performed													
1h	Enable	Range reduction is performed													
27:24	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ									
Project:	All														
Format:	MBZ														

MFD_VC1_SHORT_PIC_STATE		
23:22	Progressive Pic Type	
	This field is used in both DXVA2 VC1 VLD mode and IT mode. It is the same parameter as bPicExtrapolation in DXVA2 VC1 spec. The Picture Structure and Progressive Pic Type are used to derive the picture structure as specified in FCM, in DXVA2 VC1 VLD and IT mode.	
	Value	Name
	Description	
	0	progressive only picture
21	Reserved	
	Project:	All
	Format:	MBZ
20:16	P-Pic Ref Distance	
	Project:	All
	Access:	None
	This element defines the number of frames between the current frame and the reference frame. It is the same as the REFDIST SE in VC1 interlaced field picture header. It is present if the entry-level flag REFDIST_FLAG == 1, and if the picture type is not one of the following types: B/B, B/BI, BI/B, BI/BI. If the entry level flag REFDIST_FLAG == 0, REFDIST shall be set to the default value of 0. This field is used in DXVA2 VC1 VLD mode only, not used in IT and intel VC1 VLD Long Format modes.	
	Value	Name
0-16	unsigned integer	
0h	[Default]	
15:14	QUANTIZER	
	Value	Name
	Description	
	00b	implicit quantizer at frame level
	01b	explicit quantizer at frame level, and use PQUANTIZER SE to specify uniform or non-uniform
	10b	explicit quantizer, and non-uniform quantizer for all frames
11b	explicit quantizer, and uniform quantizer for all frames	
13	MULTIRES Present Flag (for Simple/Main Profile only)	
	Value	Name
	Description	
	0h	RESPIC Parameter is present in the picture header
1h	RESPIC Parameter is present in the picture header	

MFD_VC1_SHORT_PIC_STATE		
12	SYNCMARKER Present Flag (for Simple/Main Profile only)	
	Value	Name
		Description
0		Bitstream for Simple and Main Profile has no sync marker
1		Bitstream for Simple and Main Profile may have sync marker(s)
11	RANGERED Present Flag (for Simple/Main Profile only)	
	It is needed for Picture Header Parsing.Driver is responsible to keep RangeReductionScale, RangeReduction Enable and RANGERED Present Flag of current picture coherent.	
	Value	Name
		Description
0		Range Reduction Parameter (RANGEREDFRM) is not present in the picture header
1		Range Reduction Parameter (RANGEREDFRM) is present in the picture header.
10:8	MAXBFRAMES	
	Number of consecutive B Frames.	
7	PANSCAN Present Flag	
	Value	Name
		Description
0		Pan Scan Parameters are not present in the picture header
1		Pan Scan Parameters are present in the picture header
6	REFDIST_FLAG	
	For header parsing REFDIST.This is used in DXVA2 VC1 VLD mode only, not used in IT and intel VC1 VLD modes.	
5	LOOPFILTER Enable Flag	
	This field is the decoded syntax element LOOPFILTER in bitstream. It indicates if In-loop Deblocking is ON according to picture level bitstream syntax control. This bit affects BSD unit and also the loop filter unit.When this bit is set to 1, PostDeblockOutEnable field in MFX_PIPE_MODE_SELECT command must also be set to 1. In this case, in-loop deblocking operation follows the VC1 standard - deblocking doesn't cross slice boundary.	
	Value	Name
		Description
0		In-Loop-Deblocking-Filter is disabled
1		In-Loop-Deblocking-Filter is enabled
4	FastUVMCFlag (Fast UV Motion Compensation Flag)	
	This field specifies whether the motion vectors for UV is rounded to half or full pel position. It is identical to the variable FASTUVMC in VC1 standard.This field is used in both VLD and IT modes.It is derived from FASTUVMC = (bPicSpatialResid8 » 4) & 1 in both VLD and IT modes, and should have the same value as Motion Vector Mode LSBIt.	
	Value	Name
		Description
0h		no rounding
1h		quarter-pel offsets to half/full pel positions

MFD_VC1_SHORT_PIC_STATE			
3	EXTENDED_MV Present Flag		
	BitFieldDesc		
	Value	Name	Description
	0h		Extended_MV is not present in the picture header
	1h		Extended_MV is present in the picture header
	2:1	DQUANT	
		Project:	All
		Access:	None
		Format:	U2
		Use for Picture Header Parsing of VOPDUANT elements	
Value		Name	Description
0h		[Default]	
00b		no VOPDQUANT elements; Quantizer cannot vary in frame, same quantization step size PQUANT is used for all MBs in the frame	
01b		refer to VC1 Spec. for all the MB position dependent quantizer selection	
10b		The macroblocks located on the picture edge boundary shall be quantized with ALTPQUANT while the rest of the macroblocks shall be quantized with PQUANT.	
11b	Reserved		
0	VSTRANSFORM flag		
	Value	Name	Description
	0h	Disable	variable-sized transform coding is not enabled
	1h	Enable	variable-sized transform coding is enabled
4	31:29	Reserved	
	Format:	MBZ (for possible future change to BFraction Enumeration)	
28:24	BFraction Enumeration		
<p>This field is the scale factor for computing Direct-mode motion vectors. It is derived from the variable BFRACTION in the VC1 standard, section 8.4.5.4. There are only 21 valid values corresponding to the 21 encodings of BFRACTION as shown in the table here. Other values are reserved. The VLD decoded value of BFRACTION (from the picture header) is mapped into an enum value from 0 to 20. (??? MSB of this field can be used to determine if BFRACTION is greater than or equal to 1/2, which is used to determine Motion Prediction Type for B pictures. Effectively, condition "BFRACTION >= 1/2" is equivalent to condition "ScaleFactor >= 128". ??? How can the enum replicate this feature ???) This field is only valid for B pictures. This field is used only in DXVA2 VC1 VLD mode, it is not used in Intel VC1 VLD Long Format mode and IT mode.</p> <p>BFRACTION VLCBFRACTION Enum0001/200011/310102/320111/431003/441011/551102/5611100003/5711100014/5811100101/6911100115/61011101001/71111101012/71211101103/71311101114/71411110005/71511110016/71611110101/81711110113/81811111005/81911111017/82011111111BI Pic Indicator31 (optional)</p>			



MFD_VC1_SHORT_PIC_STATE		
23	Reserved	
	Project:	All
	Format:	MBZ Advanced Profile only; RANGE_MAPY_FLAG Range Mapping not supported
22:20	Reserved	
	Project:	All
	Format:	MBZ Advanced Profile only; RANGE_MAPY Range Mapping not supported
19	Reserved	
	Project:	All
	Format:	MBZ Advanced Profile only; RANGE_MAPUV_FLAG Range Mapping not supported
18:16	Reserved	
	Project:	All
	Format:	MBZ Advanced Profile only; RANGE_MAPUV Range Mapping not supported
15:9	Reserved	
	Project:	All
	Format:	MBZ
8	4MV Allowed Flag	
7	POSTPROC Flag	
6	PULLDOWN	
5	INTERLACE	
4	TFCNTRFLAG	
3	FINTERFLAG	
2	REFPIC Flag	
	For a BI picture, REFPIC flag must set to 0. For I and P picture, REFPIC flag must set to 0. For a B picture, REFPIC flag must set to 0, except for a B-field in interlaced field mode which can be 0 or 1 (e.g. the top B field can be used as a reference for decoding its corresponding bottom B-field in a field pair). In VLD mode, this flag cannot be used as an optimization signaling for an I or P picture that is not used as a reference picture. This field is used in both DXVA2 VC1 VLD mode and IT mode. It is the same parameter as bPicDeblockConfined[bit2] in DXVA2 VC1 spec. The Intra Picture Flag, Backward Prediction Present Flag and RefPicFlag are used to derive the picture type, as specified in PTYPE for a frame, and in FPTYPE for a field, in DXVA2 VC1 VLD and IT mode.	
	Value	Name Description
	0h	
1h		the current picture after decoded, will be used as a reference picture later
1	PSF	

MFD_VC1_SHORT_PIC_STATE		
0	EXTENDED_DMV Present Flag	
Value	Name	Description
0h	[Default]	Extended_DMV is not present in the picture header
1h		Extended_DMV is present in the picture header

MFD_VP8_BSD_OBJECT

MFD_VP8_BSD_OBJECT		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
<p>The MFD_VP8_BSD_OBJECT command is the only primitive command for the VP8 Decoding Pipeline. The Partitions of the bitstream is loaded as indirect data object. Before issuing a MFD_VP8_BSD_OBJECT command, all VP8 frame level states of the MFD Engine need to be valid. Therefore the commands used to set these states need to have been issued prior to the issue of a MFD_VP8_BSD_OBJECT command. Context switch interrupt is not supported by this command.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h PARALLEL_VIDEO_PIPE
	Format: OpCode	
	28:27	Pipeline
		Default Value: 2h MFD_VP8_BSD_OBJECT
	Format: OpCode	
	26:24	Media Command OpCode
		Default Value: 4h VP8_DEC
Format: OpCode		
23:21	subOpcodeA	
	Default Value: 1h	
Format: OpCode		
20:16	subOpcodeB	
	Default Value: 8h	
Format: OpCode		
15:12	Reserved	
Format: MBZ		
11:0	DWord Length	
	Default Value: 14h Excludes DWord (0,1)	
Format: =n Total Length - 2		
1	31:21	Reserved
		Format: MBZ
	20:16	Partition0 CPBAC Entropy Count Pass the Partition0 CPBAC State to HW. Max value is 24.
15:8	Partition0 CPBAC Entropy Range Pass the Partition0 CPBAC State to HW.	

MFD_VP8_BSD_OBJECT				
	7:6	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	5:4	Coded Num of Coeff Token Partitions Num of Partitions = $2^{\text{CodedNumCoeffTokenPartitions}}$. 0 = 1 Partition only 1 = 2 Partitions 2 = 4 Partitions 3 = 8 Partitions are present in the bitstream.		
	3	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
2:0	Partition0 FirstMBBitOffset from Frame Header Allow HW to jump to the location in the bitstream where per MB information starts in the Partition0.			
2	31:24	Partition0 CPBAC Entropy Value Pass the Partition0 CPBAC State to HW.		
	23:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
3	31:24	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	Indirect Partition0 Data Length This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.			
4	31:0	Indirect Partition0 Data Start Offset This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.		
5	31:24	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	Indirect Partition1 Data Length This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.			
6	31:0	Indirect Partition1 Data Start Offset This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.		

MFD_VP8_BSD_OBJECT				
7	31:24	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	Indirect Partition2 Data Length This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.			
8	31:0	Indirect Partition2 Data Start Offset This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.		
9	31:24	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	Indirect Partition3 Data Length This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.			
10	31:0	Indirect Partition3 Data Start Offset This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.		
11	31:24	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	Indirect Partition4 Data Length This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.			
12	31:0	Indirect Partition4 Data Start Offset This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.		
13	31:24	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			

MFD_VP8_BSD_OBJECT				
	23:0	<p>Indirect Partition5 Data Length</p> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.</p>		
14	31:0	<p>Indirect Partition5 Data Start Offset</p> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.</p>		
15	31:24	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	<p>Indirect Partition6 Data Length</p> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.</p>			
16	31:0	<p>Indirect Partition6 Data Start Offset</p> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.</p>		
17	31:24	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
23:0	<p>Indirect Partition7 Data Length</p> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.</p>			
18	31:0	<p>Indirect Partition7 Data Start Offset</p> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.</p>		
19	31:24	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			

MFD_VP8_BSD_OBJECT											
	23:0	<p>Indirect Partition8 Data Length</p> <p>This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Partition Start Offset field is ignored. The Partition is byte aligned in both ends. It is the length in bytes of the bitstream data for the current partition. It includes the first byte of the first macroblock and the last byte of the last macroblock in the partition.</p>									
20	31:0	<p>Indirect Partition8 Data Start Offset</p> <p>This field specifies the Graphics Memory starting address of the data to be fetched into BSD Unit for processing. This offset is relative to the MFD Indirect Object Base Address. Hardware ignores this field if indirect data is not present. It is a byte-aligned address for the VP8 bitstream data in each partition.</p>									
21	31	<p>Concealment Method</p> <p>This field specifies the method used for concealment when error is detected.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Intra 16x16 Prediction</td> <td>A copy from the current picture is performed using Intra 16x16 Prediction method.</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Inter P Copy</td> <td>A copy from collocated macroblock location is performed from the concealment reference indicated by the ConCeal_Pic_Id field.</td> </tr> </tbody> </table>	Value	Name	Description	0	Intra 16x16 Prediction	A copy from the current picture is performed using Intra 16x16 Prediction method.	1	Inter P Copy	A copy from collocated macroblock location is performed from the concealment reference indicated by the ConCeal_Pic_Id field.
		Value	Name	Description							
		0	Intra 16x16 Prediction	A copy from the current picture is performed using Intra 16x16 Prediction method.							
	1	Inter P Copy	A copy from collocated macroblock location is performed from the concealment reference indicated by the ConCeal_Pic_Id field.								
	30:18	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ										
17:16	<p>Conceal_Pic_Id (Concealment Picture ID)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Exists If:</td> <td>[Concealment Method] == 1</td> </tr> </table> <p>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. 00 - Last Decoded Picture 01 - Golden Reference Picture 02 - Alternate Reference Picture 03 - User provided Reference Picture</p>	Exists If:	[Concealment Method] == 1								
Exists If:	[Concealment Method] == 1										
15	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
14	<p>BSDPrematureComplete Error Handling</p> <p>It occurs in situation where the decode is completed but there are still data in the bitstream.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Set the interrupt to the driver (provide MMIO registers for MB address R/W)??</td> </tr> </tbody> </table>	Value	Name	0	Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling	1	Set the interrupt to the driver (provide MMIO registers for MB address R/W)??				
	Value	Name									
	0	Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling									
1	Set the interrupt to the driver (provide MMIO registers for MB address R/W)??										
13	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										

MFD_VP8_BSD_OBJECT			
12	MPR Error (MV out of range) Handling		
	Value	Name	
	0	Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling	
	1	Set the interrupt to the driver (provide MMIO registers for MB address R/W)??	
	11	Reserved	
		Format:	MBZ
	10	Entropy Error Handling	
		Value	Name
		0	Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling
		1	Set the interrupt to the driver (provide MMIO registers for MB address R/W)??
	9	Reserved	
		Format:	MBZ
8	MB Header Error Handling		
	Value	Name	
	0	Ignore the error and continue (masked the interrupt), assume the hardware automatically perform the error handling	
	1	Set the interrupt to the driver (provide MMIO registers for MB address R/W)??	
7:0	Reserved		
	Format:	MBZ	

MFX_AVC_DIRECTMODE_STATE

MFX_AVC_DIRECTMODE_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a picture level command and is issued once per picture. All DMV buffers are treated as standard media surfaces, in which the lower 6 bits are used for conveying surface states. Current Pic POC number is assumed to be available in POCList[32 and 33] of the MFX_AVC_DIRECTMODE_STATE Command. This command is only valid in the AVC decoding in VLD and IT modes, and AVC encoder mode. The same command supports both Long and Short DXVA2 AVC Interface. The DMV buffers are not required to be programmed for encoder mode.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_SINGLE_DW
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h AVC_COMMON
		Format:	OpCode
	23:21	SubOpcodeA	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcodeB	
		Default Value:	2h
		Format:	OpCode
	15:12	Reserved	
		Format:	MBZ
	11:0	DWord Length	
Default Value:		0045h Excludes DWord (0,1)	
Format:		=n Total Length - 2	

MFX_AVC_DIRECTMODE_STATE									
1	31:6	<p>Direct MV Buffer Base Address for Picture 0 (In Frame)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Note:This field is changed to one per frame (both top and bottom field share the same Direct MV Buffer Base Address).</p> <p>This field provides the base address of the DMV write buffer to store motion vectors decoded in the current picture (top field), which may be used later as a collocated motion information read buffer of the associated reference picture in decoding subsequent B-pictures that have MB coded in direct mode. It is a private buffer used by the MPR hardware only. Its content is not accessed by software. This buffer must be 64-byte cacheline aligned. The write buffer size is 557,056 bytes for 1 frame. Scalable with frame height, but do not scale with frame width as the hardware assumes frame width (in MBs) fixed at 128 (smallest power of 2 value larger than 120 - 1920x1088 screen resolution) It is only valid if the current picture is a progressive frame, MbAff frame, or a top field. There are a total of 32 reference picture (previously decoded) Direct MV Buffers (0 to 31, not including the DMV write buffer 32 and 33 of the current picture) to read in the corresponding collocated DMV and motion information. For reference picture, these 32 DMV read Buffers can be indexed by the frame_store_ID[4:0], which is obtained from RefPicList L0/L1[RefPicIdx]. frame_Store_IDbit[0] (indicator for Top/Bottom Field). For writing out motion information during the decoding of the current picture, all 34 DMV buffers can be addressed by [img_dec_fs_idc[4:0]«1 + img_structure[1]].</p>	Format:	GraphicsAddress[31:6]					
	Format:	GraphicsAddress[31:6]							
5:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
Project:	BDW								
Format:	MBZ								
2 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	BDW	Format:	MBZ			
	Project:	BDW							
Format:	MBZ								
15:0	<p>Direct MV Buffer Base Address for Picture 0 - Read/Write [47:32]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>This field is for the upper range of AACs Bit Vector Surface Starting Byte Address.</td> <td></td> </tr> <tr> <td>This field is used for 48-bit addressing.</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Project:	BDW	Description	Project	This field is for the upper range of AACs Bit Vector Surface Starting Byte Address.		This field is used for 48-bit addressing.	BDW
Project:	BDW								
Description	Project								
This field is for the upper range of AACs Bit Vector Surface Starting Byte Address.									
This field is used for 48-bit addressing.	BDW								
3..32	63:48	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
47:32	<p>Direct MV Buffer Base Address for Reference Frame 1 to 15 (In Frame) High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Pre-Deblocking Destination Address. This field is ignored if PreDeblockOutEnable is set to 0 (disable). This field is used for 48-bit addressing.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]				
Project:	BDW								
Format:	GraphicsAddress[47:32]								

MFX_AVC_DIRECTMODE_STATE															
	31:6	<p>Direct MV Buffer Base Address for Reference Frame 1 to 15 (In Frame)</p> <table border="1"> <tr> <td>Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Note:This field is changed to one per frame (both top and bottom field shared the same Direct MV Buffer Base Address)</p> <p>This field provides the base address of the DMV buffer for reference frame 2 to 31. They are needed if the current B-Picture has MBs coded in direct mode. It is a private buffer used by the MPR hardware only. Its content is not accessed by software. All these buffers must be 64-byte cacheline aligned. There are a total of 32 possible Direct MV Read Buffers (not including the current write buffer of the current picture) to read in the corresponding DMV. Each read buffer size is 557,056 bytes for 1 frame (the selected colPic). Scalable with frame height, but do not scale with frame width as the hardware assumes frame width (in MBs) fixed at 128 (smallest power of 2 value larger than 120 - 1920x1088 screen resolution). The adjacent DMV buffers are paired ([2 and 3], [4 and 5], [N and N+1], ..[30 and 31]).</p>	Format:	GraphicsAddress[31:6]											
	Format:	GraphicsAddress[31:6]													
5:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Format:	MBZ												
Format:	MBZ														
33 Project: BDW	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
	Format:	MBZ													
	14:13	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
Format:	MBZ														
12:11	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW														
Format:	MBZ														
10:9	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW														
Format:	MBZ														
8:7	<p>Direct MV Buffer Base Address for Reference Frame - Arbitration Priority Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	U2	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW														
Format:	U2														
Value	Name														
00b	Highest priority														
01b	Second highest priority														
10b	Third highest priority														
11b	Lowest priority														

MFX_AVC_DIRECTMODE_STATE													
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Direct MV Buffer for Reference Picture 0 to 15</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)
Project:	BDW												
Value	Name												
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)												
01b	Uncacheable (UC) - non-cacheable												
10b	Writethrough (WT)												
11b	Writeback (WB)												
4:3	<p>Target Cache (TC) for Direct MV Buffer for Reference Picture 0 to 15</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed
Project:	BDW												
Value	Name												
00b	eLLC Only - not snooped in GT (BDW)												
01b	LLC Only												
10b	LLC/eLLC Allowed												
11b	L3, LLC, eLLC Allowed												
2	Reserved												
1:0	<p>Age for QUADLRU (AGE) for Direct MV Buffer for Reference Picture 0 to 15</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW												
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												



		MFX_AVC_DIRECTMODE_STATE	
34	31:6	Direct MV Buffer Base Address for Write (Write-Only Buffer)(in frame)	
		Format:	GraphicsAddress[31:6]
		<p>This field provides the base address of the DMV write-only buffer for the current decoding frame/field. It is a private buffer used by the MPR hardware only. Its content is not accessed by software. All these buffers must be 64-byte cacheline aligned, i.e. the same as the above DMV read/write buffers. These 2 buffers can only be addressed by [img_dec_fs_idc[4:0]«1 + img_structure[1]] for the current picture being decoded.</p> <p>Each write buffer size is 557,056 bytes for 1 frame (the selected colPic). Scalable with frame height, but do not scale with frame width as the hardware assumes frame width (in MBs) fixed at 128 (smallest power of 2 value larger than 120 - 1920x1088 screen resolution).</p> <p>DMV write buffer 32 is valid only if the current picture is a progressive frame, MbAff frame, or a top field. DMV write buffer 33 is valid only if the current picture is a bottom field.</p>	
	5:0	Reserved	
		Project:	BDW
		Format:	MBZ
35 Project: BDW	31:16	Reserved	
		Project:	BDW
		Format:	MBZ
		Reserved for 64-bit address extension.	
	15:0	Direct MV Buffer Base Address for Write (Write-Only Buffer)(in frame) High	
		Project:	BDW
		Format:	GraphicsAddress[47:32]
		This field is for the upper range of Direct MV Buffer Base Address. This field is ignored if PreDeblockOutEnable is set to 0 (disable). This field is used for 48-bit addressing.	
36 Project: BDW	31:15	Reserved	
		Project:	BDW
		Format:	MBZ
	14:13	Reserved	
		Project:	BDW
		Format:	MBZ
	12:11	Reserved	
		Project:	BDW
		Format:	MBZ
	10:9	Reserved	
		Project:	BDW
		Format:	MBZ

MFX_AVC_DIRECTMODE_STATE													
8:7	<p>Direct MV Buffer Base Address for Write - Arbitration Priority Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 25%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW												
Value	Name												
00b	Highest priority												
01b	Second highest priority												
10b	Third highest priority												
11b	Lowest priority												
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Direct MV Buffer for Write</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 25%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)
Project:	BDW												
Value	Name												
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)												
01b	Uncacheable (UC) - non-cacheable												
10b	Writethrough (WT)												
11b	Writeback (WB)												
4:3	<p>Target Cache (TC) for Direct MV Buffer for Write</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC</p> <p>00b: eLLC Only (<i>"00" setting points TC selection to PTE which defaults to eLLC</i>)</p> <p>01b: LLC Only (<i>Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present</i>).</p> <p>10b: LLC/eLLC Allowed.</p> <p>11b: L3, LLC, eLLC Allowed.</p> <p>Errata BDW:A-E (FIXED BY:G0 Stepping):</p> <p>For all system that does NOT use SVM (i.e. coherent L3\$ surfaces), back snoops from LLC has to be disabled (Dis_GtCvUpdtOnRd = "1"). Than target Cache settings can be programmed as POR requirements of L3/LLC/eDRAM caching.</p> <p>For all systems that does use SVM (i.e. coherent L3\$ surfaces), the recommended setting would be "00" in target cache settings. In case of L3 surfaces, the performance has to be tuned between "00" and "11" setting based on the benefits of L3 caching outweighing the degradation of backsnoops.</p> <p>Post G0-stepping, the above w/a for coherent L3\$ surfaces is not needed.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 25%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)						
Project:	BDW												
Value	Name												
00b	eLLC Only - not snooped in GT (BDW)												



MFX_AVC_DIRECTMODE_STATE													
	<table border="1"> <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>	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed						
01b	LLC Only												
10b	LLC/eLLC Allowed												
11b	L3, LLC, eLLC Allowed												
2	Reserved												
1:0	<p>Age for QUADLRU (AGE) for Direct MV Buffer for Write</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITS, hence need to be replaced least often in caches.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW												
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												
37..70	<p>31:0 POC List, POCList[34][31:0]</p> <p>Each POC value is a signed 32-bit number. One-to-one correspondence with the 34 Direct MV Buffer Address for Reference and Current Frames/Fields There are 34 POC entries in the list. For reference picture, only the lower 32 POC [0-31] entries can be used, and POCList is indexed by the frame_store_ID[4:0], which is obtained from RefPicList L0/L1[RefPicIdx]. frame_Store_IDbit[0] (indicator for Top/Bottom Field). For current picture, all 34 POC entries [0-33] can be addressed by POCList[img_dec_fs_idc[4:0]«1 + img_structure[1]]. For frame-only mode, every other entry is skipped. For MBAFF and field-only picture, each entry is a field POC, and every two entries are paired.</p>												

MFX_AVC_IMG_STATE

MFX_AVC_IMG_STATE		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
<p>This must be the very first command to issue after the surface state, the pipe select and base address setting commands. This command supports both Long and Short VLD and IT DXVA2 AVC Decoding Interface.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h PARALLEL_VIDEO_PIPE
		Format: OpCode
	28:27	Pipeline
		Default Value: 2h MFX_AVC_IMG_STATE
		Format: OpCode
	26:24	Media Command Opcode
		Default Value: 1h AVC_COMMON
		Format: OpCode
	23:21	SubOpcode A
Default Value: 0h		
Format: OpCode		
20:16	SubOpcode B	
	Default Value: 0h	
	Format: OpCode	
15:12	Reserved	
	Format: MBZ	
11:0	DWord Length	
	Default Value: 0Ch Excludes DWord (0,1)	
	Format: =n 00Eh, used for normal decode and encode mode000h, a special case to provide a dummy image state for stitch mode operation. In this case, fields in DW1 which is part of the dummy image state command are ignored by hardware.	
1	31:16	Reserved
		Format: MBZ

MFX_AVC_IMG_STATE										
	15:0	<p>Frame Size</p> <table border="1"> <tr> <td>Format:</td> <td>U16-1 in MB unit</td> </tr> </table> <p>The value for FrameSizeInMBs must match the product of FrameWidthInMBs and FrameHeightInMBs.Max. Screen resolution is therefore limited to 256 x 256 in MB unit. It is enough to cover all the Profile-Level specified in the current HD-DVD specification. E.g., for 1920x1080, FrameSizeInMBs[15:0] = 8160 (1920/16 * 1088/16; rounded up 1080). This parameter is specified for Intel interface only, not present in the DXVA.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td></td> <td>representing Number of MBs [1,16384]</td> </tr> </tbody> </table>	Format:	U16-1 in MB unit	Value	Name	Description	[0,16383]		representing Number of MBs [1,16384]
	Format:	U16-1 in MB unit								
Value	Name	Description								
[0,16383]		representing Number of MBs [1,16384]								
2	31:24	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>(bit[31:24] must be zero to match the DXVA 16-bit definition for FrameHeightInMBsMinus1)</p>	Format:	MBZ						
	Format:	MBZ								
	23:16	<p>Frame Height</p> <table border="1"> <tr> <td>Format:</td> <td>U8-1 in MB unit</td> </tr> </table> <p>It is set to the value of (FrameHeightInMBsMinus1+ 1). Since the max value for FrameHeightInMBs is 255, the max allowed value for FrameHeightInMBsMinus1 is only 254. The min value for FrameHeightInMBs is 1.Although the max. value that can be specified for FrameHeightInMBs is 255 (in the current implementation), FrameWidthInMBs * FrameHeightInMBs must not exceed the max value of FrameSizeInMBs[14:0].e.g. for 1920x1080, FrameHeightInMBs[7:0] is equal to 68 (1080 divided by 16, and rounded up, i.e. effectively specified as 1088 instead).It is derived from FrameHeightInMbs = (2 - frame_mbs_only_flag) * PicHeightInMapUnits and PicHeightInMbs = FrameHeightInMbs / (1 + field_pic_flag) internally done. For MBAFF, PicHeightInMapUnits is in MB pair unit, so the bitstream sends only half frame height.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,255]</td> <td></td> <td>representing height [1,256]</td> </tr> </tbody> </table>	Format:	U8-1 in MB unit	Value	Name	Description	[0,255]		representing height [1,256]
Format:	U8-1 in MB unit									
Value	Name	Description								
[0,255]		representing height [1,256]								
15:8	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>(bit[15:8] must be zero to match the DXVA 16-bit definition for FrameWidthInMBsMinus1)</p>	Format:	MBZ							
Format:	MBZ									

MFX_AVC_IMG_STATE										
	7:0	<p>Frame Width</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U8-1 in MB unit</td> </tr> </table> <p>It is set to the value of (FrameWidthInMBsMinus1+ 1). Since the max value for FrameWidthInMBs is 255, the max allowed value for FrameWidthInMBsMinus1 is only 254. The min value for FrameWidthInMBs is 1. Although the max. value that can be specified for FrameWidthInMBs is 255 (in the current implementation), FrameWidthInMBs * FrameWidthInMBs must not exceed the max value of FrameSizeInMBs[14:0]. e.g. for 1920x1080, FrameHeightInMBs[7:0] is equal to 68 (1080 divided by 16, and rounded up, i.e. effectively specified as 1088 instead). It is derived from FrameWidthInMbs = (2 - frame_mbs_only_flag) * PicWidthInMapUnits and PicWidthInMbs = FrameWidthInMbs / (1 + field_pic_flag) internally done. For MBAFF, PicWidthInMapUnits is in MB pair unit, so the bitstream sends only half frame width.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,255]</td> <td></td> <td>representing width [1,256]</td> </tr> </tbody> </table>	Format:	U8-1 in MB unit	Value	Name	Description	[0,255]		representing width [1,256]
	Format:	U8-1 in MB unit								
	Value	Name	Description							
[0,255]		representing width [1,256]								
3	31:29	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table> <p>(bit[31:29] must be zero to match the DXVA2 8-bit definition for InitQpChroma[1])</p>	Format:	MBZ						
	Format:	MBZ								
	28:24	<p>Second Chroma QP Offset</p> <p>Signed integer value. It should be in the range of -12 to +12 (according to AVC spec). 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 [4:0] - chroma_qp_offset_bits (from the current active PPS) Chroma_qp_offset [9:5] - second_chroma_qp_offset_bits</p>								
	23:21	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table> <p>(bit[23:21] must be zero to match the DXVA2 8-bit definition for InitQpChroma[1])</p>	Format:	MBZ						
	Format:	MBZ								
	20:16	<p>First Chroma QP Offset</p> <p>Signed integer value. It should be in the range of -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) Chroma_qp_offset [9:5] - second_chroma_qp_offset_bits</p>								
15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ					
Project:	BDW									
Format:	MBZ									

MFX_AVC_IMG_STATE																	
	12	<p>Weighted_Pred_Flag</p> <table border="1"> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>(This field is defined differently from Gen6, Gen7 follows strictly DXVA2 AVC interface.)</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable [Default]</td> <td>specifies that weighted prediction is not used for P and SP slices</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>specifies that weighted prediction is used for P and SP slices</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This field must set to '0' for B and I pictures.</p>	Format:	Enable	Value	Name	Description	0	Disable [Default]	specifies that weighted prediction is not used for P and SP slices	1	Enable	specifies that weighted prediction is used for P and SP slices				
	Format:	Enable															
	Value	Name	Description														
	0	Disable [Default]	specifies that weighted prediction is not used for P and SP slices														
	1	Enable	specifies that weighted prediction is used for P and SP slices														
	11:10	<p>Weighted_BiPred_Idx</p> <p>(This field is defined differently from DevSNB; DevIVB follows strictly DXVA2 AVC interface.)</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>DEFAULT [Default]</td> <td>Specifies that the default weighted prediction is used for B slices</td> </tr> <tr> <td>1</td> <td>EXPLICIT</td> <td>Specifies that explicit weighted prediction is used for B slices</td> </tr> <tr> <td>2</td> <td>IMPLICIT</td> <td>Specifies that implicit weighted prediction is used for B slices.</td> </tr> <tr> <td>3</td> <td>Reserved</td> <td>Illegal value</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This field must set to 0 for P and I pictures.</p>	Value	Name	Description	0	DEFAULT [Default]	Specifies that the default weighted prediction is used for B slices	1	EXPLICIT	Specifies that explicit weighted prediction is used for B slices	2	IMPLICIT	Specifies that implicit weighted prediction is used for B slices.	3	Reserved	Illegal value
	Value	Name	Description														
	0	DEFAULT [Default]	Specifies that the default weighted prediction is used for B slices														
	1	EXPLICIT	Specifies that explicit weighted prediction is used for B slices														
	2	IMPLICIT	Specifies that implicit weighted prediction is used for B slices.														
	3	Reserved	Illegal value														
	9:8	<p>ImgStruct - Image Structure, img_structure[1:0]</p> <p>The current encoding picture structure can only takes on 3 possible values</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Frame Picture</td> </tr> <tr> <td>01b</td> <td>Top Field Picture</td> </tr> <tr> <td>11b</td> <td>Bottom Field Picture</td> </tr> <tr> <td>10b</td> <td>Invalid, not allowed.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>img_structure[0] can be used as a flag to distinguish between frame and field structure. It must be consistent with the field_pic_flag setting in the Slice Header. This parameter is specified for Intel interface only, not present in the DXVA as a separate state (instead the img_structure[1] is embedded inside the DXVA picture definition).</p>	Value	Name	00b	Frame Picture	01b	Top Field Picture	11b	Bottom Field Picture	10b	Invalid, not allowed.					
Value	Name																
00b	Frame Picture																
01b	Top Field Picture																
11b	Bottom Field Picture																
10b	Invalid, not allowed.																
7:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
Format:	MBZ																

MFX_AVC_IMG_STATE												
4	31:16	<p>MinFrameWSize</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p>Minimum Frame Size [15:0] (in Word, 16-bit)(Encoder Only) Minimum Frame Size is specified to compensate for intel Rate Control Currently zero fill (no need to perform emulation byte insertion) is done only to the end of the CABAC_ZERO_WORD insertion (if any) at the last slice of a picture. Intel encoder parameter, not part of DXVA. The caller should always make sure that the value, represented by Minimum Frame Size, is always less than maximum frame size FrameBitRateMax (DWORD 10 bits 29:16). This field is reserved in Decode mode. The programmable range is $0 \dots 2^{18} - 1$ When MinFrameWSizeUnits is 00. Programmable range is $0 \dots 2^{20} - 1$ when MinFrameWSizeUnits is 01. Programmable range is $0 \dots 2^{26} - 1$ when MinFrameWSizeUnits is 10. Programmable range is $0 \dots 2^{32} - 1$ when MinFrameWSizeUnits is 11.</p>	Default Value:	0h	Format:	U16						
	Default Value:	0h										
	Format:	U16										
15	<p>MbStatEnabled</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>Enable reading in MB status buffer (a.k.a. encoding stream-out buffer) Note: For multi-pass encoder, all passes except the first one need to set this value to 1. By setting the first pass to 0, it does save some memory bandwidth. In VDenc mode this must be set to zero as no MB level rate control is used.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Disable</td> <td>Disable Reading of Macroblock Status Buffer</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Enable</td> <td>Enable Reading of Macroblock Status Buffer</td> </tr> </tbody> </table>	Format:	Enable	Value	Name	Description	0	Disable	Disable Reading of Macroblock Status Buffer	1	Enable	Enable Reading of Macroblock Status Buffer
Format:	Enable											
Value	Name	Description										
0	Disable	Disable Reading of Macroblock Status Buffer										
1	Enable	Enable Reading of Macroblock Status Buffer										
14	<p>LoadSlicePointerFlag</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>Enable</td> </tr> </table> <p>LoadBitStreamPointerPerSlice (Encoder-only) To support multiple slice picture and additional header/data insertion before and after an encoded slice. When this field is set to 0, bitstream pointer is only loaded once for the first slice of a frame. For subsequent slices in the frame, bitstream data are stitched together to form a single output data stream. When this field is set to 1, bitstream pointer is loaded for each slice of a frame. Basically bitstream data for different slices of a frame will be written to different memory locations.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Disable</td> <td>Load BitStream Pointer only once for the first slice of a frame</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Enable</td> <td>Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field</td> </tr> </tbody> </table>	Format:	Enable	Value	Name	Description	0	Disable	Load BitStream Pointer only once for the first slice of a frame	1	Enable	Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field
Format:	Enable											
Value	Name	Description										
0	Disable	Load BitStream Pointer only once for the first slice of a frame										
1	Enable	Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field										
13	<p>Reserved</p>											

MFX_AVC_IMG_STATE				
12	MvUnpackedFlag			
	MVUnPackedEnable (Encoder Only) This field is reserved in Decode mode.			
	Value	Name	Description	
	0	PACKED	use packed MV format (compliant to DXVA)	
	1	UNPACKED	use unpacked 8MV/32MV format only	
	11:10	ChromaFormatIdc		
		Chroma Format IDC, ChromaFormatIdc[1:0] It specifies the sampling of chroma component (Cb, Cr) in the current picture as follows :		
		Value	Name	Description
		00b	monochrome picture	Desc
		01b	4:2:0 picture	Desc
10b		4:2:2 picture (not supported)		
11b	4:4:4 picture (not supported)			
Programming Notes				
It is set to the value of the syntax element read from the current active SPS. The corresponding Monochrome Flag (monochrome_flag) can be derived from this field.				
9	Reserved			
	Format:	MBZ		
8	MbMvFormatFlag			
	Use MB level MvFormat flag (Encoder Only)			
	Value	Name	Description	Project
	0	IGNORE	HW PAK ignore MvFormat in the MB data. When bit 12 == 0, all MBs use packed MV format When bit 12 == 1, each MB data must use unpacked MV format, 8MV when there is no minor MV involved, and 32MV if there are some minor MVs.	
	1	FOLLOW	HW PAK will follow MvFormat value set within each MB data.	BDW
Programming Notes				
They must take one of the two values: the 8MV unpacked format (MvFormat =101b), and the 32MV unpacked format (MvFormat =110b). This bit can be set only when MvUnpackedFlag (bit 12 of this register) is set otherwise system could hang.				

MFX_AVC_IMG_STATE											
	7	<p>EntropyCodingFlag Entropy Coding Flag, entropy_coding_flag</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>CAVLC bit-serial encoding mode</td> <td>Desc</td> </tr> <tr> <td style="text-align: center;">1</td> <td>CABAC bit-serial encoding mode.</td> <td>Desc</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>It specifies one of the two possible bit stream encoding modes in the AVC. It is set to the value of the syntax element read from the current active PPS.</p>	Value	Name	Description	0	CAVLC bit-serial encoding mode	Desc	1	CABAC bit-serial encoding mode.	Desc
	Value	Name	Description								
	0	CAVLC bit-serial encoding mode	Desc								
	1	CABAC bit-serial encoding mode.	Desc								
	6	<p>ImgDisposableFlag Current Img Disposable Flag or Non-Reference Picture Flag</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>REFERENCE</td> <td>the current decoding picture may be used as a reference picture for others</td> </tr> <tr> <td style="text-align: center;">1</td> <td>DISPOSABLE</td> <td>the current decoding picture is not used as a reference picture (e.g. a B-picture cannot be a reference picture for any subsequent decoding)</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>It is derived from <code>ImgDisposableFlag = (nal_ref_idc == 0)</code>. <code>nal_ref_idc</code> is a syntax element from a NAL unit. When this flag is set, no reference picture and DMV are written out. This field is only valid for VLD decoding mode.</p>	Value	Name	Description	0	REFERENCE	the current decoding picture may be used as a reference picture for others	1	DISPOSABLE	the current decoding picture is not used as a reference picture (e.g. a B-picture cannot be a reference picture for any subsequent decoding)
	Value	Name	Description								
	0	REFERENCE	the current decoding picture may be used as a reference picture for others								
	1	DISPOSABLE	the current decoding picture is not used as a reference picture (e.g. a B-picture cannot be a reference picture for any subsequent decoding)								
	5	<p>ConstrainedIPredFlag Constrained Intra Prediction Flag, constrained_ipred_flagIt is set to the value of the syntax element in the current active PPS.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>INTRA_AND_INTER</td> <td>allows both intra and inter neighboring MB to be used in the intra-prediction encoding of the current MB.</td> </tr> <tr> <td style="text-align: center;">1</td> <td>INTRA_ONLY</td> <td>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.</td> </tr> </tbody> </table>	Value	Name	Description	0	INTRA_AND_INTER	allows both intra and inter neighboring MB to be used in the intra-prediction encoding of the current MB.	1	INTRA_ONLY	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.
	Value	Name	Description								
0	INTRA_AND_INTER	allows both intra and inter neighboring MB to be used in the intra-prediction encoding of the current MB.									
1	INTRA_ONLY	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.									

MFX_AVC_IMG_STATE										
4	<p>Direct8x8InfFlag Direct 8x8 Inference Flag, direct_8x8_inference_flagIt is set to the value of the syntax element in the current active SPS.It specifies the derivation process for luma motion vectors in the Direct MV coding modes (B_Skip, B_Direct_16x16 and B_Direct_8x8). When frame_mbs_only_flag is equal to 0, direct_8x8_inference_flag shall be equal to 1.It must be consistent with the frame_mbs_only_flag and transform_8x8_mode_flag settings.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>SUBBLOCK</td> <td>allows subpartitioning to go below 8x8 block size (i.e. 4x4, 8x4 or 4x8)</td> </tr> <tr> <td>1</td> <td>BLOCK</td> <td>allows processing only at 8x8 block size. MB Info is stored for 8x8 block size.</td> </tr> </tbody> </table>	Value	Name	Description	0	SUBBLOCK	allows subpartitioning to go below 8x8 block size (i.e. 4x4, 8x4 or 4x8)	1	BLOCK	allows processing only at 8x8 block size. MB Info is stored for 8x8 block size.
Value	Name	Description								
0	SUBBLOCK	allows subpartitioning to go below 8x8 block size (i.e. 4x4, 8x4 or 4x8)								
1	BLOCK	allows processing only at 8x8 block size. MB Info is stored for 8x8 block size.								
3	<p>Transform8x8Flag 8x8 IDCT Transform Mode Flag, trans8x8_mode_flagSpecifies 8x8 IDCT transform may be used in this pictureIt is set to the value of the syntax element in the current active PPS.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>4x4</td> <td>no 8x8 IDCT Transform, only 4x4 IDCT transform blocks are present</td> </tr> <tr> <td>1</td> <td>8x8</td> <td>8x8 Transform is allowed</td> </tr> </tbody> </table>	Value	Name	Description	0	4x4	no 8x8 IDCT Transform, only 4x4 IDCT transform blocks are present	1	8x8	8x8 Transform is allowed
Value	Name	Description								
0	4x4	no 8x8 IDCT Transform, only 4x4 IDCT transform blocks are present								
1	8x8	8x8 Transform is allowed								
2	<p>FrameMbOnlyFlag Frame MB only flag, frame_mbs_only_flagIt is set to the value of the syntax element in the current active SPS.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>FALSE</td> <td>not true ; effectively enables the possibility of MBAFF mode.</td> </tr> <tr> <td>1</td> <td>TRUE</td> <td>true, only frame MBs can occur in this sequence, hence disallows the MBAFF mode and field picture.</td> </tr> </tbody> </table>	Value	Name	Description	0	FALSE	not true ; effectively enables the possibility of MBAFF mode.	1	TRUE	true, only frame MBs can occur in this sequence, hence disallows the MBAFF mode and field picture.
Value	Name	Description								
0	FALSE	not true ; effectively enables the possibility of MBAFF mode.								
1	TRUE	true, only frame MBs can occur in this sequence, hence disallows the MBAFF mode and field picture.								
1	<p>MbaffFlameFlag MBAFF mode is active, mbaff_frame_flag.It is derived from MbaffFrameFlag = (mb_adaptive_frame_field_flag && ! field_pic_flag). mb_adaptive_frame_field_flag is a syntax element in the current active SPS and field_pic_flag is a syntax element in the current Slice Header. They both are present only if frame_mbs_only_flag is 0. Although mbaff_frame_flag is a Slice Header parameter, its value is expected to be the same for all the slices of a picture.It must be consistent with the mb_adaptive_frame_field_flag, the field_pic_flag and the frame_mbs_only_flag settings.This bit is valid only when the img_structure[1:0] indicates the current picture is a frame.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>FALSE</td> <td>not in MBAFF mode</td> </tr> <tr> <td>1</td> <td>TRUE</td> <td>in MBAFF mode</td> </tr> </tbody> </table>	Value	Name	Description	0	FALSE	not in MBAFF mode	1	TRUE	in MBAFF mode
Value	Name	Description								
0	FALSE	not in MBAFF mode								
1	TRUE	in MBAFF mode								

		MFX_AVC_IMG_STATE																																		
	0	<p>FieldPicFlag Field picture flag, field_pic_flag, specifies the current slice is a coded field or not. It is set to the same value as the syntax element in the Slice Header. It must be consistent with the img_structure[1:0] and the frame_mbs_only_flag settings. Although field_pic_flag is a Slice Header parameter, its value is expected to be the same for all the slices of a picture.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th colspan="2"></th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>FRAME</td> <td>a slice of a coded frame</td> <td colspan="2"></td> </tr> <tr> <td>1h</td> <td>FIELD</td> <td>a slice of a coded field</td> <td colspan="2"></td> </tr> </tbody> </table>				Value	Name	Description			0h	FRAME	a slice of a coded frame			1h	FIELD	a slice of a coded field																		
Value	Name	Description																																		
0h	FRAME	a slice of a coded frame																																		
1h	FIELD	a slice of a coded field																																		
5 [ExistsIf]Encode Only	31	<p>Trellis Quantization Enabled (TQEnb)</p> <table border="1"> <tr> <td>Format:</td> <td colspan="3">Enable</td> </tr> </table> <p>The TQ improves output video quality of AVC CABAC encoder by selecting quantized values for each non-zero coefficient so as to minimize the total R-D cost. This flag is only valid AVC CABAC mode. Otherwise, this flag should be disabled.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Use Normal</td> <td></td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Use Trellis quantization</td> <td>BDW</td> </tr> </tbody> </table>				Format:	Enable			Value	Name	Description	Project	0h	Disable	Use Normal		1h	Enable	Use Trellis quantization	BDW															
		Format:	Enable																																	
	Value	Name	Description	Project																																
	0h	Disable	Use Normal																																	
1h	Enable	Use Trellis quantization	BDW																																	
30:28	<p>Trellis Quantization Rounding (TQR) This rounding scheme is only applied to the quantized coefficients ranging from 0 to 1 when TQEnb is set to 1 in AVC CABAC mode. One of the following values is added to quantized coefficients before truncating fractional part.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td></td> <td>Add 1/8</td> <td>BDW</td> </tr> <tr> <td>001b</td> <td></td> <td>Add 2/8</td> <td>BDW</td> </tr> <tr> <td>010b</td> <td></td> <td>Add 3/8</td> <td>BDW</td> </tr> <tr> <td>011b</td> <td></td> <td>Add 4/8 (rounding 0.5)</td> <td>BDW</td> </tr> <tr> <td>100b</td> <td></td> <td>Add 5/8</td> <td>BDW</td> </tr> <tr> <td>101b</td> <td></td> <td>Add 6/8</td> <td>BDW</td> </tr> <tr> <td>110b</td> <td>Default</td> <td>Add 7/8 (Default rounding 0.875)</td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Description	Project	000b		Add 1/8	BDW	001b		Add 2/8	BDW	010b		Add 3/8	BDW	011b		Add 4/8 (rounding 0.5)	BDW	100b		Add 5/8	BDW	101b		Add 6/8	BDW	110b	Default	Add 7/8 (Default rounding 0.875)	BDW
	Value	Name	Description	Project																																
	000b		Add 1/8	BDW																																
	001b		Add 2/8	BDW																																
	010b		Add 3/8	BDW																																
	011b		Add 4/8 (rounding 0.5)	BDW																																
	100b		Add 5/8	BDW																																
101b		Add 6/8	BDW																																	
110b	Default	Add 7/8 (Default rounding 0.875)	BDW																																	
27	<p>Trellis Quantization Chroma Disable (TQChromaDisable) This signal is used to disable chroma TQ. To enable TQ for both luma and chroma, TQEnb=1, TQChromaDisable=0. To enable TQ only for luma, TQEnb=1, TQChromaDisable=1.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Enable Trellis Quantization chroma</td> <td>BDW</td> </tr> <tr> <td>1h</td> <td>Default</td> <td>Disable Trellis Quantization chroma</td> <td>BDW</td> </tr> </tbody> </table>				Value	Name	Description	Project	0h		Enable Trellis Quantization chroma	BDW	1h	Default	Disable Trellis Quantization chroma	BDW																				
	Value	Name	Description	Project																																
	0h		Enable Trellis Quantization chroma	BDW																																
1h	Default	Disable Trellis Quantization chroma	BDW																																	
<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td colspan="3">MBZ</td> </tr> </table>				Format:	MBZ																															
Format:	MBZ																																			
26:17																																				

MFX_AVC_IMG_STATE				
16	NonFirstPassFlag			
	This signals the current pass is not the first pass. It will imply designate HW behavior: e.g			
	Value	Name	Description	
	0h	Disable	Always use the MbQpY from initial PAK inline object for all passes of PAK	
	1h	Enable	Use MbQpY from stream-out buffer if MbRateCtrlFlag is set to 1	
	Reserved			
	Format:		MBZ	
	Reserved			
	Project:		BDW	
	Format:		MBZ	
	11:10	MinFrameWSizeUnits		
		This field is the Minimum Frame Size Units		
		Value	Name	Description
		00b	compatibility mode	Minimum Frame Size is in old mode (words, 2bytes)
		01b	16 byte	Minimum Frame Size is in 16bytes
		10b	4Kb	Minimum Frame Size is in 4Kbytes
11b	16Kb	Minimum Frame Size is in 16Kbytes		
9	MbRateCtrlFlag - MB level Rate Control Enabling Flag			
	MB Rate Control conformance mask			
	In VDenc mode, this field must be zero as frame level rate control is used.			
	Value	Name	Description	
	0h	Disable	Apply accumulative delta QP for consecutive passes on top of the macroblock QP values in inline data	
	1h	Enable	Apply RC QP delta to suggested QP values in Macroblock Status Buffer except the first pass.	
Programming Notes				
This field is ignored when MacroblockStatEnable is disabled or MB level Rate control flag for the current MB is disable in Macroblock Status Buffer.				
8	Reserved			
	Format:		MBZ	

MFX_AVC_IMG_STATE			
7	Intra/InterMblpcmFlag - ForceIPCMControlMask This field is to Force IPCM for Intra or Inter Macroblock size conformance mask.		
	Value	Name	Description
	0h	Disable	Do not change intra or Inter macroblocks even
	1h	Enable	Change intra or Inter macroblocks MB_type to IPCM
Programming Notes			
This field is ignored when MacroblockStatEnable is disabled or MB level Intra MB conformance flag for the current MB is disable in Macroblock Status Buffer.			
6:4	Reserved		
	Format:		MBZ
3	FrameSzUnderFlag - FrameBitRateMinReportMask This is a mask bit controlling if the condition of frame level bit count is less than FrameBitRateMin		
	Value	Name	Description
	0h	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.
	1h	Enable	set bit0 and bit 1of MFC_IMAGE_STATUS control register if the total frame level bit counter is less than or equal to Frame Bit rate Minimum limit.
2	FrameSzOverFlag - FrameBitRateMaxReportMask This is a mask bit controlling if the condition of frame level bit count exceeds FrameBitRateMax.		
	Value	Name	Description
	0	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.
	1	Enable	Set bit0 and bit 1 of MFC_IMAGE_STATUS control register if the total frame level bit counter is greater than or equal to Frame Bit rate Maximum limit.
1	InterMbMaxBitFlag - InterMBMaxSizeReportMask This is a mask bit controlling if the condition of any inter MB in the frame exceeds InterMBMaxSize.		
	Value	Name	Description
	0	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.
	1	Enable	Set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Inter MB Conformance Max size limit.

MFX_AVC_IMG_STATE												
	0	<p>IntraMbMaxBitFlag - IntraMBMaxSizeReportMask This is a mask bit controlling if the condition of any intra MB in the frame exceeds IntraMBMaxSize.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Do not update bit0 of MFC_IMAGE_STATUS control register.</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.	1	Enable	set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.	
Value	Name	Description										
0h	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.										
1	Enable	set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.										
6 [ExistsIf]Encode Only	31:28	Reserved										
	27:16	<p>InterMbMaxSz</p> <table border="1"> <tr> <td>Format:</td> <td>U12</td> </tr> </table> <p>This field, Inter MB Conformance Max size limit, indicates the allowed max bit count size for Inter MB</p>	Format:	U12								
	Format:	U12										
	15:12	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ											
11:0	<p>IntraMbMaxSz</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Intra Only</td> </tr> <tr> <td>Format:</td> <td>U12</td> </tr> </table> <p>This field, Intra MB Conformance Max size limit, indicates the allowed max bit count size for Intra MB</p> <p>All IPCM MBs should ignore this Max size limit.</p>	Exists If:	//Intra Only	Format:	U12							
Exists If:	//Intra Only											
Format:	U12											
7 [ExistsIf]Encode Only	31:1	Reserved										
	0	<p>VSL Top MB Trans8x8flag</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable [Default]</td> <td>VSL will only fetch the current MB data.</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>When this bit is set VSL will make extra fetch to memory to fetch the MB data for top MB.</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	Description	0	Disable [Default]	VSL will only fetch the current MB data.	1	Enable
Project:	BDW											
Value	Name	Description										
0	Disable [Default]	VSL will only fetch the current MB data.										
1	Enable	When this bit is set VSL will make extra fetch to memory to fetch the MB data for top MB.										

MFX_AVC_IMG_STATE			
<p>8</p> <p>[ExistsIf]Encode Only</p>	<p>31:24 SliceDeltaQpMax[3]</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>S7</td> </tr> </table> <p>Range: [0:MAX_QP_DELTA]</p> <p>This field is the Slice level delta QP for total bit-count above FrameBitRateMax - first 1/8 region This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame exceeds FrameBitRateMax but is within 1/8 of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of (FrameBitRateMax, (FrameBitRateMax+ FrameBitRateMaxDelta»3).</p>	Format:	S7
	Format:	S7	
	<p>23:16 SliceDeltaQpMax[2]</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>Range: [0:MAX_QP_DELTA]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/8 and below 1/ 4 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between 1/8 and ¼ of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»3), (FrameBitRateMax+ FrameBitRateMaxDelta»2).</p>	Format:	U8
	Format:	U8	
<p>15:8 SliceDeltaQpMax[1]</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>S7</td> </tr> </table> <p>Range: [0:MAX_QP_DELTA]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/ 4 and below 1/2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between ¼ and ½ of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»2), (FrameBitRateMax+ FrameBitRateMaxDelta»1).</p>	Format:	S7	
Format:	S7		
<p>7:0 SliceDeltaQpPMax[0]</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>S7</td> </tr> </table> <p>Range: [0:MAX_QP_DELTA]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/ 2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is above FrameBitRateMax by more than half the distance of FrameBitRateMaxDelta , i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»1), infinite).</p>	Format:	S7	
Format:	S7		

MFX_AVC_IMG_STATE	
9 [ExistsIf]Encode Only	31:24 SliceDeltaQpMin[3] Format: S7 <hr/> Range: [0:MAX_QP_DELTA] This field is the Slice level delta QP for total bit-count below FrameBitRateMin - first 1/8 region This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is less than FrameBitRateMin and greater than or equal to 1/8 the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of [(FrameBitRateMin-FrameBitRateMinDelta»3), FrameBitRateMin).
	23:16 SliceDeltaQpMin[2] Format: S7 <hr/> Range: [0:MAX_QP_DELTA] This field is the Slice level delta QP for bit-count below FrameBitRateMin - below 1/ 8 and above 1/ 4 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between one-eighth and quarter the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of [(FrameBitRateMin- FrameBitRateMinDelta»2), (FrameBitRateMin- FrameBitRateMinDelta»3)).
	15:8 SliceDeltaQpMin[1] Format: S7 <hr/> Range: [0:MAX_QP_DELTA] This field is the Slice level delta QP for bit-count below FrameBitRateMin- below 1/4 and above 1/ 2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between quarter and half the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of [(FrameBitRateMin- FrameBitRateMinDelta»1), (FrameBitRateMin- FrameBitRateMinDelta»2)).
	7:0 SliceDeltaQpMin[0] Format: S7 <hr/> Range: [0:MAX_QP_DELTA] This field is the Slice Level Delta QP for bit-count below FrameBitRateMin - below 1/ 2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is below FrameBitRateMin by more than half the distance of FrameBitRateMinDelta , i.e., in the range of [0, (FrameBitRateMin- FrameBitRateMinDelta»1).

MFX_AVC_IMG_STATE											
10 [ExistsIf]Encode Only	31	<p>FrameBitrateMaxUnit This field is the Frame Bitrate Maximum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Byte</td> <td>FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0</td> </tr> <tr> <td>1</td> <td>Kilo Byte</td> <td>FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0</td> </tr> </tbody> </table>	Value	Name	Description	0	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0	1	Kilo Byte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0
	Value	Name	Description								
	0	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0								
	1	Kilo Byte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0								
	30	<p>FrameBitrateMaxUnitMode This field is the Frame Bitrate Maximum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>compatibility mode</td> <td>FrameBitRateMaxUnit is in old mode (128b/16Kb)</td> </tr> <tr> <td>1h</td> <td>New mode</td> <td>FrameBitRateMaxUnit is in new mode (32byte/4Kb)</td> </tr> </tbody> </table>	Value	Name	Description	0h	compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)	1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)
	Value	Name	Description								
	0h	compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)								
	1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)								
	29:16	<p>FrameBitRateMax This field is the Frame Bitrate Maximum Limit. This field along with FrameBitrateMaxUnit determines maximum allowed bits in a frame before multi-pass gets triggered (when enabled). In other words, multi-pass is triggered when the actual frame byte count exceeds this value. When FrameBitrateMaxUnitMode is 0(compatibility mode) bits 16:27 should be used, bits 28 and 29 should be 0..</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0-512KB</td> <td></td> <td>The programmable range is 0-512KB when FrameBitrateMaxUnit is 0.</td> </tr> <tr> <td>0-8190KB</td> <td></td> <td>The programmable range is 0-8190KB when FrameBitrateMaxUnit is 1.</td> </tr> </tbody> </table>	Value	Name	Description	0-512KB		The programmable range is 0-512KB when FrameBitrateMaxUnit is 0.	0-8190KB		The programmable range is 0-8190KB when FrameBitrateMaxUnit is 1.
	Value	Name	Description								
0-512KB		The programmable range is 0-512KB when FrameBitrateMaxUnit is 0.									
0-8190KB		The programmable range is 0-8190KB when FrameBitrateMaxUnit is 1.									
15	<p>FrameBitrateMinUnit This field is the Frame Bitrate Minimum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Byte</td> <td>FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0</td> </tr> <tr> <td>1</td> <td>Kilo Byte</td> <td>FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0</td> </tr> </tbody> </table>	Value	Name	Description	0	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0	1	Kilo Byte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0	
Value	Name	Description									
0	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0									
1	Kilo Byte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0									
14	<p>FrameBitrateMinUnitMode This field is the Frame Bitrate Minimum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Compatibility mode</td> <td>FrameBitRateMaxUnit is in old mode (128b/16Kb)</td> </tr> <tr> <td>1h</td> <td>New mode</td> <td>FrameBitRateMaxUnit is in new mode (32byte/4Kb)</td> </tr> </tbody> </table>	Value	Name	Description	0h	Compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)	1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)	
Value	Name	Description									
0h	Compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)									
1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)									

MFX_AVC_IMG_STATE																
	13:0	<p>FrameBitRateMin</p> <p>RangeThe programmable range 0-512KB When FrameBitrateMinUnit is in 0.Programmable range is 0-8190 KB when FrameBitrateMinUnit is in 1.This field is the Frame Bitrate Minimum Limit ()This field along with FrameBitrateMinUnit determines minimum allowed bits in a Frame before Multi-Pass gets triggered (when enabled). In other words, multi-pass is triggered when the actual frame byte count is less than this value. When FrameBitrateMinUnitMode is 0 (compatibility mode) bits 0:11 should be used, bits 12 and 13 should be 0.</p>														
11	31	Slice Stats Streamout Enable														
	30:16	<p>FrameBitRateMaxDelta</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U15</td> </tr> </table> <p>This field is used to select the slice delta QP when FrameBitRateMax Is exceeded. It shares the same FrameBitrateMaxUnit. When FrameBitrateMaxUnitMode is 0(compatibility mode) bits 16:27 should be used, bits 28, 29 and 30 should be 0.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>0-1024KB</td> <td></td> <td>The Programmable range 0-1024KB when FrameBitRateMaxUnit is 0.</td> </tr> <tr> <td>0-16380KB</td> <td></td> <td>The Programmable range is 0-16380KB when FrameBitRateMaxUnit is 1.</td> </tr> <tr> <td>0h</td> <td style="text-align: center;">[Default]</td> <td></td> </tr> </tbody> </table>	Format:	U15	Value	Name	Description	0-1024KB		The Programmable range 0-1024KB when FrameBitRateMaxUnit is 0.	0-16380KB		The Programmable range is 0-16380KB when FrameBitRateMaxUnit is 1.	0h	[Default]	
		Format:	U15													
		Value	Name	Description												
		0-1024KB		The Programmable range 0-1024KB when FrameBitRateMaxUnit is 0.												
0-16380KB		The Programmable range is 0-16380KB when FrameBitRateMaxUnit is 1.														
0h	[Default]															
15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ													
Format:	MBZ															
14:0	<p>FrameBitRateMinDelta</p> <p>Range: The programmable range 0-1024KB When FrameBitrateMinUnit is in 32Bytes.Programmable range is 0-16380KB when FrameBitrateMinUnit is in 4Kbytes.</p> <p>This field is used to select the slice delta QP when FrameBitRateMin Is exceeded. It shares the same FrameBitrateMinUnit. When FrameBitrateMinUnitMode is 0(compatibility mode) bits 0:11 should be used, bits 12, 13 and 14 should be 0.Note: HW requires the following condition $FrameBitRateMinDelta \leq 2 * FrameBitRateMin$ Must be true, otherwise it may cause unpredicted behavior.</p>															
12	31:21	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
	Format:	MBZ														
	20	<p>VMD Error Logic</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW												
Project:		BDW														
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable [Default]</td> <td></td> </tr> <tr> <td>1</td> <td>Enable</td> <td>Error Handling</td> </tr> </tbody> </table>		Value	Name	Description	0	Disable [Default]		1	Enable	Error Handling						
Value	Name	Description														
0	Disable [Default]															
1	Enable	Error Handling														

MFX_AVC_IMG_STATE											
	19	Reserved Format: MBZ									
	18	VAD Error Logic Project: BDW <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Enable [Default]</td> <td>Error reporting ON in case of premature Slice done</td> </tr> <tr> <td>1</td> <td>Disable</td> <td>CABAC Engine will auto decode the bitstream in case of premature slice done.</td> </tr> </tbody> </table>	Value	Name	Description	0	Enable [Default]	Error reporting ON in case of premature Slice done	1	Disable	CABAC Engine will auto decode the bitstream in case of premature slice done.
	Value	Name	Description								
	0	Enable [Default]	Error reporting ON in case of premature Slice done								
	1	Disable	CABAC Engine will auto decode the bitstream in case of premature slice done.								
17	Reserved Project: BDW										
16	Reserved										
15:0	Reserved Format: MBZ										
13	31:30	Reserved Project: All Format: MBZ									
	29	Current Picture Has Performed MMC05 Set to 1 if the current Pic has performed the memory_management_control_operation = = 5.									
	28:24	Number of Reference Frames Format: U5 Range: Range 0 to MaxDpbSize (= 16 for Level 4.1) Specifies the maximum number of reference frames (frames, field pairs, unpaired field) existed in the current DBP for decoding the current picture.									
	23:22	Reserved Format: MBZ									
	21:16	Number of Active Reference Pictures from L1 Format: U6-1 Specifies the initial maximum reference index value minus 1 to access the L1 Reference List. It is extracted from PPS. It corresponds to the number of active reference pictures from L1 to decode the current picture. It can be modified by the slice header if num_ref_idx_active_override_flag is set. Only valid for B picture. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> </tr> </tbody> </table>	Value	Name	[0,31]						
	Value	Name									
	[0,31]										
15:14	Reserved Format: MBZ										

MFX_AVC_IMG_STATE								
	13:8	<p>Number of Active Reference Pictures from L0</p> <table border="1"> <tr> <td>Format:</td> <td>U6-1</td> </tr> </table> <p>Specifies the initial maximum reference index value minus 1 to access the L0 Reference List. It is extracted from PPS. It corresponds to the number of active reference pictures from L0 to decode the current picture. It can be modified by the slice header if num_ref_idx_active_override_flag is set. Valid for both P and B pictures.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[0,31]</td> <td></td> </tr> </tbody> </table>	Format:	U6-1	Value	Name	[0,31]	
	Format:	U6-1						
Value	Name							
[0,31]								
	7:0	<p>Initial QP Value</p> <table border="1"> <tr> <td>Format:</td> <td>S7</td> </tr> </table> <p>Range: [-26,25]</p> <p>Initial QP value for a Slice, extracted from PPS. It may further get modified by slice_qp_delta in slice header and mb_qp_delta in MB header.</p>	Format:	S7				
Format:	S7							
14 [ExistsIf] Short Format only	31:24	<p>Log2_max_pic_order_cnt_lsb_minus4</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a SPS syntax element, used to determine how many bits in the bitstream are used to represent pic_order_cnt_lsb syntax element in the slice header. Unsigned</p>	Exists If:	//Short Format Only				
	Exists If:	//Short Format Only						
	23:16	<p>Log2_max_frame_num_minus4</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a SPS syntax element, used to determine how many bits in the bitstream are used to represent frame_num syntax element in the slice header. Unsigned.</p>	Exists If:	//Short Format Only				
	Exists If:	//Short Format Only						
	15	<p>deblocking_filter_control_present_flag</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a PPS syntax element, indicates if more deblocking filter control syntax elements are present in the slice header.</p>	Exists If:	//Short Format Only				
Exists If:	//Short Format Only							
14:12	<p>num_slice_groups_minus1</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>BitField It is a PPS syntax element. Use for Slice Header parsing only, to read in slice_group_change_cycle, if any, but is not used by H/W, i.e. no slice group support. Desc</p>	Exists If:	//Short Format Only					
Exists If:	//Short Format Only							
11	<p>redundant_pic_cnt_present_flag</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a PPS syntax element. Use for Slice Header parsing only, to read-in redundant_pic_cnt, if any, but is not used by H/W, i.e. no support for redundant slice processing.</p>	Exists If:	//Short Format Only					
Exists If:	//Short Format Only							

MFX_AVC_IMG_STATE									
	10:8	<p>slice_group_map_type</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a PPS syntax element. Use for Slice Header parsing only, to read in slice_group_change_cycle, if any, but is not used by H/W, i.e. no slice group support.</p>	Exists If:	//Short Format Only					
	Exists If:	//Short Format Only							
	7:4	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>IDR flag is decoded from NAL Header Byte</p>	Format:	MBZ					
	Format:	MBZ							
	3:2	<p>Pic_order_cnt_type</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a SPS syntax element. Use for Slice Header parsing only.</p>	Exists If:	//Short Format Only					
Exists If:	//Short Format Only								
1	<p>Delta_pic_order_always_zero_flag</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a SPS syntax element. Use for Slice Header parsing only.</p>	Exists If:	//Short Format Only						
Exists If:	//Short Format Only								
0	<p>Pic_order_present_flag</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a PPS syntax element. Use for Slice Header parsing only.</p>	Exists If:	//Short Format Only						
Exists If:	//Short Format Only								
15 [ExistsIf] Short Format only	31:16	<p>Curr Pic Frame Num</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p>Derived from Slice Header syntax element</p>	Exists If:	//Short Format Only	Format:	U16			
	Exists If:	//Short Format Only							
Format:	U16								
15:0	<p>Slice Group Change Rate</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> <tr> <td>Format:</td> <td>U16-1</td> </tr> </table> <p>It is a PPS syntax element Use for Slice Header parsing only, to read in slice_group_change_cycle, if any, but is not used by H/W, i.e. no slice group support.</p>	Exists If:	//Short Format Only	Format:	U16-1				
Exists If:	//Short Format Only								
Format:	U16-1								
16 [ExistsIf]: Short Format only	31	<p>Inter View Order Disable</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It indicates how to append inter-view picture into initial sorted reference list. (due to ambiguity in the MVC Spec)</p>	Project:	BDW	Exists If:	//Short Format Only			
		Project:	BDW						
		Exists If:	//Short Format Only						
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Default [Default]</td> <td>View Order Ascending</td> </tr> <tr> <td>1h</td> <td>Disable</td> <td>View ID Ascending</td> </tr> </tbody> </table>	Value	Name	Description	0h	Default [Default]	View Order Ascending	1h
Value	Name	Description							
0h	Default [Default]	View Order Ascending							
1h	Disable	View ID Ascending							



MFX_AVC_IMG_STATE						
	30:22	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
	Project:	BDW				
	Format:	MBZ				
	21:18	<p>Max View IDX1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It is a PPS syntax element corresponding to Anchor/Non-Anchor Reference ListL1 It indicates the maximum number of inter-view picture for Reference List L1</p>	Project:	BDW	Exists If:	//Short Format Only
	Project:	BDW				
	Exists If:	//Short Format Only				
17:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
15:12	<p>Max View IDXLO</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>Reference ListL0 It indicates the maximum number of inter-view picture for Reference List L0</p>	Project:	BDW	Exists If:	//Short Format Only	
Project:	BDW					
Exists If:	//Short Format Only					
11:10	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
9:0	<p>Current Frame View ID</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Short Format Only</td> </tr> </table> <p>It indicates the View ID of the current decoding frame</p>	Project:	BDW	Exists If:	//Short Format Only	
Project:	BDW					
Exists If:	//Short Format Only					

MFX_AVC_REF_IDX_STATE

MFX_AVC_REF_IDX_STATE				
Project:	BDW			
Source:	VideoCS			
Length Bias:	2			
<p>This is a slice level command and can be issued multiple times within a picture that is comprised of multiple slices. The same command is used for AVC encoder (PAK mode) and decoder (VLD mode); it is not need in decoder IT mode.</p> <p>The inline data of this command is interpreted differently for encoder as for decoder. For decoder, it is interpreted as RefIdx List L0/L1 as in AVC spec., and it matches with the DXVA2 AVC API data structure for decoder in VLD mode : RefPicList[2][32] (L0:L1, 0:31 RefPic). But for encoder, it is interpreted as a Reference Index Mapping Table for L0 and L1 reference pictures. For packing the bits at the output of PAK, the syntax elements must follow the definition of RefIdxL0/L1 list according to the AVC spec. However, the decoder pipeline was designed to use a variation of that standard definition, as such a conversion (mapping) is needed to support the hardware design.</p> <p>The Reference lists are needed in processing both P and B slice in AVC codec. For P-MB, only L0 list is used; for B-MB both L0 and L1 lists are needed. For a B-MB that is coded in L1-only Prediction, only L1 list is used.</p>				
Programming Notes				
<p>DXVA2 specifies that an application will create the RefPicList L0 and L1 and pass onto the driver. The content of each entry of RefPicList L0/L1 is a 7-bit picture index. This picture index is the same as that of RefFrameList content. This picture index, however, is not defined the same as the frame store ID (0 to 16, 5-bits) we have implemented in H/W. Hence, driver is required to manage a table to convert between DXVA2 picture index and intel frame store ID. As such, the final RefPicList L0/L1 that the driver passes onto the H/W is not the same as that defined in the DXVA2.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h PARALLEL_VIDEO_PIPE	
		Format:	OpCode	
	28:27	28:27	Pipeline	
			Default Value:	2h MFX_AVC_REF_IDX_STATE
			Format:	OpCode
	26:24	26:24	Command Opcode	
			Default Value:	1h AVC
			Format:	OpCode
	23:21	23:21	SubOpcodeA	
			Default Value:	0h MFX_AVC_REF_IDX_STATE
			Format:	OpCode
	20:16	20:16	SubOpcodeB	
			Default Value:	4h MFX_AVC_REF_IDX_STATE
			Format:	OpCode



MFX_AVC_REF_IDX_STATE											
	15:12	Reserved Format: MBZ									
	11:0	DWord Length Default Value: 0008h Format: =n Excludes DWords 0,1									
	1	31:1 Reserved Format: MBZ									
	0	RefPicList Select Num_ref_idx_l1_active is resulted from the specifications in both PPS and Slice Header for the current slice. However, since the full reference list L0 and/or L1 are always sent, only present flags are specified instead. This parameter is specified for Intel interface only, not present in the DXVA. <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>RefPicList 0</td> <td>The list that followed represents RefList L0 (Decoder VLD mode) or Ref Idx Mapping Table L0 (Encoder PAK mode)</td> </tr> <tr> <td style="text-align: center;">1</td> <td>RefPicList1</td> <td>The list that followed represents RefList L1 (Decoder VLD mode) or Ref Idx Mapping Table L1 (Encoder PAK mode)</td> </tr> </tbody> </table>	Value	Name	Description	0	RefPicList 0	The list that followed represents RefList L0 (Decoder VLD mode) or Ref Idx Mapping Table L0 (Encoder PAK mode)	1	RefPicList1	The list that followed represents RefList L1 (Decoder VLD mode) or Ref Idx Mapping Table L1 (Encoder PAK mode)
Value	Name	Description									
0	RefPicList 0	The list that followed represents RefList L0 (Decoder VLD mode) or Ref Idx Mapping Table L0 (Encoder PAK mode)									
1	RefPicList1	The list that followed represents RefList L1 (Decoder VLD mode) or Ref Idx Mapping Table L1 (Encoder PAK mode)									

MFX_AVC_REF_IDX_STATE

2..9	31:0	<p>Reference List Entry</p> <p>This set of fields is always present whenever this command is issued. It always specifies the full 32 reference pictures in the selected list, regardless they are "existing picture" or not. If a picture is non-existing, the corresponding entry should be set to all ones. Each list entry is 1 byte. A 32-bit DW can hold 4 list entries in the following format</p> <ul style="list-style-type: none"> • 31:24 entry X+3 (e.g. listY_3) • 23:16 entry X+2 (e.g. listY_2) • 15:8 entry X+1 (e.g. listY_1) • 7:0 entry X (e.g. listY_0) <p>X is replaced by the $\text{paddr}[2:0] * 4$; $\text{paddr}[5:0]$ with 0x20 and 0x27, and Y is replaced by 0 or 1. The byte definition for a reference picture :</p> <ul style="list-style-type: none"> • Bit 7 : Non-Existing - indicates that frame store index that should have been at this entry did not exist and was replaced by an index 0 (a valid entry) for error concealment • Bit 6 : Long term bit - set this reference picture to be used as long term reference • Bit 5 : Field picture flag - indicates frame/field • Bit 4:0 : Frame store index or Frame Store ID (Bit 4:1 is used to form the binding table index in intel implementation) <p>This is the final Reference List L0 or L1 after any reordering specified in the Slice Header as well as modified by the driver, and its indices values are all translated to the intel specification. If the reference picture is a frame (Bit5 = 1), frame store ID is always an even number. This list is used in outputting MV information by the BSD unit in VLD mode. DMV access also reads and writes Mvlist0 using this frame store ID. If this set of fields is interpreted as Reference Index Mapping Table L0/L1, the same field alignment is followed, i.e. 4 mapping entries per DW. Each mapping entry is one byte in size, but only the least significant 5 bits [4:0] is relevant. Driver should zero all the upper bits [7:5] for each entry.</p>
------	------	--

MFX_AVC_SLICE_STATE

MFX_AVC_SLICE_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
This is a slice level command and can be issued multiple times within a picture that is comprised of multiple slices. The same command is used for AVC encoder (PAK mode) and decoder (VLD and IT modes).			
Programming Notes			
MFX_AVC_SLICE_STATE command is not issued for AVC DXVA2 Short Format Bitstream decode, instead MFD_AVC_SLICEADDR command is executed to retrieve the next slice MB Start Address X and Y by H/W itself.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_AVC_SLICE_STATE
		Format:	OpCode
	26:24	Command Opcode	
		Default Value:	1h AVC
		Format:	OpCode
	23:21	SubOpcodeA	
Default Value:		0h MFX_AVC_SLICE_STATE	
Format:		OpCode	
20:16	Command SubOpcodeB		
	Default Value:	3h MFX_AVC_SLICE_STATE	
	Format:	OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	11:0	DWord Length	
		Default Value:	8h DWORD_COUNT_n
	Format:	=n	
	Excludes DWords 0,1		
1	31:4	Reserved	
		Format:	MBZ

MFX_AVC_SLICE_STATE											
3:0	<p>Slice Type It is set to the value of the syntax element read from the Slice Header.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0000b</td> <td>P Slice</td> </tr> <tr> <td style="text-align: center;">0001b</td> <td>B Slice</td> </tr> <tr> <td style="text-align: center;">0010b</td> <td>I Slice</td> </tr> <tr> <td style="text-align: center;">0011b-1111b</td> <td>Reserved</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Bits[3:2] must be 0</p>	Value	Name	0000b	P Slice	0001b	B Slice	0010b	I Slice	0011b-1111b	Reserved
	Value	Name									
	0000b	P Slice									
	0001b	B Slice									
	0010b	I Slice									
	0011b-1111b	Reserved									
	2	<p>31:30 Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:		MBZ									
<p>29:24 Number of Reference Pictures in Inter-prediction List 1</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is valid only for encoding a B Slice, for which it is expected to have at least one entry in the reference list L1; otherwise (if Slice Type is not a B Slice), this field must be set to 0. This field can be derived for a B Slice from the Slice Header syntax element NumRefIdxActiveMinus1 as, Num_Ref_Idx_L1 = NumRefIdxActiveMinus1[1] + 1.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0-32</td> <td></td> </tr> </tbody> </table>		Format:	U6	Value	Name	0-32					
Format:		U6									
Value		Name									
0-32											
<p>23:22 Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>		Format:	MBZ								
Format:		MBZ									
<p>21:16 Number of Reference Pictures in Inter-prediction List 0</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is valid for encoding a P or B Slice, for which it is expected to have at least one entry in the reference list L0; otherwise (if Slice Type is not a P or B Slice), this field must be set to 0. This field can be derived for a P or B Slice from the Slice Header syntax element NumRefIdxActiveMinus1 as, Num_Ref_Idx_L0 = NumRefIdxActiveMinus1[0] + 1.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0-32</td> <td></td> </tr> </tbody> </table>		Format:	U6	Value	Name	0-32					
Format:		U6									
Value	Name										
0-32											
<p>15:11 Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
Format:	MBZ										
<p>10:8 Log 2 Weight Denom Chroma</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>U3</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0-7</td> <td></td> </tr> </tbody> </table>	Format:	U3	Value	Name	0-7						
Format:	U3										
Value	Name										
0-7											
<p>7:3 Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
Format:	MBZ										



MFX_AVC_SLICE_STATE							
2:0	<p>Log 2 Weight Denom Luma</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U3</td> </tr> </table> <p>It is the base 2 logarithm of the denominator for all Luma weighting factors. It is set to the value of the syntax element read from the Slice Header Pred_Weight_Table().</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0-7</td> <td></td> </tr> </tbody> </table>	Format:	U3	Value	Name	0-7	
Format:	U3						
Value	Name						
0-7							
3	<p>31:30 Weighted Prediction Indicator</p> <p>This field indicates the Weighted Prediction mode for a P or B Slice. It is a combined field corresponding to the syntax element WeightedBiPredIdc or WeightedPredFlag read from the current active PPS.</p> <ul style="list-style-type: none"> If it is a B-Slice, these bits are interpreted as: <ul style="list-style-type: none"> 00b - Specifies the default weighted inter-prediction to be applied 01b - Specifies the explicit weighted inter-prediction to be applied 10b - Specifies the implicit weighted inter-prediction to be applied 11b - Reserved (not allowed) If it is a P Slice, these bits are interpreted as: <ul style="list-style-type: none"> 00b - Disables weighted inter-prediction (Default weighted) 01b - Enables weighted inter-prediction (Explicit weighted) 10b - 11b - Reserved <p style="text-align: center;">Programming Notes</p> <p>Only when in B Slice with Weighted_Pred_Idc = 1 (explicit weighted prediction), will there be a L1 and/or a L0 weight+offset tables being sent to the BSD unit through the Slice_State command. Only when in P Slice with Weighted_Pred_Idc = 1, will there be a L0 weight+offset table being sent to the BSD.</p> <p>If Weighted_Pred_Idc != 1 for B Slice or Weighted_Pred_Idc =0 for P Slice, no Slice_State command should be issued to send these tables. If still being issued, the data is read but ignored.</p> <p>DXVA specifies Weighted_Bipred and Weighted_Pred in frame-level state. However, these two flags are combined and specified in slice level for both P and B slice type.</p>						
29	<p>Direct Prediction Type</p> <p>Type of direct prediction used for B Slices. This field is valid only for Slice_Type = B Slice; otherwise, it must be set to 0.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Temporal</td> </tr> <tr> <td>1</td> <td>Spatial</td> </tr> </tbody> </table>	Value	Name	0	Temporal	1	Spatial
Value	Name						
0	Temporal						
1	Spatial						

MFX_AVC_SLICE_STATE		
28:27	Disable Deblocking Filter Indicator	
	Value	Description
	00b	FilterInternalEdgesFlag is set equal to 1
	01b	Disable all deblocking operation, no deblocking parameter syntax element is read; filterInternalEdgesFlag is set equal to 0
	10b	Macroblocks in different slices are considered not available; filterInternalEdgesFlag is set equal to 1
11b	Reserved	Not defined in AVC
26	Reserved	
	Format:	MBZ
25:24	Cabac Init Idc[1:0]	
	Specifies the index for determining the initialization table used in the context variable initialization process.	
	Value	Name
	0-2	
Programming Notes		
Cabac initialization is also dependent on the field/frame picture type, Slice type, and the current SliceQP value.		
23:22	Reserved	
	Format:	MBZ
21:16	Slice Quantization Parameter	
	Quantization Parameter for current slice. Derived from PPS and slice_delta_qp syntax element in Slice Header. It is needed for CABAC context initialization and deblocking filter control. And it is also used as the starting QP value in the very first MB of a slice. It is in the range of unsigned integer 0 to 51, for 8-bit pixel bit-depth.	
15:12	Reserved	
	Format:	MBZ
11:8	Slice Beta Offset Div2	
	Format:	S3 2's Complement
	Range: [-6, 6] Inclusive	
	Specifies the offset used in accessing the deblocking filter strength tables.	
7:4	Reserved	
	Format:	MBZ

MFX_AVC_SLICE_STATE		
	3:0	Slice Alpha C0 Offset Div2 Format: S3 2's Complement Range: [-6, 6] Inclusive Specifies the offset used in accessing the deblocking filter strength tables.
		Slice Vertical Position This field specifies the position in y-direction of the first macroblock in the Slice in unit of macroblocks. The fields (Slice_MB_Start_Hor_Pos, Slice_MB_Start_Vert_Pos) are valid in VLD (decoding) mode only. They are ignored by hardware in decoding IT mode and encoding mode (whereas the position is provided by the per-macroblock object command). Derived <div style="text-align: center;">Programming Notes</div> Error Handling: Driver needs to check if FirstMbY starts at 0 on the first slice of frame. If not, driver needs to add a phantom slice with FirstMbX and FirstMbY set to 0.
4	31:24	Slice Horizontal Position This field specifies the position in x-direction of the first macroblock in the Slice in unit of macroblocks. Derived <div style="text-align: center;">Programming Notes</div> Error Handling: Driver needs to check if FirstMbY starts at 0 on the first slice of frame. If not, driver needs to add a phantom slice with FirstMbX and FirstMbY set to 0.
	23:16	Slice Horizontal Position This field specifies the position in x-direction of the first macroblock in the Slice in unit of macroblocks. Derived <div style="text-align: center;">Programming Notes</div> Error Handling: Driver needs to check if FirstMbY starts at 0 on the first slice of frame. If not, driver needs to add a phantom slice with FirstMbX and FirstMbY set to 0.
	15	Reserved Format: MBZ
	14:0	Slice Start Mb Num Exists If: //Decoder Only The MB number (linear MB address in a picture) at the start of a Slice, it must match with the Slice Horizontal Position (Slice_MB_Start_Hor_Pos) and Vertical Position (Slice_MB_Start_Vert_Pos) in the picture. <div style="text-align: center;">Programming Notes</div> In creating the Phantom Slice for error concealment, this field should set to the total number of MB in the current picture + 1.
		Reserved Format: MBZ
5	31:24	Reserved Format: MBZ
	23:16	Next Slice Vertical Position This field specifies the position in y-direction of the first macroblock in the next Slice in unit of macroblocks. 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 picture (since y-direction is zero-based numbering).
	15:8	Reserved Format: MBZ

MFX_AVC_SLICE_STATE																	
	7:0	<p>Next Slice Horizontal Position</p> <p>This field specifies the position in x-direction of the first macroblock in the next Slice in unit of macroblocks. This field is primarily used for error concealment. In the case that current slice is the last slice, this field should set to 0.</p>															
Encoder Only	31	<p>Rate Control Counter Enable</p> <p>To enable the accumulation of bit allocation for rate control This field enables hardware Rate Control logic. The rest of the RC control fields are only valid when this field is set to 1. Otherwise, hardware ignores these fields.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Disable</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Enable</td> </tr> </tbody> </table>	Value	Name	0	Disable	1	Enable									
		Value	Name														
		0	Disable														
	1	Enable															
	30	<p>ResetRateControlCounter</p> <p>To reset the bit allocation accumulation counter to 0 to restart the rate control.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Not Reset</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Reset</td> </tr> </tbody> </table>	Value	Name	0	Not Reset	1	Reset									
		Value	Name														
	0	Not Reset															
	1	Reset															
	29:28	<p>RC Triggler Mode</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">00b</td> <td>Always Rate Control</td> <td>Whereas RC becomes active if $sum_act > sum_target$ or $sum_act < sum_target$</td> </tr> <tr> <td style="text-align: center;">01b</td> <td>Gentle Rate Control</td> <td>whereas RC becomes active if $sum_act > upper_midpt$ or $sum_act < lower_midpt$</td> </tr> <tr> <td style="text-align: center;">10b</td> <td>Loose Rate Control</td> <td>whereas RC becomes active if $sum_act > sum_max$ or $sum_act < sum_min$</td> </tr> <tr> <td style="text-align: center;">11b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	00b	Always Rate Control	Whereas RC becomes active if $sum_act > sum_target$ or $sum_act < sum_target$	01b	Gentle Rate Control	whereas RC becomes active if $sum_act > upper_midpt$ or $sum_act < lower_midpt$	10b	Loose Rate Control	whereas RC becomes active if $sum_act > sum_max$ or $sum_act < sum_min$	11b	Reserved	
		Value	Name	Description													
00b		Always Rate Control	Whereas RC becomes active if $sum_act > sum_target$ or $sum_act < sum_target$														
01b		Gentle Rate Control	whereas RC becomes active if $sum_act > upper_midpt$ or $sum_act < lower_midpt$														
10b		Loose Rate Control	whereas RC becomes active if $sum_act > sum_max$ or $sum_act < sum_min$														
11b	Reserved																
27:24	<p>RC Stable Tolerance</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>This field specifies the tolerance required to deactivate RC once it has been triggered.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0-15</td> <td></td> </tr> </tbody> </table>	Format:	U4	Value	Name	0-15											
	Format:	U4															
Value	Name																
0-15																	
23	<p>RC Panic Enable</p> <p>If this field is set to 1, RC enters panic mode when $sum_act > sum_max$. RC Panic Type field controls what type of panic behavior is invoked.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Disable</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Enable</td> </tr> </tbody> </table>	Value	Name	0	Disable	1	Enable										
	Value	Name															
	0	Disable															
1	Enable																

MFX_AVC_SLICE_STATE			
22	RC Panic Type		
	This field selects between two RC Panic methods		
	Value	Name	
	0	QP Panic	
	1	CBP Panic	
Programming Notes			
If it is set to 0, in panic mode, the macroblock QP is maxed out, setting to requested QP + QP_max_pos_mod. If it is set to 1, for an intra macroblock, AC CBPs are set to zero (note that DC CBPs are not modified). For inter macroblocks, AC and DC CBPs are forced to zero.			
21	MB Type Direct Conversion Disable		
	Exists If:	//B-Slice	
	For all Macroblock type conversions in different slices, refer to Section "Macroblock Type Conversion Rules" in the same volume.		
	Value	Name	
	0	Enable direct mode conversion	
	1	Disable direct mode conversion	
Programming Notes			
This field is zero for all other slices other than B-Slice.			
20	MB Type Skip Conversion Disable		
	Exists If:	//P-Slice or B-Slice	
	For all Macroblock type conversions in different slices, refer to Section "Macroblock Type Conversion Rules" in the same volume.		
	Value	Name	
	0	Enable skip type conversion	
	1	Disable skip type conversion	
Programming Notes			
This field is zero for all other slices other than P_Slice or B-Slice. \			
19	Is Last Slice		
	It is used by the zero filling in the Minimum Frame Size test.		
	Value	Name	Description
	1		Current slice is the last slice of a picture
	0		Current slice is NOT the last slice of a picture
18	Reserved		

MFX_AVC_SLICE_STATE		
17	Header Insertion Present in Bitstream	
	Value	Name
	Description	
	0	No header insertion into the output bitstream buffer, in front of the current slice encoded bits.
1	Header insertion into the output bitstream buffer is present, and is in front of the current slice encoded bits.	
Programming Notes		
Note: In VDEnc mode, the slice header PAK object maximum size is 25 DWs.		
16	SliceData Insertion Present in Bitstream	
	Value	Name
	Description	
	0	No Slice Data insertion into the output bitstream buffer
1	Slice Data insertion into the output bitstream buffer is present.	
15	Tail Insertion Present in bitstream	
	Value	Name
	Description	
	0	No tail insertion into the output bitstream buffer, after the current slice encoded bits
1	Tail insertion into the output bitstream buffer is present, and is after the current slice encoded bits.	
14	Reserved	
	Format:	MBZ
13	EmulationByteSliceInsertEnable	
	To have PAK outputting SODB or EBSP to the output bitstream buffer	
	Value	Name
	Description	
0	outputting RBSP	
1	outputting EBSP	
12	CabacZeroWordInsertionEnable	
	To pad the end of a SliceLayer RBSP to meet the encoded size requirement.	
	Value	Name
	Description	
0	No Cabac_Zero_Word Insertion	
1	Allow internal Cabac_Zero_Word generation and append to the end of RBSP (effectively can be used as an indicator for last slice of a picture, if the assumption is only the last slice of a picture needs to insert CABAC_ZERO_WORDS.	
11:8	Reserved	
	Format:	MBZ
7:4	Slice ID [3:0]	
	To identify the output data (coding information record) returned for rate control from PAK to ENC and VPP.	

MFX_AVC_SLICE_STATE						
	3:2	Reserved Format: MBZ				
	1:0	Stream ID [1:0] To identify the output data (coding information record) returned for rate control from PAK to ENC and VPP.				
Encoder Only	31:29	Reserved Format: MBZ				
	28:0	Indirect PAK-BSE Data Start Address (Write) Exists If: //AVC Encode Mode This field specifies the memory starting address (offset) to write out the compressed bitstream data from the BSE processing. This pointer is relative to the MFC Indirect PAK-BSE Object Base Address. It is a byte-aligned address for the AVC bitstream data in both CABAC/CAVLC Modes. For Write, there is no need to have a data length field. It is assumed the global memory bound check specified in the IND_OBJ_BASE_ADDRESS command (Indirect PAK-BSE Object Access Upper Bound) will take care of any illegal write access. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 512MB</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 512MB	
Value	Name					
0 - 512MB						
Encoder Only	31:24	Magnitude of QP Max Negative Modifier Format: U8 This field specifies the lower limit of the QP modifier. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> </thead> <tbody> <tr> <td>0-51</td> <td></td> </tr> </tbody> </table>	Value	Name	0-51	
		Value	Name			
		0-51				
		23:16	Magnitude of QP Max Positive Modifier Format: U8 This field specifies the upper limit of the QP modifier. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15
	Value		Name			
	0 - 15					
	15:12		Shrink Param - Shrink Resistance Format: U4 This field specifies the additional points added each time decreased correction is invoked. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15
		Value	Name			
		0 - 15				
		11:8	Shrink Param - Shrink Init Format: U4 This field specifies the initial points required to trip decreased control. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%;">Value</th> <th style="width: 40%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15
	Value		Name			
	0 - 15					

MFX_AVC_SLICE_STATE																					
	7:4	<p>Grow Param - Grow Resistance</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td style="text-align: right;">U4</td> </tr> </table> <p>This field specifies the additional points added each time increased correction is invoked.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Format:	U4	Value	Name	0 - 15														
	Format:	U4																			
Value	Name																				
0 - 15																					
	3:0	<p>Grow Param - Grow Init</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td style="text-align: right;">U4</td> </tr> </table> <p>This field specifies the initial points required to trip increased control.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Format:	U4	Value	Name	0 - 15														
Format:	U4																				
Value	Name																				
0 - 15																					
Encoder Only	9																				
	31	<p>RoundInterEnable</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td style="text-align: right;">Enable</td> </tr> </table> <p>When this bit is not set, RoundInter defaults to 2 to match SNB.</p>	Format:	Enable																	
	Format:	Enable																			
30:28	<p>RoundInter</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td style="text-align: right;">U3</td> </tr> </table> <p>Rounding precision for Inter quantized coefficients</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>+1/16 [Default]</td> </tr> <tr> <td>001b</td> <td>+2/16</td> </tr> <tr> <td>010b</td> <td>+3/16</td> </tr> <tr> <td>011b</td> <td>+4/16</td> </tr> <tr> <td>100b</td> <td>+5/16</td> </tr> <tr> <td>101b</td> <td>+6/16</td> </tr> <tr> <td>110b</td> <td>+7/16</td> </tr> <tr> <td>111b</td> <td>+8/16</td> </tr> </tbody> </table>	Format:	U3	Value	Name	000b	+1/16 [Default]	001b	+2/16	010b	+3/16	011b	+4/16	100b	+5/16	101b	+6/16	110b	+7/16	111b	+8/16
Format:	U3																				
Value	Name																				
000b	+1/16 [Default]																				
001b	+2/16																				
010b	+3/16																				
011b	+4/16																				
100b	+5/16																				
101b	+6/16																				
110b	+7/16																				
111b	+8/16																				
	27	<p>RoundIntraEnable</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td style="text-align: right;">Enable</td> </tr> </table> <p>When this bit is not set, RoundIntra defaults to 4 to match SNB.</p>	Format:	Enable																	
Format:	Enable																				

MFX_AVC_SLICE_STATE																			
26:24	RoundIntra Format: U3 Rounding precision for Intra quantized coefficients <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>+1/16 [Default]</td> </tr> <tr> <td>001b</td> <td>+2/16</td> </tr> <tr> <td>010b</td> <td>+3/16</td> </tr> <tr> <td>011b</td> <td>+4/16</td> </tr> <tr> <td>100b</td> <td>+5/16</td> </tr> <tr> <td>101b</td> <td>+6/16</td> </tr> <tr> <td>110b</td> <td>+7/16</td> </tr> <tr> <td>111b</td> <td>+8/16</td> </tr> </tbody> </table>	Value	Name	000b	+1/16 [Default]	001b	+2/16	010b	+3/16	011b	+4/16	100b	+5/16	101b	+6/16	110b	+7/16	111b	+8/16
	Value	Name																	
	000b	+1/16 [Default]																	
	001b	+2/16																	
	010b	+3/16																	
	011b	+4/16																	
	100b	+5/16																	
	101b	+6/16																	
	110b	+7/16																	
	111b	+8/16																	
23:20	Correct 6 Format: U4 This field specifies the points used in the lowermost RC region when $sum_act \leq sum_min$. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15															
	Value	Name																	
	0 - 15																		
19:16	Correct 5 Format: U4 This field specifies the points used in the fifth RC region when $sum_act > sum_min$ but $\leq lower_midpt$. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15															
	Value	Name																	
	0 - 15																		
15:12	Correct 4 Format: U4 This field specifies the points used in the fourth RC region when $sum_act > lower_midpt$ but $\leq sum_target$. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15															
	Value	Name																	
	0 - 15																		
11:8	Correct 3 Format: U4 This field specifies the points used in the third RC region when $sum_act > sum_target$ but $\leq upper_midpt$. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name	0 - 15															
	Value	Name																	
	0 - 15																		

MFX_AVC_SLICE_STATE				
	7:4	Correct 2		
		Format: U4		
		This field specifies the points used in the second RC region when sum_act > upper_midpt but <= sum_max.		
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name
Value	Name			
0 - 15				
	3:0	Correct 1		
		Format: U4		
		This field specifies the points used in the topmost RC region when sum_act > sum_max.		
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0 - 15</td> <td></td> </tr> </tbody> </table>	Value	Name
Value	Name			
0 - 15				
Encoder Only	31:28	ClampValues - CV7		
	27:24	CV6		
	23:20	CV5		
	19:16	CV4		
	15:12	CV3		
	11:8	CV2		
	7:4	CV1		

MFX_AVC_SLICE_STATE

3:0

CV0 - Clamp Value 0

Format:	U4
---------	----

If the magnitude of coefficients at locations assigned with CV0 (mapping shown below) exceeds 2^{CV0-1} , they are replaced with 2^{CV0-1} . For coefficients at locations marked as 'none', no clamping is performed. The following mappings are only applied to luma and chroma blocks/subblocks containing AC coefficients (blocks/subblocks with only DC coeffs will not be clamped).

For 4x4 frame block, each coefficient is mapped to one of the eight CV values as following:

none	CV7	CV5	CV4
CV7	CV6	CV4	CV3
CV5	CV4	CV2	CV1
CV4	CV3	CV1	CV0

For 8x8 frame block, each coefficient is mapped to one of the eight CV values as following:

none	none	CV7	CV6	CV5	CV4	CV3	CV3
none	CV7	CV6	CV5	CV4	CV3	CV3	CV2
CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV2
CV6	CV5	CV4	CV3	CV3	CV2	CV2	CV1
CV5	CV4	CV3	CV3	CV2	CV2	CV1	CV1
CV4	CV3	CV3	CV2	CV2	CV1	CV1	CV0
CV3	CV3	CV2	CV2	CV1	CV1	CV0	CV0
CV3	CV2	CV2	CV1	CV1	CV0	CV0	CV0

For 4x4 field block, each coefficient is mapped to one of the eight CV values as following:

none	CV6	CV3	CV1
CV7	CV6	CV3	CV1
CV5	CV4	CV2	CV0
CV5	CV4	CV2	CV0

For 8x8 field block, each coefficient is mapped to one of the eight CV values as following:

none	none	CV6	CV5	CV4	CV3	CV2	CV1
none	CV7	CV6	CV5	CV4	CV3	CV2	CV1
CV7	CV6	CV5	CV4	CV3	CV3	CV2	CV1
CV7	CV6	CV5	CV4	CV3	CV2	CV2	CV1
CV6	CV5	CV4	CV4	CV3	CV2	CV1	CV0
CV6	CV5	CV4	CV3	CV2	CV2	CV1	CV0
CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0
CV5	CV5	CV4	CV3	CV2	CV1	CV1	CV0

Value	Name
0 - 15	

MFX_AVC_WEIGHTOFFSET_STATE

MFX_AVC_WEIGHTOFFSET_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a slice level command and can be issued multiple times within a picture that is comprised of multiple slices. The same command is used for AVC encoder (PAK mode) and decoder (VLD and IT modes). However, since for AVC decoder VLD and IT modes, and AVC encoder mode, the implicit weights are computed in hardware, this command is not issued. For encoder, regardless of the type of weight calculation is active for the current slice (default, implicit or explicit), they are all sent to the PAK as if they were all in explicit mode. However, for implicit weight and offset, each entry contains only a 16-bit weight and no offset (offset = 0 always in implicit mode and can be hard-coded inside the hardware).The weights (and offsets) are needed in processing both P and B slice in AVC codec. For P-MB, at most only L0 list is used; for B-MB both L0 and L1 lists may be needed. For a B-MB that is coded in L1-only Prediction, only L1 list is sent.The content of this command matches with the DXVA2 AVC API data structure for explicit prediction mode only : Weights[2][32][3][2] (L0:L1, 0:31 RefPic, Y:Cb:Cr, W:0)</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_AVC_WEIGHTOFFSET_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	1h AVC_COMMON
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	5h
Format:		OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	60h Excludes DWord (0,1)	
	Format:	=n Total Length - 2	

MFX_AVC_WEIGHTOFFSET_STATE										
1	31:1	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
	0	<p>Weight and Offset Select</p> <p>It must be set in consistent with the WeightedPredFlag and WeightedBiPredIdc in the Img_State command. This parameter is specified for Intel interface only, not present in the DXVA. For implicit even though only one entry may be used, still loading the whole 32-entry table.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Weight and Offset L0 table</td> <td>The list that followed is associated with the weight and offset for RefPicList L0</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Weight and Offset L1 table</td> <td>The list that followed is associated with the weight and offset for RefPicList L1</td> </tr> </tbody> </table>	Value	Name	Description	0	Weight and Offset L0 table	The list that followed is associated with the weight and offset for RefPicList L0	1	Weight and Offset L1 table
Value	Name	Description								
0	Weight and Offset L0 table	The list that followed is associated with the weight and offset for RefPicList L0								
1	Weight and Offset L1 table	The list that followed is associated with the weight and offset for RefPicList L1								
2..97	31:0	<p>WeightOffset</p> <p>WeightOffset[L=L0=0 or L1=1][i=0 to 31][Y=0/Cb=1/Cr=2][weight=0/offset=1] WeightOffset[L][i=0][Y=0][Weight=0], WeightOffset[L][i=0][Y=0][Offset=1] WeightOffset[L][i=0][Cb=1][Weight=0], WeightOffset[L][i=0][Cb=1][Offset=1] WeightOffset[L][i=0][Cr=2][Weight=0], WeightOffset[L][i=0][Cr=2][Offset=1]: WeightOffset[L][i=31][Y=0][Weight=0], WeightOffset[L][i=31][Y=0][Offset=1] WeightOffset[L][i=31][Cb=1][Weight=0], WeightOffset[L][i=31][Cb=1][Offset=1] WeightOffset[L][i=31][Cr=2][Weight=0], WeightOffset[L][i=31][Cr=2][Offset=1]</p> <p>Format for explicit: Both Weight and Offset are S15 in two's compliment, with a valid range from -128 to 128 Format for implicit: S15</p> <p>This set of fields is always present whenever this command is issued. The full table, one entry for each reference picture, is always specified. Any reference list L0/L1[i] that does not exist, the corresponding weight and offset are set to 0. Weight and Offset are 2 byte each. Apair of Weight and Offset forms a dword, with Weight in the LOWER word and Offset in the HIGHER word. WeightOffset[L0=0][i=0 to 31][Y=0] (i.e. luma_weight_10[i]) are specified for the weighting and offset factors applied to the luma prediction value for list 0 prediction using RefPicList0[i] (one-to-one correspondence in i). When luma_weight_10_flag (Slice Header syntax element) is equal to 1, the value of luma_weight_10[i] shall be in the range of -128 to 127. When luma_weight_10_flag is equal to 0, luma_weight_10[i] shall be inferred to be equal to 2luma_log2_weight_denom for RefPicList0[i]. luma_log2_weight_denom is a Slice Header syntax element. WeightOffset[L0=0][i=0 to 31][Cb=1] (i.e. chromaCb_weight_10[i]) are specified for the weighting and offset factors applied to the chroma Cb prediction values for list 0 prediction using RefPicList0[i] (one-to-one correspondence in i). When chroma_weight_10_flag (Slice Header syntax element) is equal to 1, the value of chromaCb_weight_10[i] shall be in the range of -128 to 127. When chroma_weight_10_flag is equal to 0, chromaCb_weight_10[i] shall be inferred to be equal to 2chroma_log2_weight_denom for RefPicList0[i]. chroma_log2_weight_denom is a Slice Header syntax element. WeightOffset[L0=0][i=0 to 31][Cr=2] (i.e. chromaCr_weight_10[i]) are specified for the weighting and offset factors applied to the chroma Cr prediction values for list 0 prediction using RefPicList0[i] (one-to-one correspondence in i). When chroma_weight_10_flag (Slice Header syntax element) is equal to 1, the value of chromaCr_weight_10[i] shall be in the range of -128 to 127. When chroma_weight_10_flag is equal to 0, chromaCr_weight_10[i] shall be inferred to be equal to 2chroma_log2_weight_denom for RefPicList0[i].</p>								

MFX_BSP_BUF_BASE_ADDR_STATE

MFX_BSP_BUF_BASE_ADDR_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This frame-level state command is used to specify all the buffer base addresses needed for the operation of the AVC Bit Stream Processing Units (for decoder, it is BSD Unit; for encoder, it is BSE Unit) For both encoder and decoder, currently it is assumed that all codec standards can share the same BSP_BUF_BASE_STATE. The simplicity of this command is the result of moving all the direct MV related processing into the ENC Subsystem. Since all implicit weight calculations and directMV calculations are done in ENC and all picture buffer management are done in the Host, there is no need to provide POC (POC List - FieldOrderCntList, CurrPic POC - CurrFieldOrderCnt) information to PAK. For decoder, all the direct mode information are sent in a separate slice-level command (AVC_DIRECTMODE_STATE command). In addition, in Encoder, the row stores for CABAC encoding and MB Parameters Construction (MPC) are combined into one single row store. The row stores specified in this command do not combine with those specified in the MFC_PIPE_BUF_ADDR_STATE command for hardware simplification reason.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Pipeline
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MFX_COMMON_STATE
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	4h
		Format:	OpCode
	15:12	Reserved	
		Project:	All
		Format:	MBZ

MFx_BSP_BUF_BASE_ADDR_STATE		
	11:0	DWord Length
		Default Value: 8h Excludes DWord (0,1)
		Project: All Format: =n Total Length - 2
1	31:6	BSD/MPC Row Store Scratch Buffer Base Address - Read/Write
		<p>This field provides the base address of the scratch buffer used by BSD (decoder) and MPC (encoder) unit to store MB information of the previous row for coding each macroblock in the current row. It is a private buffer used by the BSD (decoder) and MPC (encoder) hardware only. Its content is not accessible by software. This Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of the current macroblock to address this Row Store.</p> <p>For AVC BSD, 2 cacheline (CL) per MB when in MBAFF mode (row of MB pair); 1 CL per MB for non-MBAFF. So, to support 256 MBs per row (4K screen resolution), 2 * 256 * 64 bytes = 32,768 bytes are required. Cacheline alignment should be followed. For AVC MPC, 1 cacheline for non-MBAFF, 2 cachelines for MBAFF per MB. For VC1, the BSD row store is 512-bit (one cacheline) per MB, times the number of MBs per picture MB row.</p>
	5:0	Reserved
		Project: BDW Format: MBZ
2 Project: BDW	31:16	Reserved
		Project: BDW Format: MBZ Reserved for 64-bit address extension.
	15:0	BSD/MPC Row Store Scratch Buffer Base Address - Read/Write [47:32]
		Project: BDW
		Description
		This field is for the upper range of BSD/MPC Row Store Scratch Buffer Base Address. This field is used for 48-bit addressing.
	12	Reserved
		Project: BDW Format: MBZ
3 Project: BDW	31:15	Reserved
		Project: BDW Format: MBZ
	14:13	Reserved
		Project: BDW Format: MBZ
	12	Reserved
		Project: BDW Format: MBZ

MFX_BSP_BUF_BASE_ADDR_STATE																
	11	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
	Project:	BDW														
	Format:	MBZ														
	10:9	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
	Project:	BDW														
	Format:	MBZ														
	8:7	BSD/MPC Row Store Scratch Buffer - Arbitration Priority Control <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	U2	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
	Project:	BDW														
	Format:	U2														
	Value	Name														
	00b	Highest priority														
	01b	Second highest priority														
10b	Third highest priority															
11b	Lowest priority															
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for BSD/MPC Row Store Scratch Buffer Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)			
Project:	BDW															
Value	Name															
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)															
01b	Uncacheable (UC) - non-cacheable															
10b	Writethrough (WT)															
11b	Writeback (WB)															
4:3	Target Cache (TC) for BSD/MPC Row Store Scratch Buffer Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed			
Project:	BDW															
Value	Name															
00b	eLLC Only - not snooped in GT (BDW)															
01b	LLC Only															
10b	LLC/eLLC Allowed															
11b	L3, LLC, eLLC Allowed															
2	Reserved															



MFX_BSP_BUF_BASE_ADDR_STATE																
	1:0	<p>Age for QUADLRU (AGE) BSD/MPC Row Store Scratch Buffer Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW														
	Format:	Enable														
	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															
4	31:6	<p>MPR Row Store Scratch Buffer Base Address - Read/Write (Decoder Only)</p> <p>This field provides the base address of the scratch buffer used by decoder's MPR unit to store MB information of the previous row for decoding each macroblock in the current row. It is a private buffer used by the MPR hardware only. Its content is not accessible by software.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td>The MPR Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of each macroblock to address the MPR Row Store. Except ILDB Control Data, all other operations does not cross slice boundary. This field is specified in frame-level.2 cacheline (CL) per MB when in MBAFF mode (row of MB pair); 1 CL per MB for non-MBAFF, So, to support 256 MBs per row (4K screen resolution), 2 * 256 * 64 bytes = 32,768 bytes are required. Cacheline alignment should be followed. This field is only valid for AVC decoder mode</td> </tr> </tbody> </table>	Programming Notes	The MPR Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of each macroblock to address the MPR Row Store. Except ILDB Control Data, all other operations does not cross slice boundary. This field is specified in frame-level.2 cacheline (CL) per MB when in MBAFF mode (row of MB pair); 1 CL per MB for non-MBAFF, So, to support 256 MBs per row (4K screen resolution), 2 * 256 * 64 bytes = 32,768 bytes are required. Cacheline alignment should be followed. This field is only valid for AVC decoder mode												
	Programming Notes															
The MPR Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of each macroblock to address the MPR Row Store. Except ILDB Control Data, all other operations does not cross slice boundary. This field is specified in frame-level.2 cacheline (CL) per MB when in MBAFF mode (row of MB pair); 1 CL per MB for non-MBAFF, So, to support 256 MBs per row (4K screen resolution), 2 * 256 * 64 bytes = 32,768 bytes are required. Cacheline alignment should be followed. This field is only valid for AVC decoder mode																
5:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ											
Project:	BDW															
Format:	MBZ															
5 Project: BDW	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	All	Format:	MBZ										
		Project:	All													
	Format:	MBZ														
15:0	<p>MPR Row Store Scratch Buffer Base Address - Read/Write [47:32]</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <p>This field is for the upper range of MPR Row Store Scratch Buffer Base Address. This field is used for 48-bit addressing.</p>	Project:	All													
Project:	All															
6 Project: BDW	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
		Project:	BDW													
Format:	MBZ															

MFX_BSP_BUF_BASE_ADDR_STATE														
14:13	Reserved													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													
12	Reserved													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													
11	Reserved													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													
10:9	Reserved													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													
8:7	MPR Row Store Scratch Buffer - Arbitration Priority Control													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	U2	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b
Project:	BDW													
Format:	U2													
Value	Name													
00b	Highest priority													
01b	Second highest priority													
10b	Third highest priority													
11b	Lowest priority													
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MPR Row Store Scratch Buffer Base Address													
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)	
Project:	BDW													
Value	Name													
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
01b	Uncacheable (UC) - non-cacheable													
10b	Writethrough (WT)													
11b	Writeback (WB)													

MFX_BSP_BUF_BASE_ADDR_STATE																
	4:3	<p>Target Cache (TC) MPR Row Store Scratch Buffer Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed		
	Project:	BDW														
	Value	Name														
	00b	eLLC Only - not snooped in GT (BDW)														
	01b	LLC Only														
	10b	LLC/eLLC Allowed														
	11b	L3, LLC, eLLC Allowed														
	2	Reserved														
	1:0	<p>Age for QUADLRU (AGE) MPR Row Store Scratch Buffer Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW														
Format:	Enable															
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															
7	31:6	<p>Bitplane Read Buffer Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <p>It must be cacheline aligned (i.e. 64 bytes address boundary), so lower bit 0 to 5 are used for controlling information.(In Cantiga, this field must be dword aligned.)Bitplane buffer is a linear buffer. In VC1 Long format, it is written by an application. In VC1 Short Format, it is written and read by H/W only.For VC1 intel Long Format : it is a read-only bufferFor VC1 DXVA2 Short Format : it is a write and a read bufferThis field is only valid for VC1 decoder mode.</p>	Project:	All												
	Project:	All														
5:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ											
Project:	BDW															
Format:	MBZ															
8 Project: BDW	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	BDW	Format:	MBZ										
	Project:	BDW														
Format:	MBZ															
15:0	<p>Bitplane Read Buffer Base Address - Read/Write [47:32]</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <p>This field is for the upper range of Bitplane Read Buffer Base Address. This field is used for 48-bit addressing.</p>	Project:	All													
Project:	All															

MFX_BSP_BUF_BASE_ADDR_STATE			
9 Project: BDW	31:15	Reserved	
		Project: BDW	
		Format: MBZ	
	14:13	Reserved	
		Project: BDW	
		Format: MBZ	
	12:11	Reserved	
		Project: BDW	
		Format: MBZ	
	10:9	Reserved	
		Project: BDW	
		Format: MBZ	
	8:7	Bitplane Read Buffer - Arbitration Priority Control	
		Project: BDW	
Format: U2			
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	
6:5		Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Bitplane Read Buffer Base Address	
	Project: BDW		
	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.		
	Value	Name	
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	
	01b	Uncacheable (UC) - non-cacheable	
	10b	Writethrough (WT)	
	11b	Writeback (WB)	

MFX_BSP_BUF_BASE_ADDR_STATE

4:3	Target Cache (TC) Bitplane Read Buffer Base Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC</p> <p>00b: eLLC Only ("00" setting points TC selection to PTE which defaults to eLLC)</p> <p>01b: LLC Only (Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present).</p> <p>10b: LLC/eLLC Allowed.</p> <p>11b: L3, LLC, eLLC Allowed.</p> <p>Errata BDW:A-E (FIXED BY:G0 Stepping):</p> <p>For all system that does NOT use SVM (i.e. coherent L3\$ surfaces), back snoops from LLC has to be disabled (Dis_GtCvUpdtOnRd = "1"). Than target Cache settings can be programmed as POR requirements of L3/LLC/eDRAM caching.</p> <p>For all systems that does use SVM (i.e. coherent L3\$ surfaces), the recommended setting would be "00" in target cache settings. In case of L3 surfaces, the performance has to be tuned between "00" and "11" setting based on the benefits of L3 caching outweighing the degradation of backsnoops.</p> <p>Post G0-stepping, the above w/a for coherent L3\$ surfaces is not needed.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed		
Project:	BDW															
Value	Name															
00b	eLLC Only - not snooped in GT (BDW)															
01b	LLC Only															
10b	LLC/eLLC Allowed															
11b	L3, LLC, eLLC Allowed															
2	Reserved															
1:0	Age for QUADLRU (AGE) Bitplane Read Buffer Base Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITS, hence need to be replaced least often in caches.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
Project:	BDW															
Format:	Enable															
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															

MFX_DBK_OBJECT

MFX_DBK_OBJECT		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 3h PARALLEL_VIDEO_PIPE Format: OpCode
	28:27	Pipeline
		Default Value: 2h MFX_DBK_OBJECT Format: OpCode
	26:24	Media Command Opcode
		Default Value: 0h Common Format: OpCode
	23:21	SubOpcode A
		Default Value: 0h Format: OpCode
20:16	SubOpcode B	
	Default Value: 9h Format: OpCode	
15:12	Reserved	
11:0	DWord Length	
	Default Value: 0Bh Excludes DWord (0,1) Format: =n	
	Note: Regardless of the mode, inline data must be present in this command	
1	31:6	Pre Deblocking Source Address
		Format: GraphicsAddress[31:6] Specifies the 4K byte aligned frame buffer address for outputting the non-filtered reconstructed YUV picture (i.e. output of final adder in each codec standard, and prior to the deblocking filter unit).
	5:0	Reserved
	Project:	BDW
	Format:	MBZ

MFX_DBK_OBJECT			
2 Project: BDW	31:16	Reserved	
		Project: BDW Format: MBZ	
	15:0	Pre Deblocking Source Address High	
		Project: BDW Format: GraphicsAddress[47:32]	
This field is for the upper range of Pre-Deblocking Source Address. This field is used for 48-bit addressing.			
3 Project: BDW	31:15	Reserved	
		Project: BDW Format: MBZ	
	14:13	Reserved	
		Project: BDW Format: MBZ	
	12:11	Reserved	
		Project: BDW Format: MBZ	
	10:9	Reserved	
		Project: BDW Format: MBZ	
	8:7	Pre Deblocking Source - Arbitration Priority Control	Project: BDW
			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		

MFX_DBK_OBJECT															
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Pre Deblocking Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
	Project:	BDW													
	Value	Name													
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
	01b	Uncacheable (UC) - non-cacheable													
	10b	Writethrough (WT)													
	11b	Writeback (WB)													
	4:3	<p>Target Cache (TC) for Pre Deblocking Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed	
		Project:	BDW												
		Value	Name												
00b		eLLC Only - not snooped in GT (BDW)													
01b		LLC Only													
10b		LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) for Pre Deblocking Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW													
	Format:	Enable													
	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														
4	<p>31:6 Deblocking Control Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Specifies the 4K byte aligned frame buffer address as input MB-level deblocking parameters to control the way hardware deblock the each micro-block. One 512-bit cacheline is allocated for each Macroblock in raster scan order.</p>	Format:	GraphicsAddress[31:6]												
	Format:	GraphicsAddress[31:6]													
<p>5:0 Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ											
Project:	BDW														
Format:	MBZ														

MFX_DBK_OBJECT			
5 Project: BDW	31:16	Reserved	
		Project: BDW Format: MBZ	
	15:0	Deblocking Control Address High	
		Project: BDW Format: GraphicsAddress[47:32]	
		This field is for the upper range of Deblocking Control Address (DeblockCntrlAddr). This field is used for 48-bit addressing.	
6 Project: BDW	31:15	Reserved	
		Project: BDW Format: MBZ	
	14:13	Reserved	
		Project: BDW Format: MBZ	
	12:11	Reserved	
		Project: BDW Format: MBZ	
	10:9	Reserved	
		Project: BDW Format: MBZ	
	8:7	Deblocking Control - Arbitration Priority Control	Project: BDW
			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		

MFX_DBK_OBJECT															
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Deblocking Control Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
	Project:	BDW													
	Value	Name													
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
	01b	Uncacheable (UC) - non-cacheable													
	10b	Writethrough (WT)													
11b	Writeback (WB)														
4:3	<p>Target Cache (TC) for Deblocking Control Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed		
	Project:	BDW													
	Value	Name													
	00b	eLLC Only - not snooped in GT (BDW)													
	01b	LLC Only													
	10b	LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) for Deblocking Control Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW													
	Format:	Enable													
	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														
7	<p>31:6 Deblocking Destination Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Specifies the 4K byte aligned frame buffer address for outputting the post-loop filtered reconstructed YUV picture (i.e. output of the deblocking filter unit)</p>	Format:	GraphicsAddress[31:6]												
	Format:	GraphicsAddress[31:6]													
	5:0	Reserved													
<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	BDW	Format:	MBZ										
Project:	BDW														
Format:	MBZ														

MFX_DBK_OBJECT											
8 Project: BDW	31:16	Reserved									
		Project: BDW									
		Format: MBZ									
	15:0	Deblocking Destination Address High									
		Project: BDW									
Format: GraphicsAddress[47:32]											
This field is for the upper range of Deblocking Destination Address. This field is used for 48-bit addressing.											
9 Project: BDW	31:15	Reserved									
		Project: BDW									
		Format: MBZ									
	14:13	Reserved									
		Project: BDW									
		Format: MBZ									
	12:11	Reserved									
		Project: BDW									
		Format: MBZ									
	10:9	Reserved									
		Project: BDW									
		Format: MBZ									
	8:7	Deblocking Destination - Arbitration Priority Control	Project: BDW								
			This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.								
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b
Value		Name									
00b		Highest priority									
01b		Second highest priority									
10b		Third highest priority									
11b	Lowest priority										
00b	Highest priority										
01b	Second highest priority										
10b	Third highest priority										
11b	Lowest priority										

MFX_DBK_OBJECT															
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Deblocking Destination Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
	Project:	BDW													
	Value	Name													
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
	01b	Uncacheable (UC) - non-cacheable													
	10b	Writethrough (WT)													
	11b	Writeback (WB)													
	4:3	<p>Target Cache (TC) for Deblocking Destination Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed	
		Project:	BDW												
		Value	Name												
00b		eLLC Only - not snooped in GT (BDW)													
01b		LLC Only													
10b		LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) for Deblocking Destination Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW													
	Format:	Enable													
	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														
10	<p>31:6 Deblock Row Store Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>This field provides the base address of the scratch buffer (read and write) used by the deblocking filter unit to store MB information of the previous row for filtering of each macroblock in the current row. The Deblocking Filter Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of the current macroblock to address the Deblocking Filter Row Store.</p>	Format:	GraphicsAddress[31:6]												
Format:	GraphicsAddress[31:6]														

MFX_DBK_OBJECT												
	5:0	Reserved										
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
Project:	BDW											
Format:	MBZ											
11 Project: BDW	31:16	Reserved										
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
		Project:	BDW									
	Format:	MBZ										
Reserved for 64-bit address extension.												
15:0	Deblock Row Store Address High											
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Deblock Row Store Address (DeblockRowStoreAddr). This field is used for 48-bit addressing.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]							
Project:	BDW											
Format:	GraphicsAddress[47:32]											
12 Project: BDW	31:15	Reserved										
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
	Format:	MBZ										
	14:13	Reserved										
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ						
	Project:	BDW										
	Format:	MBZ										
12	Reserved											
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW											
Format:	MBZ											
11	Reserved											
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW											
Format:	MBZ											
10:9	Reserved											
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW											
Format:	MBZ											
8:7		Deblock Row Store - Arbitration Priority Control										
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p>	Project:	BDW								
		Project:	BDW									
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
		Value	Name									
		00b	Highest priority									
01b	Second highest priority											
10b	Third highest priority											
11b	Lowest priority											

MFX_DBK_OBJECT

6:5		<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Deblock Row Store Address</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)
Project:	BDW													
Value	Name													
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
01b	Uncacheable (UC) - non-cacheable													
10b	Writethrough (WT)													
11b	Writeback (WB)													
4:3		<p>Target Cache (TC) for Deblock Row Store Address</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC</p> <p>00b: eLLC Only ("00" setting points TC selection to PTE which defaults to eLLC)</p> <p>01b: LLC Only (Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present).</p> <p>10b: LLC/eLLC Allowed.</p> <p>11b: L3, LLC, eLLC Allowed.</p> <p>Errata BDW:A-E (FIXED BY:G0 Stepping):</p> <p>For all system that does NOT use SVM (i.e. coherent L3\$ surfaces), back snoops from LLC has to be disabled (Dis_GtCvUpdtOnRd = "1"). Than target Cache settings can be programmed as POR requirements of L3/LLC/eDRAM caching.</p> <p>For all systems that does use SVM (i.e. coherent L3\$ surfaces), the recommended setting would be "00" in target cache settings. In case of L3 surfaces, the performance has to be tuned between "00" and "11" setting based on the benefits of L3 caching outweighing the degradation of backsnoops.</p> <p>Post G0-stepping, the above w/a for coherent L3\$ surfaces is not needed.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed
Project:	BDW													
Value	Name													
00b	eLLC Only - not snooped in GT (BDW)													
01b	LLC Only													
10b	LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed													
2		Reserved												



MFX_DBK_OBJECT

1:0

Age for QUADLRU (AGE) for Deblock Row Store Address

Project:	BDW
Format:	Enable

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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITS, hence need to be replaced least often in caches.

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

MFX_FQM_STATE

MFX_FQM_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a common state command for AVC encoder modes. For encoder, it represents both the forward QM matrices as well as the decoding QM matrices. This is a Frame-level state. Only Scaling Lists specified by an application are being sent to the hardware. The driver is responsible for determining the final set of scaling lists to be used for decoding the current slice, based on the AVC Spec Table 7-2 (Fall-Back Rules A and B). In MFX AVC PAK mode, PAK needs both forward Q scaling lists and IQ scaling lists. The IQ scaling lists are sent as in MFD in raster scan order. But the Forward Q scaling lists are sent in column-wise raster order (column-by-column) to simplify the H/W. Driver will perform all the scan order conversion for both ForwardQ and IQ.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MULTI_DW
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MFX_COMMON_STATE
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	8h
Format:		OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	20h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:2	Reserved	
		Format:	MBZ

MFX_FQM_STATE												
	1:0	AVC										
		Exists If: //AVC- Decoder Only										
		For AVC QM Type: This field specifies which Quantizer Matrix is loaded.										
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)</td> </tr> <tr> <td>1</td> <td>AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)</td> </tr> <tr> <td>2</td> <td>AVC_8x8_Intra_MATRIX</td> </tr> <tr> <td>3</td> <td>AVC_8x8_Inter_MATRIX</td> </tr> </tbody> </table>	Value	Name	0	AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)	1	AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)	2	AVC_8x8_Intra_MATRIX	3	AVC_8x8_Inter_MATRIX
		Value	Name									
	0	AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)										
	1	AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)										
	2	AVC_8x8_Intra_MATRIX										
	3	AVC_8x8_Inter_MATRIX										
	1:0	MPEG2										
Exists If: //MPEG2- Decoder Only												
For MPEG2 QM Type: This field specifies which Quantizer Matrix is loaded.												
<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>MPEG_INTRA_QUANTIZER_MATRIX</td> </tr> <tr> <td>1</td> <td>MPEG_NON_INTRA_QUANTIZER_MATRIX</td> </tr> <tr> <td>2-3</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	0	MPEG_INTRA_QUANTIZER_MATRIX	1	MPEG_NON_INTRA_QUANTIZER_MATRIX	2-3	Reserved				
Value	Name											
0	MPEG_INTRA_QUANTIZER_MATRIX											
1	MPEG_NON_INTRA_QUANTIZER_MATRIX											
2-3	Reserved											
2..33	31:0	Forward Quantizer Matrix										
Project:		All										
Format:		U32										
The format of a Quantizer Matrix is an 8x8 matrix in raster order. Each element is an unsigned byte.												

MFX_IND_OBJ_BASE_ADDR_STATE

MFX_IND_OBJ_BASE_ADDR_STATE								
Project:	BDW							
Source:	VideoCS							
Length Bias:	2							
<p>This state command provides the memory base addresses for all row stores, StreamOut buffer and reconstructed picture output buffers required by the MFD or MFC Engine (that are in addition to the row stores of the Bit Stream Decoding/Encoding Unit (BSD/BSE) and the reference picture buffers). This is a picture level state command and is common among all codec standards and for both encoder and decoder operating modes. However, some fields may only applicable to a specific codec standard. All Pixel Surfaces (original, reference frame and reconstructed frame) in the Encoder are programmed with the same surface state (NV12 and TileY format), except each has its own frame buffer base address. In the tile format, there is no need to provide buffer offset for each slice; since from each MB address, the hardware can calculated the corresponding memory location within the frame buffer directly.</p>								
<p>The MFX_IND_OBJ_BASE_ADDR command sets the memory base address pointers for the corresponding Indirect Object Data Start Addresses (Offsets) specified in each OBJECT commands. The characteristic of these indirect object data is their variable size (per MB or per Slice). Hence, each OBJECT command must specify the indirect object data offset from the base address to start fetching or writing object data.</p>								
<p>While the use of base address is unconditional, the indirection can be effectively disabled by setting the base address to zero.</p> <p>For decoder, there are:</p> <ul style="list-style-type: none"> • 1 read-only per-slice indirect object in the BSD_OBJECT Command, and • 2 read-only per-MB indirect objects in the IT_OBJECT Command. <p>For decoder: the Video Command Streamer (VCS) will perform the memory access bound check automatically using the corresponding MFC Indirect Object Access Upper Bound specification. If any access is at or beyond the upper bound, zero value is returned. The request to memory is still being sent, but the corresponding codec's BSD unit will detect this condition and perform the zeroing return. If the Upper Bound is turned off, the beyond bound request will return whatever on the bus (invalid data).</p> <p>For encoder, there are:</p> <ul style="list-style-type: none"> • 1 read-only per-MB indirect object in the PAK_OBJECT Command, and • 1 write-only per-slice indirect object in the PAK Slice_State Command <p>For encoder: whenever an out of bound address accessing request is generated, VMX will detect such requests and snap the address to the corresponding [indirect object base address + indirect data start address]. VMX will return all 0s as the data to the requestor. NotationDefinitionPhysicalAddress[n:m] Corresponding bits of a physical graphics memory byte address (not mapped by a GTT) GraphicsAddress[n:m] Corresponding bits of an absolute, virtual graphics memory byte address (mapped by a GTT).</p>								
DWord	Bit	Description						
0	31:29	<table border="1"> <tr> <td colspan="2">Command Type</td> </tr> <tr> <td>Default Value:</td> <td>3h PARALLEL_VIDEO_PIPE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Command Type		Default Value:	3h PARALLEL_VIDEO_PIPE	Format:	OpCode
Command Type								
Default Value:	3h PARALLEL_VIDEO_PIPE							
Format:	OpCode							

MFX_IND_OBJ_BASE_ADDR_STATE						
	28:27	Pipeline <table border="1"> <tr> <td>Default Value:</td> <td>2h MFX_IND_OBJ_BASE_ADDR_STATE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	2h MFX_IND_OBJ_BASE_ADDR_STATE	Format:	OpCode
	Default Value:	2h MFX_IND_OBJ_BASE_ADDR_STATE				
	Format:	OpCode				
	26:24	Common Opcode <table border="1"> <tr> <td>Default Value:</td> <td>0h MFX_IND_OBJ_BASE_ADDR_STATE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MFX_IND_OBJ_BASE_ADDR_STATE	Format:	OpCode
	Default Value:	0h MFX_IND_OBJ_BASE_ADDR_STATE				
	Format:	OpCode				
	23:21	Sub OpcodeA <table border="1"> <tr> <td>Default Value:</td> <td>0h MFX_IND_OBJ_BASE_ADDR_STATE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MFX_IND_OBJ_BASE_ADDR_STATE	Format:	OpCode
Default Value:	0h MFX_IND_OBJ_BASE_ADDR_STATE					
Format:	OpCode					
20:16	SubOpcodeB <table border="1"> <tr> <td>Default Value:</td> <td>3h MFX_IND_OBJ_BASE_ADDR_STATE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	3h MFX_IND_OBJ_BASE_ADDR_STATE	Format:	OpCode	
Default Value:	3h MFX_IND_OBJ_BASE_ADDR_STATE					
Format:	OpCode					
15:12	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
11:0	DWord Length <table border="1"> <tr> <td>Default Value:</td> <td>0018h Excludes DWord (0,1)</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0018h Excludes DWord (0,1)	Format:	=n Total Length - 2	
Default Value:	0018h Excludes DWord (0,1)					
Format:	=n Total Length - 2					
1	31:12	MFX Indirect Bitstream Object - Base Address (Decoder and Stitch Modes) <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>Specifies the 4K-byte aligned memory base address for the read-only indirect data object pointed in the MFD_XXX_BSD_OBJECT command for fetching (reading) the compressed Slice Data. This field is only valid in MPEG2, AVC and VC1 decoder VLD mode.</p>	Project:	All	Format:	GraphicsAddress[31:12]
	Project:	All				
	Format:	GraphicsAddress[31:12]				
11:6	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
5:0	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
2 Project: BDW	31:16	Reserved <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	All	Format:	MBZ
		Project:	All			
		Format:	MBZ			

MFX_IND_OBJ_BASE_ADDR_STATE				
	15:0	MFX Indirect Bitstream Object - Destination Address (Decoder and Stitch Modes)[47:32]		
		Project:	All	
		Description		
		This field is for the upper range of MFX Indirect Bitstream Object Base Address.		
		This field is used for 48-bit addressing.		
3 Project: BDW	31:15	Reserved		
		Project:	BDW	
		Format:	MBZ	
	14:13	Reserved		
		Project:	BDW	
		Format:	MBZ	
	12:11	Reserved		
		Project:	BDW	
		Format:	MBZ	
	10:9	Reserved		
		Project:	BDW	
		Format:	MBZ	
	8:7	MFX Indirect Bitstream ObjectBase - Arbitration Priority Control		
			Project:	BDW
			Format:	U2
		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		

MFX_IND_OBJ_BASE_ADDR_STATE															
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MFX Indirect Bitstream ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
	Project:	BDW													
	Value	Name													
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
	01b	Uncacheable (UC) - non-cacheable													
	10b	Writethrough (WT)													
	11b	Writeback (WB)													
	4:3	<p>Target Cache (TC) MFX Indirect Bitstream ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed	
		Project:	BDW												
		Value	Name												
00b		eLLC Only - not snooped in GT (BDW)													
01b		LLC Only													
10b		LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) MFX Indirect Bitstream ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW													
	Format:	Enable													
	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														
4	<p>31:12 MFX Indirect Bitstream Object - Access Upper Bound (Decoder and Stitch Modes)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>This field specifies the 4K-byte aligned (exclusive) maximum Graphics Memory address access by the indirect data object in the MFD_XXX_BSD_OBJECT command for the compressed Slice Data. Indirect data accessed at this address and beyond will return as 0 by the hardware. Setting this field to 0 will cause this range check to be ignored.If non-zero, this address must be greater than the MFX Indirect Bitstream ObjectBase Address state.Hardware ignores this field if indirect data is not present, i.e. the Indirect Data Length field of the MFD_XXX_BSD_OBJECT command is set to 0.This field is only valid in MPEG2, AVC, VP8, and VC1 decoder VLD mode.</p>	Format:	GraphicsAddress[31:12]												
Format:	GraphicsAddress[31:12]														

		MFX_IND_OBJ_BASE_ADDR_STATE		
	11:0	Reserved		
		Format:	MBZ	
5 Project: BDW	31:16	Reserved		
		Project:	BDW	
		Format:	MBZ	
	Reserved for 64-bit address extension.			
	15:0	MFX Indirect Bitstream Object UpperBound (Decoder and Stitch Modes)[47:32]		
	Project:	BDW		
		Description	Project	
		This field is for the upper range of MFX Indirect Bitstream Object UpperBound.		
		This field is used for 48-bit addressing.	BDW	
6	31:12	MFX Indirect MV Object - Base Address		
		Format:	GraphicsAddress[31:12]	
	Specifies the 4K-byte aligned memory base address for the read-only indirect data object pointed in the encoder MFC_AVC_PAK_OBJECT command or the decoder MFD_IT_OBJECT command for fetching the per-MB MV data.This field is only valid in AVC encoder mode or in AVC decoder IT mode			
	11:6	Reserved		
		Format:	MBZ	
5:0	Reserved			
	Project:	BDW		
	Format:	MBZ		
7 Project: BDW	31:16	Reserved		
		Project:	BDW	
		Format:	MBZ	
	Reserved for 64-bit address extension.			
	15:0	MFX Indirect MV Object Base Address [47:32]		
	Project:	All		
		Description	Project	
		This field is for the upper range of MFX Indirect MV Object Base Address.		
		This field is used for 48-bit addressing.	BDW	
8 Project: BDW	31:15	Reserved		
		Project:	BDW	
		Format:	MBZ	

MFX_IND_OBJ_BASE_ADDR_STATE					
14:13	Reserved				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
12:11	Reserved				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
10:9	Reserved				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
8:7	MFX Indirect MV Object - Arbitration Priority Control				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table>	Project:	BDW	Format:	U2
	Project:	BDW			
	Format:	U2			
	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				
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MFX Indirect MV ObjectBase Address				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW		
	Project:	BDW			
	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.				
	Value	Name			
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)			
	01b	Uncacheable (UC) - non-cacheable			
10b	Writethrough (WT)				
11b	Writeback (WB)				
4:3	Target Cache (TC) MFX Indirect MV ObjectBase Address				
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW		
	Project:	BDW			
	This field allows the choice of LLC vs eLLC for caching				
	Value	Name			
	00b	eLLC Only - not snooped in GT (BDW)			
01b	LLC Only				
10b	LLC/eLLC Allowed				
11b	L3, LLC, eLLC Allowed				

MFX_IND_OBJ_BASE_ADDR_STATE																
	2	Reserved														
	1:0	<p>Age for QUADLRU (AGE) MFX Indirect MV ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
Project:	BDW															
Format:	Enable															
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															
9	31:12	<p>MFX Indirect MV Object Access Upper Bound</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>This field specifies the 4K-byte aligned (exclusive) maximum Graphics Memory address access by the indirect data object in the MFC_AVC_PAK_OBJECT / MFD_IT_OBJECT command for the per-MB MV data. Indirect data accessed at this address and beyond will return as 0 by the hardware. Setting this field to 0 will cause this range check to be ignored. If non-zero, this address must be greater than the MFX Indirect MV Object Base Address state. Hardware ignores this field if indirect data is not present, i.e. the Indirect Data Length field of the MFC_AVC_PAK_OBJECT / MFD_IT_OBJECT command is set to 0. This field is only valid in AVC encoder mode or in AVC decoder IT mode.</p>	Format:	GraphicsAddress[31:12]												
	Format:	GraphicsAddress[31:12]														
11:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ													
Format:	MBZ															
10 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	All	Format:	MBZ										
	Project:	All														
Format:	MBZ															
	15:0	<p>MFX Indirect MV Object UpperBound [47:32]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> </table> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 80%;">Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>This field is for the upper range of MFX Indirect MV Object Base Address.</td> <td></td> </tr> <tr> <td>This field is used for 48-bit addressing.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	All	Description	Project	This field is for the upper range of MFX Indirect MV Object Base Address.		This field is used for 48-bit addressing.	BDW						
Project:	All															
Description	Project															
This field is for the upper range of MFX Indirect MV Object Base Address.																
This field is used for 48-bit addressing.	BDW															
11	31:12	<p>MFD Indirect IT-COEFF Object - Base Address (Decoder Only)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>Specifies the 4K-byte aligned memory base address for the read-only indirect data object pointed in the MFD_IT_OBJECT command for fetching (reading) the per-MB non-scaled coefficient data (all inverse scaling and quantization are done in hardware). This field is only valid in MPEG2, AVC and VC1 decoder IT mode.</p>	Format:	GraphicsAddress[31:12]												
Format:	GraphicsAddress[31:12]															

MFX_IND_OBJ_BASE_ADDR_STATE			
	11:6	Reserved Format: MBZ	
	5:0	Reserved Project: BDW Format: MBZ	
12 Project: BDW	31:16	Reserved Project: BDW Format: MBZ Reserved for 64-bit address extension.	
		15:0	MFD Indirect IT-COEFF Object Base Address [47:32] Project: BDW
			Description
	This field is for the upper range of MFX Indirect IT-COEFF Object Base Address.		
	This field is for the upper range of MFX Indirect MV Object Base Address.		
This field is used for 48-bit addressing.		BDW	
13 Project: BDW	31:15	Reserved Project: BDW Format: MBZ	
		14:13	Reserved Project: BDW Format: MBZ
	12:11		Reserved Project: BDW Format: MBZ
		10:9	Reserved Project: BDW Format: MBZ
	8:7		MFD Indirect IT-COEFF Object Desitnation - Arbitration Priority Control Project: BDW Format: U2 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	

MFX_IND_OBJ_BASE_ADDR_STATE															
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MFD Indirect IT-COEFF ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
	Project:	BDW													
	Value	Name													
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)													
	01b	Uncacheable (UC) - non-cacheable													
	10b	Writethrough (WT)													
	11b	Writeback (WB)													
	4:3	<p>Target Cache (TC) MFD Indirect IT-COEFF ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed	
		Project:	BDW												
		Value	Name												
00b		eLLC Only - not snooped in GT (BDW)													
01b		LLC Only													
10b		LLC/eLLC Allowed													
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) MFD Indirect IT-COEFF ObjectBase Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
	Project:	BDW													
	Format:	Enable													
	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														
14	<p>31:12 MFD Indirect IT-COEFF Object - Access Upper Bound (Decoder Only)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>This field specifies the 4K-byte aligned (exclusive) maximum Graphics Memory address access by the indirect data object in the MFD_IT_OBJECT command for the per-MB non-scaled coefficient data. Indirect data accessed at this address and beyond will return as 0 by the hardware. Setting this field to 0 will cause this range check to be ignored.If non-zero, this address must be greater than the MFD Indirect IT-COEFF Object Base Address state.Hardware ignores this field if indirect data is not present, i.e. the Indirect COEFF Data Length field of the MFD_IT_OBJECT command is set to 0.This field is only valid in MPEG2, AVC and VC1 decoder IT mode.</p>	Format:	GraphicsAddress[31:12]												
Format:	GraphicsAddress[31:12]														

		MFX_IND_OBJ_BASE_ADDR_STATE		
	11:0	Reserved		
		Format:	MBZ	
15 Project: BDW	31:16	Reserved		
		Project:	All	
		Format:	MBZ	
		Reserved for 64-bit address extension.		
	15:0	MFD Indirect IT-COEFF Object UpperBound [47:32]		
	Project:	BDW		
		Description	Project	
		This field is for the upper range of MFX Indirect IT-COEFF Object UpperBound.		
		This field is for the upper range of MFX Indirect MV Object Base Address.		
		This field is used for 48-bit addressing.	BDW	
16	31:12	MFD Indirect IT-DBLK Object - Base Address (Decoder Only)		
		Format:	GraphicsAddress[31:12]	
		Specifies the 4K-byte aligned memory base address for the read-only indirect data object pointed in the MFD_IT_OBJECT command for fetching (reading) the per-MB Deblocking filter control data. This field is only valid in AVC decoder IT mode.		
	11:6	Reserved		
		Format:	MBZ	
	5:0	Reserved		
		Project:	BDW	
		Format:	MBZ	
17 Project: BDW	31:16	Reserved		
		Project:	BDW	
		Format:	MBZ	
		Reserved for 64-bit address extension.		
	15:0	MFD Indirect IT-DBLK Object Base Address [47:32]		
	Project:	BDW		
		Description	Project	
		This field is for the upper range of MFX Indirect IT-DBLK Object Base Address.		
		This field is used for 48-bit addressing.	BDW	
18 Project: BDW	31:15	Reserved		
		Project:	BDW	
		Format:	MBZ	

MFX_IND_OBJ_BASE_ADDR_STATE		
14:13	Reserved	
	Project:	BDW
	Format:	MBZ
12:11	Reserved	
	Project:	BDW
	Format:	MBZ
10:9	Reserved	
	Project:	BDW
	Format:	MBZ
8:7	MFD Indirect IT-DBLK Object - Arbitration Priority Control	
	Project:	BDW
	Format:	U2
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
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MFD Indirect IT-DBLK ObjectBase Address	
	Project:	BDW
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.		
	Value	Name
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)
	01b	Uncacheable (UC) - non-cacheable
	10b	Writethrough (WT)
	11b	Writeback (WB)
4:3	Target Cache (TC) MFD Indirect IT-DBLK ObjectBase Address	
	Project:	BDW
This field allows the choice of LLC vs eLLC for caching		
	Value	Name
	00b	eLLC Only - not snooped in GT (BDW)
	01b	LLC Only
	10b	LLC/eLLC Allowed
	11b	L3, LLC, eLLC Allowed

MFX_IND_OBJ_BASE_ADDR_STATE																
	2	Reserved														
	1:0	<p>Age for QUADLRU (AGE) MFD Indirect IT-DBLK ObjectBase Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
Project:	BDW															
Format:	Enable															
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															
19	31:12	<p>MFD Indirect IT-DBLK Object - Access Upper Bound (Decoder Only)</p> <table border="1"> <tr> <td>Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>This field specifies the 4K-byte aligned (exclusive) maximum Graphics Memory address access by the indirect data object in the MFD_IT_OBJECT command for the per-MB Deblocking filter control data. Indirect data accessed at this address and beyond will return as 0 by the hardware. Setting this field to 0 will cause this range check to be ignored. If non-zero, this address must be greater than the MFD Indirect IT-DBLK Object Base Address state. Hardware ignores this field if indirect data is not present, i.e. the Indirect Deblocking Control Data Length field of the MFD_IT_OBJECT command is set to 0. This field is only valid in AVC decoder IT mode.</p>	Format:	GraphicsAddress[31:12]												
Format:	GraphicsAddress[31:12]															
	11:0	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ															
20	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	All	Format:	MBZ										
Project:	All															
Format:	MBZ															
Project: BDW	15:0	<p>MFD Indirect IT-DBLK Object UpperBound [47:32]</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>This field is for the upper range of MFX Indirect IT-DBLK Object UpperBound.</td> <td></td> </tr> <tr> <td>This field is used for 48-bit addressing.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	All	Description	Project	This field is for the upper range of MFX Indirect IT-DBLK Object UpperBound.		This field is used for 48-bit addressing.	BDW						
Project:	All															
Description	Project															
This field is for the upper range of MFX Indirect IT-DBLK Object UpperBound.																
This field is used for 48-bit addressing.	BDW															
21	31:12	<p>MFC Indirect PAK-BSE Object - Base Address (Encoder Only)</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:12]</td> </tr> </table> <p>Specifies the 4K-byte aligned memory base address for the write-only indirect data object pointed in the PAK_SLICE_STATE command for writing out the compressed bitstream. This field is only valid in AVC encoder mode.</p>	Project:	All	Format:	GraphicsAddress[31:12]										
Project:	All															
Format:	GraphicsAddress[31:12]															

MFX_IND_OBJ_BASE_ADDR_STATE						
	11:6	Reserved				
		Project: All Format: MBZ				
	5:0	Reserved				
		Project: BDW Format: MBZ				
22 Project: BDW	31:16	Reserved				
		Project: BDW Format: MBZ Reserved for 64-bit address extension.				
	15:0	MFC Indirect PAK-BSE Object Base Address [47:32]				
		Project: BDW				
<table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>This field is for the upper range of MFX Indirect PAK-BSE Object Base Address.</td> <td></td> </tr> <tr> <td>This field is used for 48-bit addressing.</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>		Description	Project	This field is for the upper range of MFX Indirect PAK-BSE Object Base Address.		This field is used for 48-bit addressing.
Description	Project					
This field is for the upper range of MFX Indirect PAK-BSE Object Base Address.						
This field is used for 48-bit addressing.	BDW					
23 Project: BDW	31:15	Reserved				
		Project: BDW Format: MBZ				
	14:13	Reserved				
		Project: BDW Format: MBZ				
	12:11	Reserved				
		Project: BDW Format: MBZ				
	10:9	Reserved				
		Project: BDW Format: MBZ				

MFX_IND_OBJ_BASE_ADDR_STATE		
8:7	MFC Indirect PAK-BSE Object Desitnation - Arbitration Priority Control	
	Project:	BDW
	Format:	U2
	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
	6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for MFC Indirect PAK-BSE ObjectBase Address
Project:		BDW
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.		
Value		Name
00b		Use Cacheability Controls from page table / UC with Fence (if coherent cycle)
01b		Uncacheable (UC) - non-cacheable
10b		Writethrough (WT)
11b		Writeback (WB)
4:3	Target Cache (TC) MFC Indirect PAK-BSE ObjectBase Address	
	Project:	BDW
	This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC	
	00b: eLLC Only ("00" setting points TC selection to PTE which defaults to eLLC)	
	01b: LLC Only (Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present).	
	10b: LLC/eLLC Allowed.	
	11b: L3, LLC, eLLC Allowed.	
	Value	Name
	00b	eLLC Only - not snooped in GT (BDW)
	01b	LLC Only
10b	LLC/eLLC Allowed	
11b	L3, LLC, eLLC Allowed	
2	Reserved	

MFX_IND_OBJ_BASE_ADDR_STATE												
	1:0	Age for QUADLRU (AGE) MFC Indirect PAK-BSE ObjectBase Address										
		Project: BDW										
		Format: Enable										
		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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITs, hence need to be replaced least often in caches.										
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	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
	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											
24 Project: BDW	31:12	MFC Indirect PAK-BSE Object - Access Upper Bound (Eecoder Only)										
		Project: BDW										
		Format: GraphicsAddress[31:12]										
		This field specifies the 4K-byte aligned (exclusive) maximum Graphics Memory address access by the indirect data object in the PAK_SLICE_STATE command for the per-slice output bitstream. Indirect data accessed at this address and beyond will be blocked by the hardware and ignored. Setting this field to 0 will cause this range check to be ignored. If non-zero, this address must be greater than the MFC Indirect PAK-BSE Object Base Address state. This field is only valid in AVC encoder mode.										
	11:0	Reserved										
		Project: BDW										
		Format: MBZ										
25 Project: BDW	31:16	Reserved										
		Project: All										
		Format: MBZ										
		Reserved for 64-bit address extension.										
	15:0	MFC Indirect PAK-BSE Object UpperBound [47:32]										
		Project: BDW										
		<table border="1"> <thead> <tr> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>This field is for the upper range of MFC Indirect PAK-BSE Object UpperBound</td> <td></td> </tr> <tr> <td>This field is used for 48-bit addressing.</td> <td>BDW</td> </tr> </tbody> </table>	Description	Project	This field is for the upper range of MFC Indirect PAK-BSE Object UpperBound		This field is used for 48-bit addressing.	BDW				
	Description	Project										
	This field is for the upper range of MFC Indirect PAK-BSE Object UpperBound											
This field is used for 48-bit addressing.	BDW											

MFX_JPEG_HUFF_TABLE_STATE

MFX_JPEG_HUFF_TABLE_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This Huffman table commands contains both DC and AC tables for either luma or chroma. Once a Huffman table has been defined for a particular destination, it replaces the previous tables stored in that destination and shall be used in the remaining Scans of the current image. A Huffman table will be sent to H/W only when it is loaded from bitstream.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MULTI_DW
	26:24	Media Command Opcode	
		Default Value:	7h JPEG_COMMON
	23:21	SubOpcode A	
		Default Value:	0h
	20:16	SubOpcode B	
Default Value:		2h	
15:12	Reserved		
	Project:	All	
11:0	DWord Length		
	Default Value:	033Dh Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:1	Reserved	
		Format:	MBZ

MFX_JPEG_HUFF_TABLE_STATE								
	0	<p>HuffTableID (1-bit) Identifies the huffman table.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">Y</td> <td>Huffman table for Y</td> </tr> </tbody> </table>	Value	Name	Description	0	Y	Huffman table for Y
Value	Name	Description						
0	Y	Huffman table for Y						
2..4	31:0	<p>DC_BITS (12 8-bit array) The number of DC Huffman codes of length i, where i is 1~12</p>						
5..7	31:0	<p>DC_HUFFVAL (12 8-bit array) The value associated with each DC Huffman code of length i.</p>						
8..11	31:0	<p>AC_BITS (16 8-bit array) the list of Li, number of Huffman codes of length i, where i is 1~16</p>						
12..51	31:0	<p>AC_HUFFVAL (160 8-bit array) the list of Vi,j, the value associated with each Huffman code of length i</p>						
52	31:16	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </tbody> </table>	Project:	All	Format:	MBZ		
		Project:	All					
Format:	MBZ							
15:0	<p>AC_HUFFVAL(2-8 bit array) In AC table, BITS can have up to 16-bit codeword. Li can be 0 ~ 162. HUFFVAL will have a list of likely random distributed values</p>							

MFX_JPEG_PIC_STATE

MFX_JPEG_PIC_STATE							
Project:		BDW					
Source:		VideoCS					
Length Bias:		2					
DWord	Bit	Description					
0	31:29	Command Type					
		Default Value: 3h PARALLEL_VIDEO_PIPE Format: OpCode					
	28:27	Pipeline					
		Default Value: 2h MFX_MULTI_DW Format: OpCode					
	26:24	Media Command Opcode					
		Default Value: 7h JPEG Format: OpCode					
	23:21	SubOpcode A					
Default Value: 0h Common Format: OpCode							
20:16	SubOpcode B						
	Default Value: 0h MEDIA_ Format: OpCode						
15:12	Reserved						
11:0	Format: MBZ						
	DWord Length						
	Project: All Format: =n Total Length - 2						
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0001h</td> <td>[Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	0001h	[Default]	Excludes DWord (0,1)
Value	Name	Description					
0001h	[Default]	Excludes DWord (0,1)					
1	31:21	Reserved					
		Exists If: //Decoder Only Format: MBZ					

MFX_JPEG_PIC_STATE										
20	Vertical Up-Sampling Enable									
	Project:	BDW								
	Exists If:	//Decoder Only								
	<p>Only applied to chroma blocks. This flag is used for 2:1 vertical up-sampling for chroma 420 and outputting chroma422 YUY2 or UYVY format. To enable this flag, the input should be interleaved Scan, InputFormatYUV should be set to YUV420, and OutputFormatYUV should be set to YUY2 or UYVY.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td></td> <td>no up-sampling</td> </tr> <tr> <td>1b</td> <td></td> <td>2:1 vertical up-sampling</td> </tr> </tbody> </table>		Value	Name	Description	0b		no up-sampling	1b	
Value	Name	Description								
0b		no up-sampling								
1b		2:1 vertical up-sampling								
19	Reserved									
	Project:	BDW								
	Exists If:	//Decoder Only								
18	Horizontal Down-Sampling Enable									
	Project:	BDW								
	Exists If:	//Decoder Only								
	<p>Only applied to chroma blocks. This flag is used for 2:1 horizontal down-sampling for chroma 422 and outputting chroma420 NV21 format. To enable this flag, the input should be interleaved Scan, InputFormatYUV should be set to YUV422V_2Y or YUV422V_4Y, and OutputFormatYUV should be set to NV12.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td></td> <td>no down-sampling</td> </tr> <tr> <td>1b</td> <td></td> <td>2:1 horizontal down-sampling</td> </tr> </tbody> </table>		Value	Name	Description	0b		no down-sampling	1b	
Value	Name	Description								
0b		no down-sampling								
1b		2:1 horizontal down-sampling								
17	Vertical Down-Sampling Enable									
	Project:	BDW								
	Exists If:	//Decoder Only								
	<p>Only applied to chroma blocks. This flag is used for 2:1 vertical down-sampling for chroma 422 and outputting chroma420 NV21 format. To enable this flag, the input should be interleaved Scan, InputFormatYUV should be set to YUV422H_2Y or YUV422H_4Y, and OutputFormatYUV should be set to NV12.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td></td> <td>no down-sampling</td> </tr> <tr> <td>1b</td> <td></td> <td>2:1 vertical down-sampling</td> </tr> </tbody> </table>		Value	Name	Description	0b		no down-sampling	1b	
Value	Name	Description								
0b		no down-sampling								
1b		2:1 vertical down-sampling								

MFX_JPEG_PIC_STATE			
16	Average Down Sampling		
	Project:	BDW	
	Exists If:	//Decoder Only	
	This flag is used to select a down-sampling method when VertDownSamplingEnb or HoriDownSamplingEnb is set to 1.		
	Value	Name	Description
0b		Drop every other line (or column) pixels	
1b		Average neighboring two pixels	
15:12	Reserved		
	Exists If:	//Decoder Only	
	Format:	MBZ	
11:8	Output Format YUV		
	Project:	BDW	
	Exists If:	//Decoder Only	
	This field specifies the surface format to write the decoded JPEG image. Note that any non-interleaved JPEG input should be set to "0000". For the interleaved input Scan data, it can be set either "0000" or the corresponding format.		
	Value	Name	Description
	0000b		3 separate plane for Y, U, and V respectively
	0001b		NV12 for chroma 4:2:0
	0010b		UYVY for chroma 4:2:2
	0011b		YUY2 for chroma 4:2:2
	Programming Notes		
The MFX_SURFACE_STATE command should be set accordingly for each OutputFormatYUV . For NV12, Surface Format = 4 (PLANAR_420_8) For YUY2, Surface Format = 0 (YCRCB_NORMAL) For UYVY, Surface Format = 3 (YCRCB_SWAPY) NV12 (0001b) can be set only when Y, U, V are interleaved in a single Scan data with the following cases <ul style="list-style-type: none"> • InputFormatYUV is YUV420 and VertDownSamplingEnb is disabled • InputFormatYUV is YUV422H_2Y or YUV422H_4Y, and VertDownSamplingEnb is enabled UYVY (0010b) and YUY2 (0011b) can be set only when Y, U, V are interleaved in a single Scan data with the following cases <ul style="list-style-type: none"> • InputFormatYUV is YUV420 and VertUpSamplingEnb is enabled • InputFormatYUV is YUV422H_2Y or YUV422H_4Y and VertUpSamplingEnb is disabled 			

MFX_JPEG_PIC_STATE																																
	7:6	Reserved <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ																										
	Exists If:	//Decoder Only																														
	Format:	MBZ																														
	5:4	Rotation <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <th style="text-align: left;">Value</th> <th style="text-align: left;">Name</th> <th style="text-align: left;">Description</th> </tr> <tr> <td>00b</td> <td></td> <td>no rotation</td> </tr> <tr> <td>01b</td> <td></td> <td>rotate clockwise 90 degree</td> </tr> <tr> <td>10b</td> <td></td> <td>rotate counter-clockwise 90 degree (same as rotating 270 degree clockwise)</td> </tr> <tr> <td>11b</td> <td></td> <td>rotate 180 degree (NOT the same as flipped on the x-axis)</td> </tr> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> <tr> <td colspan="2">Rotation can be set to 01b, 10b, or 11b when OutputFormatYUV is set to 0000b. For other OutputFormatYUV, Rotation is not allowed.</td> <td>BDW</td> </tr> </table>	Exists If:	//Decoder Only	Value	Name	Description	00b		no rotation	01b		rotate clockwise 90 degree	10b		rotate counter-clockwise 90 degree (same as rotating 270 degree clockwise)	11b		rotate 180 degree (NOT the same as flipped on the x-axis)	Programming Notes		Project	Rotation can be set to 01b, 10b, or 11b when OutputFormatYUV is set to 0000b. For other OutputFormatYUV, Rotation is not allowed.		BDW							
	Exists If:	//Decoder Only																														
	Value	Name	Description																													
	00b		no rotation																													
	01b		rotate clockwise 90 degree																													
	10b		rotate counter-clockwise 90 degree (same as rotating 270 degree clockwise)																													
	11b		rotate 180 degree (NOT the same as flipped on the x-axis)																													
Programming Notes		Project																														
Rotation can be set to 01b, 10b, or 11b when OutputFormatYUV is set to 0000b. For other OutputFormatYUV, Rotation is not allowed.		BDW																														
3	Reserved <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ																											
Exists If:	//Decoder Only																															
Format:	MBZ																															
2:0	Input Format YUV <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> <tr> <th style="text-align: left;">Value</th> <th style="text-align: left;">Name</th> <th style="text-align: left;">Description</th> </tr> <tr> <td>0</td> <td>[Default]</td> <td>YUV400 (grayscale image)</td> </tr> <tr> <td>1</td> <td></td> <td>YUV420</td> </tr> <tr> <td>2</td> <td></td> <td>YUV422H_2Y (Horizontally chroma 2:1 subsampled) - horizontal 2 Y-block, 1U and 1V</td> </tr> <tr> <td>3</td> <td></td> <td>YUV444</td> </tr> <tr> <td>4</td> <td></td> <td>YUV411</td> </tr> <tr> <td>5</td> <td></td> <td>YUV422V_2Y (Vertically chroma 2:1 subsampled) - vertical 2 Y-blocks, 1U and 1V</td> </tr> <tr> <td>6</td> <td></td> <td>YUV422H_4Y - 2x2 Y-blocks, vertical 2U and 2V</td> </tr> <tr> <td>7</td> <td></td> <td>YUV422V_4Y - 2x2 Y-blocks, horizontal 2U and 2V</td> </tr> </table>	Exists If:	//Decoder Only	Format:	U3	Value	Name	Description	0	[Default]	YUV400 (grayscale image)	1		YUV420	2		YUV422H_2Y (Horizontally chroma 2:1 subsampled) - horizontal 2 Y-block, 1U and 1V	3		YUV444	4		YUV411	5		YUV422V_2Y (Vertically chroma 2:1 subsampled) - vertical 2 Y-blocks, 1U and 1V	6		YUV422H_4Y - 2x2 Y-blocks, vertical 2U and 2V	7		YUV422V_4Y - 2x2 Y-blocks, horizontal 2U and 2V
Exists If:	//Decoder Only																															
Format:	U3																															
Value	Name	Description																														
0	[Default]	YUV400 (grayscale image)																														
1		YUV420																														
2		YUV422H_2Y (Horizontally chroma 2:1 subsampled) - horizontal 2 Y-block, 1U and 1V																														
3		YUV444																														
4		YUV411																														
5		YUV422V_2Y (Vertically chroma 2:1 subsampled) - vertical 2 Y-blocks, 1U and 1V																														
6		YUV422H_4Y - 2x2 Y-blocks, vertical 2U and 2V																														
7		YUV422V_4Y - 2x2 Y-blocks, horizontal 2U and 2V																														
2	31:30	Reserved <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ																										
Exists If:	//Decoder Only																															
Format:	MBZ																															

MFX_JPEG_PIC_STATE		
29	Reserved	
	Project: BDW	
	Exists If: //Decoder Only	
	Format: MBZ	
	28:16 Frame Height In Blocks Minus 1	
	Exists If: //Decoder Only	
	Format: U13-1	
	Description	Project
	(The number of blocks in height) - 1. This value is calculated using the number of lines Y and vertical sampling factor of the first component V ₁ in Frame header. See the note following this table. For interleaved components, $((Y + (V_1 * 8 - 1)) / (V_1 * 8)) * V_1 - 1$, where "/" is integer division. For non-interleaved components, $(Y + 7) / 8 - 1$.	
	Workaround: For interleaved components, when Input Format YUV is set to YUV422H_2Y , OutputFormatYUV is set to NV12 , If $(((((Y + (V_1 * 8 - 1)) / (V_1 * 8)) * V_1) - 1) \% 2) == 0$, then Frame Height In Blocks Minus 1 = $((Y + (V_1 * 8 - 1)) / (V_1 * 8)) * V_1$ else then Frame Height In Blocks Minus 1 = $((Y + (V_1 * 8 - 1)) / (V_1 * 8)) * V_1 - 1$	BDW
	15:13 Reserved	
	Exists If: //Decoder Only	
Format: MBZ		
12:0 Frame Width In Blocks Minus 1		
Exists If: //Decoder Only		
Format: U13-1		
(The number of blocks in width) - 1. This value is calculated using the number of samples per line X and horizontal sampling factor of the first component H ₁ in Frame header. See the note following this table. For interleaved components, $((X + (H_1 * 8 - 1)) / (H_1 * 8)) * H_1 - 1$. For non-interleaved components, $(X + 7) / 8 - 1$.		

MFX_MPEG2_PIC_STATE

MFX_MPEG2_PIC_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This must be the very first command to issue after the surface state, the pipe select and base address setting commands. For MPEG-2 the encoder is called per slice-group, however the picture state is called per picture. Notice that a slice-group is a group of consecutive slices that no non-trivial slice headers are inserted in between.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MPEG2_PIC_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	3h MPEG2_COMMON
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		0h	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	0h	
	Format:	OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)= 00Bh, used for normal decode and encode mode000h, a special case to provide a dummy image state for stitch mode operation. In this case, fields in DW1 which is part of the dummy image state command are ignored by hardware.	
	Format:	=n Total Length - 2	
1	31:28	f_code[1][1]. Used for backward motion vector prediction. See ISO/IEC 13818-2 7.6.3.1 for details	
	27:24	f_code[1][0]. Used for backward motion vector prediction. See ISO/IEC 13818-2 7.6.3.1 for details	

MFX_MPEG2_PIC_STATE												
23:20	<p>f_code[0][1] Used for forward motion vector prediction. See ISO/IEC 13818-2 7.6.3.1 for details</p>											
19:16	<p>f_code[0][0] Used for forward motion vector prediction. See ISO/IEC 13818-2 7.6.3.1 for details</p>											
15:14	<p>Intra DC Precision</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>See ISO/IEC 13818-2 6.3.10 for details.</p>	Project:	All	Format:	U2							
Project:	All											
Format:	U2											
13:12	<p>Picture Structure This field specifies whether the picture is encoded in the form of a frame picture or one field (top or bottom) picture. See ISO/IEC 13818-2 6.3.10 for details.Format = MPEG_PICTURE_STRUCTURE00 = Reserved01 = MPEG_TOP_FIELD10 = MPEG_BOTTOM_FIELD11 = MPEG_FRAME</p>											
11	<p>TFF (Top Field First) When two fields are stored in a picture, this bit indicates if the top field is the first field.For a frame P picture, the value 1 indicates that the top field of the reconstructed frame is the first field output by the decoding process, the same as defined in ISO/IEC 13818-2 6.3.10. Particularly, it is used by the hardware to calculate derivative motion vectors from the dual-prime motion vectors.For a field P picture, hardware uses this bit together with the Picture Structure to determine if the current picture is the Second Field. In this case, the definition of this bit differs from ISO/IEC 13818-2 6.3.10 - software must derive the value for this bit according to the following relation:Picture Structure = top fieldPicture Structure = bottom fieldSecond Field = 0TFF = 1TFF = 0Second Field = 1TFF = 0TFF = 1</p>											
10	<p>Frame Prediction Frame DCT This field provides constraints on the DCT type and prediction type. It affects the syntax of the bitstream.</p>											
9	<p>Concealment Motion Vector Flag This field indicates if the concealment motion vectors are coded in intra macroblocks. It affects the syntax of the bitstream.</p>											
8	<p>Quantizer Scale Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>MPEG_Q_SCALE_TYPE</td> </tr> </table> <p>This field specifies the quantizer scaling type.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>MPEG_QSCALE_LINEAR</td> </tr> <tr> <td>1h</td> <td></td> <td>D MPEG_QSCALE_NONLINEAR esc</td> </tr> </tbody> </table>	Format:	MPEG_Q_SCALE_TYPE	Value	Name	Description	0h		MPEG_QSCALE_LINEAR	1h		D MPEG_QSCALE_NONLINEAR esc
Format:	MPEG_Q_SCALE_TYPE											
Value	Name	Description										
0h		MPEG_QSCALE_LINEAR										
1h		D MPEG_QSCALE_NONLINEAR esc										
7	<p>Intra VLC Format This field is used by VLD</p>											

MFX_MPEG2_PIC_STATE																	
	6	<p>Scan Order</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>MPEG_INVERSESCAN_TYPE</td> </tr> </table> <p>This field specifies the Inverse Scan method for the DCT-domain coefficients in the blocks of the current picture.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>MPEG_ZIGZAG_SCAN</td> </tr> <tr> <td>1h</td> <td></td> <td>MPEG_ALTERNATE_VERTICAL_SCAN</td> </tr> </tbody> </table>	Format:	MPEG_INVERSESCAN_TYPE	Value	Name	Description	0h		MPEG_ZIGZAG_SCAN	1h		MPEG_ALTERNATE_VERTICAL_SCAN				
	Format:	MPEG_INVERSESCAN_TYPE															
Value	Name	Description															
0h		MPEG_ZIGZAG_SCAN															
1h		MPEG_ALTERNATE_VERTICAL_SCAN															
	5:0	Reserved															
2	31	<p>I Slice Concealment Mode</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder</td> </tr> </table> <p>This field controls how MPEG decoder handles MB concealment in I Slice</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Intra Concealment</td> <td>Using Coefficient values to handle MB concealment</td> </tr> <tr> <td>1h</td> <td>Inter Concealment</td> <td>Using Motion Vectors to handle MB concealment</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 100%; text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">If this field is set to "1", driver must provide a valid forward reference picture (both top and bottom Field must be valid)</td> </tr> </tbody> </table>	Project:	BDW	Exists If:	//Decoder	Value	Name	Description	0h	Intra Concealment	Using Coefficient values to handle MB concealment	1h	Inter Concealment	Using Motion Vectors to handle MB concealment	Programming Notes	If this field is set to "1", driver must provide a valid forward reference picture (both top and bottom Field must be valid)
	Project:	BDW															
Exists If:	//Decoder																
Value	Name	Description															
0h	Intra Concealment	Using Coefficient values to handle MB concealment															
1h	Inter Concealment	Using Motion Vectors to handle MB concealment															
Programming Notes																	
If this field is set to "1", driver must provide a valid forward reference picture (both top and bottom Field must be valid)																	
	30	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ											
Project:	BDW																
Format:	MBZ																

MFX_MPEG2_PIC_STATE			
29:28	P/B Slice Concealment Mode		
	Project:	BDW	
	Exists If:	//Decoder	
	This field controls how MPEG decoder handles MB concealment in P/B Slice.		
	Value	Name	Description
	00b	INTER	If left MB is NOT Intra MB type (including skipMB), use left MB inter prediction mode [frame/field or forward/backward/bi] and MV final values as concealment. Otherwise (left MB is Intra MB), use forward reference (same polarity for field pic) with MV final values set to 0.
	01b	LEFT	If left MB is NOT Intra MB type (including skipMB), use left MB inter prediction mode [frame/field or forward/backward/bi] and MV final values as concealment. Otherwise (left MB is Intra MB), use left MB dct_dc_pred[cc] values for concealment (Macroblock is concealed as INTRA MB and dct_dc_pred[cc] are DC predictor for Luma, Cr, Cb data)
	10b	ZERO	Always use forward reference (same polarity for field pic) with MV final values set to 0 (Macroblock is concealed as INTER coded)
	11b	INTRA	Use left MB dct_dc_pred[cc] values for concealment (Macroblock is concealed as INTRA MB and dct_dc_pred[cc] are DC predictor for Luma, Cr, Cb data)
	27	Reserved	
Project:		BDW	
Format:		MBZ	
26:25	P/B Slice Predicted BiDir Motion Type Override - Bi-direction MV Type Override		
	Project:	BDW	
	Exists If:	//Decoder	
	This field is only applicable if the Concealment Motion Type is predicted to be Bi-directional. (It is only possible if "P/B Slice Concealment Mode" is set to "00" or "01" and left MB is a bi-directional MB).		
	Value	Name	Description
	0h	BID	Keep Bi-direction Prediction
	1h	RESERVED	
2h	FWD	Only use Forward Prediction (Backward MV is forced to invalid)	
3h	BWD	Only use Backward Prediction (Forward MV is forced to invalid)	

MFX_MPEG2_PIC_STATE															
	24	<p>P/B Slice Predicted Motion Vector Override Final MV value Override</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder</td> </tr> </table> <p>This field is only applicable if the Concealment Motion Vectors are non-zero. It is only possible if "P/B Slice Concealment Mode" is set to "00" or "01" and left MB has non-zero motion vectors).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Predicted</td> <td>Motion Vectors use predicted values</td> </tr> <tr> <td>1h</td> <td>ZERO</td> <td>Motion Vectors force to 0</td> </tr> </tbody> </table>	Project:	BDW	Exists If:	//Decoder	Value	Name	Description	0h	Predicted	Motion Vectors use predicted values	1h	ZERO	Motion Vectors force to 0
	Project:	BDW													
	Exists If:	//Decoder													
	Value	Name	Description												
	0h	Predicted	Motion Vectors use predicted values												
	1h	ZERO	Motion Vectors force to 0												
	23:15	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
	Format:	MBZ													
	14	<p>LoadSlicePointerFlag - LoadBitStreamPointerPerSlice</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder</td> </tr> </table> <p>To support multiple slice picture and additional header/data insertion before and after an encoded slice. When this field is set to 0, bitstream pointer is only loaded once for the first slice of a frame. For subsequent slices in the frame, bitstream data are stitched together to form a single output data stream. When this field is set to 1, bitstream pointer is loaded for each slice of a frame. Basically bitstream data for different slices of a frame will be written to different memory locations.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Load BitStream Pointer only once for the first slice of a frame</td> </tr> <tr> <td>1h</td> <td></td> <td>Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field</td> </tr> </tbody> </table>	Exists If:	//Encoder	Value	Name	Description	0h		Load BitStream Pointer only once for the first slice of a frame	1h		Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field		
	Exists If:	//Encoder													
Value	Name	Description													
0h		Load BitStream Pointer only once for the first slice of a frame													
1h		Load/reload BitStream Pointer only once for the each slice, reload the start location of the bitstream buffer from the Indirect PAK-BSE Object Data Start Address field													
13	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ														
12	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ														
11	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ														

MFX_MPEG2_PIC_STATE																
10:9	<p>Picture Coding Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>MPEG_PICTURE_CODING_TYPE</td> </tr> </table> <p>This field identifies whether the picture is an intra-coded picture (I), predictive-coded picture (P) or bi-directionally predictive-coded picture (B). See ISO/IEC 13818-2 6.3.9 for details.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Reserved</td> </tr> <tr> <td>01b</td> <td>MPEG_I_PICTURE</td> </tr> <tr> <td>10b</td> <td>10 = MPEG_P_PICTURE</td> </tr> <tr> <td>11b</td> <td>MPEG_B_PICTURE</td> </tr> </tbody> </table>	Format:	MPEG_PICTURE_CODING_TYPE	Value	Name	00b	Reserved	01b	MPEG_I_PICTURE	10b	10 = MPEG_P_PICTURE	11b	MPEG_B_PICTURE			
	Format:	MPEG_PICTURE_CODING_TYPE														
	Value	Name														
	00b	Reserved														
01b	MPEG_I_PICTURE															
10b	10 = MPEG_P_PICTURE															
11b	MPEG_B_PICTURE															
8:2	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 80%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ													
Format:	MBZ															
1	<p>MismatchControlDisabled</p> <p>These 2 bits flag disables mismatch control of the inverse transformation for some specific cases during reference reconstruction.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td></td> <td>Mismatch control applies to all MBs</td> </tr> <tr> <td>01b</td> <td></td> <td>Disable mismatch control to all intra MBs whose all AC-coefficients are zero.</td> </tr> <tr> <td>10b</td> <td></td> <td>Disable mismatch control to all MBs whose all AC-coefficients are zero.</td> </tr> <tr> <td>11b</td> <td></td> <td>Disable mismatch control to all MBs.</td> </tr> </tbody> </table>	Value	Name	Description	00b		Mismatch control applies to all MBs	01b		Disable mismatch control to all intra MBs whose all AC-coefficients are zero.	10b		Disable mismatch control to all MBs whose all AC-coefficients are zero.	11b		Disable mismatch control to all MBs.
Value	Name	Description														
00b		Mismatch control applies to all MBs														
01b		Disable mismatch control to all intra MBs whose all AC-coefficients are zero.														
10b		Disable mismatch control to all MBs whose all AC-coefficients are zero.														
11b		Disable mismatch control to all MBs.														
0	<p>Disable Mismatch</p> <p>To disable MPEG2 IDCT fixed point arithmetic correction</p>															
3	<p style="text-align: center;">31</p> <p>Slice Concealment Disable Bit</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decode</td> </tr> </table> <p>If VINunit detects the next slice starting position is either out-of-bound or smaller than or equal to the current slice starting position, VIN will set the current slice to be 1 MB and force VMDunit to do slice concealment on the next slice. This bit will disable this feature and the MB data from the next slice will be decoded from bitstream.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Enable [Default]</td> <td>VIN will force next slice to be concealment if detects slice boundary error</td> </tr> <tr> <td>1h</td> <td>Disable</td> <td>VIN will not force next slice to be in concealment</td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center;">Programming Notes</p> <p>Driver has an option to detect the scenario given in description (above) and remove the second (out-of-order) slice. In this case, hardware will decode the first slice in completion and do concealment till the third slice. It should yield a picture with better quality this way.</p> </div>	Project:	BDW	Exists If:	//Decode	Value	Name	Description	0h	Enable [Default]	VIN will force next slice to be concealment if detects slice boundary error	1h	Disable	VIN will not force next slice to be in concealment		
Project:	BDW															
Exists If:	//Decode															
Value	Name	Description														
0h	Enable [Default]	VIN will force next slice to be concealment if detects slice boundary error														
1h	Disable	VIN will not force next slice to be in concealment														

MFX_MPEG2_PIC_STATE																							
	30:29	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																			
	Format:	MBZ																					
	28:24	Reserved																					
	23:16	FrameHeightInMBsMinus1[7:0] (Picture Height in Macroblocks) <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U8</td> </tr> </table>	Format:	U8																			
	Format:	U8																					
15:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>MBZ for future supporting width > 4K</td> </tr> </table>	Format:	MBZ for future supporting width > 4K																				
Format:	MBZ for future supporting width > 4K																						
7:0	FrameWidthInMBsMinus1[7:0] (Picture Width in Macroblocks) <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table>	Project:	All	Format:	U8																		
Project:	All																						
Format:	U8																						
4	31:16 MinFrameWSize <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U16</td> </tr> </table> <p>- Minimum Frame Size [15:0] (16-bit) (Encoder Only) Minimum Frame Size is specified to compensate for intel Rate Control Currently zero fill (no need to perform emulation byte insertion) is done only to the end of the CABAC_ZERO_WORD insertion (if any) at the last slice of a picture. Intel encoder parameter, not part of DXVA. The caller should always make sure that the value, represented by Minimum Frame Size, is always less than maximum frame size FrameBitRateMax (DWORD 10 bits 29:16). This field is reserved in Decode mode.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,0003FFFFh]</td> <td></td> <td>The programmable range when MinFrameWSizeUnits is 00.</td> </tr> <tr> <td>[0,000FFFFFFh]</td> <td></td> <td>The Programmable range when MinFrameWSizeUnits is 01.</td> </tr> <tr> <td>[0,03FFFFFFh]</td> <td></td> <td>The Programmable range when MinFrameWSizeUnits is 10.</td> </tr> <tr> <td>[0, FFFFFFFFh]</td> <td></td> <td>The Programmable range when MinFrameWSizeUnits is 11.</td> </tr> <tr> <td>0h</td> <td>[Default]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U16	Value	Name	Description	[0,0003FFFFh]		The programmable range when MinFrameWSizeUnits is 00.	[0,000FFFFFFh]		The Programmable range when MinFrameWSizeUnits is 01.	[0,03FFFFFFh]		The Programmable range when MinFrameWSizeUnits is 10.	[0, FFFFFFFFh]		The Programmable range when MinFrameWSizeUnits is 11.	0h	[Default]	
Project:	All																						
Format:	U16																						
Value	Name	Description																					
[0,0003FFFFh]		The programmable range when MinFrameWSizeUnits is 00.																					
[0,000FFFFFFh]		The Programmable range when MinFrameWSizeUnits is 01.																					
[0,03FFFFFFh]		The Programmable range when MinFrameWSizeUnits is 10.																					
[0, FFFFFFFFh]		The Programmable range when MinFrameWSizeUnits is 11.																					
0h	[Default]																						
	15	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ																	
	Project:	All																					
	Format:	MBZ																					
14:12	RoundInterAC, rounding precision for non-Intra AC000: +1/16001: +2/16010: +3/16011: +4/16100: +5/16101: +6/16110: +7/16111: +8/16																						
	11	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																			
Format:	MBZ																						

MFX_MPEG2_PIC_STATE															
	10:8	RoundIntraAC <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3</td> </tr> </table> rounding precision for Intra AC000: +1/16001: +2/16010: +3/16011: +4/16100: +5/16101: +6/16110: +7/16111: +8/16	Project:	All	Format:	U3									
	Project:	All													
	Format:	U3													
	7	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
	Format:	MBZ													
	6:4	RoundInterDC rounding Precision for non-Intra-DC000: +1/16001: +2/16010: +3/16011: +4/16100: +5/16101: +6/16110: +7/16111: +8/16													
3	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ														
2:1	RoundIntraDC rounding Precision for Intra-DC00: +1/801: +2/810: +3/811: +4/8														
0	Reserved														
5	31:17	Reserved (for future Mask bits)													
	16	FrameSizeControlMask Frame size conformance maskThis field is used when MacroblockStatEnable is set to 1. <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th style="width: 80%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Do not change Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control</td> </tr> <tr> <td>1h</td> <td></td> <td>Replace Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control values in MFC_IMAGE_STATUS control register.</td> </tr> </tbody> </table>	Value	Name	Description	0h		Do not change Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control	1h		Replace Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control values in MFC_IMAGE_STATUS control register.				
	Value	Name	Description												
	0h		Do not change Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control												
1h		Replace Slice Quantization Parameter values in MFC_MPEG2_SLICEGROUP_STATE with suggested slice QP value for frame level Rate control values in MFC_IMAGE_STATUS control register.													
15:13	Reserved														
12	InterMBForceCBPZeroControlMask <table border="1" style="width: 100%; margin-top: 5px;"> <tr> <td style="width: 70%;">Format:</td> <td>U1</td> </tr> </table> Inter MB Force CBP ZERO mask. <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0, FFFFFFFFh]</td> <td></td> <td></td> </tr> <tr> <td>0h</td> <td></td> <td>No effect</td> </tr> <tr> <td>1h</td> <td></td> <td>Zero out all A/C coefficients for the inter MB violating Inter Conformance</td> </tr> </tbody> </table>	Format:	U1	Value	Name	Description	[0, FFFFFFFFh]			0h		No effect	1h		Zero out all A/C coefficients for the inter MB violating Inter Conformance
Format:	U1														
Value	Name	Description													
[0, FFFFFFFFh]															
0h		No effect													
1h		Zero out all A/C coefficients for the inter MB violating Inter Conformance													

MFX_MPEG2_PIC_STATE			
11:10	MinFrameWSizeUnits		
	This field is the Minimum Frame Size Units		
	Value	Name	Description
	00b	compatibility mode	Minimum Frame Size is in old mode (words, 2bytes)
	01b	16 byte	Minimum Frame Size is in 16bytes
	10b	4Kb	Minimum Frame Size is in 4Kbytes
	11b	16Kb	Minimum Frame Size is in 16Kbytes
	9	MBRateControlMask	
		MB Rate Control conformance maskThis field is ignored when MacroblockStatEnable is disabled or MB level Rate control flag for the current MB is disable in Macroblock Status Buffer.	
		Value	Name
0h			Do not change QP values of inter macroblock with suggested QP values in Macroblock Status Buffer
1h		Apply RC QP delta for all macroblock	
8	Reserved		
7	Reserved		
	Format:	MBZ	
6:4	Reserved		
3	FrameBitRateMinReportMask		
	This is a mask bit controlling if the condition of frame level bit count is less than FrameBitRateMin.		
	Value	Name	Description
	0h	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.
1h	Enable	set bit0 and bit 1of MFC_IMAGE_STATUS control register if the total frame level bit counter is less than or equal to Frame Bit rate Minimum limit.	
2	FrameBitRateMaxReportMask		
	This is a mask bit controlling if the condition of frame level bit count exceeds FrameBitRateMax.		
	Value	Name	Description
	0h	Disable	Do not update bit0 of MFC_IMAGE_STATUS control register.
1h	Enable	set bit0 and bit 1 of MFC_IMAGE_STATUS control register if the total frame level bit counter is greater than or equal to Frame Bit rate Maximum limit.	

MFX_MPEG2_PIC_STATE														
	1	<p>InterMBMaxSizeReportMask This is a mask bit controlling if the condition of any inter MB in the frame exceeds InterMBMaxSize.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Do not update bit0 of MFC_IMAGE_STATUS control register.</td> </tr> <tr> <td>1h</td> <td></td> <td>set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Inter MB Conformance Max size limit.</td> </tr> </tbody> </table>	Value	Name	Description	0h		Do not update bit0 of MFC_IMAGE_STATUS control register.	1h		set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Inter MB Conformance Max size limit.			
	Value	Name	Description											
0h		Do not update bit0 of MFC_IMAGE_STATUS control register.												
1h		set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Inter MB Conformance Max size limit.												
	0	<p>IntraMBMaxSizeReportMask This is a mask bit controlling if the condition of any intra MB in the frame exceeds IntraMBMaxSize.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Do not update bit0 of MFC_IMAGE_STATUS control register.</td> </tr> <tr> <td>1h</td> <td></td> <td>set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.</td> </tr> </tbody> </table>	Value	Name	Description	0h		Do not update bit0 of MFC_IMAGE_STATUS control register.	1h		set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.			
Value	Name	Description												
0h		Do not update bit0 of MFC_IMAGE_STATUS control register.												
1h		set bit0 of MFC_IMAGE_STATUS control register if the total bit counter for the current MB is greater than the Intra MB Conformance Max size limit.												
6 [ExistsIf]Encode Only	31:28	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
	27:16	<p>InterMBMaxSize</p> <table border="1"> <tr> <td>Default Value:</td> <td>FFFh</td> </tr> </table> <p>This field, Inter MB Conformance Max size limit, indicates the allowed max bit count size for Inter MB</p>	Default Value:	FFFh										
	Default Value:	FFFh												
15:12	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
Format:	MBZ													
11:0	<p>IntraMBMaxSize</p> <table border="1"> <tr> <td>Default Value:</td> <td>FFFh</td> </tr> </table> <p>This field, Intra MB Conformance Max size limit, indicates the allowed max bit count size for Intra MB</p>	Default Value:	FFFh											
Default Value:	FFFh													
7	31:1	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
	0	<p>VSL top MB Trans8x8flag</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Encode Only</td> </tr> </table> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable</td> <td>VSL will only fetch the current MB data.</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>When this bit is set VSL will make extra fetch to memory to fetch the MB data for top MB.</td> </tr> </tbody> </table>	Project:	BDW	Exists If:	//Encode Only	Value	Name	Description	0	Disable	VSL will only fetch the current MB data.	1	Enable
Project:	BDW													
Exists If:	//Encode Only													
Value	Name	Description												
0	Disable	VSL will only fetch the current MB data.												
1	Enable	When this bit is set VSL will make extra fetch to memory to fetch the MB data for top MB.												

MFX_MPEG2_PIC_STATE										
8 [ExistsIf]Encode Only	31:24	<p>SliceDeltaQPMax[3]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>This field is the Slice level delta QP for total bit-count above FrameBitRateMax - first 1/8 region This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame exceeds FrameBitRateMax but is within 1/8 of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of (FrameBitRateMax, (FrameBitRateMax+ FrameBitRateMaxDelta»3).</p> <p>Range: [-30,30]</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> </tr> <tr> <td>1h</td> <td>Enable</td> </tr> </tbody> </table>	Format:	S7	Value	Name	0h	Disable	1h	Enable
	Format:	S7								
	Value	Name								
	0h	Disable								
1h	Enable									
23:16	<p>SliceDeltaQPMax[2]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/8 and below 1/4 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between 1/8 and ¼ of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»3), (FrameBitRateMax+ FrameBitRateMaxDelta»2).</p>	Format:	S7							
Format:	S7									
15:8	<p>SliceDeltaQPMax[1]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/4 and below 1/2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between ¼ and ½ of FrameBitRateMaxDelta above FrameBitRateMax, i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»2), (FrameBitRateMax+ FrameBitRateMaxDelta»1).</p>	Format:	S7							
Format:	S7									
7:0	<p>SliceDeltaQPMax[0]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for bit-count above FrameBitRateMax - above 1/2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is above FrameBitRateMax by more than half the distance of FrameBitRateMaxDelta , i.e., in the range of ((FrameBitRateMax+ FrameBitRateMaxDelta»1), infinite).</p>	Format:	S7							
Format:	S7									

MFX_MPEG2_PIC_STATE				
9 [ExistsIf]Encode Only	31:24	<p>SliceDeltaQPMin[3]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for total bit-count below FrameBitRateMin - first 1/8 region This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is less than FrameBitRateMin and greater than or equal to 1/8 the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of $[(\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 3), \text{FrameBitRateMin}]$.</p>	Format:	S7
	Format:	S7		
	23:16	<p>SliceDeltaQPMin[2]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for bit-count below FrameBitRateMin - below 1/8 and above 1/4 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between one-eighth and quarter the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of $[(\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 2), (\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 3)]$.</p>	Format:	S7
	Format:	S7		
15:8	<p>SliceDeltaQPMin[1]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice level delta QP for bit-count below FrameBitRateMin - below 1/4 and above 1/2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is between quarter and half the distance of FrameBitRateMinDelta from FrameBitRateMin, i.e., in the range of $[(\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 1), (\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 2)]$.</p>	Format:	S7	
Format:	S7			
7:0	<p>SliceDeltaQPMin[0]</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">S7</td> </tr> </table> <p>Range: [-30,30]</p> <p>This field is the Slice Level Delta QP for bit-count below FrameBitRateMin - below 1/2 This field is used to calculate the suggested slice QP into the MFC_IMAGE_STATUS control register when total bit count for the entire frame is below FrameBitRateMin by more than half the distance of FrameBitRateMinDelta , i.e., in the range of $[0, (\text{FrameBitRateMin} - \text{FrameBitRateMinDelta} \gg 1)]$.</p>	Format:	S7	
Format:	S7			

MFX_MPEG2_PIC_STATE											
10 [ExistsIf]Encode Only	31	<p>FrameBitrateMaxUnit This field is the Frame Bitrate Maximum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Byte</td> <td>FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0</td> </tr> <tr> <td>1h</td> <td>Kilobyte</td> <td>FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0</td> </tr> </tbody> </table>	Value	Name	Description	0h	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0	1h	Kilobyte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0
	Value	Name	Description								
	0h	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMaxUnitMode is 1 and in units of 128 Bytes if FrameBitrateMaxUnitMode is 0								
	1h	Kilobyte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0								
	30	<p>FrameBitrateMaxUnitMode BitFiel This field is the Frame Bitrate Maximum Limit Units.dDesc</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Compatibility mode</td> <td>FrameBitRateMaxUnit is in old mode (128b/16Kb)</td> </tr> <tr> <td>1h</td> <td>New mode</td> <td>FrameBitRateMaxUnit is in new mode (32byte/4Kb)</td> </tr> </tbody> </table>	Value	Name	Description	0h	Compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)	1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)
	Value	Name	Description								
	0h	Compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)								
	1h	New mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)								
	29:16	<p>FrameBitRateMax This field is the Frame Bitrate Maximum Limit. This field along with FrameBitrateMaxUnit determines maximum allowed bits in a frame before multi-pass gets triggered (when enabled). In other words, multi-pass is triggered when the actual frame byte count exceeds this value. When FrameBitrateMaxUnitMode is 0(compatibility mode) bits 16:27 should be used, bits 28 and 29 should be 0.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0-512KB</td> <td></td> <td>The programmable range 0-512KB when FrameBitrateMaxUnit is 0.</td> </tr> <tr> <td>0-8190KB</td> <td></td> <td>The programmable range 0-8190KB when FrameBitrateMaxUnit is 1.</td> </tr> </tbody> </table>	Value	Name	Description	0-512KB		The programmable range 0-512KB when FrameBitrateMaxUnit is 0.	0-8190KB		The programmable range 0-8190KB when FrameBitrateMaxUnit is 1.
	Value	Name	Description								
	0-512KB		The programmable range 0-512KB when FrameBitrateMaxUnit is 0.								
	0-8190KB		The programmable range 0-8190KB when FrameBitrateMaxUnit is 1.								
15	<p>FrameBitrateMinUnit This field is the Frame Bitrate Minimum Limit Units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Byte</td> <td>FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0</td> </tr> <tr> <td>1h</td> <td>KiloByte</td> <td>FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0</td> </tr> </tbody> </table>	Value	Name	Description	0h	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0	1h	KiloByte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0	
Value	Name	Description									
0h	Byte	FrameBitRateMax is in units of 32 Bytes when FrameBitrateMinUnitMode is 1 and in units of 128 Bytes if FrameBitrateMinUnitMode is 0									
1h	KiloByte	FrameBitRateMax is in units of 4KBytes Bytes when FrameBitrateMaxUnitMode is 1 and in units of 16KBytes if FrameBitrateMaxUnitMode is 0									
14	<p>FrameBitrateMinUnitMode This field is the Frame Bitrate Minimum Limit Units.ValueNameDescriptionProject</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>compatibility mode</td> <td>FrameBitRateMaxUnit is in old mode (128b/16Kb)</td> </tr> <tr> <td>1h</td> <td>New Mode</td> <td>FrameBitRateMaxUnit is in new mode (32byte/4Kb)</td> </tr> </tbody> </table>	Value	Name	Description	0h	compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)	1h	New Mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)	
Value	Name	Description									
0h	compatibility mode	FrameBitRateMaxUnit is in old mode (128b/16Kb)									
1h	New Mode	FrameBitRateMaxUnit is in new mode (32byte/4Kb)									

MFX_MPEG2_PIC_STATE										
	13:0	<p>FrameBitRateMin</p> <p>This field is the Frame Bitrate Minimum Limit ()This field along with FrameBitrateMinUnit determines minimum allowed bits in a Frame before Multi-Pass gets triggered (when enabled). In other words, multi-pass is triggered when the actual frame byte count is less than this value. When FrameBitrateMinUnitMode is 0 (compatibility mode) bits 0:11 should be used, bits 12 and 13 should be 0. Range: The programmable range 0-512KB When FrameBitrateMinUnit is in 0. Programmable range is 0-8190 KB when FrameBitrateMinUnit is in 1</p>								
11 [ExistsIf]Encode Only	31	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
	30:16	<p>FrameBitRateMaxDelta</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Access:</td> <td>None</td> </tr> <tr> <td>Format:</td> <td>U15</td> </tr> </table> <p>This field is used to select the slice delta QP when FrameBitRateMax Is exceeded. It shares the same FrameBitrateMaxUnit. The programmable range is either 0- 512KB or 4MBB in FrameBitrateMaxUnit of 128 Bytes or 16KB respectively.</p> <p>This field is used to select the slice delta QP when FrameBitRateMax Is exceeded. It shares the same FrameBitrateMaxUnit. When FrameBitrateMaxUnitMode is 0(compatibility mode) bits 16:27 should be used, bits 28, 29 and 30 should be 0.</p>	Default Value:	0h	Project:	All	Access:	None	Format:	U15
	Default Value:	0h								
	Project:	All								
Access:	None									
Format:	U15									
15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ					
Project:	All									
Format:	MBZ									
14:0	<p>FrameBitRateMinDelta</p> <p>This field is used to select the slice delta QP when FrameBitRateMin Is exceeded. It shares the same FrameBitrateMinUnit. When FrameBitrateMinUnitMode is 0(compatibility mode) bits 0:11 should be used, bits 12, 13 and 14 should be 0.Note: HW requires the following condition $FrameBitRateMinDelta \leq 2 * FrameBitRateMin$ Must be true, otherwise it may cause unpredicted behavior.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0-1024KB</td> <td></td> <td>The programmable range 0-1024KB When FrameBitrateMinUnit is in 32Bytes.</td> </tr> <tr> <td>0-16380KB</td> <td></td> <td>Programmable range is 0-16380KB when FrameBitrateMinUnit is in 4Kbytes.</td> </tr> </tbody> </table>	Value	Name	Description	0-1024KB		The programmable range 0-1024KB When FrameBitrateMinUnit is in 32Bytes.	0-16380KB		Programmable range is 0-16380KB when FrameBitrateMinUnit is in 4Kbytes.
Value	Name	Description								
0-1024KB		The programmable range 0-1024KB When FrameBitrateMinUnit is in 32Bytes.								
0-16380KB		Programmable range is 0-16380KB when FrameBitrateMinUnit is in 4Kbytes.								
12	31:21	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									

MFX_MPEG2_PIC_STATE			
20	VMD Error Logic		
	Project: BDW		
	Value	Name	
	0	Disable [Default]	
	1	Enable	
	Error Handling		
	Reserved		
	Format: MBZ		
	18	VAD Error Logic	
		Project: BDW	
Value		Name	
0		Enable [Default]	
1		Disable	
Error reporting ON in case of premature Slice done CABAC Engine will auto decode the bitstream in case of premature slice done.			
17	Reserved		
	Project: BDW		
16	Reserved		
	Format: MBZ		
15:0	Reserved		
	Format: MBZ		



MFX_PAK_INSERT_OBJECT

MFX_PAK_INSERT_OBJECT	
Project:	BDW
Source:	VideoCS
Length Bias:	2
Description	
<p>The MFX_PAK_INSERT_OBJECT command is the first primitive command for the AVC and MPEG2 Encoding Pipeline.</p> <p>This command is issued to setup the control and parameters of inserting a chunk of compressed/encoded bits into the current bitstream output buffer starting at the specified bit location to perform the actual insertion by transferring the command inline data to the output buffer max, 32 bits at a time.</p> <p>It is a variable length command as the data to be inserted are presented as inline data of this command. It is a multiple of 32-bit (1 DW), as the data bus to the bitstream buffer is 32-bit wide.</p> <p>Multiple insertion commands can be issued back to back in a series. It is host software's responsibility to make sure their corresponding data will properly stitch together to form a valid H.264 bitstream.</p> <p>Internally, MFX hardware will keep track of the very last two bytes' (the very last byte can be a partial byte) values of the previous insertion. It is required that the next Insertion Object Command or the next PAK Object Command to perform the start code emulation sequence check and prevention 0x03 byte insertion with this end condition of the previous insertion.</p> <p>Hardware will keep track of an output bitstream buffer current byte position and the associated next bit insertion position index. Data to be inserted can be a valid H.264 NAL units or a partial NAL unit. Certain NAL unit has a minimum byte size requirement. As such the hardware will optionally (enabled by STATE Command) determines the number of CABAC_ZERO_WORD to be inserted to the end of the current NAL, based on the minimum byte size of a NAL and the actual bin count of the encoded Slice. Since prior to the CABAC_ZERO_WORD insertion, the RBSP or EBSP is already byte-aligned, so each CABAC_ZERO_WORD insertion is actually a 3-byte sequence 0x00 00 03. The inline data may have already been processed for start code emulation byte insertion, except the possibility of the last 2 bytes plus the very last partial byte (if any). Hence, when hardware performing the concatenation of multiple consecutive insertion commands, or concatenation of an insertion command and a PAK object command, it must check and perform the necessary start code emulation byte insert at the junction. The inline data is required to be byte aligned on the left (first transmitted bit order) and may or may not be byte aligned on the right (last transmitted bits).</p> <p>The command will specify the bit offset of the last valid DW. Each insertion state command defines a chunk of bits (compressed data) to be inserted at a specific location of the output compressed bitstream in the output buffer. Depend on CABAC or CAVLC encoding mode (from Slice State), PAK Object Command is always ended in byte aligned output bitstream except for CABAC header insertion which is bit aligned. In the aligned cases, PAK will perform 0 filling in CAVLC mode, and 1 filling in CABAC mode.</p> <p>Insertion data can include: any encoded syntax elements bit data before the encoded Slice Data (PAK Object Command) of the current Slice SPS NAL PPS NAL SEI NAL Other Non-Slice NAL Leading_Zero_8_bits (as many bytes as there is) Start Code Prefix NAL Header Byte Slice Header Any encoded syntax elements bit data after the encoded Slice Data (PAK Object Command) of the current Slice and prior to the next encoded Slice Data of the next Slice or prior to the end of the bitstream, whichever comes first Cabac_Zero_Word or Trailing_Zero_8bits (as many bytes as there is).</p> <p>Anything listed above before a Slice Data Context switch interrupt is not supported by this command.</p>	
DWord	Bit
	Description

MFX_PAK_INSERT_OBJECT										
0	31:29	Command Type <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>3h PARALLEL_VIDEO_PIPE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	3h PARALLEL_VIDEO_PIPE	Format:	OpCode				
	Default Value:	3h PARALLEL_VIDEO_PIPE								
	Format:	OpCode								
	28:27	Pipeline <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>2h MFX_PAK_INSERT_OBJECT</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	2h MFX_PAK_INSERT_OBJECT	Format:	OpCode				
	Default Value:	2h MFX_PAK_INSERT_OBJECT								
	Format:	OpCode								
	26:24	Media Command Opcode <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h MFX_COMMON</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MFX_COMMON	Format:	OpCode				
	Default Value:	0h MFX_COMMON								
	Format:	OpCode								
	23:21	SubOpcode A <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>2h</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	2h	Format:	OpCode				
Default Value:	2h									
Format:	OpCode									
20:16	SubOpcode B <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>8h</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	8h	Format:	OpCode					
Default Value:	8h									
Format:	OpCode									
15:12	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
11:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0h Excludes DWord (0,1) = Variable Length in DW</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0h Excludes DWord (0,1) = Variable Length in DW	Format:	=n Total Length - 2					
Default Value:	0h Excludes DWord (0,1) = Variable Length in DW									
Format:	=n Total Length - 2									
1	31:18	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
	17:16	DataByteOffset - SrcDataStartingByteOffset[1:0] Source Data Starting Byte Position within the very first inline DW.								
15	HeaderLengthExcludeFrmSize In case this flag is on, bits are NOT accumulated during current access unit coding neither for Cabac Zero Word insertion bits counting or for output in MMIO register MFC_BITSTREAM_BYTECOUNT_FRAME_NO_HEADER. When using HeaderLengthExcludeFrmSize for header insertion, the software needs to make sure that data comes already with inserted start code emulation bytes. SW shouldn't set EmulationFlag bit (Bit 3 of DWORD1 of MFX_PAK_INSERT_OBJECT). <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>NO_ACCUMULATION</td> <td>Bits during current call are not accumulated</td> </tr> <tr> <td>0</td> <td>ACCUMULATE</td> <td>All bits accumulated</td> </tr> </tbody> </table>	Value	Name	Description	1	NO_ACCUMULATION	Bits during current call are not accumulated	0	ACCUMULATE	All bits accumulated
Value	Name	Description								
1	NO_ACCUMULATION	Bits during current call are not accumulated								
0	ACCUMULATE	All bits accumulated								

MFX_PAK_INSERT_OBJECT											
14	<p>Slice Header Indicator</p> <p>This bit indicates if the insert object is a slice header. In the VDEnc mode, PAK only gets this command at the beginning of the frame for slice position X=0, Y=0. It internally generates the header that needs to be inserted per slice. For VDEnc mode, this bit should always be set.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>SLICE_HEADER</td> <td>Insertion Object is a Slice Header. The command is stored internally by HW and is used for inserting slice headers.</td> </tr> <tr> <td>0</td> <td>LEGACY</td> <td>Legacy Insertion Object command. The PAK Insertion Object command is not stored in HW.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>In VDENC mode, we support only Slice layer without partitioning RBSP syntax. The payload for PAK_INS_OBJ should contain only start code for Slice header followed by NAL_type and slice header (slice_header() in AVC spec). The payload for PAK_INS_OBJ shouldn't contain CABAC Byte alignment bits. HW adds these alignment bits which are part of slice_data. Example PAK_INS_OBJ payload : 00 00 01 <NAL_type> <slice_header_Byte0><slice_header_Byte LAST> Any zero_bytes that are added before slice header can be inserted by any preceding general PAK_INS_OBJ.</p>		Value	Name	Description	1	SLICE_HEADER	Insertion Object is a Slice Header. The command is stored internally by HW and is used for inserting slice headers.	0	LEGACY	Legacy Insertion Object command. The PAK Insertion Object command is not stored in HW.
Value	Name	Description									
1	SLICE_HEADER	Insertion Object is a Slice Header. The command is stored internally by HW and is used for inserting slice headers.									
0	LEGACY	Legacy Insertion Object command. The PAK Insertion Object command is not stored in HW.									
13:8	<p>DataBitsInLastDW - SrCDataEndingBitInclusion[5:0]</p> <p>Source Data to be included in the very last inline DW. Follows the MSBit is the upper bit of each byte within the DW. The lower byte is actually processed first. For example, SrCDataEndingBitInclusion = 9, bit 7:0 and bit 15 are included as valid header data.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>[1,32]</td> <td></td> </tr> </tbody> </table>		Value	Name	[1,32]						
Value	Name										
[1,32]											
7:4	<p>SkipEmulByteCnt - Skip Emulation Byte Count</p> <p>Skip emulation check for number of starting bytes. It can be programmed from 0 to 15 bytes. For example, to skip the start code that has already prefixed in the bitstream.</p>										
3	<p>EmulationFlag - EmulationByteBitsInsertEnable</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>NONE</td> <td>No emulation</td> </tr> <tr> <td>1</td> <td>EMULATE</td> <td>Instruct the hardware to perform Start Code Prefix (0x 00 00 01/02/03/00) Search and Prevention Byte (0x 03) insertion on the insertion data of this command. It is required that hardware will handle a start code prefix crossing the boundary between insertion commands, or an insertion command followed by a PAK Object command.</td> </tr> </tbody> </table>		Value	Name	Description	0	NONE	No emulation	1	EMULATE	Instruct the hardware to perform Start Code Prefix (0x 00 00 01/02/03/00) Search and Prevention Byte (0x 03) insertion on the insertion data of this command. It is required that hardware will handle a start code prefix crossing the boundary between insertion commands, or an insertion command followed by a PAK Object command.
Value	Name	Description									
0	NONE	No emulation									
1	EMULATE	Instruct the hardware to perform Start Code Prefix (0x 00 00 01/02/03/00) Search and Prevention Byte (0x 03) insertion on the insertion data of this command. It is required that hardware will handle a start code prefix crossing the boundary between insertion commands, or an insertion command followed by a PAK Object command.									
2	<p>LastHeaderFlag - LastSrcHeaderDataInsertCommandFlag</p> <p>To process a series of consecutive insertion commands, this flag (=1) indicates the current command is the last 'header' insertion in the series. In CABAC, hardware must perform the "1" insert for byte align for Slice Header before Slice Data comes in in the next PAK-OBJECT command. In CAVLC, hardware ignores this bit</p>										

MFX_PAK_INSERT_OBJECT											
	1	<p>EndOfSliceFlag - LastDstDataInsertCommandFlag No more insertion command and no more PAK-OBJECT command follows.Flush data out to memory</p>									
	0	<p>BitstreamStartReset - ResetBitStreamStartingPos</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>RESET</td> <td>Reset the bitstream buffer insertion position to the bitstream buffer starting position.</td> </tr> <tr> <td>0</td> <td>INSERT</td> <td>Insert the current command inline data starting at the current bitstream buffer insertion position</td> </tr> </tbody> </table>	Value	Name	Description	1	RESET	Reset the bitstream buffer insertion position to the bitstream buffer starting position.	0	INSERT	Insert the current command inline data starting at the current bitstream buffer insertion position
Value	Name	Description									
1	RESET	Reset the bitstream buffer insertion position to the bitstream buffer starting position.									
0	INSERT	Insert the current command inline data starting at the current bitstream buffer insertion position									
2..n	31:0	<p>Insert Data Payload Actual Data to be inserted to the output bitstream buffer.</p>									

MFX_PIPE_BUF_ADDR_STATE

MFX_PIPE_BUF_ADDR_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This state command provides the memory base addresses for all row stores, StreamOut buffer and reconstructed picture output buffers required by the MFD or MFC Engine (that are in addition to the row stores of the Bit Stream Decoding/Encoding Unit (BSD/BSE) and the reference picture buffers). This is a picture level state command and is common among all codec standards and for both encoder and decoder operating modes. However, some fields may only applicable to a specific codec standard. All Pixel Surfaces (original, reference frame and reconstructed frame) in the Encoder are programmed with the same surface state (NV12 and TileY format), except each has its own frame buffer base address. In the tile format, there is no need to provide buffer offset for each slice; since from each MB address, the hardware can calculated the corresponding memory location within the frame buffer directly.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_PIPE_BUF_ADDR_STATE
		Format:	OpCode
	26:24	Common Opcode	
		Default Value:	0h MFX_COMMON_STATE
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	2h
Format:		OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Format:	=n	
	Fixed Length		
	Value	Name	Description
3Bh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)	BDW

MFX_PIPE_BUF_ADDR_STATE													
1	31:6	<p>Pre Deblocking Destination Address</p> <table border="1"> <tr> <td>Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Specifies the 4K byte aligned frame buffer address for outputting the non-filtered reconstructed YUV picture (i.e. output of final adder in each codec standard, and prior to the deblocking filter unit). This field is ignored if PreDeblockOutEnable is set to 0 (disable).</p>	Format:	GraphicsAddress[31:6]									
	Format:	GraphicsAddress[31:6]											
5:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW										
Project:	BDW												
2	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
15:0	<p>Pre Deblocking Destination Address High</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Pre-Deblocking Destination Address. This field is ignored if PreDeblockOutEnable is set to 0 (disable).</p>	Project:	BDW	Format:	GraphicsAddress[47:32]								
Project:	BDW												
Format:	GraphicsAddress[47:32]												
3	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	14:13	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
		Project:	BDW										
	Format:	MBZ											
12:11	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
10:9	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
	Project:	BDW											
Format:	MBZ												
8:7	<p>Pre Deblocking - Arbitration Priority Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW												
Value	Name												
00b	Highest priority												
01b	Second highest priority												
10b	Third highest priority												
11b	Lowest priority												

		MFX_PIPE_BUF_ADDR_STATE	
	6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Pre Deblocking Destination Address	
		Project:	BDW
		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.	
		Value	Name
		00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)
		01b	UC
		10b	WT
		11b	WB
	4:3	Target Cache (TC) Pre Deblocking Destination Address	
		Project:	BDW
		This field allows the choice of LLC vs eLLC for caching.	
		Value	Name
		00b	eLLC Only - not snooped in GT (BDW)
		01b	LLC Only
		10b	LLC/eLLC Allowed
		11b	L3, LLC, eLLC Allowed
	2	Reserved	
	1:0	Age for QUADLRU (AGE) Pre Deblocking Destination Address	
		Project:	BDW
		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.	
		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
4	31:6	Post Deblocking Destination Address	
		Format:	GraphicsAddress[31:6]
		Specifies the 4K byte aligned frame buffer address for outputting the post-loop filtered reconstructed YUV picture (i.e. output of the deblocking filter unit)This field is ignored if PostDeblockOutEnable is set to 0 (disable).	
		Programming Notes	

MFX_PIPE_BUF_ADDR_STATE		
	5:0	Reserved
		Project: BDW
5	31:16	Reserved
		Format: MBZ
	15:0	Post Deblocking Destination Address High
		Project: BDW Format: GraphicsAddress[47:32] This field is for the upper range of Post-Deblocking Destination Address. This field is ignored if PostDeblockOutEnable is set to 0 (disable).
6	31:15	Reserved
		Format: MBZ
	14:13	Reserved
		Project: BDW
		Format: MBZ
	12:11	Reserved
		Format: MBZ
	10:9	Reserved
		Project: BDW
		Format: MBZ
8:7	Post Deblocking - Arbitration Priority Control	Project: BDW
		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	

MFX_PIPE_BUF_ADDR_STATE

6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Post Deblocking Destination Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 55%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW																		
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																	
4:3	Target Cache (TC) for Post Deblocking Destination Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC</p> <p>00b: eLLC Only ("00" setting points TC selection to PTE which defaults to eLLC)</p> <p>01b: LLC Only (Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present).</p> <p>10b: LLC/eLLC Allowed.</p> <p>11b: L3, LLC, eLLC Allowed.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed					
Project:	BDW																		
Value	Name																		
00b	eLLC Only - not snooped in GT (BDW)																		
01b	LLC Only																		
10b	LLC/eLLC Allowed																		
11b	L3, LLC, eLLC Allowed																		
2	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table>	Project:	BDW															
Project:	BDW																		

MFX_PIPE_BUF_ADDR_STATE														
	1:0	<p>Age for QUADLRU (AGE) for Post Deblocking Destination Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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 good chance of generating hits</td> </tr> <tr> <td>00b</td> <td>Poor good chance of generating hits</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	11b	Good chance of generating hits.	10b	Next good chance of generating hits	01b	Decent good chance of generating hits	00b	Poor good chance of generating hits
	Project:	BDW												
	Value	Name												
	11b	Good chance of generating hits.												
	10b	Next good chance of generating hits												
	01b	Decent good chance of generating hits												
00b	Poor good chance of generating hits													
7	31:6	<p>Original Uncompressed Picture Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>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.</p>	Format:	GraphicsAddress[31:6]										
	Format:	GraphicsAddress[31:6]												
5:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													
8	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
15:0	<p>Original Uncompressed Picture Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Original Uncompressed Picture Source Address. This field is valid for encoding mode only.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]									
Project:	BDW													
Format:	GraphicsAddress[47:32]													
9	31:15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
	14:13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:		BDW												
Format:	MBZ													
12:11	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ											
Format:	MBZ													

MFX_PIPE_BUF_ADDR_STATE			
8:7	Original Uncompressed Picture Source - Arbitration Priority Control		
	Project: BDW		
	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	
	6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Original Uncompressed Picture Source Address	
		Project: BDW	
		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.	
		Value	Name
		00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)
		01b	UC
		10b	WT
11b	WB		
4:3	Target Cache (TC) for Original Uncompressed Picture Source Address		
	Project: BDW		
	This field allows the choice of LLC vs eLLC for caching		
	Value	Name	
	00b	eLLC Only - not snooped in GT (BDW)	
	01b	LLC Only	
2	Reserved		
	Project: BDW		

		MFX_PIPE_BUF_ADDR_STATE	
	1:0	Age for QUADLRU (AGE) for Original Uncompressed Picture Source Address	
		Project:	BDW
		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.	
		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
10	31:6	StreamOut Data Destination Base Address	
		Format:	GraphicsAddress[31:6]
		Specifies the 64 byte aligned address for outputting the per-MB indirect data to memory when StreamOutEnable is set in the MFX_PIPE_MODE_SELECT command. For Decoder : This field is used for transcoding purpose. For Encoder : This field is used for dynamic repeat of frame in PAK for Rate Control. Also used for feeding coding information back to the Host, Video Preprocessing Unit and ENC Unit. All data are written in fixed formats, and therefore all record sizes are known in the hardware. Hardware can calculate the offset into this base address for per-MB data.	
	5:0	Reserved	
		Project:	BDW
		Format:	MBZ
11	31:16	Reserved	
		Format:	MBZ
	15:0	StreamOut Data Destination Base Address High	
		Project:	BDW
		Format:	GraphicsAddress[47:32]
		This field is for the upper range of Original Uncompressed Picture Source Address	
12	31:15	Reserved	
		Format:	MBZ
	14:13	Reserved	
		Project:	BDW
		Format:	MBZ
	12:11	Reserved	
Format:		MBZ	

MFX_PIPE_BUF_ADDR_STATE			
8:7	StreamOut Data Destination - Arbitration Priority Control		
	Project: BDW		
	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	
	6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for StreamOut Data Destination Base Address	
		Project: BDW	
		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.	
		Value	Name
		00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)
		01b	UC
		10b	WT
11b	WB		
4:3	Target Cache (TC) for StreamOut Data Destination Base Address		
	Project: BDW		
	This field allows the choice of LLC vs eLLC for caching		
	Value	Name	
	00b	eLLC Only - not snooped in GT (BDW)	
	01b	LLC Only	
2	Reserved		
	Project: BDW		

MFX_PIPE_BUF_ADDR_STATE														
	1:0	<p>Age for QUADLRU (AGE) for StreamOut Data Destination Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
	Project:	BDW												
	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													
13	31:6	<p>Intra Row Store Scratch Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>This field provides the base address of the scratch buffer (read/write) used by the AVC/VP8 IntraPrediction unit to store MB information of the previous row for processing of each macroblock in the current row. The Intra Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of the current macroblock to address the Intra Row Store. This field is ignored in MPEG2 and VC1 mode. Max 256 cachelines for 4K pixels (1 cacheline for either MBAFF or non-MBAFF) Intra Row Store Scratch Buffer - Arbitration Priority Control</p>	Format:	GraphicsAddress[31:6]										
	Format:	GraphicsAddress[31:6]												
	5:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW													
Format:	MBZ													
14	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
15:0	<p>Intra Row Store Scratch Buffer Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Intra RowStore/Scratch Buffer Base Address This field is ignored in MPEG2 and VC1 mode.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]									
Project:	BDW													
Format:	GraphicsAddress[47:32]													
15	31:15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
	14:13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
		Project:	BDW											
	Format:	MBZ												
	12	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:		BDW												
Format:	MBZ													

MFX_PIPE_BUF_ADDR_STATE																		
	11	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
	Project:	BDW																
	Format:	MBZ																
	10:9	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
	Project:	BDW																
	Format:	MBZ																
	8:7	<p>Intra Row Store Scratch Buffer - Arbitration Priority Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority				
	Project:	BDW																
	Value	Name																
	00b	Highest priority																
	01b	Second highest priority																
	10b	Third highest priority																
11b	Lowest priority																	
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Intra Row Store Scratch Buffer Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW																	
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																
4:3	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW															
Project:	BDW																	
2	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW															
Project:	BDW																	

MFX_PIPE_BUF_ADDR_STATE														
	1:0	<p>Age for QUADLRU (AGE) for Intra Row Store Scratch Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
	Project:	BDW												
	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													
16	31:6	<p>Deblocking Filter Row Store Scratch Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Deblocking Filter Row Store is needed for:</p> <ul style="list-style-type: none"> • AVC and VC1 In-Loop Deblocking Filter • VC1 Overlap-smoothing Filter <p>This field provides the 64-byte aligned base address of the scratch buffer (read and write) used by the deblocking filter unit to store MB information of the previous row for filtering of each macroblock in the current row. The Deblocking Filter Row Store buffer must be 64-byte cacheline aligned. Hardware uses the horizontal address of the current macroblock to address the Deblocking Filter Row Store. Max 6 cachelines for VC1 and MPEG2, and max 4 for AVC (for MBAFF, 2 for non-MBAFF)</p>	Format:	GraphicsAddress[31:6]										
	Format:	GraphicsAddress[31:6]												
	5:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW													
Format:	MBZ													
17	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
15:0	<p>Deblocking Filter Row Store Scratch Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Deblocking Filter Row Store Scratch Buffer Address.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]									
Project:	BDW													
Format:	GraphicsAddress[47:32]													
18	31:15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
	Format:	MBZ												
14:13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
Project:	BDW													
Format:	MBZ													

MFX_PIPE_BUF_ADDR_STATE			
12	Reserved		
	Project:	BDW	
	Format:	MBZ	
	11	Reserved	
		Project:	BDW
	Format:	MBZ	
	10:9	Reserved	
		Project:	BDW
	Format:	MBZ	
	8:7	Deblocking Filter Row Store Scratch - Arbitration Priority Control	
		Project:	BDW
		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	
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Deblocking Filter Row Store Scratch Base Address		
	Project:	BDW	
	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.		
	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	
4:3	Target Cache (TC) for Deblocking Filter Row Store Scratch Base Address		
	Project:	BDW	
	This field allows the choice of LLC vs eLLC for caching		
	Value	Name	
	00b	eLLC Only - not snooped in GT (BDW)	
	01b	LLC Only	
10b	LLC/eLLC Allowed		
11b	L3, LLC, eLLC Allowed		

MFX_PIPE_BUF_ADDR_STATE												
	2	Reserved										
		Project: BDW										
	1:0	Age for QUADLRU (AGE) for Deblocking Filter Row Store Scratch Base Address										
		Project: BDW										
		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 border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	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
	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											
19..50	63:48	Reserved										
		Project: BDW										
		Format: MBZ										
	47:32	Reference Picture Address [n] High										
		Project: BDW										
		Format: Address[47:32]										
		This field is for the upper range of Reference Picture Addresses										
	31:6	Reference Picture Address [n]										
		Format: Address[31:6]										
		<p>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> <ul style="list-style-type: none"> • 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) <p>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.</p>										
	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.											

MFX_PIPE_BUF_ADDR_STATE																		
	5:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
Project:	BDW																	
Format:	MBZ																	
51	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
	Format:	MBZ																
	14:13	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
	Project:	BDW																
	Format:	MBZ																
	12:9	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
	Format:	MBZ																
	8:7	<p>Reference Picture - Arbitration Priority Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority				
	Project:	BDW																
	Value	Name																
00b	Highest priority																	
01b	Second highest priority																	
10b	Third highest priority																	
11b	Lowest priority																	
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Reference Picture Addresses</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>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. Note: There is ONLY ONE LLC/eLLC Cacheability Control (LeLLCCC) for all 16 Reference Picture Addresses (RefAddr[0-15])</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW																	
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																

MFX_PIPE_BUF_ADDR_STATE													
4:3	<p>Target Cache (TC) for Reference Picture Addresses</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching. NOTE: There is ONLY ONE Target Cache (TC) for all 16 Reference Picture Addresses (RefAddr[0-15])</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed
	Project:	BDW											
	Value	Name											
	00b	eLLC Only - not snooped in GT											
	01b	LLC Only											
10b	LLC/eLLC Allowed												
11b	L3, LLC, eLLC Allowed												
2	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW										
	Project:	BDW											
1:0	<p>Age for QUADLRU (AGE) for Reference Picture Addresses</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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. NOTE: There is ONLY ONE Age for QUADLRU (AGE) for all 16 Reference Picture Addresses (RefAddr[0-15])</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
	Project:	BDW											
	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												
52	31:6	<p>Macroblock Buffer Base Address or Decoded Picture Error/Status Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>For decoder: Specifies the 64 byte aligned buffer address for writing a single error/status record into the memory when Pic Error/Status Report Enable is set in the MFX_PIPE_MODE_SELECT Command. The error/status record is written by HW at the end of decoding one single picture. The record is written in a fixed format, total 96-bits in size always. Please refer to "Media VDBOX -> Video Codec -> Other Codec Functions -> MFX Error Handling -> Decoder" session for the output format.</p> <p>For encoder: Specifies the 64 byte aligned buffer address for reading the per-MB indirect data from memory when MacroblockStatEnable is set in the MFX_AVC_IMG_STATE Command. This field is used for dynamic repeat of frame in PAK for Rate Control. Also used for feeding coding information back to the Host, Video Preprocessing Unit, and ENC Unit. All data are written in fixed formats, and therefore all record sizes are known in the hardware. Hardware can calculate the offset into this base address for per-MB data.</p>	Project:	BDW	Format:	GraphicsAddress[31:6]							
		Project:	BDW										
		Format:	GraphicsAddress[31:6]										

MFX_PIPE_BUF_ADDR_STATE													
	5:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW												
Format:	MBZ												
53	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
15:0	Macroblock Buffer Base Address or Decoded Picture Error/Status Buffer Base Address High <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Macroblock Status Buffer Base Address</p>	Project:	BDW	Format:	GraphicsAddress[47:32]								
Project:	BDW												
Format:	GraphicsAddress[47:32]												
54	31:15	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	14:13	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
	Project:	BDW											
	Format:	MBZ											
12:11	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
10:9	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW												
Format:	MBZ												
8:7	Macroblock Status Buffer - Arbitration Priority Control <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW												
Value	Name												
00b	Highest priority												
01b	Second highest priority												
10b	Third highest priority												
11b	Lowest priority												

MFX_PIPE_BUF_ADDR_STATE

6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Macroblock Status Buffer Base Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
Project:	BDW																		
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																	
4:3	Target Cache (TC) for Macroblock Status Buffer Base Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 20%;">Value</th> <th style="width: 80%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed					
Project:	BDW																		
Value	Name																		
00b	eLLC Only - not snooped in GT																		
01b	LLC Only																		
10b	LLC/eLLC Allowed																		
11b	L3, LLC, eLLC Allowed																		
2	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table>	Project:	BDW															
Project:	BDW																		
1:0	Age for QUADLRU (AGE) for Macroblock Status Buffer Base Address	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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					
Project:	BDW																		
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																		

MFX_PIPE_BUF_ADDR_STATE													
55	31:6	<p>Macroblock ILDB StreamOut Buffer Base Address</p> <table border="1"> <tr> <td>Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>Specifies the 64 byte aligned buffer address for writing MB ILDB parameter per MB to memory when Deblocker streamout enable is set in the MFX_PIPE_MODE_SELECT Command. The ildb MB control parameters are written by HW at the end of each decoding MB. Only AVC edge information is being streamed out. It is used in AVC decode mode only.</p>	Format:	GraphicsAddress[31:6]									
	Format:	GraphicsAddress[31:6]											
	5:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW												
Format:	MBZ												
56	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	15:0	<p>Macroblock ILDB StreamOut Buffer Base Address High</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Deblocking Filter Row Store Scratch Address</p>	Project:	BDW	Format:	GraphicsAddress[47:32]							
Project:	BDW												
Format:	GraphicsAddress[47:32]												
57	31:15	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ											
	14:13	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
	Project:	BDW											
	Format:	MBZ											
12:11	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
10:9	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ								
Project:	BDW												
Format:	MBZ												
8:7	<p>Macroblock ILDB StreamOut Buffer - Arbitration Priority Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW												
Value	Name												
00b	Highest priority												
01b	Second highest priority												
10b	Third highest priority												
11b	Lowest priority												

MFX_PIPE_BUF_ADDR_STATE																		
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Macroblock ILDB StreamOut Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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
	Project:	BDW																
	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															
	4:3	<p>Target Cache (TC) for Macroblock ILDB StreamOut Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed				
		Project:	BDW															
		Value	Name															
00b		eLLC Only - not snooped in GT																
01b		LLC Only																
10b		LLC/eLLC Allowed																
11b	L3, LLC, eLLC Allowed																	
2	Reserved																	
1:0	<p>Age for QUADLRU (AGE) for Macroblock ILDB StreamOut Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	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					
	Project:	BDW																
	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																	
58	<p>31:6 Second Macroblock ILDB StreamOut Buffer Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:6]</td> </tr> </table> <p>64 byte aligned buffer. Specifies the 64 byte aligned buffer address for writing MB ILDB parameter per MB to memory when Debocker streamout enable is set in the MFX_PIPE_MODE_SELECT Command. The ildb MB control parameters are written by HW at the end of each decoding MB. Only AVC edge information is being streamed out. It is used in AVC decode mode only.</p>	Format:	GraphicsAddress[31:6]															
Format:	GraphicsAddress[31:6]																	



MFX_PIPE_BUF_ADDR_STATE																		
	5:0	Reserved																
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
Project:	BDW																	
Format:	MBZ																	
59	31:16	Reserved																
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
Format:	MBZ																	
	15:0	Second Macroblock ILDB StreamOut Buffer Base Address High																
		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field is for the upper range of Second Macroblock ILDB StreamOutBuffer Base Address.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]												
Project:	BDW																	
Format:	GraphicsAddress[47:32]																	
60	31:15	Reserved																
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ														
	Format:	MBZ																
	14:13	Reserved																
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ												
	Project:	BDW																
	Format:	MBZ																
12:11	Reserved																	
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ															
Format:	MBZ																	
10:9	Reserved																	
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ													
Project:	BDW																	
Format:	MBZ																	
8:7	Second Macroblock ILDB StreamOut Buffer - Arbitration Priority Control																	
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p>	Project:	BDW															
Project:	BDW																	
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Second Macroblock ILDB StreamOut Buffer Base Address																	
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 30%;">Description</th> </tr> </thead> <tbody> <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</td> </tr> <tr> <td>10b</td> <td>WT</td> <td>Writethrough</td> </tr> <tr> <td>11b</td> <td>WB</td> <td>Writeback</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	Description	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)		01b	UC	Uncacheable	10b	WT	Writethrough	11b	WB	Writeback
Project:	BDW																	
Value	Name	Description																
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)																	
01b	UC	Uncacheable																
10b	WT	Writethrough																
11b	WB	Writeback																

MFX_PIPE_MODE_SELECT

MFX_PIPE_MODE_SELECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>Specifies which codec and hardware module is being used to encode/decode the video data, on a per-frame basis.</p> <p>The MFX_PIPE_MODE_SELECT command specifies which codec and hardware module is being used to encode/decode the video data, on a per-frame basis. It also configures the hardware pipeline according to the active encoder/decoder operating mode for encoding/decoding the current picture. Commands issued specifically for AVC and MPEG2 are ignored when VC1 is the active codec.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 3h PARALLEL_VIDEO_PIPE	
		Format: OpCode	
	28:27	Pipeline	
		Default Value: 2h MFX_COMMON	
		Format: OpCode	
	26:24	Opcode	
		Default Value: 0h MFX_COMMON_STATE	
		Format: OpCode	
	23:21	SubOpA	
Default Value: 0h			
Format: OpCode			
20:16	SubOpB		
	Default Value: 0h MFX_PIPE_MODE_SELECT		
	Format: OpCode		
15:12	Reserved		
	Format: MBZ		
11:0	DWord Length		
	Format: =n Total Length - 2		
	Value	Name	Description
	3h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
1	31	Reserved	
	30	Reserved	
	29	Reserved	
	30	Project: BDW	

MFX_PIPE_MODE_SELECT		
28:27	Reserved	
26	Reserved	
	Project:	BDW
	Format:	MBZ
25	Reserved	
	Project:	BDW
	Format:	MBZ
24	Reserved	
	Project:	BDW
	Format:	MBZ
23:19	Reserved	
	Format:	MBZ
18	Extended stream out enable	
	Format:	U1
	<p>This bit can be set only when VDEnc_Mode is set.</p> <p>When this bit is set and MB stream out is enabled, per MB 1CL of data is streamed out. The actual contents of the stream out are listed in Media VDBOX > Encoder VDEnc mode StreamOut Data Structure Definition.</p> <p>When this bit is not set, per MB ¼ CL data is streamed out. The actual contents of the stream out are listed in Media VDBOX > Encoder StreamOut Mode Data Structure Definition.</p>	
17	Decoder Short Format Mode	
	For IT mode, this bit must be 0.	
	Value	Name
	Description	
	1	Long Format Driver Interface
	0	Short Format Driver Interface [Default]
		[BDW] AVC/VC1/MVC/VP8 Long Format Mode is in use.
		AVC/VC1/MVC/VP8 Short Format Mode is in use Note: There is no Short Format for VP8 yet, so this field must be set to 1 for VP8.

MFX_PIPE_MODE_SELECT

16:15	Decoder Mode select	<p>Each coding standard supports two entry points: VLD entry point and IT (IDCT) entry point. This field selects which one is in use. This field is only valid if Codec Select is 0 (decoder).</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 60%;">Description</th> <th style="width: 15%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>VLD Mode</td> <td>All codec minimum must support this mode Configure the MFD Engine for VLD Mode Note: All codec minimum must support this mode</td> <td></td> </tr> <tr> <td>1h</td> <td>IT Mode</td> <td>Configure the MFD Engine for IT Mode Note: Only VC1 and MPEG2 support this mode</td> <td></td> </tr> <tr> <td>2h</td> <td>Deblocker Mode</td> <td>Configure the MFD Engine for Standalone Deblocker Mode. Require streamout AVC edge control information from preceeding decoding pass.</td> <td>BDW</td> </tr> <tr> <td>3h</td> <td>Reserved</td> <td></td> <td>BDW</td> </tr> </tbody> </table>		Value	Name	Description	Project	0h	VLD Mode	All codec minimum must support this mode Configure the MFD Engine for VLD Mode Note: All codec minimum must support this mode		1h	IT Mode	Configure the MFD Engine for IT Mode Note: Only VC1 and MPEG2 support this mode		2h	Deblocker Mode	Configure the MFD Engine for Standalone Deblocker Mode. Require streamout AVC edge control information from preceeding decoding pass.	BDW	3h	Reserved		BDW
Value	Name	Description	Project																				
0h	VLD Mode	All codec minimum must support this mode Configure the MFD Engine for VLD Mode Note: All codec minimum must support this mode																					
1h	IT Mode	Configure the MFD Engine for IT Mode Note: Only VC1 and MPEG2 support this mode																					
2h	Deblocker Mode	Configure the MFD Engine for Standalone Deblocker Mode. Require streamout AVC edge control information from preceeding decoding pass.	BDW																				
3h	Reserved		BDW																				
14:13	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>		Project:	BDW	Format:	MBZ																
Project:	BDW																						
Format:	MBZ																						
12	Deblocker Stream-Out Enable	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field indicates if Deblocker information is going to be streamout during VLD decoding. For AVC, it is needed to enable the deblocker streamout as the AVC Disable_DLKFilterIdc is a slice level parameters. Driver needs to determine ahead of time if at least one slice of the current frame/ has deblocker ON.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Disable streamout of deblocking control information for standalone deblocker operation.</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td></td> </tr> </tbody> </table>		Project:	BDW	Value	Name	Description	0h	Disable	Disable streamout of deblocking control information for standalone deblocker operation.	1h	Enable										
Project:	BDW																						
Value	Name	Description																					
0h	Disable	Disable streamout of deblocking control information for standalone deblocker operation.																					
1h	Enable																						
11	Pic Error/Status Report Enable.	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field control whether the error/status reporting is enable or not. 0: Disable 1: Enable In decoder modes: Error reporting is written out once per frame. The Error Report frame ID listed in DW3 along with the VLD/IT error status bits are packed into one cache and written to the "Decoded Picture Error/Status Buffer address" listed in the MFX_PIPE_BUF_ADDR_STATE Command. Note: driver shall program different error buffer addresses between pictrue; otherwise, hardware might overwrite previous written data if driver does not read it fast enough. In encoder modes: Not used Please refer to "Media VDBOX -> Video Codec -> Other Codec Functions -> MFX Error Handling -> Decoder" session for the output format.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 40%;">Value</th> <th style="width: 60%;">Name</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> </tr> <tr> <td>1h</td> <td>Enable</td> </tr> </tbody> </table>		Project:	BDW	Value	Name	0h	Disable	1h	Enable												
Project:	BDW																						
Value	Name																						
0h	Disable																						
1h	Enable																						

MFX_PIPE_MODE_SELECT														
10	<p>Stream-Out Enable This field controls whether the macroblock parameter stream-out is enabled during VLD decoding for transcoding purpose.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">Disable</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">Enable</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>In decoder modes: The Stream-Out feature is added to support transcoding. While decoding the input compressed stream, selected decoded information may be used by the encoder for re-compression. In encoder modes: This feature used to perform dynamic Multipass of PAK for conformance pupose. Also it provides feedback to host (ENC) for future needs. Software can use this bit to disable writing PAK steam data to the streamout buffer for last pass of frame in PAK. Thus, save memory bandwidth.</p>		Value	Name	0h	Disable	1h	Enable						
Value	Name													
0h	Disable													
1h	Enable													
9	<p>Post Deblocking Output Enable (PostDeblockOutEnable) This field controls the output write for the reconstructed pixels AFTER the deblocking filter. In MPEG2 decoding mode, if this is enabled, VC1 deblocking filter is used.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">Disable</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">Enable</td> </tr> </tbody> </table>		Value	Name	0h	Disable	1h	Enable						
Value	Name													
0h	Disable													
1h	Enable													
8	<p>Pre Deblocking Output Enable (PreDeblockOutEnable) This field controls the output write for the reconstructed pixels BEFORE the deblocking filter.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">Disable</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">Enable</td> </tr> </tbody> </table>		Value	Name	0h	Disable	1h	Enable						
Value	Name													
0h	Disable													
1h	Enable													
7:6	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="text-align: center;">BDW</td> </tr> <tr> <td>Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>		Project:	BDW	Format:	MBZ								
Project:	BDW													
Format:	MBZ													
5	<p>Stitch Mode</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;">Exists If:</td> <td colspan="2">//CodecSel=Encode and StandardSel=AVC</td> </tr> <tr> <th style="width: 15%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 60%;">Description</th> </tr> <tr> <td style="text-align: center;">0h</td> <td>Not in stitch mode</td> <td></td> </tr> <tr> <td style="text-align: center;">1h</td> <td>In the special stitch mode</td> <td>This mode can be used for any Codec as long as bitfield conditions are met.</td> </tr> </table>		Exists If:	//CodecSel=Encode and StandardSel=AVC		Value	Name	Description	0h	Not in stitch mode		1h	In the special stitch mode	This mode can be used for any Codec as long as bitfield conditions are met.
Exists If:	//CodecSel=Encode and StandardSel=AVC													
Value	Name	Description												
0h	Not in stitch mode													
1h	In the special stitch mode	This mode can be used for any Codec as long as bitfield conditions are met.												
4	<p>Codec Select</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>Decode</td> <td></td> </tr> <tr> <td style="text-align: center;">1h</td> <td>Encode</td> <td>Valid only if StandardSel is AVC and MPEG2)</td> </tr> </tbody> </table>		Value	Name	Description	0h	Decode		1h	Encode	Valid only if StandardSel is AVC and MPEG2)			
Value	Name	Description												
0h	Decode													
1h	Encode	Valid only if StandardSel is AVC and MPEG2)												

MFx_PIPE_MODE_SELECT					
	3:0	Standard Select			
		Value	Name	Description	Project
		0000b	MPEG2		
		0001b	VC1		
		0010b	AVC	Covers both AVC and MVC	
		0011b	JPEG		
		0100b	Reserved		BDW
		0101b	VP8	Decoder starting from BDW	BDW
		0110b	Reserved		
		0111b	Reserved		
	1111b	UVLD	SW decoder w/ embedded micro-controller and co-processor	BDW	
2	31	Reserved			
		Format:	MBZ		
	30	Reserved			
		Project:	BDW		
	29	Reserved			
		Format:	MBZ		
	28	Reserved			
	27	Reserved			
	26	Reserved			
		Project:	BDW		
	25	Reserved			
		Project:	BDW		
	24	VHR MVC Field Reference List Logic Enable			
	Project:	BDW			
	Value	Name	Description		
	0	Disable [Default]	Disable MVC Field Logic		
	1	Enable	VHR MVC Field Enable		
23	Reserved				
	Project:	BDW			
22:21	Reserved				
20:19	Reserved				
	Project:	BDW			
	Format:	MBZ			

MFX_PIPE_MODE_SELECT										
18	Reserved									
	Format:	MBZ								
17	Reserved									
	Project:	BDW								
16	Reserved									
15	Reserved									
14	VLF 720i (Odd Height) in VC1 Mode									
	Project:	BDW								
	This bit indicates VLF write out VC1 picture with odd height (in MBs).									
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable [Default]</td> <td></td> </tr> <tr> <td>1</td> <td>Enable</td> <td>720i Enable</td> </tr> </tbody> </table>	Value	Name	Description	0	Disable [Default]		1	Enable	720i Enable
Value	Name	Description								
0	Disable [Default]									
1	Enable	720i Enable								
13	Reserved									
	Format:	MBZ								
12	Reserved									
	Project:	BDW								
11	Reserved									
10	MPC pref08x8_disable Flag (Default 0)									
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable</td> </tr> <tr> <td>1</td> <td>Enable</td> </tr> </tbody> </table>	Value	Name	0	Disable	1	Enable			
	Value	Name								
0	Disable									
1	Enable									
9	Reserved									
	Project:	BDW								
	Format:	MBZ								
8	Reserved									
	Project:	BDW								
7	Reserved									
6	Clock gate Enable at Slice-level									
	BitFieldDesc:									
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable</td> <td>Disable Slice-level Clock gating, Unit-level Clock gating will apply</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>Enable Slice-level Clock gating, overrides any Unit level Clock gating</td> </tr> </tbody> </table>	Value	Name	Description	0	Disable	Disable Slice-level Clock gating, Unit-level Clock gating will apply	1	Enable	Enable Slice-level Clock gating, overrides any Unit level Clock gating
	Value	Name	Description							
0	Disable	Disable Slice-level Clock gating, Unit-level Clock gating will apply								
1	Enable	Enable Slice-level Clock gating, overrides any Unit level Clock gating								
5	Reserved									
4	Reserved									
	Project:	BDW								
	Format:	MBZ								

MFX_PIPE_MODE_SELECT															
3	3	<p>VDS ILDB Calculation</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This bit forces all MB into INTRA MBs before doing ILDB control generation in VDS.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 35%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable [Default]</td> <td>Use original definition for ILDB calculation.</td> </tr> <tr> <td>1</td> <td>Enable</td> <td>Force neighbor Intra MB = 1 on ILDB BS calculation.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>When the bit is '0', the ILDB control generation will be the same as the original spec (AVC/VC1).</p>	Project:	BDW	Value	Name	Description	0	Disable [Default]	Use original definition for ILDB calculation.	1	Enable	Force neighbor Intra MB = 1 on ILDB BS calculation.		
	Project:	BDW													
	Value	Name	Description												
	0	Disable [Default]	Use original definition for ILDB calculation.												
1	Enable	Force neighbor Intra MB = 1 on ILDB BS calculation.													
2:1	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW												
Project:	BDW														
0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW												
Project:	BDW														
3	31:0	<p>Pic Status/Error Report ID</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Exists If:</td> <td>//Decoder Mode Only</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>In decoder modes: Error reporting is written out once per frame. This field along with the VLD error status bits are packed into one cache and written to the memory location specified by "Decoded Picture Error/Status Buffer address" listed in the MFX_PIPE_BUF_ADDR_STATE Command.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 40%;">Name</th> <th style="width: 45%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>32-bit unsigned</td> <td>Unique ID Number</td> </tr> <tr> <td>1h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Exists If:	//Decoder Mode Only	Format:	U32	Value	Name	Description	0h	32-bit unsigned	Unique ID Number	1h	Reserved	
	Exists If:	//Decoder Mode Only													
	Format:	U32													
	Value	Name	Description												
0h	32-bit unsigned	Unique ID Number													
1h	Reserved														
4	31:0	<p>Media Soft-Reset Counter (per 1000 clocks)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>In decoder modes, this indicates the number of clocks (per 1000) VINunit will wait for inactivity from MFX pipeline before issuing media soft reset. If this counter is set to 0, VINunit will never issue soft media reset. In encoder modes: This counter must be set to 0 to disable media soft reset since encoder mode is not supported.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 40%;">Name</th> <th style="width: 35%;">Project</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disable</td> <td>BDW</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	Project	0	Disable	BDW					
Project:	BDW														
Value	Name	Project													
0	Disable	BDW													

MFX_QM_STATE

MFX_QM_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This is a common state command for AVC encoder modes. For encoder, it represents both the forward QM matrices as well as the decoding QM matrices. This is a Frame-level state. Only Scaling Lists specified by an application are being sent to the hardware. The driver is responsible for determining the final set of scaling lists to be used for decoding the current slice, based on the AVC Spec Table 7-2 (Fall-Back Rules A and B). In MFX AVC PAK mode, PAK needs both forward Q scaling lists and IQ scaling lists. The IQ scaling lists are sent as in MFD in raster scan order. But the Forward Q scaling lists are sent in column-wise raster order (column-by-column) to simplify the H/W. Driver will perform all the scan order conversion for both ForwardQ and IQ.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_MULTI_DW
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MFX_COMMON_STATE
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	7h
Format:		OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	20h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:2	Reserved	
		Format:	MBZ

MFX_QM_STATE												
	1:0	AVC										
		Exists If: //AVC- Decoder Only										
		For AVC QM Type: This field specifies which Quantizer Matrix is loaded.										
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)</td> </tr> <tr> <td>1</td> <td>AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)</td> </tr> <tr> <td>2</td> <td>AVC_8x8_Intra_MATRIX</td> </tr> <tr> <td>3</td> <td>AVC_8x8_Inter_MATRIX</td> </tr> </tbody> </table>	Value	Name	0	AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)	1	AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)	2	AVC_8x8_Intra_MATRIX	3	AVC_8x8_Inter_MATRIX
		Value	Name									
	0	AVC_4x4_Intra_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)										
	1	AVC_4x4_Inter_MATRIX, (Y-4DWs, Cb-4DWs, Cr-4DWs, reserved-4DWs)										
	2	AVC_8x8_Intra_MATRIX										
	3	AVC_8x8_Inter_MATRIX										
	1:0	MPEG2										
Exists If: //MPEG2- Decoder Only												
For MPEG2 QM Type: This field specifies which Quantizer Matrix is loaded.												
<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>MPEG_INTRA_QUANTIZER_MATRIX</td> </tr> <tr> <td>1</td> <td>MPEG_NON_INTRA_QUANTIZER_MATRIX</td> </tr> <tr> <td>2-3</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Name	0	MPEG_INTRA_QUANTIZER_MATRIX	1	MPEG_NON_INTRA_QUANTIZER_MATRIX	2-3	Reserved				
Value	Name											
0	MPEG_INTRA_QUANTIZER_MATRIX											
1	MPEG_NON_INTRA_QUANTIZER_MATRIX											
2-3	Reserved											
2..33	31:0	Forward Quantizer Matrix										
Project:		All										
Format:		U32										
The format of a Quantizer Matrix is an 8x8 matrix in raster order. Each element is an unsigned byte.												

MFX_STATE_POINTER

MFX_STATE_POINTER			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFX_STATE_POINTER command, issued at picture level, is used to set up the indirect pointers for VCS to fetch all the MFX states (Image state, Slice state, etc.) needed for the encoding/decoding process in PAK/IT mode. The encoding/decoding states are presented by state commands, which are grouped into separate sets (picture level, slice level, etc.), and each is stored in its own memory buffer referred by an indirect state pointer. The content of each indirect state buffer is a list of MFX state commands with no special format requirements. The sequence of commands in each indirect state buffer is terminated by a MI_BATCH_BUFFER_END command (acts as the last command marker). Therefore, indirect state buffers can have different and variable length of command sequences.</p> <p>The indirection is designed to facilitate context switching in the middle of a codec operation. The smallest granularity of interruption is designed to be at a completed MB row in AVC/VC1/MPEG2 IT and AVC PAK operating modes as well as in VC1/MPEG2 VLD mode. There is no support for context switch in AVC VLD mode. Hardware supports up to 4 separate indirect state pointers, allowing software to manage the grouping of state commands. During context switch, hardware restores (re-issues) the latest version of each indirect state pointer, if present.</p> <p>MFX_STATE_POINTER command can only program one indirect state pointer at a time. MI_FLUSH will invalidate all indirect state buffer pointers inside VCS.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFX_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MFX_COMMON_STATE
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	6h
		Format:	OpCode
	15:12	Reserved	
		Project:	All
		Format:	MBZ

MFX_STATE_POINTER					
	11:0	DWord Length			
		Default Value:	0h DWORD_COUNT_n		
		Project:	All		
		Format:	=n Total Length - 2		
1	31:5	State Pointer			
		Format:	GeneralStateOffset[31:5]Indirect State Buffer		
	Specifies the 32-byte aligned address of an Indirect State Buffer. This pointer is relative to the General State Base Address.				
	4:2	Reserved			
		Project:	All		
		Format:	MBZ		
	1:0	State Pointer Index			
		Specifies one of the four indirect state pointers to program.			
		Value	Name	Description	Project
		00b		indirect state pointer 0 (image state)	All
01b			indirect state pointer 1 (slice state)sc	All	
10b			indirect state pointer 2		
11b		indirect state pointer 3			

MFX_STITCH_OBJECT

MFX_STITCH_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFC_STITCH_OBJECT command is used when stitch-enabled is set to 1, while CodecSel and StandardSel are set to ENCODE and AVC, respectively. This command is used, for example, to stitch multiple bitstreams to form a transport stream.</p> <p>It is a variable length command as the data to be inserted are presented as either inline data and/or indirect data of this command. Multiple insertion commands can be issued back to back in a series. It is host software's responsibility to make sure their corresponding data will properly stitch together to form a valid output. Hardware keeps track of an output bitstream buffer current byte position and the associated next bit insertion position index. Context switch interrupt is not supported by this command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFC_STITCH_OBJECT
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	0h MFX_COMMON
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		2h	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	Ah	
	Format:	OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1) = Variable Length in DW (>= 3)	
	Format:	=n Total Length - 2	
	If it is 3, it indicates the absent of inline data.		
1	31:18	Reserved	
		Format:	MBZ

MFX_STITCH_OBJECT					
17:16	Source Data Starting Byte Offset Source Data Starting Byte Position within the very first inline DW.				
	Reserved				
	Format: MBZ				
	Source Data Ending Bit Inclusion Source Data to be included in the very last inline DW. Follows the MSBit is the upper bit of each byte within the DW. The lower byte is actually processed first. For example, SrCDataEndingBitInclusion =9, bit 7:0 and bit 15 are included as valid header data.				
	<table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>[1,32]</td> <td></td> </tr> </tbody> </table>	Value	Name	[1,32]	
	Value	Name			
	[1,32]				
	Reserved				
	Reserved				
	2 Last Source Header Data Insert Command Flag To process a series of consecutive insertion commands, this flag (=1) indicates the current command is the last 'header' insertion in the series. In CABAC, hardware must perform the "1" insert for byte align for Slice Header before Slice Data comes in in the next PAK-OBJECT command. In CAVLC, hardware ignores this bit.				
1 Last Destination Data Insert Command Flag					
THIS FIELD MUST BE THE SAME AS Last Source Header Data Insert Command Flag No more insertion command and no more PAK-OBJECT command follows. Flush data out to memory					
0 Reserved					
2	Reserved				
	Format: MBZ				
	18:0 Indirect Data Length				
Project: BDW					
Format: U19					
This field provides the length in bytes of the indirect data. A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect Data Start Address field is ignored. This field must have the same alignment as the Indirect Object Data Start Address.					
3	31:0 Indirect Data Start Address				
	Format: MfxIndirectBitstreamObjectAddress[31:0]				
This field specifies the Graphics Memory starting address of the data to be loaded into the kernel for processing. This pointer is relative to the MFX Indirect Bitstream Object Base Address. Hardware ignores this field if indirect data is not present.					
4..n	31:0 Insert Data Payload Inline data to be inserted to the output bitstream buffer				

MFX_SURFACE_STATE

MFX_SURFACE_STATE	
Project:	BDW
Source:	VideoCS
Length Bias:	2
<p>This command is common for all encoding/decoding modes, to specify the uncompressed YUV picture (i.e. destination surface) or intermediate streamout in/out surface (e.g. coefficient/residual) (field, frame or interleaved frame) format for reading and writing:</p> <ul style="list-style-type: none"> • Uncompressed, original input picture to be encoded • Reconstructed non-filtered/filtered display picture (becoming reference pictures as well for subsequent temporal inter-prediction) <p>Since there is only one media surface state being active during the entire encoding/decoding process, all the uncompressed/reconstructed pictures are defined to have the same surface state. The primary difference among picture surface states is their individual programmed base addresses, which are provided by other state commands and not included in this command. MFX engine is making the association of surface states and corresponding buffer base addresses.</p> <p>MFX engine currently supports only one media surface type for video and that is the NV12 (Planar YUV420 with interleaved U (Cb) and V (Cr)). For optimizing memory efficiency based on access patterns, only TileY is supported. For JPEG decoder, only IMC1 and IMC3 are supported. Pitch can be wider than the Picture Width in pixels and garbage will be there at the end of each line. The following describes all the different formats that are supported and not supported in Gen7 MFX :</p> <ul style="list-style-type: none"> • NV12 - 4:2:0 only; UV interleaved; Full Pitch, U and V offset is set to 0 (the only format supported for video codec); vertical UV offset is MB aligned; UV xoffsets = 0. JPEG does not support NV12 format because non-interleave JPEG has performance issue with partial write (in interleaved UV format) • IMC 1 & 3 - Full Pitch, U and V are separate plane; (JPEG only; U plane + garbage first in full pitch followed by V plane + garbage in full pitch). U and V vertical offsets are block aligned; U and V xoffset = 0; there is no gap between Y, U and V planes. IMC1 and IMC3 are different by a swap of U and V. This is the only format supported in JPEG for all video subsampling types (4:4:4, 4:2:2 and 4:2:0) • We are not supporting IMC 2 & 4 - Full Pitch, U and V are separate plane (JPEG only; U plane first in full pitch followed by V plane in full pitch - U and V plane are side-by-side). U and V vertical offsets are 16-pixel aligned; V xoffset is half-pitch aligned; U xoffset is 0; there is no gap between Y, U and V planes. IMC2 and IMC4 are different by a swap of U and V. • We are not supporting YV12 - half pitch for each U and V plane, and separate planes for Y, U and V (U plane first in half pitch followed by V plane in half pitch). For YV12, U and V vertical offsets are block aligned; U and V xoffset = 0; there is no gap between Y, U and V planes <p>Note that the following data structures are not specified through the media surface state</p> <ul style="list-style-type: none"> • 1D buffers for row-store and other miscellaneous information. • 2D buffers for per-MB data-structures (e.g. DMV biffer, MB info record, ILDB Control and Tcoeff/Stocoeff). <p>This surface state here is identical to the Surface State for deinterlace and sample_8x8 messages described in the Shared Function Volume and Sampler Chapter.</p> <p>For non pixel data, such as row stores, indirect data (Compressed Slice Data, AVC MV record, Coeff record and AVC ILDB record) and streamin/out and output compressed bitstream, a linear buffer is employed. For row</p>	

MFX_SURFACE_STATE

stores, the H/W is designed to guarantee legal memory accesses (read and write). For the remaining cases, indirect object base address, indirect object address upper bound, object data start address (offset) and object data length are used to fully specified their corresponding buffer. This mechanism is chosen over the pixel surface type because of their variable record sizes.

All row store surfaces are linear surface. Their addresses are programmed in Pipe_Buf_Base_State or Bsp_Buf_Base_Addr_State

Programming Notes

VC1 I picture scaling: Even though VC1 allows I reconstructed picture scaling (via RESPIC), as such scaling is only allowed at I picture. All subsequent P (and B) pictures must have the same picture dimensions with the preceding I picture. Therefore, all reference pictures for P or B picture can share the same surface state with the current P and B picture. Note : H/W is not processing RESPIC. Application is no longer expecting intel decoder pipeline and kernel to perform this function, it is going to be done in the video post-processing scaler or display controller scale as a separate step and controller.

All video codec surfaces must be NV12 Compliant, except JPEG. U/V vertical must be MB aligned for all video codec (further constrained for field picture), but JPEG can be block aligned. All video codec and JPEG uses Tiled - Y format only, for uncompressed pixel surfaces.

Even for JPEG planar 420 surface, application may provide only 1 buffers, but there is still only one single surface state for all of them. If IMC equal to 1, 2, 3 or 4, U and V have the pitch same as Y. And U and V will have different offset, each offset is block aligned.

DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h PARALLEL_VIDEO_PIPE	
		Format:	OpCode	
	28:27		Pipeline	
			Default Value:	2h MFX_COMMON
			Format:	OpCode
	26:24		Opcode	
			Default Value:	0h MFX_COMMON_STATE
			Format:	OpCode
	23:21		SubOpA	
			Default Value:	0h
			Format:	OpCode
	20:16		SubOpB	
			Default Value:	1h
			Format:	OpCode
	15:12		Reserved	
			Format:	MBZ

MFX_SURFACE_STATE																						
	11:0	DWord Length Format: =n Total Length - 2																				
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">4h</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	4h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)														
Value	Name	Description																				
4h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)																				
1	31:4	Reserved Format: MBZ																				
	3:0	Surface Id Project: BDW Format: U4 <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0000b</td> <td>Reserved</td> <td>8-bit uncompressed data</td> </tr> <tr> <td style="text-align: center;">0001b</td> <td>Reserved</td> <td>16-bit uncompressed data</td> </tr> <tr> <td style="text-align: center;">0010b</td> <td>Reserved</td> <td>8-bit uncompressed data.</td> </tr> <tr> <td style="text-align: center;">0011b</td> <td>Reserved</td> <td>16-bit uncompressed data</td> </tr> <tr> <td style="text-align: center;">0100b</td> <td>Source Input Picture (encoder)</td> <td>8-bit uncompressed data</td> </tr> <tr> <td style="text-align: center;">0101b</td> <td>Reconstructed Scaled Reference Picture</td> <td>8-bit data</td> </tr> </tbody> </table>	Value	Name	Description	0000b	Reserved	8-bit uncompressed data	0001b	Reserved	16-bit uncompressed data	0010b	Reserved	8-bit uncompressed data.	0011b	Reserved	16-bit uncompressed data	0100b	Source Input Picture (encoder)	8-bit uncompressed data	0101b	Reconstructed Scaled Reference Picture
Value	Name	Description																				
0000b	Reserved	8-bit uncompressed data																				
0001b	Reserved	16-bit uncompressed data																				
0010b	Reserved	8-bit uncompressed data.																				
0011b	Reserved	16-bit uncompressed data																				
0100b	Source Input Picture (encoder)	8-bit uncompressed data																				
0101b	Reconstructed Scaled Reference Picture	8-bit data																				
2	31:18	Height Format: U14-1 Height This field specifies the height of the Picture in units of pixels/residuals. For PLANAR surface formats, this field indicates the height of the Y (luma) plane. Note : Gen7 Video Codecs must program less than and equal to 4K.(In future, it will be ideal to have this field define in a WORD boundary.)AVC - multiple of 2 MB rows for field pictureVC1 - multiple of 4 pixels for field pictureMPEG2 - multiple of 2 MB rows for field picJPEG - multiple of integral MCU (8 or 16 pixels) per picture																				
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,16383]</td> <td></td> <td>representing heights [1,16384]</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th style="text-align: center; color: blue;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • For AVC : For frame picture is a multiple of 16; for field picture is a multiple of 32 • For VC1 : For progressive frames, the frame height and frame width is a multiple of 2 pixels. For interlaced frames, the frame height shall be a multiple of 4 pixels, and its width is a multiple of 2 pixels, based on a PLANAR_420 surface. </td> </tr> </tbody> </table>	Value	Name	Description	[0,16383]		representing heights [1,16384]	Programming Notes	<ul style="list-style-type: none"> • For AVC : For frame picture is a multiple of 16; for field picture is a multiple of 32 • For VC1 : For progressive frames, the frame height and frame width is a multiple of 2 pixels. For interlaced frames, the frame height shall be a multiple of 4 pixels, and its width is a multiple of 2 pixels, based on a PLANAR_420 surface. 												
Value	Name	Description																				
[0,16383]		representing heights [1,16384]																				
Programming Notes																						
<ul style="list-style-type: none"> • For AVC : For frame picture is a multiple of 16; for field picture is a multiple of 32 • For VC1 : For progressive frames, the frame height and frame width is a multiple of 2 pixels. For interlaced frames, the frame height shall be a multiple of 4 pixels, and its width is a multiple of 2 pixels, based on a PLANAR_420 surface. 																						

MFX_SURFACE_STATE									
17:4	<p>Width</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U14-1 Width</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>[0,16383]</td> <td></td> <td>representing widths [1,16384]</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <ul style="list-style-type: none"> • 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 • MFX 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. 	Format:	U14-1 Width	Value	Name	Description	[0,16383]		representing widths [1,16384]
Format:	U14-1 Width								
Value	Name	Description							
[0,16383]		representing widths [1,16384]							
3:2	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								
1:0	<p>Cr(V)/Cb(U) Pixel Offset V Direction</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U0.2 exactly as shown in the original spec</td> </tr> </table> <p>Specifies the distance to the U/V values with respect to the even numbered Y channels in the V direction</p> <p style="text-align: center;">Programming Notes</p> <p>This field is ignored for all formats except PLANAR_420_8</p>	Project:	All	Format:	U0.2 exactly as shown in the original spec				
Project:	All								
Format:	U0.2 exactly as shown in the original spec								

MFX_SURFACE_STATE																																														
3	31:28	<p>Surface Format</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>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. Usage: For 420 planar YUV surface, use 4; for monochrome surfaces, use 12. For monochrome surfaces, hardware ignores control fields for Chroma planes. This field must be set to 4 - PLANAR_420_8, or 12 - Y8_UNORM. Not used for MFX, and is ignored. But for JPEG decoding, this field should be programmed to the same format as JPEG_PIC_STATE. For video codec, it should set to 4 always.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 50%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr><td>0</td><td>YCRCB_NORMAL</td><td></td></tr> <tr><td>1</td><td>YCRCB_SWAPUVY</td><td></td></tr> <tr><td>2</td><td>YCRCB_SWAPUV</td><td></td></tr> <tr><td>3</td><td>YCRCB_SWAPY</td><td></td></tr> <tr><td>4</td><td>PLANAR_420_8</td><td>(NV12, IMC1,2,3,4, YV12)</td></tr> <tr><td>5</td><td>PLANAR_411_8</td><td>Deinterlace Only</td></tr> <tr><td>6</td><td>PLANAR_422_8</td><td>Deinterlace Only</td></tr> <tr><td>7</td><td>STMM_DN_STATISTICS</td><td>Deinterlace Only</td></tr> <tr><td>8</td><td>R10G10B10A2_UNORM</td><td>Sample_8x8 Only</td></tr> <tr><td>9</td><td>R8G8B8A8_UNORM</td><td>Sample_8x8 Only</td></tr> <tr><td>10</td><td>R8B8_UNORM (CrCb)</td><td>Sample_8x8 Only</td></tr> <tr><td>11</td><td>R8_UNORM (Cr/Cb)</td><td>Sample_8x8 Only</td></tr> <tr><td>12</td><td>Y8_UNORM</td><td>Sample_8x8 Only</td></tr> </tbody> </table>	Format:	U4	Value	Name	Description	0	YCRCB_NORMAL		1	YCRCB_SWAPUVY		2	YCRCB_SWAPUV		3	YCRCB_SWAPY		4	PLANAR_420_8	(NV12, IMC1,2,3,4, YV12)	5	PLANAR_411_8	Deinterlace Only	6	PLANAR_422_8	Deinterlace Only	7	STMM_DN_STATISTICS	Deinterlace Only	8	R10G10B10A2_UNORM	Sample_8x8 Only	9	R8G8B8A8_UNORM	Sample_8x8 Only	10	R8B8_UNORM (CrCb)	Sample_8x8 Only	11	R8_UNORM (Cr/Cb)	Sample_8x8 Only	12	Y8_UNORM	Sample_8x8 Only
	Format:	U4																																												
	Value	Name	Description																																											
0	YCRCB_NORMAL																																													
1	YCRCB_SWAPUVY																																													
2	YCRCB_SWAPUV																																													
3	YCRCB_SWAPY																																													
4	PLANAR_420_8	(NV12, IMC1,2,3,4, YV12)																																												
5	PLANAR_411_8	Deinterlace Only																																												
6	PLANAR_422_8	Deinterlace Only																																												
7	STMM_DN_STATISTICS	Deinterlace Only																																												
8	R10G10B10A2_UNORM	Sample_8x8 Only																																												
9	R8G8B8A8_UNORM	Sample_8x8 Only																																												
10	R8B8_UNORM (CrCb)	Sample_8x8 Only																																												
11	R8_UNORM (Cr/Cb)	Sample_8x8 Only																																												
12	Y8_UNORM	Sample_8x8 Only																																												
27	<p>Interleave Chroma</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">Enable</td> </tr> </table> <p>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. For AVC/VC1/MPEG VLD and IT modes : set to Enable to support interleave U/V only. For JPEG : set to Disable for all formats (including 4:2:0) - because JPEG does not support NV12. (This field is needed only if JPEG will support NV12; otherwise is ignored.)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr><td>1</td><td>Enable</td></tr> <tr><td>0</td><td>Disable</td></tr> </tbody> </table>	Format:	Enable	Value	Name	1	Enable	0	Disable																																					
Format:	Enable																																													
Value	Name																																													
1	Enable																																													
0	Disable																																													
26:20	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ																																											
Format:	MBZ																																													

MFX_SURFACE_STATE		
19:3	Surface Pitch	
	Format:	U17-1 pitch in Bytes
	This field specifies the surface pitch in (#Bytes).	
	Value	Name
	[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 ignored by MFX (unless we support YV12)		
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 MFX	
	Value	Name
	Description	
	0	False
1	True	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.		

MFX_SURFACE_STATE												
0	Tile Walk											
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>3D_Tilewalk</td> </tr> </table> <p>(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. This field is ignored by MFX. Internally H/W is always treated this set to 1 for all video codec and for JPEG.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th style="width: 30%;">Name</th> <th style="width: 50%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>XMAJOR</td> <td>TILEWALK_XMAJOR</td> </tr> <tr> <td>1h</td> <td>YMAJOR</td> <td>TILEWALK_YMAJOR</td> </tr> </tbody> </table>	Format:	3D_Tilewalk	Value	Name	Description	0h	XMAJOR	TILEWALK_XMAJOR	1h	YMAJOR	TILEWALK_YMAJOR
	Format:	3D_Tilewalk										
	Value	Name	Description									
0h	XMAJOR	TILEWALK_XMAJOR										
1h	YMAJOR	TILEWALK_YMAJOR										
Programming Notes												
The corresponding cache(s) must be invalidated before a previously accessed surface is accessed again with an altered state of this bit												
4	31											
	Reserved											
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ									
	Format:	MBZ										
	30:16											
	X Offset for U(Cb)											
	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U15 Pixel Offset</td> </tr> </table> <p>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)</p>	Project:	All	Format:	U15 Pixel Offset							
	Project:	All										
	Format:	U15 Pixel Offset										
	Programming Notes											
For PLANAR_420 and PLANAR_422 surface formats, this field must be zero.												
15												
Reserved												
<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ								
Project:	All											
Format:	MBZ											
14:0												
Y Offset for U(Cb)												
<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U15 Pixel Row Offset</td> </tr> </table> <p>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.</p>	Project:	All	Format:	U15 Pixel Row Offset								
Project:	All											
Format:	U15 Pixel Row Offset											
Programming Notes												
For PLANAR_420 and PLANAR_422 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs. For JPEG, this field must be a multiple of 16 pixels.												
5												
31:29												
Reserved												
<table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ											

MFX_SURFACE_STATE			
28:16	<p>X Offset for V(Cr)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U13 Offset in Pixels</td> </tr> </table> <p>This field must be zero for NV12 and IMC 1 and 3</p> <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> <p style="text-align: center;">Programming Notes</p> <p>For PLANAR_420 and PLANAR_422 surface formats, this field must indicate an even number of pixels.</p>	Format:	U13 Offset in Pixels
Format:	U13 Offset in Pixels		
15:0	<p>Y Offset for V(Cr)</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>U16 Row Offset in Pixels</td> </tr> </table> <p>This field specifies the veritical 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.</p> <p style="text-align: center;">Programming Notes</p> <p>For PLANAR_420 surface formats, this field must be multiple of 16 pixels - i.e. multiple MBs. For JPEG, this field must be a multiple of 16 pixels.</p>	Format:	U16 Row Offset in Pixels
Format:	U16 Row Offset in Pixels		

MFX_VC1_DIRECTMODE_STATE

MFX_VC1_DIRECTMODE_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
Exists If:	//VC1 decoding in VLD modes		
<p>This is a picture level command and should be issued only once, even for a multi-slices picture. There is only one DMV buffer for read (when processing a B-picture) and one for write (when processing a P-Picture). Each DMV record is 64 bits per MB, to store the top and bottom field MVs (32-bit MV_{x,y} each).</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_VC1_DIRECTMODE_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	2h VC1_COMMON
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	0h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	2h
Format:		OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0005h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	



MFX_VC1_DIRECTMODE_STATE						
1	31:6	<p>Direct MV Write Buffer Base Address for the Current Picture</p> <p>This field provides the base address of the DMV write buffer to store the motion vectors decoded in the current picture. It is a private buffer used by the MPR hardware only. Its content is not accessed by software. This buffer must be 64-byte cacheline aligned. The write buffer size is 557,056 bytes for 1 frame. Scalable with frame height, but do not scale with frame width as the hardware assumes frame width (in MBs) fixed at 128 (smallest power of 2 value larger than 120 - 1920x1088 screen resolution). This field is only valid for a P picture</p>				
	5:0	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW					
Format:	MBZ					
2 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
		Project:	BDW			
	Format:	MBZ				
	15:0	<p>Direct MV Write Buffer Base Address for the Current Picture [47:32]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field is for the upper range of Direct MV Write Buffer Base Address for the Current Picture. This field is used for 48-bit addressing.</p>	Project:	BDW		
Project:		BDW				
31:15	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
	Project:	BDW				
Format:	MBZ					
3 Project: BDW	14:13	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
		Project:	BDW			
	Format:	MBZ				
	12:11	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:		BDW				
Format:	MBZ					
10:9	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
	Project:	BDW				
Format:	MBZ					

MFX_VC1_DIRECTMODE_STATE															
8:7	<p>Direct MV Write Buffer Base Address for the Current Picture - Arbitration Priority Control</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	U2	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW														
Format:	U2														
Value	Name														
00b	Highest priority														
01b	Second highest priority														
10b	Third highest priority														
11b	Lowest priority														
6:5	<p>Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Direct MV Write Buffer for the Current Picture</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
Project:	BDW														
Value	Name														
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)														
01b	Uncacheable (UC) - non-cacheable														
10b	Writethrough (WT)														
11b	Writeback (WB)														
4:3	<p>Target Cache (TC) for Direct MV Write Buffer for the Current Picture</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field allows the choice of LLC vs eLLC for caching</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed		
Project:	BDW														
Value	Name														
00b	eLLC Only - not snooped in GT (BDW)														
01b	LLC Only														
10b	LLC/eLLC Allowed														
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) for Direct MV Write Buffer for the Current Picture</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field allows the selection of AGE parameter for a given surface in LLC or eLLC.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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
Project:	BDW														
Format:	Enable														
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														

MFX_VC1_DIRECTMODE_STATE															
4	31:6	<p>Direct MV Read Buffer Base Address for the Reference Picture</p> <p>This field provides the base address of the DMV buffer for reference picture. It is a private buffer used by the MPR hardware only. Its content is not accessed by software. All these buffers must be 64-byte cacheline aligned. This field is only valid for a B picture.</p>													
	5:0	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
Format:	MBZ														
5 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Reserved for 64-bit address extension.</p>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
	Format:	MBZ													
15:0	<p>Direct MV Read Buffer Base Address for the Current Picture [47:32]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field is for the upper range of Direct MV Read Buffer Base Address for the Current Picture. This field is used for 48-bit addressing.</p>	Project:	BDW												
Project:	BDW														
6 Project: BDW	31:15	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
	Format:	MBZ													
	14:13	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ									
	Project:	BDW													
Format:	MBZ														
12:11	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW														
Format:	MBZ														
10:9	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW														
Format:	MBZ														
8:7	<p>Direct MV Read Buffer Base Address for the Current Picture - Arbitration Priority Control</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>U2</td> </tr> </table> <p>This field controls the priority of arbitration used in the GAC/GAM pipeline for this surface.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">00b</td> <td>Highest priority</td> </tr> <tr> <td style="text-align: center;">01b</td> <td>Second highest priority</td> </tr> <tr> <td style="text-align: center;">10b</td> <td>Third highest priority</td> </tr> <tr> <td style="text-align: center;">11b</td> <td>Lowest priority</td> </tr> </tbody> </table>	Project:	BDW	Format:	U2	Value	Name	00b	Highest priority	01b	Second highest priority	10b	Third highest priority	11b	Lowest priority
Project:	BDW														
Format:	U2														
Value	Name														
00b	Highest priority														
01b	Second highest priority														
10b	Third highest priority														
11b	Lowest priority														

MFX_VC1_DIRECTMODE_STATE

6:5		Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Direct MV Read Buffer for the Current Picture		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>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.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Use Cacheability Controls from page table / UC with Fence (if coherent cycle)</td> </tr> <tr> <td>01b</td> <td>Uncacheable (UC) - non-cacheable</td> </tr> <tr> <td>10b</td> <td>Writethrough (WT)</td> </tr> <tr> <td>11b</td> <td>Writeback (WB)</td> </tr> </tbody> </table>	Project:	BDW	Value	Name	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)	01b	Uncacheable (UC) - non-cacheable	10b	Writethrough (WT)	11b	Writeback (WB)		
Project:	BDW																	
Value	Name																	
00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)																	
01b	Uncacheable (UC) - non-cacheable																	
10b	Writethrough (WT)																	
11b	Writeback (WB)																	
4:3		Target Cache (TC) for Direct MV Read Buffer for the Current Picture		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC 00b: eLLC Only ("00" setting points TC selection to PTE which defaults to eLLC) 01b: LLC Only (Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present). 10b: LLC/eLLC Allowed. 11b: L3, LLC, eLLC Allowed.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>eLLC Only - not snooped in GT (BDW)</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> </tbody> </table>	Project:	BDW	Value	Name	00b	eLLC Only - not snooped in GT (BDW)	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed		
Project:	BDW																	
Value	Name																	
00b	eLLC Only - not snooped in GT (BDW)																	
01b	LLC Only																	
10b	LLC/eLLC Allowed																	
11b	L3, LLC, eLLC Allowed																	
2		Reserved																
1:0		Age for QUADLRU (AGE) for Direct MV Read Buffer for the Current Picture		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITS, hence need to be replaced least often in caches.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #e1eef6;"> <th style="width: 15%;">Value</th> <th style="width: 85%;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Format:	Enable	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.
Project:	BDW																	
Format:	Enable																	
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.																	

MFX_VC1_PRED_PIPE_STATE

MFX_VC1_PRED_PIPE_STATE			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>This command is used to set the operating states of the MFD Engine beyond the BSD unit. It is used with both VC1 Long and Short format. Driver is responsible to take the intensity compensation enable signal, the LumScale and the LumShift provided from the DXVA2 VC1 interface, and maintain a history of these values for reference pictures. Together with these three parameters specified for the current picture being decoded, driver will derive and supply the above sets of LumScaleX, LumShiftX and intensity compensation enable (single or double, forward or backward) signals. H/W is responsible to take these state values, and use them to build the lookup table (including the derivation of iScale and iShift) for remapping the reference frame pixels, as well as performing the actual pixel remapping calculations/process.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_VC1_PRED_PIPE_STATE
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	2h VC1_COMMON
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		0h	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	1h	
	Format:	OpCode	
15:12	Reserved		
	Project:	All	
	Format:	MBZ	
11:0	DWord Length		
	Default Value:	0004h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n Total Length - 2	
1	31:16	Reserved	
		Format:	MBZ

MFX_VC1_PRED_PIPE_STATE				
	15:14	<p>vin_intensitycomp_Double_FWDen</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U2</td> </tr> </table> <p>for forward reference picture only, to enable top or/and bottom of the reference field enable for single compensation. For frame, may only need one bit. This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U2
	Format:	U2		
	13:12	<p>vin_intensitycomp_Double_BWDen</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U2</td> </tr> </table> <p>for backward reference picture only, no double for backward reference. This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U2
	Format:	U2		
	11:10	<p>vin_intensitycomp_Single_FWDen</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U2</td> </tr> </table> <p>for forward reference picture only, to enable top or/and bottom of the reference field enable for single compensation. For frame, may only need one bit. This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U2
Format:	U2			
9:8	<p>vin_intensitycomp_Single_BWDen</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U2</td> </tr> </table> <p>for backward reference picture only, no double for backward reference. This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U2	
Format:	U2			
7:4	<p>Reference Frame Boundary Replication Mode</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U4</td> </tr> </table> <p>This is a bit field with each bit indicating the corresponding picture's boundary replication mode. Bit 11: reference 3 Bit 10: reference 2 Bit 9: reference 1 Bit 8: reference 0 0 = progressive frame replication 1 = interlace frame replication This field is maintained and provided by driver for both long and short VC1 interface format.</p>	Format:	U4	
Format:	U4			
2	3:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
31:30	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			

MFX_VC1_PRED_PIPE_STATE				
	29:24	<p>LumShift2 - single - FWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
	Format:	U6		
	23:22	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	21:16	<p>LumShift1 - single - FWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
	Format:	U6		
	15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
13:8	<p>LumScale2 - single - FWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6	
Format:	U6			
7:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
5:0	<p>LumScale1 - Single - FWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6	
Format:	U6			
3	31:30	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			

MFX_VC1_PRED_PIPE_STATE	
	29:24 LumShift2- double - FWD Format: U6 This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.
	23:22 Reserved Format: MBZ
	21:16 LumShift1 - double -FWD Format: U6 This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.
	15:14 Reserved Format: MBZ
	13:8 LumScale2 - double - FWD Format: U6 This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.
	7:6 Reserved Format: MBZ
	5:0 LumScale1 - double - FWD Format: U6 This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.
4	31:30 Reserved Format: MBZ
	29:24 LumShift2- single - BWD Format: U6 This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.



MFX_VC1_PRED_PIPE_STATE				
	23:22	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	21:16	<p>LumShift1 - single - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
	Format:	U6		
	15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	13:8	<p>LumScale2 - single - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
Format:	U6			
7:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
5:0	<p>LumScale1 - Single - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6	
Format:	U6			
5	31:30	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
	29:24	<p>LumShift2- double - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCEelement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
Format:	U6			
23:22	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			

MFx_VC1_PRED_PIPE_STATE			
21:16	<p>LumShift1 - double - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
Format:	U6		
15:14	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ		
13:8	<p>LumScale2 - double - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
Format:	U6		
7:6	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ		
5:0	<p>LumScale1 - double - BWD</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U6</td> </tr> </table> <p>This field is maintained and provided by driver for both long and short VC1 interface format. And is derived from the intensity compensation enable flag, wBitstreamPCElement and wBitstreamFcodes parameters provided by the DXVA2 VC1 interface to the driver for each current picture.</p>	Format:	U6
Format:	U6		

MFX_VP8_PAK_OBJECT

MFX_VP8_PAK_OBJECT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MFX_VP8_PAK_OBJECT command is the second primitive command for the VP8 Encoding Pipeline. The MV Data portion of the bitstream is loaded as indirect data object. Before issuing a MFX_VP8_PAK_OBJECT command, all VP8 MFX states need to be valid; therefore the commands used to set these states need to have been issued prior to the issue of this command. MB record must be consecutive with no gaps, hence we do not need MB(x,y) in each MB command. Internal counter will keep track of the current MB address, starting from the first MB. MFX_VP8_PAK_OBJECT command follows the MbType definition like MFD. Encoding statistical data such as the total size of the output bitstream are provided through MMIO registers. Software may access these registers through MI_STORE_REGISTER_MEM command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h MFX_VP8_PAK_OBJECT
		Format:	OpCode
	26:24	Media Command Opcode	
		Default Value:	4h VP8_ENC
		Format:	OpCode
	23:21	SubOpcode A	
		Default Value:	2h
		Format:	OpCode
	20:16	SubOpcode B	
		Default Value:	9h
		Format:	OpCode
	15:12	Reserved	
		Project:	All
		Format:	MBZ
	11:0	DWord Length	
		Default Value:	5h DWORD_COUNT_n
		Project:	All
		Format:	=n Length -2

MFX_VP8_PAK_OBJECT					
1	31:30	Reserved			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:
	Project:	All			
	Format:	MBZ			
29	Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				
28:10	Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				
9:0	Indirect PAK-MV Data Length				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U10</td> </tr> </table> <p>This field provides the length in bytes of the indirect data, which contains all the MVs for the current MB (in any partitioning and subpartitioning form). A value zero indicates that indirect data fetching is disabled - subsequently, the Indirect PAK-MV Data Start Address field is ignored. This field must have the same alignment as the Indirect PAK-MV Data Start Address. This field must be DW aligned (since each MV is 4 bytes in size). Driver has to derived this field from MVsize (MVquantity in DXVA, exact size) *4 bytes per MV.</p>	Format:	U10		
Format:	U10				
2	31:29	Reserved			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ				
	28:0	Indirect PAK-MV Data Start Address Offset			
		<p>This field specifies the memory starting address (offset) of the MV data to be fetched into PAK Subsystem for processing. This pointer is relative to the MFC Indirect PAK-MV Object Base Address. Hardware ignores this field if indirect data is not present, i.e. the Indirect PAK-MV Data Length is set to 0. It is a Dword aligned address in all AVC encoding configuration, since each MV is 4 bytes in size.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,512MB)</td> <td></td> </tr> </tbody> </table>	Value	Name	[0,512MB)
Value	Name				
[0,512MB)					
3.6	127:0	<p>Inline Data</p> <p>All the required MB level controls and parameters for encoding are captured as Inline Data Description - VP8 PAK OBJECT. It has a fixed size of 4 DWs. Its definition is described in the next section.</p>			

MFX_VP8_PIC_STATE

MFX_VP8_PIC_STATE											
Project:	BDW										
Source:	VideoCS										
Length Bias:	2										
This must be the very first command to issue after the surface state, the pipe select and base address setting commands and must be issued before MFX_VP8_IMG_STATE.											
Programming Notes		Project									
Only able to use this instruction for decoder workloads.		BDW									
DWord	Bit	Description									
0	31:29	Command Type									
		Default Value:	3h PARALLEL_VIDEO_PIPE								
		Format:	OpCode								
	28:27	Pipeline									
		Default Value:	2h Video Codec								
		Format:	OpCode								
	26:24	Media Command OpCode									
		Default Value:	4h VP8								
		Format:	OpCode								
	23:21	Sub Opcode A									
		Default Value:	0h VP8 Common								
		Format:	OpCode								
	20:16	Sub Opcode B									
		Default Value:	0h MFX_VP8_PIC_STATE								
Format:		OpCode									
15:12	Reserved										
	Format:	MBZ									
11:0	DWord Length	Format:	=n								
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>000h</td> <td>Excludes DWord (0,1) [Default]</td> <td>A special case to provide a dummy image state for stitch mode operation. In this case, fields in DW1 which is part of the dummy image state command are ignored by hardware."</td> </tr> <tr> <td>024h</td> <td></td> <td>Used for normal decode and encode mode</td> </tr> </tbody> </table>		Value	Name	Description	000h	Excludes DWord (0,1) [Default]	A special case to provide a dummy image state for stitch mode operation. In this case, fields in DW1 which is part of the dummy image state command are ignored by hardware."	024h	
	Value	Name	Description								
	000h	Excludes DWord (0,1) [Default]	A special case to provide a dummy image state for stitch mode operation. In this case, fields in DW1 which is part of the dummy image state command are ignored by hardware."								
024h		Used for normal decode and encode mode									

MFX_VP8_PIC_STATE			
1	31:24	Reserved	
		Exists If: //Decoder / Encoder	
		Format: MBZ	
	23:16	Frame Height Minus 1	
		Exists If: //Decoder / Encoder	
		Format: U8	
			Picture Height in integer number of MBs minus 1, so the min pic height can be program is 16 rows of pixels.
	15:8	Reserved	
		Exists If: //Decoder / Encoder	
		Format: MBZ	
	7:0	Frame Width Minus 1	
		Exists If: //Decoder / Encoder	
Format: U8			
		Picture Width in integer number of MBs minus 1, so the min pic width can be program is 16 pixels.	
2	31:26	Reserved	
		Exists If: //Decoder / Encoder	
		Format: MBZ	
	25:24	Log2 Num of Partition	
		Exists If: //Decoder / Encoder	
		Format: U2	
		Value	Name
		0	1 Token partition
		1	2 Token partition
		2	4 Token partition
	3	8 Token partition	
	23:19	Reserved	
		Exists If: //Decoder / Encoder	
		Format: MBZ	

MFX_VP8_PIC_STATE			
18:16	Deblock Sharpness Level		
	Exists If:	//Decoder / Encoder	
	Format:	U3	
	Specify the sharpness level, as one of the regular deblocking strength control parameters.		
	Programming Notes		
		Set to 0 to disable the use of sharpness control.	
15:14	Reserved		
	Exists If:	//Decoder / Encoder	
	Format:	MBZ	
13	Alternate Ref Pic MV SignBias Flag		
	Exists If:	//Decoder / Encoder	
Alternate Reference Picture MV sign bias flag, specified for non-key frame only.			
12	Golden Ref Picture MV SignBias Flag		
	Exists If:	//Decoder / Encoder	
Golden Reference Picture MV sign bias flag, specified for non-key frame only.			
11	Mode Reference Loop Filter Delta Enabled		
	Exists If:	//Decoder / Encoder	
	Value	Name	Description
	0		Mode or Reference Loop Filter Delta Adjustment for current frame is disabled.
	1		Mode or Reference Loop Filter Delta Adjustment for current frame is enabled.
10	MB NoCoeff SkipFlag		
	Exists If:	//Decoder / Encoder	
	Frame level control if Skip MB (with no non-zero coefficient) is allowed or not.		
	Value	Name	Description
	0		All MBs will have its MB level signaling mb_skip_coeff forced to 0. That is, no skip of coefficient record in the bitstream (even their values are all 0s)
1		Skip MB is enabled in the per MB record.	
9	Update MBSegment Map Flag		
	Exists If:	//Decoder / Encoder	
	Value	Name	Description
	0		Disable segmentation update
	1		Enable segmentation update, and to enable reading segment_id for each MB.

MFX_VP8_PIC_STATE

	8	Segment Enable Flag Exists If: //Decoder / Encoder	
		Value	Name
		0	Disable Segmentation processing in the current frame
		1	Enable Segmentation processing in the current frame
	7	Segmentation ID StreamIn Enable Exists If: //Decoder Only	
		Value	Name
		0	StreamIn Disabled
		1	StreamIn Enabled
		Programming Notes	
		When 0, no input needed.	
	7:6	Reserved Exists If: //Encoder Only Format: MBZ	
	6	Segmentation ID StreamOut Enable Exists If: //Decoder Only	
		Value	Name
		0	StreamOut Disabled
		1	StreamOut Enabled
		Programming Notes	
		When 0, no output needed.	
	5	sKeyFrameFlag Exists If: //Decoder / Encoder	
		Value	Name
		0	Non-Key Frame (P-Frame)
		1	Key Frame (I-Frame)

MFX_VP8_PIC_STATE				
4	DBLKFilterType			
	Exists If:	//Decoder / Encoder		
	To specify VP8 Profile of operation.			
	Value	Name	Description	
	0		Use a full feature normal deblocking filter	
	1		Use a simple filter for deblocking	
	Reserved			
	Exists If:	//Decoder / Encoder		
	Format:	MBZ		
	1	Chroma Full Pixel MC Filter Mode		
		Exists If:	//Decoder / Encoder	
		To specify VP8 Profile of operation.		
Value		Name	Description	
0			Chroma MC filter operates in sub-pixel mode	
1			Chroma MC filter only operates in full pixel position, i.e. no sub-pixel interpolation.	
0	MC Filter Select			
	Exists If:	//Decoder / Encoder		
	To specify VP8 Profile of operation.			
	Value	Name	Description	
	0		6-tap filter (regular filter mode)	
	1		2-tap bilinear filter (simple profile/version mode)	
3	31:30	Reserved		
		Exists If:	//Decoder / Encoder	
		Format:	MBZ	
	29:24	DBLKFilterLevel for Segment3		
		Exists If:	//Decoder / Encoder	
		Format:	U6	
		Value	Name	Description
		0	Signifies disable in loop deblocking operation	This is used to set a VP8 profile without in loop deblocker.
		Programming Notes		
		There are max 4 segments per frame, each segment can have its own deblocking filter level. When segmentation is disabled, only segment 0 parameter is used for the entire frame.		

MFX_VP8_PIC_STATE				
23:22	Reserved			
	Exists If:	//Decoder / Encoder		
	Format:	MBZ		
	21:16	DBLKFilterLevel for Segment2		
		Exists If:	//Decoder / Encoder	
		Format:	U6	
		Value	Name	Description
		0	Signifies disable in loop deblocking operation	This is used to set a VP8 profile without in loop deblocker.
		Programming Notes		
	There are max 4 segments per frame, each segment can have its own deblocking filter level. When segmentation is disabled, only segment 0 parameter is used for the entire frame.			
	15:14	Reserved		
		Exists If:	//Decoder / Encoder	
Format:		MBZ		
13:8	DBLKFilterLevel for Segment1			
	Exists If:	//Decoder / Encoder		
	Format:	U6		
	Value	Name	Description	
	0	Signifies disable in loop deblocking operation	This is used to set a VP8 profile without in loop deblocker.	
	Programming Notes			
There are max 4 segments per frame, each segment can have its own deblocking filter level. When segmentation is disabled, only segment 0 parameter is used for the entire frame.				
7:6	Reserved			
	Exists If:	//Decoder / Encoder		
	Format:	MBZ		



		MFX_VP8_PIC_STATE			
	5:0	DBLKFiterLevel for Segment0			
		Exists If:	//Decoder / Encoder		
		Format:	U6		
		Value	Name	Description	
		0	Signifies disable in loop deblocking operation	This is used to set a VP8 profile without in loop deblocker.	
		Programming Notes			
		There are max 4 segments per frame, each segment can have its own deblocking filter level. When segmentation is disabled, only segment 0 parameter is used for the entire frame.			
		4 Project: BDW	31:25	Reserved	
				Project:	BDW
				Exists If:	//Decoder Only
24:16	Quantizer Value [0][BlockType1=Y1AC]				
	Project:		BDW		
	Exists If:		//Decoder Only		
			Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]		
15:9	Reserved				
	Project:		BDW		
	Exists If:		//Decoder Only		
		Format:	MBZ		
31:0	Reserved				
	Project:	BDW			
	Exists If:	//Encoder Only			
		Format:	MBZ		
8:0	Quantizer Value [0][BlockType0=Y1DC]				
	Project:	BDW			
	Exists If:	//Decoder Only			
		Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]			
5 Project: BDW	31:25	Reserved			
		Project:	BDW		
		Exists If:	//Decoder Only		
		Format:	MBZ		

MFX_VP8_PIC_STATE								
	24:16	Quantizer Value [0][BlockType3=UVAC] <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> <tr><td>Format:</td><td>U9</td></tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only	Format:	U9
	Project:	BDW						
	Exists If:	//Decoder Only						
	Format:	U9						
	15:9	Reserved <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> <tr><td>Format:</td><td>MBZ</td></tr> </table>	Project:	BDW	Exists If:	//Decoder Only	Format:	MBZ
	Project:	BDW						
	Exists If:	//Decoder Only						
	Format:	MBZ						
	31:0	Reserved <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Encoder Only</td></tr> <tr><td>Format:</td><td>MBZ</td></tr> </table>	Project:	BDW	Exists If:	//Encoder Only	Format:	MBZ
	Project:	BDW						
	Exists If:	//Encoder Only						
	Format:	MBZ						
8:0	Quantizer Value [0][BlockType2=UVDC] <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only			
Project:	BDW							
Exists If:	//Decoder Only							
6 Project: BDW	31:25	Reserved <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> <tr><td>Format:</td><td>MBZ</td></tr> </table>	Project:	BDW	Exists If:	//Decoder Only	Format:	MBZ
	Project:	BDW						
	Exists If:	//Decoder Only						
	Format:	MBZ						
	24:16	Quantizer Value [0][BlockType5=Y2AC] <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> <tr><td>Format:</td><td>U9</td></tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only	Format:	U9
Project:	BDW							
Exists If:	//Decoder Only							
Format:	U9							
15:9	Reserved <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Decoder Only</td></tr> <tr><td>Format:</td><td>MBZ</td></tr> </table>	Project:	BDW	Exists If:	//Decoder Only	Format:	MBZ	
Project:	BDW							
Exists If:	//Decoder Only							
Format:	MBZ							
31:0	Reserved <table border="1"> <tr><td>Project:</td><td>BDW</td></tr> <tr><td>Exists If:</td><td>//Encoder Only</td></tr> <tr><td>Format:</td><td>MBZ</td></tr> </table>	Project:	BDW	Exists If:	//Encoder Only	Format:	MBZ	
Project:	BDW							
Exists If:	//Encoder Only							
Format:	MBZ							

MFX_VP8_PIC_STATE								
	8:0	Quantizer Value [0][BlockType4=Y2DC] <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only		
	Project:	BDW						
Exists If:	//Decoder Only							
7 Project: BDW	31:25	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Exists If:	//Decoder Only	Format:	MBZ
	Project:	BDW						
	Exists If:	//Decoder Only						
	Format:	MBZ						
	24:16	Quantizer Value [1][BlockType1=Y1AC] <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only	Format:	U9
	Project:	BDW						
	Exists If:	//Decoder Only						
	Format:	U9						
	15:9	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Exists If:	//Decoder Only	Format:	MBZ
	Project:	BDW						
Exists If:	//Decoder Only							
Format:	MBZ							
31:0	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Exists If:	//Encoder Only	Format:	MBZ	
Project:	BDW							
Exists If:	//Encoder Only							
Format:	MBZ							
8:0	Quantizer Value [1][BlockType0=Y1DC] <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Project:	BDW	Exists If:	//Decoder Only			
Project:	BDW							
Exists If:	//Decoder Only							
8	31:25	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ		
	Exists If:	//Decoder Only						
	Format:	MBZ						
24:16	Quantizer Value [1][BlockType3=UVAC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only	Format:	U9			
Exists If:	//Decoder Only							
Format:	U9							

MFX_VP8_PIC_STATE						
	15:9	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	31:0	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Encoder Only	Format:	MBZ
	Exists If:	//Encoder Only				
	Format:	MBZ				
8:0	Quantizer Value [1][BlockType2=UVDC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only			
Exists If:	//Decoder Only					
9	31:25	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	24:16	Quantizer Value [1][BlockType5=Y2AC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only	Format:	U9
	Exists If:	//Decoder Only				
	Format:	U9				
15:9	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ	
Exists If:	//Decoder Only					
Format:	MBZ					
31:0	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Encoder Only	Format:	MBZ	
Exists If:	//Encoder Only					
Format:	MBZ					
8:0	Quantizer Value [1][BlockType4=Y2DC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only			
Exists If:	//Decoder Only					
10	31:25	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	24:16	Quantizer Value [2][BlockType1=Y1AC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only	Format:	U9
Exists If:	//Decoder Only					
Format:	U9					

MFX_VP8_PIC_STATE			
	15:9	Reserved	
		Exists If: //Decoder Only	
		Format: MBZ	
	31:0	Reserved	
		Exists If: //Encoder Only	
		Format: MBZ	
8:0	Quantizer Value [2][BlockType0=Y1DC]		
	Exists If: //Decoder Only		
	Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]		
11	31:25	Reserved	
		Exists If: //Decoder Only	
		Format: MBZ	
	24:16	Quantizer Value [2][BlockType3=UVAC]	
		Exists If: //Decoder Only	
		Format: U9	
	Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]		
	15:9	Reserved	
		Exists If: //Decoder Only	
		Format: MBZ	
	31:0	Reserved	
		Exists If: //Encoder Only	
		Format: MBZ	
	8:0	Quantizer Value [2][BlockType2=UVDC]	
		Exists If: //Decoder Only	
		Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	
	12	31:25	Reserved
			Exists If: //Decoder Only
Format: MBZ			
24:16		Quantizer Value [2][BlockType5=Y2AC]	
		Exists If: //Decoder Only	
		Format: U9	
Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]			

MFX_VP8_PIC_STATE						
	15:9	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	31:0	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Encoder Only	Format:	MBZ
	Exists If:	//Encoder Only				
	Format:	MBZ				
8:0	Quantizer Value [2][BlockType4=Y2DC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only			
Exists If:	//Decoder Only					
13	31:25	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	24:16	Quantizer Value [3][BlockType1=Y1AC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only	Format:	U9
	Exists If:	//Decoder Only				
	Format:	U9				
15:9	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ	
Exists If:	//Decoder Only					
Format:	MBZ					
31:0	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Encoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Encoder Only	Format:	MBZ	
Exists If:	//Encoder Only					
Format:	MBZ					
8:0	Quantizer Value [3][BlockType0=Y1DC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only			
Exists If:	//Decoder Only					
14	31:25	Reserved <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder Only	Format:	MBZ
	Exists If:	//Decoder Only				
	Format:	MBZ				
	24:16	Quantizer Value [3][BlockType3=UVAC] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U9</td> </tr> </table> Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]	Exists If:	//Decoder Only	Format:	U9
Exists If:	//Decoder Only					
Format:	U9					

MFX_VP8_PIC_STATE		
	15:9	Reserved
		Exists If: //Decoder Only
		Format: MBZ
	31:0	Reserved
		Exists If: //Encoder Only
		Format: MBZ
15	8:0	Quantizer Value [3][BlockType2=UVDC]
		Exists If: //Decoder Only
		Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]
	31:25	Reserved
		Exists If: //Decoder Only
		Format: MBZ
15	24:16	Quantizer Value [3][BlockType5=Y2AC]
		Exists If: //Decoder Only
		Format: U9
		Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]
	15:9	Reserved
		Exists If: //Decoder Only
	Format: MBZ	
15	31:0	Reserved
		Exists If: //Encoder Only
		Format: MBZ
	8:0	Quantizer Value [3][BlockType4=Y2DC]
		Exists If: //Decoder Only
		Quantizer Value [n = Segment_Id = 0..3][BlockType = 0..5]
16	31:6	CoeffProbability StreamIn Base Address
	Exists If: //Decoder Only	
	Format: StreamInAddress[31:6] 64 bytes aligned buffer in linear format. (not tile for better performance)	
		It is specified for non-key frame only. It is the final computed probability table for parsing Coeff in the bitstream. The buffer is unsigned 8-bit * 1056 entries (CoeffProbs[4][8][3][11]).

MFV_VP8_PIC_STATE			
	31:0	Reserved	
		Exists If: //Encoder Only	
		Format: MBZ	
	5:0	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
		Format: MBZ	
17 Project: BDW	31:16	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
			Format: MBZ
	31:0	Reserved	
		Project: BDW	
		Exists If: //Encoder Only	
			Format: MBZ
	15:0	CoeffProbability StreamIn Address	
		Project: BDW	
		Exists If: //Decoder Only	
			This field is for the upper range of CoeffProbability StreamIn Address
18 Project: BDW	31:15	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
			Format: MBZ
	14:13	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
			Format: MBZ
	12:11	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
			Format: MBZ
	10:9	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
			Format: MBZ

MFX_VP8_PIC_STATE				
8:7	CoeffProbability StreamIn - Arbitration Priority Control			
	Project:	BDW		
	Exists If:	//Decoder Only		
	Format:	U2		
	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		
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for CoeffProbability StreamIn Address			
	Project:	BDW		
	Exists If:	//Decoder Only		
	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.			
	Value	Name	Description	Exists If
	00b	Use Cacheability Controls from page table / UC with Fence (if coherent cycle)		//Decoder Only
	01b	UC	Uncacheable - non-cacheable	
10b	WT	Writethrough		
11b	WB	Writeback		
4:3	CoeffProbability StreamIn Address - Target Cache (TC)			
	Project:	BDW		
	Exists If:	//Decoder Only		
	This field allows the choice of LLC vs eLLC for caching			
	Value	Name		
	00b	eLLC Only - not snooped in GT		
01b	LLC Only			
10b	LLC/eLLC Allowed			
11b	L3, LLC, eLLC Allowed			
2	Reserved			
	Project:	BDW		

MFX_VP8_PIC_STATE												
	31:0	Reserved										
		Project: BDW										
		Exists If: //Encoder Only										
		Format: MBZ										
	1:0	CoeffProbability StreamIn Address - Age for QUADLRU (AGE)										
		Project: BDW										
		Exists If: //Decoder Only										
		This field allows the selection of AGE parameter for a given surface in LLC or eLLC.										
		<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	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
	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											
19	31:24	Reserved										
		Exists If: //Decoder / Encoder										
		Format: MBZ										
	23:16	MBSegmentIDTreeProbs[2]										
		Exists If: //Decoder / Encoder										
		Format: U8										
		MBSegmentIDTreeProbs[2:0] probability tree table for CPBAC parsing Segment_ID of each MB.										
	15:8	MBSegmentIDTreeProbs[1]										
		Exists If: //Decoder / Encoder										
		Format: U8										
		MBSegmentIDTreeProbs[2:0] probability tree table for CPBAC parsing Segment_ID of each MB.										
	7:0	MBSegmentIDTreeProbs[0]										
	Exists If: //Decoder / Encoder											
	Format: U8											
	MBSegmentIDTreeProbs[2:0] probability tree table for CPBAC parsing Segment_ID of each MB.											
20	31:24	MBNoCoeffSkipFalseProb										
		Exists If: //Decoder / Encoder										
		Format: U8										
		8-bit probability value for CPBAC parsing of the MBNoCoeffSkip Flag in the bistream.										

MFX_VP8_PIC_STATE						
	23:16	<p>IntraMBProb</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>8-bit probability value for CPBAC parsing of the intra or inter MB type flag in the bitstream.</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
	Format:	U8				
15:8	<p>InterPredFromLastRefProb</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>8-bit probability value for CPBAC parsing of the flag in the bitstream that determines which reference frame to be used for the current MB motion compensation.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
7:0	<p>InterPredFromGRefRefProb</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>8-bit probability value for CPBAC parsing of the flag in the bitstream that determines which reference frame to be used for the current MB motion compensation.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
21	31:24	<p>YModeProb[3]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>YModeProb[3:0] probability tree table for CPBAC parsing Luma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
	Format:	U8				
	23:16	<p>YModeProb[2]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>YModeProb[3:0] probability tree table for CPBAC parsing Luma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder					
Format:	U8					
15:8	<p>YModeProb[1]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>YModeProb[3:0] probability tree table for CPBAC parsing Luma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
7:0	<p>YModeProb[0]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>YModeProb[3:0] probability tree table for CPBAC parsing Luma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
22	31:24	<p>Reserved</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
		Exists If:	//Decoder / Encoder			
Format:	MBZ					

MFX_VP8_PIC_STATE					
	23:16 UVModeProb[2] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>UVModeProb[2:0] probability tree table for CPBAC parsing Chroma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder			
	Format:	U8			
15:8 UVModeProb[1] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>UVModeProb[2:0] probability tree table for CPBAC parsing Chroma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
7:0 UVModeProb[0] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>UVModeProb[2:0] probability tree table for CPBAC parsing Chroma MBType of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
23	31:24 MVUpdateProbs[0][3] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder			
	Format:	U8			
	23:16 MVUpdateProbs[0][2] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder				
Format:	U8				
15:8 MVUpdateProbs[0][1] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
7:0 MVUpdateProbs[0][0] <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				

MFX_VP8_PIC_STATE					
24	31:24	MVUpdateProbs[0][7]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].
	23:16	MVUpdateProbs[0][6]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].
	15:8	MVUpdateProbs[0][5]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].
	7:0	MVUpdateProbs[0][4]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].
25	31:24	MVUpdateProbs[0][11]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB.
	23:16	MVUpdateProbs[0][10]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB.
	15:8	MVUpdateProbs[0][9]	Exists If: //Decoder / Encoder	Format: U8	MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB.

MFX_VP8_PIC_STATE						
	7:0	<p>MVUpdateProbs[0][8]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB.</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder					
Format:	U8					
26	31:24	<p>MVUpdateProbs[0][15]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
	Format:	U8				
	23:16	<p>MVUpdateProbs[0][14]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder					
Format:	U8					
15:8	<p>MVUpdateProbs[0][13]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
7:0	<p>MVUpdateProbs[0][12]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
27	31:24	<p>Reserved</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
	Exists If:	//Decoder / Encoder				
Format:	MBZ					
23:16	<p>MVUpdateProbs[0][18]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					

MFX_VP8_PIC_STATE						
	15:8	<p>MVUpdateProbs[0][17]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
Format:	U8					
7:0	<p>MVUpdateProbs[0][16]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
28	31:24	<p>MVUpdateProbs[1][3]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder Only	Format:	U8
	Exists If:	//Decoder Only				
	Format:	U8				
	23:16	<p>MVUpdateProbs[1][2]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder Only	Format:	U8
	Exists If:	//Decoder Only				
Format:	U8					
15:8	<p>MVUpdateProbs[1][1]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder Only	Format:	U8	
Exists If:	//Decoder Only					
Format:	U8					
7:0	<p>MVUpdateProbs[1][0]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder Only	Format:	U8	
Exists If:	//Decoder Only					
Format:	U8					
		<p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>				
29	31:24	<p>MVUpdateProbs[1][7]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
Format:	U8					
		<p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>				

MFX_VP8_PIC_STATE					
	<p>23:16 MVUpdateProbs[1][6]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder			
	Format:	U8			
<p>15:8 MVUpdateProbs[1][5]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
<p>7:0 MVUpdateProbs[1][4]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
30	<p>31:24 MVUpdateProbs[1][11]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder			
	Format:	U8			
	<p>23:16 MVUpdateProbs[1][10]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder				
Format:	U8				
<p>15:8 MVUpdateProbs[1][9]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				
<p>7:0 MVUpdateProbs[1][8]</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder				
Format:	U8				

MFX_VP8_PIC_STATE						
31	31:24	MVUpdateProbs[1][15] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
	Format:	U8				
	23:16	MVUpdateProbs[1][14] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
Exists If:	//Decoder / Encoder					
Format:	U8					
15:8	MVUpdateProbs[1][13] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
7:0	MVUpdateProbs[1][12] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
32	31:24	Reserved <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
	Exists If:	//Decoder / Encoder				
	Format:	MBZ				
23:16	MVUpdateProbs[1][18] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					
15:8	MVUpdateProbs[1][17] <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8	
Exists If:	//Decoder / Encoder					
Format:	U8					

MFX_VP8_PIC_STATE						
33	7:0	<p>MVUpdateProbs[1][16]</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>U8</td> </tr> </table> <p>MVUpdateProbs[1:0][18:0] probability table for CPBAC parsing of MV update value of each MB. To map into DWord, it becomes MVUpdate[1:0][19:0].</p>	Exists If:	//Decoder / Encoder	Format:	U8
	Exists If:	//Decoder / Encoder				
	Format:	U8				
	31	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
	Exists If:	//Decoder / Encoder				
	Format:	MBZ				
	30:24	<p>RefLFDelta3 (for ALTREF FRAME)</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>S6 2's Compliment</td> </tr> </table> <p>Delta value for reference frame based adjustment of the MB-level's filter level value. RefLFDeltas [ref_frametype = 0 to 3]</p> <div style="background-color: #e6f2ff; padding: 5px; text-align: center;">Programming Notes</div> <p>Please note that although RefDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.</p>	Exists If:	//Decoder / Encoder	Format:	S6 2's Compliment
	Exists If:	//Decoder / Encoder				
	Format:	S6 2's Compliment				
	23	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
Exists If:	//Decoder / Encoder					
Format:	MBZ					
22:16	<p>RefLFDelta2 (for GOLDEN FRAME)</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>S6 2's Compliment</td> </tr> </table> <p>Delta value for reference frame based adjustment of the MB-level's filter level value. RefLFDeltas [ref_frametype = 0 to 3]</p> <div style="background-color: #e6f2ff; padding: 5px; text-align: center;">Programming Notes</div> <p>Please note that although RefDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.</p>	Exists If:	//Decoder / Encoder	Format:	S6 2's Compliment	
Exists If:	//Decoder / Encoder					
Format:	S6 2's Compliment					
15	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ	
Exists If:	//Decoder / Encoder					
Format:	MBZ					

MFX_VP8_PIC_STATE						
	14:8	<p>RefLFDelta1 (for LAST FRAME)</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>S6 2's Compliment</td> </tr> </table> <p>Delta value for reference frame based adjustment of the MB-level's filter level value. RefLFDeltas [ref_frametype = 0 to 3]</p> <p style="text-align: center;">Programming Notes</p> <p>Please note that although RefDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.</p>	Exists If:	//Decoder / Encoder	Format:	S6 2's Compliment
	Exists If:	//Decoder / Encoder				
	Format:	S6 2's Compliment				
	7	<p>Reserved</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ
	Exists If:	//Decoder / Encoder				
	Format:	MBZ				
	6:0	<p>RefLFDelta0 (for INTRA FRAME)</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>S6 2's Compliment</td> </tr> </table> <p>Delta value for reference frame based adjustment of the MB-level's filter level value. RefLFDeltas [ref_frametype = 0 to 3]</p> <p style="text-align: center;">Programming Notes</p> <p>Please note that although RefDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.</p>	Exists If:	//Decoder / Encoder	Format:	S6 2's Compliment
	Exists If:	//Decoder / Encoder				
	Format:	S6 2's Compliment				
	34	31	<p>Reserved</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:
Exists If:		//Decoder / Encoder				
Format:		MBZ				
30:24	<p>ModelFDelta3 (for SPLITMV mode)</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>S6 2's Compliment</td> </tr> </table> <p>Delta value for mode based adjustment of the MB-level's filter level value. ModelFDeltas[MB_Type = 0 to 3]</p> <p style="text-align: center;">Programming Notes</p> <p>Please note that although ModelFDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.</p>	Exists If:	//Decoder / Encoder	Format:	S6 2's Compliment	
Exists If:	//Decoder / Encoder					
Format:	S6 2's Compliment					
23	<p>Reserved</p> <table border="1"> <tr> <td>Exists If:</td> <td>//Decoder / Encoder</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Exists If:	//Decoder / Encoder	Format:	MBZ	
Exists If:	//Decoder / Encoder					
Format:	MBZ					

MFX_VP8_PIC_STATE	
22:16	ModelFDelta2 (for Nearest, Near and New mode)
	Exists If: //Decoder / Encoder
	Format: S6 2's Compliment
	Delta value for mode based adjustment of the MB-level's filter level value. ModelFDeltas[MB_Type = 0 to 3]
	Programming Notes
Please note that although ModelFDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.	
15	Reserved
	Exists If: //Decoder / Encoder
	Format: MBZ
14:8	ModelFDelta1 (for ZEROMV mode)
	Exists If: //Decoder / Encoder
	Format: S6 2's Compliment
	Delta value for mode based adjustment of the MB-level's filter level value. ModelFDeltas[MB_Type = 0 to 3]
	Programming Notes
Please note that although ModelFDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.	
7	Reserved
	Exists If: //Decoder / Encoder
	Format: MBZ
6:0	ModelFDelta0 (for B_PRED mode)
	Exists If: //Decoder / Encoder
	Format: S6 2's Compliment
	Delta value for mode based adjustment of the MB-level's filter level value. ModelFDeltas[MB_Type = 0 to 3]
	Programming Notes
Please note that although ModelFDelta is signed 2's complement, bitstream is sign bit + 6 bit magnitude.	



MFX_VP8_PIC_STATE			
35 Project: BDW	31:0	Segmentation ID Stream Base Address	
		Project: BDW	
		Exists If: //Decoder Only	
		Format: StreamAddress[31:0] 64 bytes linear aligned buffer	
		It is specified when SegmentationIDStreamInEnable or SegmentationIDStreamOutEnable is specified.	
Programming Notes			
Each cache has only 8 bits for 4 segmentation ID from 4 continuous MBs.			
36 Project: BDW	31:16	Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
	15:0	Segmentation ID Stream Base Address [47:32]	
		Project: BDW	
		Exists If: //Decoder Only	
	This field is for the upper range of Segmentation ID Stream Base Address		
	37 Project: BDW	31:15	Reserved
			Project: BDW
			Exists If: //Decoder Only
14:13		Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
12:11		Reserved	
		Project: BDW	
		Exists If: //Decoder Only	
10:9		Reserved	
		Project: BDW	
		Format: MBZ	

MFX_VP8_PIC_STATE

8:7	Segmentation ID Stream - Arbitration Priority Control		
	Project:	BDW	
	Exists If:	//Decoder Only	
	Format:	U2	
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	
6:5	Memory Type: LLC/eLLC Cacheability Control (LeLLCCC) for Segmentation ID Stream Base Address		
	Project:	BDW	
	Exists If:	//Decoder Only	
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.			
	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

MFX_VP8_PIC_STATE															
4:3	<p>Target Cache (TC) Segmentation ID Stream Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> <p>This field controls the L3\$, LLC and eLLC (eDRAM) cacheability for a given surface. Setting of "00" points to PTE settings which defaults to eDRAM (when present). If no eDRAM, the access will be allocated to LLC. Setting of "01", allocates into LLC and victimizes the line to eDRAM. Setting of "10" allows the line to be allocated in either LLC or eDRAM. Setting of "11" is the only option for a memory access to be allocated in L3\$ as well as LLC/eLLC 00b: eLLC Only (<i>"00" setting points TC selection to PTE which defaults to eLLC</i>) 01b: LLC Only (<i>Works at the allocation time, later victimization from LLC downgrades the line to eLLC if present</i>). 10b: LLC/eLLC Allowed. 11b: L3, LLC, eLLC Allowed.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Exists If:	//Decoder Only	Value	Name	00b	eLLC Only - not snooped in GT	01b	LLC Only	10b	LLC/eLLC Allowed	11b	L3, LLC, eLLC Allowed
Project:	BDW														
Exists If:	//Decoder Only														
Value	Name														
00b	eLLC Only - not snooped in GT														
01b	LLC Only														
10b	LLC/eLLC Allowed														
11b	L3, LLC, eLLC Allowed														
2	Reserved														
1:0	<p>Age for QUADLRU (AGE) Segmentation ID Stream Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Exists If:</td> <td>//Decoder Only</td> </tr> </table> <p>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 ("0,1,2") it tends to stay longer in the cache. This option is given to GFX software to be able to decide which surfaces are more likely to generate HITS, hence need to be replaced least often in caches.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>	Project:	BDW	Exists If:	//Decoder Only	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
Project:	BDW														
Exists If:	//Decoder Only														
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														

MFX_WAIT

MFX_WAIT	
Project:	BDW
Source:	VideoCS
Length Bias:	1

This command can be considered the same as an MI_NOOP except that the command parser will not parse the next command until the following happens

- **AVC or VC1 BSD mode:** The command will stall the parser until completion of the BSD object
- **IT, encoder, and MPEG2 BSD mode:** The command will stall the parser until the object package is sent down the pipeline. This command should be used to ensure the preemption enable window occurs during the time the object command is being executed down the pipeline.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	03h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Command Subtype	
		Default Value:	01h MFX_SINGLE_DW
		Format:	OpCode
	26:16	Sub-Opcode	
		Default Value:	0h MFX_WAIT
		Format:	OpCode
	15:10	Reserved	
		Project:	All
		Format:	MBZ
9	Reserved		
8	MFX Sync Control Flag If set, VCS will stall the parser until all prior MFX objects are completed down the MFX pipeline		
7:6	Reserved		
	Project:	All	
	Format:	MBZ	
5:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1)	
	Project:	All	
	Format:	=n	
		Total Length - 2	



MI_ARB_CHECK

MI_ARB_CHECK			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	1		
Description		Project	
The MI_ARB_CHECK is used to check for a change in arbitration. If executed as part of a Ring Buffer the command checks the UHPTR valid bit and if set the head of the ring will jump to the value of the head pointer programmed in the UHPTR.		BDW	
Programming Notes		Project	
This instruction cannot be placed in a batch buffer.			
If execlist is enabled, there is a pending execution list and this command is parsed, then the command streamer will preempt the current context and start executing the new execution list.		BDW	
DWord	Bit	Description	
0	31:29	MI Instruction Type	
		Default Value:	0h MI_INSTRUCTION
		Format:	OpCode
	28:23	MI Instruction Opcode	
		Default Value:	05h MI_ARB_CHECK
		Format:	OpCode
22:0	Reserved		
	Project:	All	
	Format:	MBZ	

MI_ARB_CHECK

MI_ARB_CHECK		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	1	
Description		Project
<p>The MI_ARB_CHECK is used to check for a change in arbitration. If executed as part of a Ring Buffer the command checks the UHPTR valid bit and if set the head of the ring will jump to the value of the head pointer programmed in the UHPTR.</p>		BDW
Programming Notes		Project
<p>This instruction cannot be placed in a batch buffer.</p>		
<p>If execlist is enabled, there is a pending execution list and this command is parsed, then the command streamer will preempt the current context and start executing the new execution list.</p>		BDW
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_INSTRUCTION
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 05h MI_ARB_CHECK
		Format: OpCode
22:0	Reserved	
	Format: MBZ	



MI_ARB_CHECK

MI_ARB_CHECK		
Project:	BDW	
Source:	RenderCS	
Length Bias:	1	
Description	Project	
The MI_ARB_CHECK instruction is used to check the ring buffer double buffered head pointer (register UHPTR). This instruction can be used to pre-empt the current execution of the ring buffer. Note that the valid bit in the updated head pointer register needs to be set for the command streamer to be pre-empted.	BDW	
Programming Notes	Project	
<p>Ring Buffer mode of scheduling:</p> <ul style="list-style-type: none"> The current head pointer is loaded with the updated head pointer register independent of the location of the updated head. If the current head pointer and the updated head pointer register are equal, hardware will automatically reset the valid bit corresponding to the UHPTR. For pre-emption, the wrap count in the ring buffer head register is no longer maintained by hardware. The hardware updates the wrap count to the value in the UHPTR register. <p>Execlist mode of scheduling: MI_ARB_CHK will be used to indicate a command boundary on which Preemption will be honored by Command Streamer in the execlist mode of operation. UHPTR is ignored when processing MI_ARB_CHK in execlist mode.</p>	BDW	
This instruction can be in either a ring buffer or batch buffer.	BDW	
MI_ARB_CHK command must not be programmed in INDIRECT_CTX and BB_PER_CTX_PTR buffers.	BDW	
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
		Default Value: 05h MI_ARB_CHECK
	Format: OpCode	
22:0	Reserved	
	Format: MBZ	

MI_ARB_CHECK

MI_ARB_CHECK			
Project:	BDW		
Source:	VideoCS		
Length Bias:	1		
Description		Project	
<p>The MI_ARB_CHECK is used to check for a change in arbitration. If executed as part of a Ring Buffer the command checks the UHPTR valid bit and if set the head of the ring will jump to the value of the head pointer programmed in the UHPTR.</p>		BDW	
Programming Notes		Project	
<p>This instruction cannot be placed in a batch buffer.</p>			
<p>If execlist is enabled, there is a pending execution list and this command is parsed, then the command streamer will preempt the current context and start executing the new execution list.</p>		BDW	
DWord	Bit	Description	
0	31:29	MI Instruction Type	
		Default Value:	0h MI_INSTRUCTION
		Format:	OpCode
	28:23	MI Instruction Opcode	
		Default Value:	05h MI_ARB_CHECK
		Format:	OpCode
22:0	Reserved		
	Format:	MBZ	



MI_ARB_ON_OFF

MI_ARB_ON_OFF						
Project:	BDW					
Source:	CommandStreamer					
Length Bias:	1					
Description						
<p>The MI_ARB_ON_OFF instruction is used to disable/enable context switching. This instruction can be used to prevent submission of a new execlist from interrupting a command sequence, however lite restore preemption is allowed with in the arbitration disabled command execution zone. Note that context switching will remain disabled until re-enabled through use of this command. This command will also prevent a switch in the case of waiting on events, running out of commands. These will effectively hang the device if allowed to occur while arbitration is off (context switching is disabled.) This command should always be used as an off-on pair with the sequence of instructions to be protected from context switch between MI_ARB_OFF and MI_ARB_ON. Software must use this arbitration control with caution since it has the potential to increase the response time of the Render Engine to pre-emption requests. This is a privileged command; it will not be effective (will be converted to a no-op) if executed from within a non-privileged batch buffer.</p>						
<p>Execution List Mode of Scheduling: The MI_ARB_ON_OFF instruction is used to disable/enable context switching. Context swiching could be either due to preemption or un-succesfull wait for events or semaphore waits. This instruction can be used to prevent submission of a new execlist from interrupting a command sequence, however lite restore preemption is allowed with in the arbitration disabled command execution zone. Note that context switching will remain disabled until re-enabled through use of this command. This command will also prevent a switch in the case of waiting on events, running out of commands. These will effectively hang the device if allowed to occur while arbitration is off (context switching is disabled.)</p> <p>Ring Buffer Mode of Scheduling: The MI_ARB_ON_OFF instruction is used to disable preemption on the preemptable commands. SW can explicitly make section of commands in a command buffer non-preemptable by sandwiching them between ARB_OFF and ARB_ON, HW will ingore preemption request (UHPTR Valid) until arbitration is enabled.</p>						
Programming Notes						
<p>This command must be always be programmed in pairs of off/on in the same command dispatch. Sequence of instructions to be protected from cntext switch or preemption must be programmed between the MI_ARB_OFF and MI_ARB_ON. Software must use this arbitration control with caution since it has the potential to increase the response time of the Render Engine to pre-emption requests. This is a privileged command; it will not be effective (will be converted to a no-op) if executed from within a non-privileged batch buffer.</p>						
<p>HW doesn't treat Arbitration Disabled as equivalent to "Inhibit Synchronous Context Switch" set in CTXT_SR_CTL register. Power management optimizations (RDOP on WT4EVT) available on setting "Inhibit Synchronous Context Switch" are not enabled by default on Arbitration Disabled. SW must explicitly program "Inhibit Synchronous Switch" when Arbitration Disabled to enable power management optimizations.</p>						
DWord	Bit	Description				
0	31:29	<p>Command Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h MI_COMMAND</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MI_COMMAND	Format:	OpCode
Default Value:	0h MI_COMMAND					
Format:	OpCode					

MI_ARB_ON_OFF			
	28:23	MI Command Opcode	
		Default Value:	08h MI_ARB_ON_OFF
	Format:	OpCode	
	22:2	Reserved	
		Format:	MBZ
	1	Reserved	
		Project:	BDW
		Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS
		Format:	MBZ
	0	Arbitration Enable	
		Format:	Enable
	This field enables or disables context switches due to pre-emption (a new execlist).		



MI_ATOMIC

MI_ATOMIC												
Project:	BDW											
Length Bias:	2											
Description												
<p>MI_ATOMIC is used to carry atomic operation on data in graphics memory. Atomic operations are supported on data granularity of 4B, 8B and 16B. The atomic operation leads to a read-modify-write operation on the data in graphics memory with the option of returning value. The data in graphics memory is modified by doing arithmetic and logical operation with the inline/indirect data provided with the MI_ATOMIC command. Inline/Indirect provided in the command can be one or two operands based on the atomic operation. Ex: Atomic-Compare operation needs two operands while Atomic-Add operation needs single operand and Atomic-increment requires no operand. Refer Vol1i L3 URB [BDW] B-spec for detailed atomic operations supported. Atomic operations can be enabled to return value by setting "Return Data Control" field in the command, return data is stored to CS_GPR registers.</p> <p>CS_GPR4/5 registers are updated with memory Return Data based on the "Data Size". Each GPR register is qword in size and occupies two MMIO registers.</p> <p>Note: Any references to CS_GPR registers in the command should be understood as the CS_GPR registers belonging to the corresponding engines *CS_GPR registers.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Engine Name</th> <th style="text-align: left;">Corresponding GPR Registers</th> </tr> </thead> <tbody> <tr> <td>RCS</td> <td>CS_GPR</td> </tr> <tr> <td>BCS</td> <td>BCS_GPR</td> </tr> <tr> <td>VCS</td> <td>VCS_GPR</td> </tr> <tr> <td>VECS</td> <td>VECS_GPR</td> </tr> </tbody> </table> <p>Indirect Source Operands: Operand1 is sourced from [CS_GPR1, CS_GPR0] Operand2 is sourced from [CS_GPR3, CS_GPR2] Read return Data is stored in [CS_GPR_5, CS_GPR4]</p> <p>When "Data Size" is QWORD or DWORD only CS_GPR4 (Qword) is updated with the qword data returned from memory. When the data size is OCTWORD CS_GPR4/5 are updated with the OCTWORD data returned from memory. CS_GPR4 is loaded with lower qword returned from memory and CS_GPR5 is loaded with upper qword returned from memory.</p>			Engine Name	Corresponding GPR Registers	RCS	CS_GPR	BCS	BCS_GPR	VCS	VCS_GPR	VECS	VECS_GPR
Engine Name	Corresponding GPR Registers											
RCS	CS_GPR											
BCS	BCS_GPR											
VCS	VCS_GPR											
VECS	VECS_GPR											
Programming Notes												
<ul style="list-style-type: none"> When Inline Data mode is not set, Dwords 3..10 must not be included as part of the command. Dword Length field in the header must be programmed accordingly. When Inline Data Mode is set, Dwords3..10 must be included based on the Data Size field of the header. Both Operand-1 and Operand-2 dwords must be programmed based on the Data Size field. Operand-2 must be programmed to 0x0 if the atomic operation doesn't require it. Dword Length field in the header must be programmed accordingly. 												
DWord	Bit	Description										

MI_ATOMIC											
0	31:29	Command Type <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h MI_COMMAND</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MI_COMMAND	Format:	OpCode					
	Default Value:	0h MI_COMMAND									
	Format:	OpCode									
	28:23	MI Command Opcode <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>2Fh MI_ATOMIC</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	2Fh MI_ATOMIC	Format:	OpCode					
	Default Value:	2Fh MI_ATOMIC									
	Format:	OpCode									
	22	Memory Type This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.									
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 25%;">Name</th> <th style="width: 60%;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>Per Process Graphics Address</td> <td></td> </tr> <tr> <td style="text-align: center;">1h</td> <td>Global Graphics Address</td> <td>This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Per Process Graphics Address		1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.
	Value	Name	Description								
	0h	Per Process Graphics Address									
1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.									
21	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Source:</td> <td>BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS	Format:	MBZ						
Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS										
Format:	MBZ										

MI_ATOMIC

21

Post-Sync Operation

Source:	RenderCS
---------	----------

Value	Name	Description
0h	No Post Sync Operation	Command is executed as usual.
1h	Post Sync Operation	<p>MI_ATOMIC command is executed as a pipelined PIPE_CONTROL flush command with Atomics operation as post sync operation. Flush completion only guarantees the workload prior to this command is pushed till Windower unit and completion of any outstanding flushes issued prior to this command.</p> <p>When this bit set following restriction apply to atomic operation:</p> <ul style="list-style-type: none"> Non-Compare atomic operations are supported on data granularity of 4B and 8B. DW3 is the lower dword of the operand and DW4 is the upper dword of the operand for the atomic operation. Compare atomic operations are supported on data granularity of 4B. DW3 is Operand-0 and DW4 is Operand-1 for the atomic operation. Atomic operations to GGTT/PPGTT memory surface are supported. Only Inline data mode for atomic operand is supported, no support for indirect data mode. No support for Return Data Control functionality. No support for atomic operations on data granularity of 16B. No support for compare atomic operations on data granularity of 8B.

Programming Notes

Any desired pipeline flush operation can be achieved by programming PIPE_CONTROL command prior to this command.

When this bit is set Command Streamer sends a flush down the pipe and the atomic operation is saved as post sync operation. Command streamer goes on executing the following commands. Atomic operation saved as post sync operation is executed at some point later on completion of corresponding flush issued.

When this bit is set atomic semaphore signal operation will be out of order with rest of the MI commands programmed in the ring buffer or batch buffer, it will be in order with respect to the post sync operations resulting due to PIPE_CONTROL command.

Workaround

Workaround : "Post Sync Operation" bit must not be set when MI_ATOMIC command is programmed by GPGPU and MEDIA workloads (i.e when PIPELINE_SELECT command is set to GPGPU or MEDIA). This is to WA FFDOP CG issue, this WA need not be implemented when FF_DOP_CG is disabled via "Fixed Function DOP Clock Gate Disable" bit in RC_PSMI_CTRL register.

MI_ATOMIC																
20:19	<p>Data Size This field indicates the size of the operand in dword/qword/octword on which atomic operation will be performed. Data size must match with the Atomic Opcode. Operation Data size could be 4B, 8B or 16B</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>DWORD</td> <td>Operand size used by Atomic Operation is DWORD.</td> </tr> <tr> <td>1h</td> <td>QWORD</td> <td>Operand Size used by Atomic Operation is QWORD.</td> </tr> <tr> <td>2h</td> <td>OCTWORD</td> <td>Operand Size used by Atomic Operation is OCTWORD.</td> </tr> <tr> <td>3h</td> <td>RESERVED</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	0h	DWORD	Operand size used by Atomic Operation is DWORD.	1h	QWORD	Operand Size used by Atomic Operation is QWORD.	2h	OCTWORD	Operand Size used by Atomic Operation is OCTWORD.	3h	RESERVED	
Value	Name	Description														
0h	DWORD	Operand size used by Atomic Operation is DWORD.														
1h	QWORD	Operand Size used by Atomic Operation is QWORD.														
2h	OCTWORD	Operand Size used by Atomic Operation is OCTWORD.														
3h	RESERVED															
18	<p>Inline Data This bit when set indicates the source operands are provided in line within the command. When reset the source operands are in CS_GPR registers.</p> <table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td colspan="2">CS_GPR registers must be programmed with appropriate values before issuing MI_ATOMIC command with this field reset.</td> </tr> </tbody> </table>	Programming Notes		CS_GPR registers must be programmed with appropriate values before issuing MI_ATOMIC command with this field reset.												
Programming Notes																
CS_GPR registers must be programmed with appropriate values before issuing MI_ATOMIC command with this field reset.																
17	<p>CS STALL This bit when set command stream waits for completion of this command before executing the next command.</p> <table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> <th>Source</th> </tr> </thead> <tbody> <tr> <td colspan="2">Render Command Streamer Only: CS will not guarantee atomic operation to be complete upon setting this bit along with Post Sync Operation set. When Post Sync Operation is set, this bit has no significance.</td> <td>RenderCS</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Workaround</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td colspan="2">Workaround : When CS STALL bit is set, Return Data Control must also be set in MI_ATOMIC command.</td> <td>BDW</td> </tr> </tbody> </table>	Programming Notes		Source	Render Command Streamer Only: CS will not guarantee atomic operation to be complete upon setting this bit along with Post Sync Operation set. When Post Sync Operation is set, this bit has no significance.		RenderCS	Workaround		Project	Workaround : When CS STALL bit is set, Return Data Control must also be set in MI_ATOMIC command.		BDW			
Programming Notes		Source														
Render Command Streamer Only: CS will not guarantee atomic operation to be complete upon setting this bit along with Post Sync Operation set. When Post Sync Operation is set, this bit has no significance.		RenderCS														
Workaround		Project														
Workaround : When CS STALL bit is set, Return Data Control must also be set in MI_ATOMIC command.		BDW														
16	<p>Return Data Control</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Source:</td> <td>RenderCS, BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> </table> <table border="1"> <thead> <tr> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>When "Data Size" is QWORD or DWORD only CS_GPR4 (Qword) is updated with the qword data returned from memory. When the data size is OCTWORD CS_GPR4/5 are updated with the OCTWORD data returned from memory. CS_GPR4 is loaded with lower qword returned from memory and CS_GPR5 is loaded with upper qword returned from memory</td> <td>BDW</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Workaround</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td colspan="2">Workaround : When Return Data Control bit is set, CS STALL must also be set in MI_ATOMIC command.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	BDW	Source:	RenderCS, BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS	Description	Project	When "Data Size" is QWORD or DWORD only CS_GPR4 (Qword) is updated with the qword data returned from memory. When the data size is OCTWORD CS_GPR4/5 are updated with the OCTWORD data returned from memory. CS_GPR4 is loaded with lower qword returned from memory and CS_GPR5 is loaded with upper qword returned from memory	BDW	Workaround		Project	Workaround : When Return Data Control bit is set, CS STALL must also be set in MI_ATOMIC command.		BDW	
Project:	BDW															
Source:	RenderCS, BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS															
Description	Project															
When "Data Size" is QWORD or DWORD only CS_GPR4 (Qword) is updated with the qword data returned from memory. When the data size is OCTWORD CS_GPR4/5 are updated with the OCTWORD data returned from memory. CS_GPR4 is loaded with lower qword returned from memory and CS_GPR5 is loaded with upper qword returned from memory	BDW															
Workaround		Project														
Workaround : When Return Data Control bit is set, CS STALL must also be set in MI_ATOMIC command.		BDW														

MI_ATOMIC												
	15:8	<p>ATOMIC OPCODE This field selects the kind of atomic operation to be performed. Refer Vol1i L3 URB [BDW] B-spec for atomic opcode corresponding to an atomic operation.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>Atomic Opcode must not be set to 0x00 (no-atomic).</td> <td>BDW</td> </tr> </tbody> </table>	Programming Notes	Project	Atomic Opcode must not be set to 0x00 (no-atomic).	BDW						
	Programming Notes	Project										
	Atomic Opcode must not be set to 0x00 (no-atomic).	BDW										
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2. Excludes DWord (0,1).</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Exists If</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>[Default]</td> <td>((Inline Data)==0)</td> </tr> <tr> <td>9h</td> <td></td> <td>((Inline Data)==1)</td> </tr> </tbody> </table>	Format:	=n	Value	Name	Exists If	1h	[Default]	((Inline Data)==0)	9h	
Format:	=n											
Value	Name	Exists If										
1h	[Default]	((Inline Data)==0)										
9h		((Inline Data)==1)										
1	31:2	<p>Memory Address</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>This field contains the graphics memory address of the data on which atomic operation has to be performed. Atomic operation can be performed on data granularity of 4B, 8B or 16B and hence the Address has to be correspondingly aligned to 4B,8B or 16B respectively.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>Memory Address must be qword aligned for all dword atomic operations. Upper Dword of the memory location should be initialized to 0x0.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	All	Format:	GraphicsAddress[31:2]	Programming Notes	Project	Memory Address must be qword aligned for all dword atomic operations. Upper Dword of the memory location should be initialized to 0x0.	BDW		
Project:	All											
Format:	GraphicsAddress[31:2]											
Programming Notes	Project											
Memory Address must be qword aligned for all dword atomic operations. Upper Dword of the memory location should be initialized to 0x0.	BDW											
	1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ											
2	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
	Format:	MBZ										
15:0	<p>Memory Address High This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>											
3	31:0	<p>Operand1 Data Dword 0</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>Dword0 of Operand1 when Inline Data mode is set.</p>	Format:	U32								
		Format:	U32									
<p>Operand2 Data Dword 0</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>Dword0 of Operand2 when Inline Data mode is set.</p>	Format:	U32										
Format:	U32											
5	31:0	<p>Operand1 Data Dword 1</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>Dword1 of Operand1 when Inline Data mode is set.</p>	Format:	U32								
		Format:	U32									

MI_ATOMIC		
6	31:0	Operand2 Data Dword 1
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Dword1 of Operand2 when Inline Data mode is set.</p>
Format:	U32	
7	31:0	Operand1 Data Dword 2
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Dword2 of Operand1 when Inline Data mode is set.</p>
Format:	U32	
8	31:0	Operand2 Data Dword 2
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Dword2 of Operand2 when Inline Data mode is set.</p>
Format:	U32	
9	31:0	Operand1 Data Dword 3
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Dword3 of Operand1 when Inline Data mode is set.</p>
Format:	U32	
10	31:0	Operand2 Data Dword 3
		<table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Dword3 of Operand2 when Inline Data mode is set.</p>
Format:	U32	



MI_BATCH_BUFFER_END

MI_BATCH_BUFFER_END		
Project: BDW		
Source: VideoEnhancementCS		
Length Bias: 1		
The MI_BATCH_BUFFER_END command is used to terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 0Ah MI_BATCH+_BUFFER_END
		Format: OpCode
	22:0	Reserved
		Project: All
		Format: MBZ

MI_BATCH_BUFFER_END

MI_BATCH_BUFFER_END		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	1	
<p>The MI_BATCH_BUFFER_END command is used to terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode
		Default Value: 0Ah MI_BATCH_BUFFER_END
	22:0	Reserved
		Project: All
Format: MBZ		



MI_BATCH_BUFFER_END

MI_BATCH_BUFFER_END		
Project:	BDW	
Source:	RenderCS	
Length Bias:	1	
The MI_BATCH_BUFFER_END command is used to terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 0Ah MI_BATCH_BUFFER_END
		Format: OpCode
	22:0	Reserved
		Format: MBZ

MI_BATCH_BUFFER_END

MI_BATCH_BUFFER_END			
Project:	BDW		
Source:	VideoCS		
Length Bias:	1		
The MI_BATCH_BUFFER_END command is used to terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	0Ah MI_BATCH+_BUFFER_END
		Format:	OpCode
	22:0	Reserved	
		Format:	MBZ



MI_BATCH_BUFFER_START

MI_BATCH_BUFFER_START											
Project:	BDW										
Source:	RenderCS										
Length Bias:	2										
<p>The MI_BATCH_BUFFER_START command is used to initiate the execution of commands stored in a <i>batch</i> buffer. For restrictions on the location of batch buffers, see Batch Buffers in the Device Programming Interface chapter of <i>MI Functions</i>. The batch buffer can be specified as privileged or non-privileged, determining the operations considered valid when initiated from within the buffer and any attached (chained) batch buffers. See Batch Buffer Protection in the Device Programming Interface chapter of <i>MI Functions</i>.</p>											
Programming Notes											
<ul style="list-style-type: none"> A batch buffer initiated with this command must end either with a MI_BATCH_BUFFER_END command or by chaining to another batch buffer with an MI_BATCH_BUFFER_START command. It is essential that the address location beyond the current page be populated inside the GTT. HW performs over-fetch of the command addresses and any over-fetch requires a valid TLB entry. A single extra page beyond the batch buffer is sufficient. 											
DWord	Bit	Description									
0	31:29	Command Type									
		Default Value:	0h MI_COMMAND								
		Format:	OpCode								
	28:23	MI Command Opcode									
		Default Value:	31h MI_BATCH_BUFFER_START								
		Format:	OpCode								
22		2nd Level Batch Buffer									
		The command streamer contains three storage elements; one for the ring head address, one for the batch head address, and one for the 2nd level batch head address. When performing batch buffer chaining, hardware simply updates the head pointer of the 1st level batch address storage. There is no stack in hardware. When this bit is set, hardware uses the 2nd level batch head address storage element. Upon MI_BATCH_BUFFER_END, it will automatically return to the 1st (traditional) level batch buffer address. this allows hardware to mimic a simple 3-level stack.									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>1st level batch</td> <td>Place the batch buffer address in the 1st (traditional) level batch address storage element.</td> </tr> <tr> <td>1h</td> <td>2nd level batch</td> <td>Place the batch buffer address in the 2nd-level batch address storage element.</td> </tr> </tbody> </table>	Value	Name	Description	0h	1st level batch	Place the batch buffer address in the 1st (traditional) level batch address storage element.	1h	2nd level batch	Place the batch buffer address in the 2nd-level batch address storage element.
		Value	Name	Description							
		0h	1st level batch	Place the batch buffer address in the 1st (traditional) level batch address storage element.							
		1h	2nd level batch	Place the batch buffer address in the 2nd-level batch address storage element.							
Programming Notes											
Within a second level batch buffer there can't be any chained batch buffers.											
MI_BATCH_BUFFER_START command is not allowed inside a second level batch buffer.											

MI_BATCH_BUFFER_START															
21:17	Reserved	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">MBZ</td></tr></table>		MBZ											
	MBZ														
16	Add Offset Enable	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">Enable</td></tr></table> If this bit is set then the value stored in the BB_OFFSET MMIO register will be added to the Batch Buffer Start Address and the summation will be used as the address to fetch from memory. Specific to the render command stream only.		Enable											
	Enable														
15	Predication Enable	This bit is used to enable predication of this command. If this bit is set and Bit 0 of the Predicate Result-1 register is clear, this command is ignored. Otherwise the command is performed normally. Specific to the Render command stream only.													
14:12	Reserved	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">MBZ</td></tr></table>		MBZ											
	MBZ														
11	Reserved	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">MBZ</td></tr></table>		MBZ											
	MBZ														
10	Resource Streamer Enable	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">Enable</td></tr></table> When this bit is set, the Resource Streamer will execute the batch buffer. When this bit is clear the Resource Streamer will not execute the batch buffer. Specific to the Render command stream only.		Enable											
	Enable														
9	Reserved	Format: <table border="1" style="width: 100%;"><tr><td style="width: 80%;"></td><td style="width: 20%; text-align: center;">MBZ</td></tr></table>		MBZ											
	MBZ														
8	Address Space Indicator	<p>Batch buffers accessed via PPGTT are considered as non-privileged. Certain operations (e.g., MI_STORE_DATA_IMM commands to GGTT memory) are prohibited within non-privileged buffers. More details mentioned in User Mode Privileged command section. When MI_BATCH_BUFFER_START command is executed from within a batch buffer (i.e., is a "chained" or "second level" batch buffer command), the current active batch buffer's "Address Space Indicator" and this field determine the "Address Space Indicator" of the next buffer in the chain.</p> <ul style="list-style-type: none"> Chained or Second level batch buffer can be in GGTT or PPGTT if the parent batch buffer is in GGTT. Chained or Second level batch buffer can only be in PPGTT if the parent batch buffer is in PPGTT. This is enforced by Hardware. <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td style="text-align: center;">GGTT</td> <td>This batch buffer is located in GGTT memory and is privileged.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td style="text-align: center;">PPGTT</td> <td>This batch buffer is located in PPGTT memory and is Non-Privileged.</td> </tr> </tbody> </table> <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="width: 80%;">This field must be '0' unless the Per-Process GTT Enable is '1'</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.	Programming Notes	Project	This field must be '0' unless the Per-Process GTT Enable is '1'	
Value	Name	Description													
0h	GGTT	This batch buffer is located in GGTT memory and is privileged.													
1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.													
Programming Notes	Project														
This field must be '0' unless the Per-Process GTT Enable is '1'															

MI_BATCH_BUFFER_START						
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td style="text-align: right;">1h</td> </tr> <tr> <td>Format:</td> <td style="text-align: right;">=n</td> </tr> </table> <p>Total - Bias. Excludes DWord (0,1).</p>	Default Value:	1h	Format:	=n
Default Value:	1h					
Format:	=n					
1	31:2	<p>Batch Buffer Start Address</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]BatchBuffer</td> </tr> </table> <p>This field specifies Bits 31:2 of the starting address of the batch buffer.</p>	Format:	GraphicsAddress[31:2]BatchBuffer		
	Format:	GraphicsAddress[31:2]BatchBuffer				
1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
2	31:16	<p>Reserved</p>				
	15:0	<p>Batch Buffer Start Address High</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]BatchBuffer</td> </tr> </table> <p>This field specifies the 4GB aligned base address of Gfx 4GB virtual address space within the hosts 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]BatchBuffer		
Format:	GraphicsAddress[47:32]BatchBuffer					

MI_BATCH_BUFFER_START

MI_BATCH_BUFFER_START				
Project:	BDW			
Source:	VideoEnhancementCS			
Length Bias:	2			
<p>The MI_BATCH_BUFFER_START command is used to initiate the execution of commands stored in a <i>batch buffer</i>. For restrictions on the location of batch buffers, see Batch Buffers in the Device Programming Interface chapter of <i>MI Functions</i>.</p> <p>The batch buffer can be specified as secure or non-secure, determining the operations considered valid when initiated from within the buffer and any attached (chained) batch buffers. See Batch Buffer Protection in the Device Programming Interface chapter of <i>MI Functions</i>.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	31h MI_BATCH_BUFFER_START	
		Format:	OpCode	
	22	2nd Level Batch Buffer		
		<p>The command streamer contains 3 storage elements; 1 for the ring head address, 1 for the batch head address, and 1 for the 2nd level batch head address. When performing batch buffer chaining, hardware simply updates the head pointer of the 1st level batch address storage. There is no stack in hardware.</p> <p>When this bit is set, hardware uses the 2nd level batch head address storage element. Upon MI_BATCH_BUFFER_END, it will automatically return to the 1st (traditional) level batch buffer address. this allows hardware to mimic a simple 3 level stack.</p>		
		Value	Name	Description
		0h	1st level batch	Place the batch buffer address in the 1st (traditional) level batch address storage element
1h		2nd level batch	Place the batch buffer address in the 2nd level batch address storage element	
21:13	Reserved			
	Format:	MBZ		
12	Reserved			
11:9	Reserved			
	Format:	MBZ		

MI_BATCH_BUFFER_START													
	8	<p>Address Space Indicator</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>Batch buffers accessed via PPGTT are considered as non-privileged. Certain operations (e.g., MI_STORE_DATA_IMM commands to GGTT memory) are prohibited within non-privileged buffers. More details mentioned in User Mode Privileged command section. When MI_BATCH_BUFFER_START command is executed from within a batch buffer (i.e., is a "chained" or "second level" batch buffer command), the current active batch buffer's "Address Space Indicator" and this field determine the "Address Space Indicator" of the next buffer in the chain.</p> <ul style="list-style-type: none"> Chained or Second level batch buffer can be in GGTT or PPGTT if the parent batch buffer is in GGTT. Chained or Second level batch buffer can only be in PPGTT if the parent batch buffer is in PPGTT. This is enforced by Hardware. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>GGTT</td> <td>This batch buffer is located in GGTT memory and is privileged.</td> </tr> <tr> <td>1h</td> <td>PPGTT</td> <td>This batch buffer is located in PPGTT memory and is Non-Privileged.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This field must be '0' unless the Per-Process GTT Enable is '1'</p>	Project:	BDW	Value	Name	Description	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.
	Project:	BDW											
Value	Name	Description											
0h	GGTT	This batch buffer is located in GGTT memory and is privileged.											
1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.											
	7:0	<p>DWord Length (Excludes D-Word 0,1) = 0</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	Project	1h	Excludes DWord (0,1) [Default]	BDW					
Value	Name	Project											
1h	Excludes DWord (0,1) [Default]	BDW											
1	31:2	<p>Batch Buffer Start Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <ul style="list-style-type: none"> A batch buffer initiated with this command must end either with a MI_BATCH_BUFFER_END command or by chaining to another batch buffer with an MI_BATCH_BUFFER_START command. The selection of PPGTT vs. GGTT for the batch buffer is determined by the Buffer Security Indicator (bit 8). 	Format:	GraphicsAddress[31:2]									
	Format:	GraphicsAddress[31:2]											
1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
2	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW												
Format:	MBZ												
2		<p>Project: BDW</p>											

MI_BATCH_BUFFER_START					
15:0	<p>Batch Buffer Start Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]BatchBuffer</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]BatchBuffer
Project:	BDW				
Format:	GraphicsAddress[47:32]BatchBuffer				



MI_BATCH_BUFFER_START

MI_BATCH_BUFFER_START	
Project:	BDW
Source:	BlitterCS
Length Bias:	2

The MI_BATCH_BUFFER_START command is used to initiate the execution of commands stored in a batch buffer. For restrictions on the location of batch buffers, see Batch Buffers in the Device Programming Interface chapter of MI Functions. The batch buffer can be specified as secure or non-secure, determining the operations considered valid when initiated from within the buffer and any attached (chained) batch buffers. See Batch Buffer Protection in the Device Programming Interface chapter of MI Functions.

Programming Notes

- Batch buffers referenced with physical addresses must not extend beyond the end of the starting physical page (can't span physical pages). However, a batch buffer initiated using a physical address can chain to another buffer in another physical page.
- A batch buffer initiated with this command must end either with a MI_BATCH_BUFFER_END command or by chaining to another batch buffer with an MI_BATCH_BUFFER_START command.

DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	31h MI_BATCH_BUFFER_START	
		Format:	OpCode	
	22	2nd Level Batch Buffer		
		Project:	BDW	
		The command streamer contains 3 storage elements; 1 for the ring head address, 1 for the batch head address, and 1 for the 2nd level batch head address. When performing batch buffer chaining, hardware simply updates the head pointer of the 1st level batch address storage. There is no stack in hardware.		
		When this bit is set, hardware uses the 2nd level batch head address storage element. Upon MI_BATCH_BUFFER_END, it will automatically return to the 1st (traditional) level batch buffer address. this allows hardware to mimic a simple 3 level stack.		
Value		Name	Description	Project
0h		1st level batch	Place the batch buffer address in the 1st (traditional) level batch address storage element	BDW
1h		2nd level batch	Place the batch buffer address in the 2nd level batch address storage element	BDW
21:9	Reserved			
	Format:	MBZ		

MI_BATCH_BUFFER_START													
	8	<p>Address Space Indicator</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>Batch buffers accessed via PPGTT are considered as non-privileged. Certain operations (e.g., MI_STORE_DATA_IMM commands to GGTT memory) are prohibited within non-privileged buffers. More details mentioned in User Mode Privileged command section. When MI_BATCH_BUFFER_START command is executed from within a batch buffer (i.e., is a "chained" or "second level" batch buffer command), the current active batch buffer's "Address Space Indicator" and this field determine the "Address Space Indicator" of the next buffer in the chain.</p> <ul style="list-style-type: none"> • Chained or Second level batch buffer can be in GGTT or PPGTT if the parent batch buffer is in GGTT. • Chained or Second level batch buffer can only be in PPGTT if the parent batch buffer is in PPGTT. This is enforced by Hardware. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>GGTT</td> <td>This batch buffer is located in GGTT memory and is privileged.</td> </tr> <tr> <td>1h</td> <td>PPGTT</td> <td>This batch buffer is located in PPGTT memory and is Non-Privileged.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This field must be '0' unless the Per-Process GTT Enable is '1'</p>	Project:	BDW	Value	Name	Description	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.
	Project:	BDW											
	Value	Name	Description										
	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.										
	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.										
7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">=n</td> </tr> </table> <p>Total - Bias</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Format:	=n	Value	Name	Project	1h	Excludes DWord (0,1) [Default]	BDW				
Format:	=n												
Value	Name	Project											
1h	Excludes DWord (0,1) [Default]	BDW											
1	31:2	<p>Batch Buffer Start Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">GraphicsAddress[31:2]BatchBuffer</td> </tr> </table> <p>This field specifies Bits 31:2 of the starting address of the batch buffer.</p>	Format:	GraphicsAddress[31:2]BatchBuffer									
	Format:	GraphicsAddress[31:2]BatchBuffer											
1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ										
Format:	MBZ												
2 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
	Project:	BDW											
	Format:	MBZ											
15:0	<p>Batch Buffer Start Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td style="width: 70%;">BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]BatchBuffer</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]BatchBuffer								
Project:	BDW												
Format:	GraphicsAddress[47:32]BatchBuffer												

MI_BATCH_BUFFER_START

MI_BATCH_BUFFER_START			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MI_BATCH_BUFFER_START command is used to initiate the execution of commands stored in a batch buffer. For restrictions on the location of batch buffers, see Batch Buffers in the Device Programming Interface chapter of MI Functions. The batch buffer can be specified as secure or non-secure, determining the operations considered valid when initiated from within the buffer and any attached (chained) batch buffers. See Batch Buffer Protection in the Device Programming Interface chapter of MI Functions.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	31h MI_BATCH_BUFFER_START
		Format:	OpCode
	22	2nd Level Batch Buffer	
		Project:	BDW
		<p>The command streamer contains 3 storage elements; 1 for the ring head address, 1 for the batch head address, and 1 for the 2nd level batch head address. When performing batch buffer chaining, hardware simply updates the head pointer of the 1st level batch address storage. There is no stack in hardware. When this bit is set, hardware uses the 2nd level batch head address storage element. Upon MI_BATCH_BUFFER_END, it will automatically return to the 1st (traditional) level batch buffer address. this allows hardware to mimic a simple 3 level stack.</p>	
		Value	Name
0h		1st level batch	Place the batch buffer address in the 1st (traditional) level batch address storage element
1h		2nd level batch	Place the batch buffer address in the 2nd level batch address storage element
Programming Notes			
<ul style="list-style-type: none"> 2nd level batch buffer chaining is not supported. 			
21:10	Reserved		
	Format:	MBZ	
9	Reserved		

MI_BATCH_BUFFER_START													
	8	<p>Address Space Indicator</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>Batch buffers accessed via PPGTT are considered as non-privileged. Certain operations (e.g., MI_STORE_DATA_IMM commands to GGTT memory) are prohibited within non-privileged buffers. More details mentioned in User Mode Privileged command section. When MI_BATCH_BUFFER_START command is executed from within a batch buffer (i.e., is a "chained" or "second level" batch buffer command), the current active batch buffer's "Address Space Indicator" and this field determine the "Address Space Indicator" of the next buffer in the chain.</p> <ul style="list-style-type: none"> • Chained or Second level batch buffer can be in GGTT or PPGTT if the parent batch buffer is in GGTT. • Chained or Second level batch buffer can only be in PPGTT if the parent batch buffer is in PPGTT. This is enforced by Hardware. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>GGTT</td> <td>This batch buffer is located in GGTT memory and is privileged.</td> </tr> <tr> <td>1h</td> <td>PPGTT</td> <td>This batch buffer is located in PPGTT memory and is Non-Privileged.</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>This field must be '0' unless the Per-Process GTT Enable is '1'.</p>	Project:	BDW	Value	Name	Description	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.
	Project:	BDW											
	Value	Name	Description										
	0h	GGTT	This batch buffer is located in GGTT memory and is privileged.										
	1h	PPGTT	This batch buffer is located in PPGTT memory and is Non-Privileged.										
7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td style="width: 60%;">=n Total Length - 2</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	Project	1h	Excludes DWord (0,1) [Default]	BDW				
Format:	=n Total Length - 2												
Value	Name	Project											
1h	Excludes DWord (0,1) [Default]	BDW											
1	31:2	<p>Batch Buffer Start Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td style="width: 60%;">GraphicsAddress[31:2]</td> </tr> </table> <p style="text-align: center;">Programming Notes</p> <ul style="list-style-type: none"> • A batch buffer initiated with this command must end either with a MI_BATCH_BUFFER_END command or by chaining to another batch buffer with an MI_BATCH_BUFFER_START command. • The selection of PPGTT vs. GGTT for the batch buffer is determined by the Buffer Security Indicator (bit8). 	Format:	GraphicsAddress[31:2]									
	Format:	GraphicsAddress[31:2]											
	1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ									
Format:	MBZ												
2	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW												
Format:	MBZ												
2	31:16	<p>Project: BDW</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ							
Project:	BDW												
Format:	MBZ												



MI_BATCH_BUFFER_START									
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center;">15:0</td> <td>Batch Buffer Start Address High</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]BatchBuffer</td> </tr> <tr> <td colspan="2"> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p> </td> </tr> </table>	15:0	Batch Buffer Start Address High	Project:	BDW	Format:	GraphicsAddress[47:32]BatchBuffer	<p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	
15:0	Batch Buffer Start Address High								
Project:	BDW								
Format:	GraphicsAddress[47:32]BatchBuffer								
<p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>									

MI_CLFLUSH

		MI_CLFLUSH		
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
Flushes out the page given in the command out to system memory. This command is specific to the render engine and is not privileged.				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	27h Store DW MI_CLFLUSH	
		Format:	OpCode	
	22	Use Global GTT		
		Value	Name	Description
		0h	Per Process Graphics Address	
		1h	Global Graphics Address	This command will use the global GTT to translate the Address.
21:10	Reserved			
	Format:	MBZ		
9:0	DWord Length			
	Format:	n Total Length - 2		
	Value	Name	Description	
	1h	[Default]	Excludes DWord (0,1)	
	Programming Notes			Project
The value of this field must not exceed a value 3Fh when programmed in a batch buffer with resource streamer enabled.			BDW	
1	31:12	Page Base Address		
		Format:	GraphicsAddress[31:12] 4KB aligned Page Address which software requires hardware to flush to DRAM.	
	11:6	Starting Cacheline Offset		
		Format:	U6 Zero based starting cacheline offset in to the Page Base Address	
	5:0	Reserved		
Format:		MBZ		



MI_CLFLUSH				
2	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Page Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
3..n	31:0	<p>DW Representing a Half Cache Line</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table> <p>The information given to hardware is the DW itself, not the contents. Hardware uses the DW count of the command to determine the offset from the base to flush out. The offset is ½ cache line (8 DW = 1HW) granular so for a full page, the command will need 4096 bytes / 4 bytes per DW / 8 DW per HW = 128 DW.</p>	Format:	MBZ
		Format:	MBZ	
		<p>Programming Notes</p>		
		<p>Always even number of "DW Representing 1/2 cacheline" terms must be programmed.</p>		

MI_CONDITIONAL_BATCH_BUFFER_END

MI_CONDITIONAL_BATCH_BUFFER_END			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>The MI_CONDITIONAL_BATCH_BUFFER_END command is used to conditionally terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command. Termination of second level batch buffer due to this command will also terminate the parent/first level batch buffer.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	36h MI_CONDITIONAL_BATCH_BUFFER_END
	22	Format:	OpCode
			Use Global GTT
		Default Value:	0h
	21	Format:	Boolean
			If set, this command uses the global GTT to translate the Compare Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT is used to translate the Compare Address.
Compare Semaphore			
20	Format:	0h	
		Boolean	
19:8	Format:	If set, the value from the Compare Data Dword is compared to the value from the Compare Address in memory. If the value at Compare Address is greater than the Compare Data Dword, execution of the current command buffer should continue. If clear, the parser will continue to the next command and not exit the batch buffer.	
		Reserved	
	7:0	Format:	MBZ
DWord Length			
Default Value:		2h Excludes DWord (0,1)	
1	31:0	Format:	=n Total Length - 2. Excludes DWord (0,1).
		Compare Data Dword	
		Data DWord to compare to memory. The Data DWord is supplied by software to control execution of the command buffer. If the compare is enabled and the data at Semaphore Address is greater than this DWord, the execution of the command buffer should continue.	

MI_CONDITIONAL_BATCH_BUFFER_END					
2	31:3	<p>Compare Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:3]</td> </tr> </table> <p>Qword address to fetch Data Dword(DW0) from memory. HW will compare the Data Dword(DW0) with Compare Data Dword</p>	Format:	GraphicsAddress[31:3]	
	Format:	GraphicsAddress[31:3]			
2:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				
3	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
	Format:	MBZ			
15:0	<p>Compare Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4 GB-aligned base address of GFX 4 GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]		
Format:	GraphicsAddress[47:32]				

MI_CONDITIONAL_BATCH_BUFFER_END

MI_CONDITIONAL_BATCH_BUFFER_END							
Project:	BDW						
Source:	VideoEnhancementCS						
Length Bias:	2						
<p>The MI_CONDITIONAL_BATCH_BUFFER_END command is used to conditionally terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command. Termination of second level batch buffer due to this command will also terminate the parent/first level batch buffer.</p>							
Programming Notes							
This command is only valid with a 1st level batch buffer (bit 22 in MI_BATCH_BUFFER_START is set to '0')							
DWord	Bit	Description					
0	31:29	Command Type					
		Default Value:	0h MI_COMMAND				
		Format:	OpCode				
	28:23	MI Command Opcode					
		Default Value:	36h MI_CONDITIONAL_BATCH_BUFFER_END				
		Format:	OpCode				
	22	Use Global GTT					
		Default Value:	0h				
		Format:	Boolean				
	<p>If set, this command will use the global GTT to translate the Compare Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT will be used to translate the Compare Address.</p>						
21	Compare Semaphore						
	Default Value:	0h					
	Format:	Boolean					
<p>If set, the value from the Compare Data Dword is compared to the value from the Compare Address in memory. If the value at Compare Address is greater than the Compare Data Dword, execution of current command buffer should continue. If clear, the parser will continue to the next command and not exit the batch buffer.</p>							
20	Reserved						
19:8	Reserved						
	Format:	MBZ					
7:0	DWord Length						
	Format:	=n Total Length - 2					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>		Value	Name	Project	1h	Excludes DWord (0,1) [Default]
Value	Name	Project					
1h	Excludes DWord (0,1) [Default]	BDW					



MI_CONDITIONAL_BATCH_BUFFER_END				
1	31:0	<p>Compare Data Dword Data dword to compare memory. The Data dword is supplied by software to control execution of the command buffer. If the compare is enabled and the data at Semaphore Address is greater than this dword, the execution of the command buffer should continue.</p>		
2	31:3	<p>Compare Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:3]</td> </tr> </table> <p>Qword address to fetch Data Dword(DW0) from memory. HW will compare the Data Dword(DW0) with Compare Data Dword</p>	Format:	GraphicsAddress[31:3]
	Format:	GraphicsAddress[31:3]		
2:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
3 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW
		Project:	BDW	
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
	Format:	MBZ		
15:0	<p>Compare Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW	
	Project:	BDW		
<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space</p>	Format:	GraphicsAddress[47:32]		
Format:	GraphicsAddress[47:32]			

MI_CONDITIONAL_BATCH_BUFFER_END

DWord		Bit	Description
Project:		BDW	
Source:		VideoCS	
Length Bias:		2	
<p>The MI_CONDITIONAL_BATCH_BUFFER_END command is used to conditionally terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command. Termination of second level batch buffer due to this command will also terminate the parent/first level batch buffer.</p>			
Programming Notes			
This command is only valid with a 1st level batch buffer (bit 22 in MI_BATCH_BUFFER_START is set to 0).			
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	36h MI_CONDITIONAL_BATCH_BUFFER_END
		Format:	OpCode
	22	Use Global GTT	
		Default Value:	0h DefaultVaueDesc
		Format:	Boolean
		Format:	U1 FormatDesc
Description		Project	
If set, this command will use the global GTT to translate the Compare Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT will be used to translate the Compare Address.		BDW	
21	Compare Semaphore		
	Default Value:	0h DefaultVaueDesc	
	Format:	Boolean	
If set, the value from the Compare Data Dword is compared to the value from the Compare Address in memory. If the value at Compare Address is greater than the Compare Data Dword, execution of current command buffer should continue.If clear, no comparison takes place.			
20	Reserved		
19:8	Reserved		
	Format:	MBZ	

MI_CONDITIONAL_BATCH_BUFFER_END										
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	Project	1h	Excludes DWord (0,1) [Default]	BDW
Format:	=n Total Length - 2									
Value	Name	Project								
1h	Excludes DWord (0,1) [Default]	BDW								
1	31:0	<p>Compare Data Dword</p> <p>Data dword to compare memory. The Data dword is supplied by software to control execution of the command buffer. If the compare is enabled and the data at Semaphore Address is greater than this dword, the execution of the command buffer should continue.</p>								
2	31:3	<p>Compare Address</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GraphicsAddress[31:3]</td> </tr> </table> <p>Qword address to fetch compare Mask (DW0) and Data Dword(DW1) from memory. HW will do AND operation on Mask(DW0) with Data Dword(DW1) and then compare the result against Semaphore Data Dword</p>	Format:	GraphicsAddress[31:3]						
Format:	GraphicsAddress[31:3]									
	2:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									
3 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ				
		Project:	BDW							
	Format:	MBZ								
	<p>Compare Address High</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space</p>	Project:	BDW	Format:	GraphicsAddress[47:32]					
Project:	BDW									
Format:	GraphicsAddress[47:32]									
15:0										

MI_CONDITIONAL_BATCH_BUFFER_END

MI_CONDITIONAL_BATCH_BUFFER_END			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The MI_CONDITIONAL_BATCH_BUFFER_END command is used to conditionally terminate the execution of commands stored in a batch buffer initiated using a MI_BATCH_BUFFER_START command. Termination of second level batch buffer due to this command will also terminate the parent/first level batch buffer.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	36h MI_CONDITIONAL_BATCH_BUFFER_END
		Format:	OpCode
	22	Use Global GTT	
		Default Value:	0h
	<p>If set, this command will use the global GTT to translate the Compare Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT will be used to translate the Compare Address.</p>		
	21	Compare Semaphore	
Default Value:		0h	
<p>If set, the value from the Compare Data Dword is compared to the value from the Compare Address in memory. If the value at Compare Address is greater than the Compare Data Dword, execution of current command buffer should continue. If clear, the parser will continue to the next command and not exit the batch buffer.</p>			
20	Reserved		
19:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Format:	=n Total Length - 2. Excludes DWord (0,1).	
	Value	Name	Project
	1h	[Default]	BDW
1	31:0	Compare Data Dword Data dword to compare memory. The Data dword is supplied by software to control execution of the command buffer. If the compare is enabled and the data at Compare Address is greater than this dword, the execution of the command buffer should continue.	



MI_CONDITIONAL_BATCH_BUFFER_END						
2	31:3	<p>Compare Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:3]</td> </tr> </table> <p>Qword address to fetch Data Dword(DW0) from memory. HW will compare the Data Dword(DW0) with Compare Data Dword</p>	Format:	GraphicsAddress[31:3]		
	Format:	GraphicsAddress[31:3]				
2:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ			
Format:	MBZ					
3 Project: BDW	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW		
	Project:	BDW				
	15:0	<p>Compare Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of Gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]
	Project:	BDW				
Format:	GraphicsAddress[47:32]					

MI_COPY_MEM_MEM

MI_COPY_MEM_MEM				
Project:	BDW			
Source:	BlitterCS			
Length Bias:	2			
<p>The MI_COPY_MEM_MEM command reads a DWord from memory and stores the value of that DWord to back to memory. The source and destination addresses are specified in the command. The command temporarily halts command execution.</p>				
Programming Notes				
<p>This command should not be used within a "non_privilege"batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or privilege batch buffers to access global virtual space.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	2Eh MI_MEM_TO_MEM	
		Format:	OpCode	
	22	Use Global GTT Source		
		It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.		
		Value	Name	Description
		0h	Per Process Graphics Address	
1h		Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	
21	Use Global GTT Destination			
	It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.			
	Value	Name	Description	
	0h	Per Process Graphics Address		
	1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	

MI_COPY_MEM_MEM						
	20:8	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ		
	Format:	MBZ				
	7:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>3</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	3	Format:	=n Total Length - 2
Default Value:	3					
Format:	=n Total Length - 2					
1	<p>31:2</p> <p>Destination Memory Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value fetched specified in the DWord address above will be written. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]	
Project:	All					
Format:	GraphicsAddress[31:2]					
	1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					
2	31:16	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
Format:	MBZ					
	15:0	<p>Destination Memory Address High</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[47:32]
Project:	All					
Format:	GraphicsAddress[47:32]					
3	31:2	<p>Source Memory Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]
	Project:	All				
Format:	GraphicsAddress[31:2]					
	1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					

MI_COPY_MEM_MEM			
4	31:16	Reserved	
		Project: All	
		Format: MBZ	
	15:0	Source Memory Address High	
		Project: All	
		Format: GraphicsAddress[47:32]	
Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register			



MI_COPY_MEM_MEM

MI_COPY_MEM_MEM		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>The MI_COPY_MEM_MEM command reads a DWord from memory and stores the value of that DWord to back to memory. The source and destination addresses are specified in the command. The command temporarily halts command execution.</p>		
Programming Notes		
<p>This command should not be used within a "non_privilege"batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or privilege batch buffers to access global virtual space.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 2Eh MI_MEM_TO_MEM
		Format: OpCode
22	Use Global GTT Source	
	It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.	
	Value	Name
	0h	Per Process Graphics Address
1h	Global Graphics Address	It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.

MI_COPY_MEM_MEM											
	21	<p>Use Global GTT Destination</p> <p>This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit <i>must</i> be '1' if the Per Process GTT Enable bit is clear. This bit will determine write to memory uses Global or Per Process GTT.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Per Process Graphics Address</td> <td></td> </tr> <tr> <td>1h</td> <td>Global Graphics Address</td> <td>This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Per Process Graphics Address		1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.
	Value	Name	Description								
	0h	Per Process Graphics Address									
	1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.								
20:8	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
7:0	<p>Dword Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>3</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	3	Format:	=n Total Length - 2						
Default Value:	3										
Format:	=n Total Length - 2										
1..2	63:2	<p>Destination Memory Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value fetched specified in the DWord address above will be written. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[63:2] for a DWord register GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	All	Format:	GraphicsAddress[63:2]					
	Project:	All									
	Format:	GraphicsAddress[63:2]									
1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All										
Format:	MBZ										
3..4	63:2	<p>Source Memory Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[63:2] for a DWord register GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	All	Format:	GraphicsAddress[63:2]					
	Project:	All									
	Format:	GraphicsAddress[63:2]									
1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All										
Format:	MBZ										

MI_COPY_MEM_MEM

MI_COPY_MEM_MEM				
Project:	BDW			
Source:	VideoCS			
Length Bias:	2			
<p>The MI_COPY_MEM_MEM command reads a DWord from memory and stores the value of that DWord to back to memory. The source and destination addresses are specified in the command. The command temporarily halts command execution.</p>				
Programming Notes				
<p>This command should not be used within a "non_privilege"batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or privilege batch buffers to access global virtual space.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	2Eh MI_MEM_TO_MEM	
		Format:	OpCode	
	22	Use Global GTT Source		
		It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.		
		Value	Name	Description
		0h	Per Process Graphics Address	
1h		Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	
21	Use Global GTT Destination			
	It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.			
	Value	Name	Description	
	0h	Per Process Graphics Address		
	1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	

MI_COPY_MEM_MEM						
	20:8	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ		
	Format:	MBZ				
7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>3</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	3	Format:	=n Total Length - 2	
Default Value:	3					
Format:	=n Total Length - 2					
1	31:2	Destination Memory Address <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value fetched specified in the DWord address above will be written. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]
	Project:	All				
Format:	GraphicsAddress[31:2]					
1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
2	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
Format:	MBZ					
15:0	Destination Memory Address High <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[47:32]	
Project:	All					
Format:	GraphicsAddress[47:32]					
3	31:2	Source Memory Address <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]
	Project:	All				
Format:	GraphicsAddress[31:2]					
1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ	
Project:	All					
Format:	MBZ					
4	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
Format:	MBZ					



MI_COPY_MEM_MEM					
15:0	<p>Source Memory Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[47:32]
Project:	All				
Format:	GraphicsAddress[47:32]				

MI_COPY_MEM_MEM

MI_COPY_MEM_MEM				
Project:	BDW			
Source:	VideoEnhancementCS			
Length Bias:	2			
<p>The MI_COPY_MEM_MEM command reads a DWord from memory and stores the value of that DWord to back to memory. The source and destination addresses are specified in the command. The command temporarily halts command execution.</p>				
Programming Notes				
<p>This command should not be used within a "non_privilege"batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or privilege batch buffers to access global virtual space.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	2Eh MI_MEM_TO_MEM	
		Format:	OpCode	
	22	Use Global GTT Source		
		It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.		
		Value	Name	Description
		0h	Per Process Graphics Address	
1h		Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	
21	Use Global GTT Destination			
	It is allowed for this bit to be set when executing this command from a privileged (secure) batch buffer or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear.			
	Value	Name	Description	
	0h	Per Process Graphics Address		
	1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	

MI_COPY_MEM_MEM						
	20:8	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ		
	Format:	MBZ				
	7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>3</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	3	Format:	=n Total Length - 2
Default Value:	3					
Format:	=n Total Length - 2					
1	31:2 Destination Memory Address <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value fetched specified in the DWord address above will be written. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]	
Project:	All					
Format:	GraphicsAddress[31:2]					
	1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					
2	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
Format:	MBZ					
	15:0	Destination Memory Address High <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[47:32]
Project:	All					
Format:	GraphicsAddress[47:32]					
3	31:2	Source Memory Address <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:2]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[31:2] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[31:2]
	Project:	All				
Format:	GraphicsAddress[31:2]					
	1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					
4	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					

MI_COPY_MEM_MEM						
	15:0	<p>Source Memory Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Surface Type: MMIO Register This field specifies the address of the memory location where the value is located that will be written to the address below. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[47:32] for a DWord register</p>	Project:	All	Format:	GraphicsAddress[47:32]
Project:	All					
Format:	GraphicsAddress[47:32]					

MI_DISPLAY_FLIP

MI_DISPLAY_FLIP	
Project:	BDW
Source:	BlitterCS
Length Bias:	2
<p>The MI_DISPLAY_FLIP command is used to request a specific display plane to switch (flip) to display a new buffer. The buffer is specified with a starting address and pitch. The tiled attribute of the buffer start address is programmed as part of the packet.</p> <p>The operation this command performs is also known as a "display flip request" operation - in that the flip operation itself will occur at some point in the future. This command specifies when the flip operation is to occur: either synchronously with vertical retrace to avoid tearing artifacts</p>	
Programming Notes	
<p>This command simply requests a display flip operation -- command execution then continues normally. There is no guarantee that the flip (even if asynchronous) will occur prior to subsequent commands being executed. (Note that completion of the MI_FLUSH_DW command does not guarantee that outstanding flip operations have completed). The MI_WAIT_FOR_EVENT command must be used to provide this synchronization to avoid back to back MI_DISPLAY_FLIP commands to the same display plane - by pausing command execution until a pending flip has actually completed. This synchronization can also be performed by use of the Display Flip Pending hardware status. See Display Flip Synchronization in the Device Programming Interface chapter of MI Functions.</p>	
<p>After a display flip operation is requested, software is responsible for initiating any required synchronization with subsequent buffer clear or blitter operations. For multi-buffering (e.g., double buffering) operations, this will typically require updating SURFACE_STATE or the binding table to change the blitter (back) buffer. In addition, prior to any subsequent clear or blitter operations, software must typically ensure that the new blitter buffer is not actively being displayed. Again, the MI_WAIT_FOR_EVENT command or Display Flip Pending hardware status can be used to provide this synchronization. See Display Flip Synchronization in the Device Programming Interface chapter of MI Functions.</p>	
<p>The display buffer command uses the X and Y offset for the tiled buffers from the Display Interface registers. Software is allowed to change the offset via the MMIO interface irrespective of the flip commands enqueued in the command stream. For tiled buffers, the display subsystem uses the X and Y offset in generation of the final request to memory. The offset is always updated on the next vblank for both Synchronous and Asynch Flips. It is not necessary to have a flip enqueued to update the X and Y offset</p>	
<p>The display buffer command uses the linear DWord offset for the linear buffers from the Display Interface registers. Software is allowed to change the offset via the MMIO interface irrespective of the flip commands enqueued in the command stream. For linear buffers, the display subsystem uses the Dword offset in generation of the final request to memory.</p> <ul style="list-style-type: none"> • For synchronous flips the offset is updated on the next vblank. It is not necessary to have a sync flip enqueued to update the DWord offset. • Linear memory does not support asynchronous flips. 	
<p>Events must be unmasked in the Display Engine Render Response Mask Register (DE RRMR 0x44050) prior to waiting for them with a MI_WAIT_FOR_EVENT command, or in the case of flips or scanlines, prior to starting the flip or loading the scanline. Unmasked events will wake command streamer as they occur, so for improved</p>	

MI_DISPLAY_FLIP

power savings it is recommended to only unmask events that are required. Programming the DE RRMR register can be done through MMIO or a LOAD_REGISTER_IMMEDIATE command.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 0h MI_COMMAND	
		Format: OpCode	
	28:23	MI Command Opcode	
		Default Value: 14h MI_DISPLAY_FLIP	
		Format: OpCode	
	22	Async Flip Indicator	
		Format: Enable	
	21:19	Display (Plane) Select	
		This field selects which display plane is to perform the flip operation.	
		Value	Name
		0h	Display Plane A
1h		Display Plane B	
2h		Display Sprite A	
3h		Display Sprite B	
4h		Display Plane C	
5h	Display Sprite C		
18:17	Reserved		
	Reserved		
16	Project: BDW		
	Format: MBZ		
15:13	Reserved		
	Format: MBZ		
12:8	Reserved		
	Project: BDW		
	Format: MBZ		

		MI_DISPLAY_FLIP		
	7:0	DWord Length		
		Format:	=n Total Length - 2	
		For Synchronous Flips and Asynchronous Flips, this field must be programmed to 1h for a total length of 3.		
		Value	Name	
		0h	Excludes DWord (0,1) [Default]	
1h		Exists If	(([Flip Type]!='Stereo 3D Flip')	
2h		Exists If	(([Flip Type]='Stereo 3D Flip')	
1	31	Reserved		
		Project:	BDW	
	30:16	Reserved		
		Project:	All	
		Format:	MBZ	
		Reserved		
	15:6	Project:	All	
		Reserved		
	5:1	Project:	All	
		Format:	MBZ	
	0	Tile Parameter		
		Project:	BDW	
Format:		Enable		
For Asynchronous Flips, this parameter cannot be changed. All the flips in a flip chain should maintain the same tile parameter as programmed with the last synchronous flip or direct thru MMIO.				
Value		Name	Description	
0h		Linear [Default]	For Synchronous Flips Only	
1h		Tiled X		
Programming Notes				
Performing a synchronous or asynchronous flip will drop any previous synchronous flip that has not yet completed.				
2	31:12	Display Buffer Base Address		
		Project:	All	
	Format:	GraphicsAddress[31:12]		
	This field specifies Bits 31:12 of the Graphics Address of the new display buffer.			
	Programming Notes			
	The Display buffer must reside completely in Main Memory.			
This address is always translated via the global (rather than per-process) GTT				

MI_DISPLAY_FLIP																
	11:3	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ										
	Project:	All														
	Format:	MBZ														
	2	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW												
	Project:	BDW														
	1:0	Flip Type <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field specifies whether the flip operation should be performed asynchronously to vertical retrace.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 20%;">Name</th> <th style="width: 65%;">Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Sync Flip [Default]</td> <td>The flip will occur during the vertical blanking interval - thus avoiding any tearing artifacts.</td> </tr> <tr> <td>01b</td> <td>Async Flip</td> <td>The flip will occur "as soon as possible" - and may exhibit tearing artifacts</td> </tr> <tr> <td>1b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p style="text-align: center; color: blue; margin: 0;">Programming Notes</p> <ul style="list-style-type: none"> The Display Buffer Pitch and Tile parameter cannot be changed for asynchronous flips (i.e., the new buffer must have the same pitch/tile format as the previous buffer). Async flips are supported on X-Tiled Frame buffers only. For Asynch Flips the Buffers used must be 32KB aligned. Asynch flips are supported on Display Planes A and B and C only. </div>	Project:	BDW	Value	Name	Description	00b	Sync Flip [Default]	The flip will occur during the vertical blanking interval - thus avoiding any tearing artifacts.	01b	Async Flip	The flip will occur "as soon as possible" - and may exhibit tearing artifacts	1b	Reserved	
	Project:	BDW														
	Value	Name	Description													
	00b	Sync Flip [Default]	The flip will occur during the vertical blanking interval - thus avoiding any tearing artifacts.													
	01b	Async Flip	The flip will occur "as soon as possible" - and may exhibit tearing artifacts													
1b	Reserved															
3 Project: BDW	31:12	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW												
	Project:	BDW														
	11:3	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW															
Format:	MBZ															
2	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW													
Project:	BDW															

MI_DISPLAY_FLIP			
1:0	Flip Type		
	Project:	BDW	
	This field specifies whether the flip operation should be performed asynchronously to vertical retrace.		
	Value	Name	Description
	00b	Sync Flip [Default]	The flip will occur during the vertical blanking interval - thus avoiding any tearing artifacts.
01b	Async Flip	The flip will occur "as soon as possible" - and may exhibit tearing artifacts	
Programming Notes			
<ul style="list-style-type: none"> • The Display Buffer Pitch and Tile parameter cannot be changed for asynchronous flips (i.e., the new buffer must have the same pitch/tile format as the previous buffer). • Async flips are supported on X-Tiled Frame buffers only. • For Async Flips the Buffers used must be 32KB aligned. • Async flips are supported on Display Planes A and B and C only. 			

MI_FLUSH_DW

MI_FLUSH_DW							
Project:	BDW						
Source:	VideoEnhancementCS						
Length Bias:	2						
<p>The MI_FLUSH_DW command is used to perform an internal "flush" operation. The parser pauses on an internal flush until all drawing engines have completed any pending operations. In addition, this command can also be used to:</p> <ul style="list-style-type: none"> • Flush any dirty data to memory. • Invalidate the TLB cache inside the hardware <p>Usage note: After this command is completed with a Store DWord enabled, CPU access to graphics memory will be coherent (assuming the Render Cache flush is not inhibited).</p>							
DWord	Bit	Description					
0	31:29	Command Type <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h MI_COMMAND</td> </tr> </table>	Default Value:	0h MI_COMMAND			
	Default Value:	0h MI_COMMAND					
	28:23	MI Command Opcode <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>26h MI_FLUSH_DW</td> </tr> </table>	Default Value:	26h MI_FLUSH_DW			
	Default Value:	26h MI_FLUSH_DW					
	22	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> </table>	Project:	All			
	Project:	All					
	21	Store Data Index <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Description</th> </tr> </table> <p>Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT).</p> <p>Execlist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).</p>	Project:	All	Format:	U1	Description
	Project:	All					
	Format:	U1					
	Description						
20:19	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ		
Project:	All						
Format:	MBZ						

MI_FLUSH_DW		
18	TLB Invalidate	
	Project:	All
	Format:	U1
Description		
<p>If ENABLED, all TLBs belonging to Video Enhancement Engine will be invalidated once the flush operation is complete.</p> <p>This bit is only valid when the Post-Sync Operation field is a value of 1h or 3h.</p> <p>If GFX_MODE (0x229c) bit 13, this command will cause a config write to MMIO register space with the address 0x4f100.</p>		
17	Reserved	
	Project:	BDW
	Format:	MBZ
16	Reserved	
	Project:	All
	Format:	MBZ
15:14	Post-Sync Operation	
	Project:	All
	Value	Name Description
	0h	No Write No write occurs as a result of this instruction. This can be used to implement a "trap" operation, etc.
	1h	Write Immediate Data Write the QWord containing Immediate Data Low, High DWs to the Destination Address
	2h	Reserved Reserved
	3h	Write TIMESTAMP register Write the TIMESTAMP register to the Destination Address. The upper 28 bits of the TIMESTAMP register are tied to '0'.
Programming Notes		
<p>If executed in non-secure batch buffer, the address given will be in a PPGTT address space. If in a secure ring or batch, address given will be in GGTT space</p>		
13:10	Reserved	
	Project:	All
	Format:	MBZ
9	Reserved	
	Project:	BDW
	Format:	MBZ

		MI_FLUSH_DW			
	8	Notify Enable			
		Project:	All		
		Format:	U1		
	If ENABLED, a Sync Completion Interrupt will be generated (if enabled by the MI Interrupt Control registers) once the sync operation is complete. See Interrupt Control Registers in Memory Interface Registers for details.				
	7	Reserved			
		Project:	All		
		Format:	MBZ		
	6	Reserved			
		Project:	BDW		
	5:0	DWord Length			
		Project:	All		
		Format:	=n Total Length - 2		
		Value	Name	Project	
		3h	Excludes DWord (0,1) = 2 for DWord, 3 for QWord [Default]	BDW	
1	31:3	Address			
		Project:	All		
		Format:	GraphicsAddress[31:3]U28		
	This field specifies Bits 31:3 of the Address where the DWord or QWord will be stored. Note that the address can only be QWord aligned, irrespective of data size.				
	2	Destination Address Type			
		Project:	All		
	Defines address space of Destination Address				
		Value	Name	Description	Project
		0h	PPGTT	Use PPGTT address space for DW write	All
		1h	GGTT	Use GGTT address space for DW write	All
	Programming Notes				
	Ignored if "No write" is the selected in Operation.				
	1:0	Reserved			
		Project:	All		
		Format:	MBZ		
2 Project: BDW	31:16	Reserved			
		Project:	BDW		
		Format:	MBZ		



MI_FLUSH_DW						
	15:0	<p>Address High</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]U64</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space</p>	Project:	BDW	Format:	GraphicsAddress[47:32]U64
Project:	BDW					
Format:	GraphicsAddress[47:32]U64					
3..4 Project: BDW	31:0	<p>Immediate Data</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. DW2 is the lower DW if QW is desired. Only valid when 15:14 in header is set to 1h</p> <p>To avoid hitting a known hardware bug, drivers cannot send a QW write when bit 5 of the address is '1'</p>	Project:	BDW		
Project:	BDW					

MI_FLUSH_DW

MI_FLUSH_DW		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	2	
<p>The MI_FLUSH_DW command is used to perform an internal "flush" operation. The parser pauses on an internal flush until all drawing engines have completed any pending operations. In addition, this command can also be used to: Flush any dirty data to memory. Invalidate the TLB cache inside the hardware</p> <p>Usage note: After this command is completed with a Store DWord enabled, CPU access to graphics memory will be coherent (assuming the Render Cache flush is not inhibited).</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode
		Default Value: 26h MI_FLUSH_DW
22		Reserved
		Project: All
		Format: U1
21		Store Data Index
		Project: BDW
		Format: U1
	Description	
<p>Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT).</p> <p>Execlst Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).</p>		
20:19		Reserved
		Project: All
		Format: MBZ

MI_FLUSH_DW			
18	TLB Invalidate		
	Project:	BDW	
	Format:	U1	
Description			
If ENABLED, all TLBs belonging to Blitter Engine will be invalidated once the flush operation is complete. This bit is only valid when the Post-Sync Operation field is a value of 1h or 3h.			
If GFX_MODE (0x229c) bit 13, this command will cause a config write to MMIO register space with the address 0x4f100.			
17	Reserved		
	Project:	BDW	
	Format:	MBZ	
16	Reserved		
	Project:	All	
	Format:	MBZ	
15:14	Post-Sync Operation		
	Project:	BDW	
	BitFieldDesc		
	Value	Name	Description
	0h	No Write	No write occurs as a result of this instruction. This can be used to implement a "trap" operation, etc.
	1h	Write Immediate Data QWord	Write the QWord containing Immediate Data Low, High DWs to the Destination Address
	2h	Reserved	Reserved
3h	Write TIMESTAMP Register	Write the TIMESTAMP register to the Destination Address. The upper 28 bits of the TIMESTAMP register are tied to '0'.	
13:10	Reserved		
	Project:	All	
	Format:	MBZ	
9	Reserved		
	Project:	BDW	
	Format:	MBZ	
8	Notify Enable		
	Project:	BDW	
	Format:	U1	
If ENABLED, a Sync Completion Interrupt will be generated (if enabled by the MI Interrupt Control registers) once the sync operation is complete. See Interrupt Control Registers in Memory Interface Registers for details.			

MI_FLUSH_DW				
	7:6	Reserved		
		Project:	All	
		Format:	MBZ	
	5:0	DWord Length		
	Project:	All		
	Format:	=n Total Length - 2		
		Value	Name	
		3h	Excludes DWord (0,1) = 2 for DWord, 3 for QWord [Default]	
			Project	
			BDW	
1	31:3	Address		
		Project:	BDW	
		Format:	GraphicsAddress[31:3]U28	
		This field specifies Bits 31:3 of the Address where the DWord or QWord will be stored. Note that the address can only be QWord aligned, irrespective of data size.		
	2	Destination Address Type		
		Project:	All	
		Defines address space of Destination Address		
		Value	Name	Description
		0h	PPGTT	Use PPGTT address space for DW write
		1h	GGTT	Use GGTT address space for DW write
		Programming Notes		
	Ignored if "No write" is the selected in Operation.			
	1:0	Reserved		
		Project:	All	
		Format:	MBZ	
2 Project: BDW	31:16	Reserved		
		Project:	BDW	
		Format:	MBZ	
	15:0	Address High		
	Project:	BDW		
	Format:	GraphicsAddress[47:32]U64		
	This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space			



MI_FLUSH_DW				
3..4 Project: BDW	31:0	<p>Immediate Data</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td style="width: 40%;">BDW</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. DW2 is the lower DW if QW is desired. Only valid when 15:14 in header is set to 1h</p> <p>To avoid hitting a known hardware bug, drivers cannot send a QW write when bit 5 of the address is '1'</p>	Project:	BDW
Project:	BDW			

MI_FLUSH_DW

MI_FLUSH_DW						
Project:	BDW					
Source:	VideoCS					
Length Bias:	2					
<p>The MI_FLUSH_DW command is used to perform an internal "flush" operation. The parser pauses on an internal flush until all drawing engines have completed any pending operations. In addition, this command can also be used to: Flush any dirty data to memory. Invalidate the TLB cache inside the hardware Usage note: After this command is completed with a Store DWord enabled, CPU access to graphics memory will be coherent (assuming the Render Cache flush is not inhibited).</p>						
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value: 0h MI_COMMAND				
	28:23	MI Command Opcode				
		Default Value: 26h MI_FLUSH_DW				
	22	Reserved				
		Project: BDW				
	21	Store Data Index				
		Project: BDW				
		Format: U1				
		<table border="1"> <thead> <tr> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td> Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT). Exclist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT). </td> <td>BDW</td> </tr> </tbody> </table>		Description	Project	Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT). Exclist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).
Description		Project				
Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT). Exclist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).	BDW					
Reserved						
20:19	Format: MBZ					
18	TLB Invalidate					
	Project: BDW					
	Format: U1					
<p>If ENABLED, all TLBs belonging to Video Engine will be invalidated once the flush operation is complete. This bit is only valid when the Post-Sync Operation field is a value of 1h or 3h.</p>						

MI_FLUSH_DW				
	17	Reserved		
		Project:	BDW	
		Format:	MBZ	
	16	Reserved		
		Format:	MBZ	
	15:14	Post-Sync Operation		
		Project:	BDW	
		BitFieldDesc		
		Value	Name	Description
				Project
		0h	No Write	No write occurs as a result of this instruction. This can be used to implement a "trap" operation, etc.
		1h	Write Immediate Data	HW implicitly detects the Data size to be Qword or Dword to be written to memory based on the command dword length programmed . When Dword Length indicates Qword, Writes the QWord containing Immediate Data Low, High DWs to the Destination Address . When Dword Length indicates Dword, Writes the DWord containing Immediate Data Low to the Destination Address
		2h	Reserved	Reserved
		3h		Write the TIMESTAMP register to the Destination Address. The upper 28 bits of the TIMESTAMP register are tied to '0'.
				BDW
	13:10	Reserved		
		Project:	All	
	Format:	MBZ		
9	Reserved			
	Project:	BDW		
	Format:	MBZ		
8	Notify Enable			
	Project:	BDW		
	Format:	U1		
	If ENABLED, a Sync Completion Interrupt will be generated (if enabled by the MI Interrupt Control registers) once the sync operation is complete. See Interrupt Control Registers in Memory Interface Registers for details.			
7	Video Pipeline Cache invalidate			
	Project:	BDW		
	Format:	U1		
	Enable the invalidation of the video cache at the end of this flush			

MI_FLUSH_DW													
	6	Reserved Project: BDW											
	5:0	DWord Length Format: =n Total Length - 2											
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 60%;">Name</th> <th style="width: 25%;">Project</th> </tr> </thead> <tbody> <tr> <td>3h</td> <td>Excludes DWord (0,1) = 2 for DWord, 3 for QWord [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	Project	3h	Excludes DWord (0,1) = 2 for DWord, 3 for QWord [Default]	BDW					
Value	Name	Project											
3h	Excludes DWord (0,1) = 2 for DWord, 3 for QWord [Default]	BDW											
1	31:3	Address Format: GraphicsAddress[31:3]U28 This field specifies Bits 31:3 of the Address where the DWord or QWord will be stored. Note that the address can only be QWord aligned, irrespective of data size.											
	2	Destination Address Type Defines address space of Destination Address <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>PPGTT</td> <td>Use PPGTT address space for DW write</td> </tr> <tr> <td>1h</td> <td>GGTT</td> <td>Use GGTT address space for DW write</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; color: blue;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">Ignored if "No write" is the selected in Operation.</td> </tr> </tbody> </table>	Value	Name	Description	0h	PPGTT	Use PPGTT address space for DW write	1h	GGTT	Use GGTT address space for DW write	Programming Notes	Ignored if "No write" is the selected in Operation.
	Value	Name	Description										
	0h	PPGTT	Use PPGTT address space for DW write										
1h	GGTT	Use GGTT address space for DW write											
Programming Notes													
Ignored if "No write" is the selected in Operation.													
1:0	Reserved Format: MBZ												
2	Reserved Project: BDW Format: MBZ												
2 Project: BDW	31:16	Reserved Project: BDW Format: MBZ											
	15:0	Address High Project: BDW Format: GraphicsAddress[47:32]U64 This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space											
3..4 Project: BDW	31:0	Immediate Data Project: BDW											
		This field specifies the DWord value to be written to the targeted location. DW2 is the lower DW if QW is desired. Only valid when 15:14 in header is set to 1h											
		To avoid hitting a known hardware bug, drivers cannot send a QW write when bit 5 of the address is '1'											



MI_LOAD_REGISTER_IMM

MI_LOAD_REGISTER_IMM			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	2		
<p>The MI_LOAD_REGISTER_IMM command requests a write of up to a DWord constant supplied in the command to the specified Register Offset (i.e., offset into Memory-Mapped Register Range). The register is loaded before the next command is executed.</p> <ul style="list-style-type: none"> The behavior of this command is controlled by Dword 3, Bit 8 (Disable Register Access) of the RINGBUF register. If this command is disallowed then the command stream converts it to a NOOP. If this command is executed from a batch buffer then the behavior of this command is controlled by Dword 0, Bit 8 (Security Indicator) of the BATCH_BUFFER_START Command. If the batch buffer is non-secure then the command stream converts this command to a NOOP. The following addresses should NOT be used for LRIs <ol style="list-style-type: none"> 0x8800 - 0x88FF >= 0x40000 <p>Any offset that is to a destination outside of the GT core will allow the parser to continue once the cycle is at the GT boundry and not destination. Any other address will ensure the destination is updated prior to parsing the next command</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	22h MI_LOAD_REGISTER_IMM
		Format:	OpCode
	22:12	Reserved	
		Project:	All
		Format:	MBZ
	11:8	Byte Write Disables	
		Project:	All
		Format:	Enable[4] (bit 8 corresponds to Data DWord [7:0]).
		Range: Must specify a valid register write operation	
		If [11:8] is '1111b', then the register write will not occur. If [11:8] is '0000b', then the register DW will be updated. Any other value, the behavior will be specifically specified by the register or the behavior is undefined.	

MI_LOAD_REGISTER_IMM			
	7:0	DWord Length	
		Default Value: 0h Excludes DWord (0,1)	
		Project: All Format: =n Total Length - 2	
1	31:23	Reserved	
		Project: All	
		Format: MBZ	
	22:2	Register Offset	
		Project: All	
		Format: MmioAddress[22:2]	
		This field specifies bits [22:2] of the offset into the Memory Mapped Register Range (i.e., this field specifies a DWord offset).Mapped	
		Programming Notes	Project
	Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.		BDW
	1:0	Reserved	
Project: All			
Format: MBZ			
2	31:0	Data DWord	
		Project: All	
		Format: U32	
This field specifies the DWord value to be written to the targeted location.			

MI_LOAD_REGISTER_IMM

MI_LOAD_REGISTER_IMM				
Project:	BDW			
Source:	BlitterCS			
Length Bias:	2			
<p>The MI_LOAD_REGISTER_IMM command requests a write of up to a DWord constant supplied in the command to the specified Register Offset (i.e., offset into Memory-Mapped Register Range). The register is loaded before the next command is executed.</p> <p>Any offset that is to a destination outside of the GT core will allow the parser to continue once the cycle is at the GT boundry and not destination. Any other address will ensure the destination is updated prior to parsing the next command</p>				
DWord	Bit	Description		
0	31:29	Command Type Default Value: 0h MI_COMMAND		
	28:23	MI Command Opcode Default Value: 22h MI_LOAD_REGISTER_IMM		
	22:12	Reserved Format: MBZ		
	11:8	Byte Write Disables Format: Enable[4] Bit 8 corresponds to Data DWord [7:0] Range: Must specify a valid register write operation If [11:8] is '1111b', then the register write will not occur. If [11:8] is '0000b', then the register DW will be updated. Any other value, the behavior will be specifically specified by the register or the behavior is undefined.		
	7:0	DWord Length Default Value: 1h Excludes DWord (0,1) Format: =n Total Length - 2		
1	31:23	Reserved Format: MBZ		
	22:2	Register Offset Format: U21 Format: MmioAddress[22:2]		
		This field specifies bits [22:2] of the offset into the Memory Mapped Register Range (i.e., this field specifies a DWord offset). <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.</td> <td>BDW</td> </tr> </tbody> </table>	Programming Notes	Project
Programming Notes	Project			
Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW			

MI_LOAD_REGISTER_IMM						
	1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					
2	31:0	Data DWord <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Mask:</td> <td>Bytes Write Disables</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location.</p>	Mask:	Bytes Write Disables	Format:	U32
Mask:	Bytes Write Disables					
Format:	U32					



MI_LOAD_REGISTER_IMM

MI_LOAD_REGISTER_IMM	
Project:	BDW
Source:	RenderCS
Length Bias:	2
<p>The MI_LOAD_REGISTER_IMM command requests a write of up to a DWord constant supplied in the command to the specified Register Offset (i.e., offset into Memory-Mapped Register Range). Any offset that is to a destination outside of the GT core will allow the parser to continue once the cycle is at the GT boundry and not destination. Any other address will ensure the destination is updated prior to parsing the next command</p>	
Programming Notes	
<p>A stalling flush must be sent down pipeline before issuing this command. The behavior of this command is controlled by Dword 3, Bit 8 (Disable Register Access) of the RINGBUF register. If this command is disallowed then the command stream converts it to a NOOP.</p> <p>If this command is executed from a BB then the behavior of this command is controlled by Dword 0, Bit 8 (Security Indicator) of the BATCH_BUFFER_START Command. If the batch buffer is insecure then the command stream converts this command to a NOOP. Note that the corresponding ring buffer must allow a register update for this command to execute.</p> <p>To ensure this command gets executed before upcoming commands in the ring, either a stalling pipeControl should be sent after this command, or MMIO 0x20C0 bit 7 should be set to 1.</p> <p>When base address of 0x180000 is added to the Register Offset, when executed will result in updating of the register in the other GT in GTB mode of operation then the GT from which this instruction is executed. When this instruction is executed by Command Streamer with COREID-0 will result in updating the register in GT with COREID-1 and vice versa, when base address of 0x180000 is added to the register offset.</p> <p>The following addresses should NOT be used for LRIs:</p> <ol style="list-style-type: none"> 1. 0x8800 - 0x88FF 2. >= 0xC0000 <p>Limited LRI cycles to the Display Engine 0x40000-0xBFFFF) are allowed, but must be spaced to allow only one pending at a time. This can be done by issuing an SRM to the same address immediately after each LRI.</p> <p>Programming an MMIO register is equivalent to programming a non-pipeline state to the hardware and hence an explicit stalling flush needs to be programmed prior to programming this command. However for certain MMIO registers based on their functionality doing an explicit stalling flush is exempted. Listed below are the exempted registers.</p> <ul style="list-style-type: none"> • 3DPRIM_END_OFFSET - Auto Draw End Offset [BDW] • 3DPRIM_START_VERTEX - Load Indirect Start Vertex [BDW] • 3DPRIM_VERTEX_COUNT - Load Indirect Vertex Count [BDW] • 3DPRIM_INSTANCE_COUNT - Load Indirect Instance Count [BDW] • 3DPRIM_START_INSTANCE - Load Indirect Start Instance [BDW] • 3DPRIM_BASE_VERTEX - Load Indirect Base Vertex [BDW] <p>Writes to the range 0x9400-0x97FF must be either be avoided, or serialized with a read (e.g. STORE_REGISTER_MEM) between them.</p>	

DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND Format: OpCode
	28:23	MI Command Opcode
		Default Value: 22h MI_LOAD_REGISTER_IMM Format: OpCode
	22:13	Reserved Format: MBZ
	12	Reserved Project: BDW
	11:8	Byte Write Disables Format: Enable[4] Bit 8 corresponds to Data DWord [7:0]
Range: Must specify a valid register write operation If [11:8] is '1111b', then this command will behave as a NOOP. Otherwise, the value is forwarded to the destination register.		
7:0	DWord Length	
	Default Value: 1h Excludes DWord (0,1) Format: =n Total Length - 2. Excludes DWord (0,1).	
1	31:23	Reserved Format: MBZ
	22:2	Register Offset Format: MmioAddress[22:2] This field specifies bits [22:2] of the offset into the Memory Mapped Register Range (i.e., this field specifies a DWord offset).
	1:0	Reserved Format: MBZ
2	31:0	Data DWord
		Mask: Bytes Write Disables Format: U32 This field specifies the DWord value to be written to the targeted location.

MI_LOAD_REGISTER_IMM

MI_LOAD_REGISTER_IMM		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
<p>The MI_LOAD_REGISTER_IMM command requests a write of up to a DWord constant supplied in the command to the specified Register Offset (i.e., offset into Memory-Mapped Register Range). The register is loaded before the next command is executed.</p> <p>Any offset that is to a destination outside of the GT core will allow the parser to continue once the cycle is at the GT boundry and not destination. Any other address will ensure the destination is updated prior to parsing the next command</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
		Default Value: 22h MI_LOAD_REGISTER_IMM
	Format: OpCode	
22:12	Reserved	
Format: MBZ		
11:8	Byte Write Disables	
	Format: Enable[4] (bit 8 corresponds to Data DWord [7:0]).	
	<p>Range: Must specify a valid register write operation</p> <p>If [11:8] is '1111b', then the register write will not occur. If [11:8] is '0000b', then the register DW will be updated. Any other value, the behavior will be specifically specified by the register or the behavior is undefined.</p>	
7:0	DWord Length	
	Default Value: 0h Excludes DWord (0,1)	
Format: =n Total Length - 2		
1	31:23	Reserved
		Format: MBZ
	22:2	Register Offset
Format: MmioAddress[22:2]		
<p>This field specifies bits [22:2] of the offset into the Memory Mapped Register Range (i.e., this field specifies a DWord offset).Mapped</p>		
Programming Notes		Project
<p>Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.</p>		BDW

MI_LOAD_REGISTER_IMM				
	1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
2	31:0	Data DWord <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>U32 FormatDesc</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location.</p>	Format:	U32 FormatDesc
Format:	U32 FormatDesc			



MI_LOAD_REGISTER_MEM

MI_LOAD_REGISTER_MEM		
Project:	BDW	
Source:	RenderCS, BlitterCS, VideoCS, VideoEnhancementCS	
Length Bias:	2	
The MI_LOAD_REGISTER_MEM command requests from a memory location and stores that DWord to a register.		
Programming Notes		
The command temporarily halts commands that will cause cycles down the 3D pipeline.		
The following addresses should NOT be used for MMIO writes: <ul style="list-style-type: none"> • 0x8800 - 0x88FF • >= 0xC0000 Limited MMIO writes cycles to the Display Engine 0x40000-0xBFFFF) are allowed, but must be spaced to allow only one pending at a time. This can be done by issuing an SRM to the same address immediately after each MMIO write.		
Any updates to the memory location exercised by this command must be ensured to be coherent in memory prior to programming of this command. This must be achieved by programming MI_ATOMIC (write to scratch space) with "CS STALL" set prior to programming of this command. Example: MI_STORE_REGISTER_MEM (0x2288, 0x2CF0_0000) MI_ATOMIC (MOV, Dummy data, Scratch Address) MI_LOAD_REGISTER_MEM(0x2288, 0x2CF0_0000)		
This command should not be used within a non-privilege batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation.		
This command is not allowed to update the privilege register range when executed from a non-privilege batch buffer.		
Writes to the range 0x9400-0x97FF must be either be avoided, or serialized with a read (e.g. STORE_REGISTER_MEM) between them.		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
Default Value: 29h MI_LOAD_REGISTER_MEM		
Format: OpCode		

MI_LOAD_REGISTER_MEM										
	22	<p>Use Global GTT</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>Boolean</td> </tr> </table> <p>This bit if set when executing from a non-privileged batch buffer will be treated as privilege access violation. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer or ring buffer. This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</p>	Format:	Boolean						
	Format:	Boolean								
	21	<p>Async Mode Enable</p> <p>If this bit is set then the command stream will not wait for completion of this command before executing the next command. Please refer to the LOAD_INDIRECT and Predicate registers for usage of this bit.</p>								
	20	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW						
	Project:	BDW								
	19	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW						
Project:	BDW									
18:8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td colspan="2">=n Total Length - 2. Excludes DWord (0,1).</td> </tr> <tr> <td style="text-align: center;">Value</td> <td style="text-align: center;">Name</td> <td style="text-align: center;">Project</td> </tr> <tr> <td>2h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </table>	Format:	=n Total Length - 2. Excludes DWord (0,1).		Value	Name	Project	2h	Excludes DWord (0,1) [Default]	BDW
Format:	=n Total Length - 2. Excludes DWord (0,1).									
Value	Name	Project								
2h	Excludes DWord (0,1) [Default]	BDW								
1	31:23	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
	22:2	<p>Register Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MMIOAddress[22:2]</td> </tr> </table> <p>This field specifies Bits 22:2 of the Register offset the DWord will be written to. As the register address must be DWord-aligned, Bits 1:0 of that address MBZ.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> <th style="text-align: center;">Source</th> </tr> </thead> <tbody> <tr> <td>Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.</td> <td>BDW</td> <td>BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> </tbody> </table>	Format:	MMIOAddress[22:2]	Programming Notes	Project	Source	Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS
Format:	MMIOAddress[22:2]									
Programming Notes	Project	Source								
Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS								
1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									

MI_LOAD_REGISTER_MEM					
2..3 Project: BDW	63:2	Memory Address			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:2]</td> </tr> </table> <p>This field specifies the address of the memory location where the register value specified in the DWord above will read from. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[63:2] for a DWord register GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	BDW	Format:
	Project:	BDW			
	Format:	GraphicsAddress[63:2]			
1:0	Reserved				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ
Project:	BDW				
Format:	MBZ				

MI_LOAD_REGISTER_REG

MI_LOAD_REGISTER_REG			
Project:	BDW		
Source:	CommandStreamer		
Length Bias:	2		
<p>The MI_LOAD_REGISTER_REG command reads from a source register location and writes that value to a destination register location.</p> <p>Any offset that is to a destination outside of the GT core will allow the parser to continue once the cycle is at the GT boundry and not destination. Any other address will ensure the destination is updated prior to parsing the next command</p>			
Programming Notes			
The command temporarily halts commands that will cause cycles down the 3D pipeline.			
Destination register with mask implemented will not get updated unless the value read from source register has the bits corresponding to the mask bits set. Note that any mask implemented register when read returns "0" for the bits corresponding to mask location. When the source and destination are mask implemented registers, destination register will not get updated with the source register contents.			
This command is not allowed to update the privilege register range when executed from a non-privilege batch buffer.			
Writes to the range 0x9400-0x97FF must be either be avoided, or serialized with a read (e.g. STORE_REGISTER_MEM) between them.			
Project			
BDW			
BDW			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	2Ah MI_LOAD_REGISTER_REG
		Format:	OpCode
	22:20	Reserved	
		Format:	MBZ
	19:18	Reserved	
		Project:	BDW
		Format:	MBZ
	17:8	Reserved	
Format:		MBZ	
7:0	DWord Length		
	Default Value:	1h	
	Format:	=n Total Length - 2. Excludes DWord (0,1).	

MI_LOAD_REGISTER_REG								
1	31:23	Reserved Format: _____ MBZ						
	22:2	Source Register Address Format: _____ MMIOAddress[22:2]MMIO_Register This field specifies Bits 22:2 of the Register offset the DWord will be written to. As the register address must be DWord-aligned, Bits 1:0 of that address MBZ.						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> <th style="text-align: center;">Source</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.</td> <td style="text-align: center; padding: 5px;">BDW</td> <td style="padding: 5px;">BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> </tbody> </table>		Programming Notes	Project	Source	Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS
	Programming Notes	Project	Source					
Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS						
1:0	Reserved Format: _____ MBZ							
2	31:23	Reserved Format: _____ MBZ						
	22:2	Destination Register Address Format: _____ MMIOAddress[22:2]MMIO_Register This field specifies Bits 22:2 of the Register offset the DWord will be written to. As the register address must be DWord-aligned, Bits 1:0 of that address MBZ.						
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> <th style="text-align: center;">Source</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.</td> <td style="text-align: center; padding: 5px;">BDW</td> <td style="padding: 5px;">BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> </tbody> </table>		Programming Notes	Project	Source	Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS
	Programming Notes	Project	Source					
Bits 22:18 must be zero. Setting these bits could cause a hang due to PM requesting a stop at the same time the request is going to a MMIO space outside the GT core.	BDW	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS						
1:0	Reserved Format: _____ MBZ							

MI_LOAD_SCAN_LINES_EXCL

MI_LOAD_SCAN_LINES_EXCL	
Project:	BDW
Source:	BlitterCS
Length Bias:	2

The MI_LOAD_SCAN_LINES_EXCL command is used to initialize the Scan Line Window registers for a specific Display Pipe. If the display refresh is *outside* this window the Display Engine asserts a signal that is used by the command parser to process the WAIT_FOR_EVENT command (i.e., the parser will wait while outside). This command overrides the Scan Line Window defined by any previous MI_LOAD_SCAN_LINES_INCL or MI_LOAD_SCAN_LINES_EXCL commands targeting the specific display pipe.

Note: The two scan-line numbers are inclusive. If programmed to the same values, that single line defines the region in question.

Always place an even number of MI_LOAD_SCAN_LINES_EXCL/INCL at a time into the ring buffer. If only a single MI_LOAD_SCAN_LINES_EXCL/INCL is desired, just add a second identical MI_LOAD_SCAN_LINES_EXCL/INCL command.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	13h MI_LOAD_SCAN_LINES_EXCL
		Format:	OpCode
	22	Reserved	
		Project:	All
		Format:	MBZ
	21:19	Display Pipe Select	
		Project:	All
		Format:	U3
This field selects which Display Engine (pipe) this command is targeting.			
Value		Name	Project
0h		Display Pipe A	All
1h		Display Pipe B	All
2h, 3h		Reserved	All
4h		Display Pipe C	All
5h		Reserved	BDW
6h, 7h	Reserved	All	
18:17	Reserved		
	Project:	BDW	

MI_LOAD_SCAN_LINES_EXCL					
	16:6	Reserved			
		<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
	Project:	BDW			
	Format:	MBZ			
5:0	DWord Length				
	<table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Default Value:</td> <td>0h Excludes DWord (0,1)</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0h Excludes DWord (0,1)	Format:	=n Total Length - 2
Default Value:	0h Excludes DWord (0,1)				
Format:	=n Total Length - 2				
1	31:16	Start Scan Line Number <table border="1" style="width: 100%;"> <tr> <td style="width: 10%;">Format:</td> <td>U16 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> </table> <p>This field specifies the starting scan line number of the Scan Line Window. Range: [0, Display Buffer height in lines-1]</p>	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.	
	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.			
15:0	End Scan Line Number <table border="1" style="width: 100%;"> <tr> <td style="width: 10%;">Format:</td> <td>U16 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> </table> <p>This field specifies the ending scan line number of the Scan Line Window. Range: [0, Display Buffer height in lines-1]</p>	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.		
Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.				

MI_LOAD_SCAN_LINES_EXCL

MI_LOAD_SCAN_LINES_EXCL			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The MI_LOAD_SCAN_LINES_EXCL command is used to initialize the Scan Line Window registers for a specific Display Pipe. If the display refresh is outside this window the Display Engine asserts a signal that is used by the command parser to process the WAIT_FOR_EVENT command (i.e., the parser will wait while outside). This command overrides the Scan Line Window defined by any previous MI_LOAD_SCAN_LINES_INCL or MI_LOAD_SCAN_LINES_EXCL commands targeting the specific display pipe. Note: The two scan-line numbers are inclusive. If programmed to the same values, that single line defines the region in question. Always place an even number of MI_LOAD_SCAN_LINES_EXCL/INCL at a time into the ring buffer. If only a single MI_LOAD_SCAN_LINES_EXCL/INCL is desired, just add a second identical MI_LOAD_SCAN_LINES_EXCL/INCL command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	13h MI_LOAD_SCAN_LINES_EXCL
		Format:	OpCode
	22	Reserved	
		Format:	MBZ
	21:19	Display (Plane) Select	
		Format:	U3
		This field selects which display plane is to perform the scanline operation.	
		Value	Name
0h		Display Plane A	
1h		Display Plane B	
2h		Reserved	
3h		Reserved	
4h	Display Plane C		
5h	Reserved	BDW	
18:17	Reserved		
	Project:	BDW	
16:6	Reserved		
	Project:	BDW	
	Format:	MBZ	

MI_LOAD_SCAN_LINES_EXCL								
	5:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2. Excludes DWord (0,1).</td> </tr> </table>	Default Value:	0h	Format:	=n Total Length - 2. Excludes DWord (0,1).		
Default Value:	0h							
Format:	=n Total Length - 2. Excludes DWord (0,1).							
1	31:29	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
	28:16	<p>Start Scan Line Number</p> <table border="1"> <tr> <td>Format:</td> <td>U13 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> <tr> <td colspan="2">Range: [0, Display Buffer height in lines-1]</td> </tr> <tr> <td colspan="2">This field specifies the starting scan line number of the Scan Line Window.</td> </tr> </table>	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.	Range: [0, Display Buffer height in lines-1]		This field specifies the starting scan line number of the Scan Line Window.	
	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.						
Range: [0, Display Buffer height in lines-1]								
This field specifies the starting scan line number of the Scan Line Window.								
15:13	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
12:0	<p>End Scan Line Number</p> <table border="1"> <tr> <td>Format:</td> <td>U13 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> <tr> <td colspan="2">This field specifies the ending scan line number of the Scan Line Window.</td> </tr> <tr> <td colspan="2">Range: [0, Display Buffer height in lines-1]</td> </tr> </table>	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.	This field specifies the ending scan line number of the Scan Line Window.		Range: [0, Display Buffer height in lines-1]		
Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.							
This field specifies the ending scan line number of the Scan Line Window.								
Range: [0, Display Buffer height in lines-1]								

MI_LOAD_SCAN_LINES_INCL

MI_LOAD_SCAN_LINES_INCL	
Project:	BDW
Source:	BlitterCS
Length Bias:	2

The MI_LOAD_SCAN_LINES_INCL command is used to initialize the Scan Line Window registers for a specific Display Engine. If the display refresh is *within* this window the Display Engine asserts a signal that is used by the command parser to process the WAIT_FOR_EVENT command (i.e., the parser will wait while inside of the window). This command overrides the Scan Line Window defined by any previous MI_LOAD_SCAN_LINES_INCL or MI_LOAD_SCAN_LINES_EXCL commands targeting the specific display. Always place an even number of MI_LOAD_SCAN_LINES_EXCL/INCL at a time into the ring buffer. If only a single MI_LOAD_SCAN_LINES_EXCL/INCL is desired, just add a second identical

DWord	Bit	Description																					
0	31:29	Command Type																					
		Default Value:	0h MI_COMMAND																				
		Format:	OpCode																				
	28:23	MI Command Opcode																					
		Default Value:	12h MI_LOAD_SCAN_LINES_INCL																				
		Format:	OpCode																				
	22	Reserved																					
		Project:	All																				
		Format:	MBZ																				
	21:19	Display Pipe Select	Project:	All																			
Format:			U3																				
This field selects which Display Engine (pipe) this command is targeting.																							
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Display Pipe A</td> <td></td> </tr> <tr> <td>1h</td> <td>Display Pipe B</td> <td></td> </tr> <tr> <td>2h, 3h</td> <td>Reserved</td> <td>All</td> </tr> <tr> <td>4h</td> <td>Display Pipe C</td> <td></td> </tr> <tr> <td>5h</td> <td>Reserved</td> <td>BDW</td> </tr> <tr> <td>6h, 7h</td> <td>Reserved</td> <td>All</td> </tr> </tbody> </table>		Value	Name	Project	0h	Display Pipe A		1h	Display Pipe B		2h, 3h	Reserved	All	4h	Display Pipe C		5h	Reserved	BDW	6h, 7h	Reserved	All	
Value		Name	Project																				
0h		Display Pipe A																					
1h		Display Pipe B																					
2h, 3h		Reserved	All																				
4h		Display Pipe C																					
5h		Reserved	BDW																				
6h, 7h	Reserved	All																					
18:17	Reserved																						
	Project:	BDW																					
16:6	Reserved																						
	Project:	BDW																					
	Format:	MBZ																					

MI_LOAD_SCAN_LINES_INCL						
	5:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>0h Excludes DWord (0,1)</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0h Excludes DWord (0,1)	Format:	=n Total Length - 2
Default Value:	0h Excludes DWord (0,1)					
Format:	=n Total Length - 2					
1	31:16	<p>Start Scan Line Number</p> <table border="1"> <tr> <td>Format:</td> <td>U16 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> </table> <p>This field specifies the starting scan line number of the Scan Line Window. Range: [0, Display Buffer height in lines-1]</p>	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.		
	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.				
15:0	<p>End Scan Line Number</p> <table border="1"> <tr> <td>Format:</td> <td>U16 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> </table> <p>This field specifies the ending scan line number of the Scan Line Window. Range: [0, Display Buffer height in lines-1]</p>	Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.			
Format:	U16 In scan lines, where scan line 0 is the first line of the display frame.					

MI_LOAD_SCAN_LINES_INCL

MI_LOAD_SCAN_LINES_INCL			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The MI_LOAD_SCAN_LINES_INCL command is used to initialize the Scan Line Window registers for a specific Display Engine. If the display refresh is within this window the Display Engine asserts a signal that is used by the command parser to process the WAIT_FOR_EVENT command (i.e., the parser will wait while inside the window). This command overrides the Scan Line Window defined by any previous MI_LOAD_SCAN_LINES_INCL or MI_LOAD_SCAN_LINES_EXCL commands targeting the specific display.</p> <p>Always place an even number of MI_LOAD_SCAN_LINES_EXCL/INCL at a time into the ring buffer. If only a single MI_LOAD_SCAN_LINES_EXCL/INCL is desired, just add a second identical MI_LOAD_SCAN_LINES_EXCL/INCL command.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	12h MI_LOAD_SCAN_LINES_INCL
		Format:	OpCode
	22	Reserved	
		Format:	MBZ
	21:19	Display (Plane) Select	
		Project:	BDW
		Format:	U3
		This field selects which display plane is to perform the scanline operation.	
		Value	Name
		0h	Display Plane A
1h		Display Plane B	
2h		Reserved	
3h		Reserved	
4h		Display Plane C	
5h	Reserved		
18:17	Reserved		
16:6	Reserved		
	Project:	BDW	
	Format:	MBZ	

MI_LOAD_SCAN_LINES_INCL								
	5:0	DWord Length						
		<table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2. Excludes DWord (0,1).</td> </tr> </table>	Default Value:	0h	Format:	=n Total Length - 2. Excludes DWord (0,1).		
Default Value:	0h							
Format:	=n Total Length - 2. Excludes DWord (0,1).							
1	31	Reserved						
		<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ		
		Project:	BDW					
	Format:	MBZ						
	Reserved							
	30	Reserved						
		<table border="1"> <tr> <td>Default Value:</td> <td>1h</td> </tr> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Must Be One</td> </tr> </table>	Default Value:	1h	Project:	BDW	Format:	Must Be One
		Default Value:	1h					
	Project:	BDW						
	Format:	Must Be One						
	Reserved							
	29	Reserved						
28:16	Start Scan Line Number							
	<table border="1"> <tr> <td>Format:</td> <td>U13 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> <tr> <td colspan="2">Range: [0, Display Buffer height in lines-1]</td> </tr> <tr> <td colspan="2">This field specifies the starting scan line number of the Scan Line window.</td> </tr> </table>	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.	Range: [0, Display Buffer height in lines-1]		This field specifies the starting scan line number of the Scan Line window.		
	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.						
	Range: [0, Display Buffer height in lines-1]							
This field specifies the starting scan line number of the Scan Line window.								
Reserved								
15:13	Reserved							
12:0	End Scan Line Number							
	<table border="1"> <tr> <td>Format:</td> <td>U13 In scan lines, where scan line 0 is the first line of the display frame.</td> </tr> <tr> <td colspan="2">Range: [0, Display Buffer height in lines-1]</td> </tr> <tr> <td colspan="2">This field specifies the ending scan line number of the Scan Line Window.</td> </tr> </table>	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.	Range: [0, Display Buffer height in lines-1]		This field specifies the ending scan line number of the Scan Line Window.		
	Format:	U13 In scan lines, where scan line 0 is the first line of the display frame.						
	Range: [0, Display Buffer height in lines-1]							
This field specifies the ending scan line number of the Scan Line Window.								
Reserved								
Reserved								

MI_LOAD_URB_MEM

MI_LOAD_URB_MEM						
Project:	BDW					
Source:	RenderCS					
Length Bias:	2					
The MI_LOAD_URB_MEM command requests from a memory location and stores that DWord to the URB.						
Programming Notes						
The command temporarily halts commands that will cause cycles down the 3D pipeline.						
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value:	0h MI_COMMAND			
		Format:	OpCode			
	28:23	MI Command Opcode				
		Default Value:	2Ch MI_LOAD_URB_MEM			
22:8	Reserved					
	Format:	MBZ				
7:0	DWord Length					
	Format:	=n				
	Total Length - 2. Excludes DWord (0,1).					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>2h</td> <td>[Default]</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	Project	2h	[Default]
Value	Name	Project				
2h	[Default]	BDW				
1	31:15	Reserved				
		Format:	MBZ			
	14:2	URB Address This field specifies Bits 14:2 of the URB offset the DWord will be written in the URB. This command only supports writing below 32KB of the URB space.				
2..3 Project: BDW	63:6	Memory Address				
		<table border="1"> <tbody> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:6]</td> </tr> </tbody> </table> <p>This field specifies the address of the location of where the value will be read from memory. The value must be in the first DW location of the cache line. Range = GraphicsVirtualAddress[47:6] GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	BDW	Format:	GraphicsAddress[63:6]
Project:	BDW					
Format:	GraphicsAddress[63:6]					



MI_LOAD_URB_MEM		
	5:0	Reserved
		Project: BDW
		Format: MBZ

MI_MATH

MI_MATH		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	2	
<p>The MI_MATH command allows software to send instructions to the ALU in the Command Streamer. This command is the means by which the ALU is accessed. ALU instructions form the data payload of the MI_MATH command. An ALU instruction takes one DWord in size. The MI_MATH DWord Length is programmed based on the number of ALU instructions included, limited only by the max DWord Length supported. When the command streamer parses an MI_MATH command, it sends the included ALU instructions to the ALU. The ALU processes any instruction in a single clock. See the ALU section for more details.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND Format: OpCode
	28:23	MI Command Opcode
		Default Value: 1Ah MI_MATH Format: OpCode
22:8	Reserved	
	Format: MBZ	
7:0	DWord Length	
	Default Value: 0h Format: =n Total Length - 2. Excludes DWord (0,1).	
1	31:0	ALU INSTRUCTION 1
		Format: Table Entry
2	31:0	ALU INSTRUCTION 2
		Format: Table Entry
3..n	31:0	ALU INSTRUCTION n
		Format: Table Entry

MI_MATH

MI_MATH		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
<p>The MI_MATH command allows software to send instructions to the ALU in the Command Streamer. This command is the means by which the ALU is accessed. ALU instructions form the data payload of the MI_MATH command. An ALU instruction takes one DWord in size. The MI_MATH DWord Length is programmed based on the number of ALU instructions included, limited only by the max DWord Length supported. When the command streamer parses an MI_MATH command, it sends the included ALU instructions to the ALU. The ALU processes any instruction in a single clock. See the ALU section for more details.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND Format: OpCode
	28:23	MI Command Opcode
		Default Value: 1Ah MI_MATH Format: OpCode
22:8	Reserved	
	Format: MBZ	
7:0	DWord Length	Default Value: 0h
		Format: =n Total Length - 2. Excludes DWord (0,1).
1	31:0	ALU INSTRUCTION 1
		Format: Table Entry
2	31:0	ALU INSTRUCTION 2
		Format: Table Entry
3..n	31:0	ALU INSTRUCTION n
		Format: Table Entry

MI_MATH

MI_MATH					
Project:	BDW				
Source:	VideoEnhancementCS				
Length Bias:	2				
<p>The MI_MATH command allows software to send instructions to the ALU in the Command Streamer. This command is the means by which the ALU is accessed. ALU instructions form the data payload of the MI_MATH command. An ALU instruction takes one DWord in size. The MI_MATH DWord Length is programmed based on the number of ALU instructions included, limited only by the max DWord Length supported. When the command streamer parses an MI_MATH command, it sends the included ALU instructions to the ALU. The ALU processes any instruction in a single clock. See the ALU section for more details.</p>					
DWord	Bit	Description			
0	31:29	Command Type			
		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h MI_COMMAND</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	0h MI_COMMAND	Format:
	Default Value:	0h MI_COMMAND			
	Format:	OpCode			
28:23	MI Command Opcode				
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>1Ah MI_MATH</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </table>	Default Value:	1Ah MI_MATH	Format:	OpCode
Default Value:	1Ah MI_MATH				
Format:	OpCode				
22:8	Reserved				
	Format:	MBZ			
7:0	7:0	DWord Length			
		Default Value:	0h		
		Format:	=n Total Length - 2. Excludes DWord (0,1).		
1	31:0	ALU INSTRUCTION 1			
		Format:	Table Entry		
2	31:0	ALU INSTRUCTION 2			
		Format:	Table Entry		
3..n	31:0	ALU INSTRUCTION n			
		Format:	Table Entry		

MI_MATH

MI_MATH		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>The MI_MATH command allows SW to send instruction to ALU in Render Command Streamer. MI_MATH command is the means by which ALU can be accessed. ALU instructions form the data payload of MI_MATH command, ALU instruction is dword in size. MI_MATH Dword Length should be programmed based on the number of ALU instruction packed, max number is limited by the max Dword Length supported. When MI_MATH command is parsed by command streamer it outputs the payload dwords (ALU instructions) to the ALU. ALU takes single clock to process any given instruction. Refer to B-spec "Command Streamer (CS) ALU Programming" section in Command Streamer Programming.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 1Ah MI_MATH
		Format: OpCode
	22:8	Reserved
		Format: MBZ
	7:6	Reserved
		Project: BDW
		Format: MBZ
	5:0	DWord Length
		Default Value: 0h
		Project: BDW
		Format: =n Total Length - 2. Excludes DWord (0,1).
1	31:0	ALU INSTRUCTION 1
		Format: Table Entry
2	31:0	ALU INSTRUCTION 2
		Format: Table Entry
3..n	31:0	ALU INSTRUCTION n
		Format: Table Entry

MI_NOOP

MI_NOOP			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	1		
<p>The MI_NOOP command basically performs a "no operation" in the command stream and is typically used to pad the command stream (e.g., in order to pad out a batch buffer to a QWord boundary). However, there is one minor (optional) function this command can perform - a 22-bit value can be loaded into the MI NOPID register. This provides a general-purpose command stream tagging ("breadcrumb") mechanism (e.g., to provide sequencing information for a subsequent breakpoint interrupt).</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	00h MI_NOOP
		Format:	OpCode
	22	Identification Number Register Write Enable	
		Format:	Enable
		This field enables the value in the Identification Number field to be written into the MI NOPID register. If disabled, that register is unmodified - making this command an effective "no operation" function.	
		Value	Name
		1	Write th NOP_ID Register
		0	Do not write the NOP_ID register
21:0	Identification Number		
	Project:	All	
	Format:	U22	
		This field contains a 22-bit number which can be written to the MI NOPID register.	



MI_NOOP

MI_NOOP					
Project:	BDW				
Source:	BlitterCS				
Length Bias:	1				
<p>The MI_NOOP command basically performs a "no operation" in the command stream and is typically used to pad the command stream (e.g., in order to pad out a batch buffer to a QWord boundary). However, there is one minor (optional) function this command can perform - a 22-bit value can be loaded into the MI NOPID register. This provides a general-purpose command stream tagging ("breadcrumb") mechanism (e.g., to provide sequencing information for a subsequent breakpoint interrupt).</p>					
DWord	Bit	Description			
0	31:29	Command Type			
		Default Value:	0h MI_COMMAND		
	28:23	MI Command Opcode			
		Default Value:	0h MI_NOOP		
	22	Identification Number Register Write Enable			
		Project:	All		
		Format:	Enable		
		This field enables the value in the Identification Number field to be written into the MI NOPID register. If disabled, that register is unmodified - making this command an effective "no operation" function.			
		Value	Name	Description	Project
		0h	Disable	Do not write the NOP_ID register.	All
1h	Enable	Write the NOP_ID register.	All		
21:0	Identification Number				
	Project:	All			
	Format:	U22			
This field contains a 22-bit number which can be written to the MI NOPID register.					

MI_NOOP

MI_NOOP												
Project:	BDW											
Source:	RenderCS											
Length Bias:	1											
<p>The MI_NOOP command basically performs a "no operation" in the command stream and is typically used to pad the command stream (e.g., in order to pad out a batch buffer to a QWord boundary). However, there is one minor (optional) function this command can perform - a 22-bit value can be loaded into the MI NOPID register. This provides a general-purpose command stream tagging ("breadcrumb") mechanism (e.g., to provide sequencing information for a subsequent breakpoint interrupt).</p>												
Performance		Project										
<p>The MI_NOOP process time is reduced to 1 clock. An example use of the improved NOOP throughput is for some multi-pass media applications where some unwanted media object commands are replaced by MI_NOOP commands without repacking the commands in a batch buffer.</p>		BDW										
DWord	Bit	Description										
0	31:29	Command Type <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td style="width: 50%;">0h MI_COMMAND</td> </tr> </table>	Default Value:	0h MI_COMMAND								
	Default Value:	0h MI_COMMAND										
	28:23	MI Command Opcode <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td style="width: 50%;">0h MI_NOOP</td> </tr> </table>	Default Value:	0h MI_NOOP								
	Default Value:	0h MI_NOOP										
22	Identification Number Register Write Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td style="width: 50%;">Enable</td> </tr> </table> <p>This field enables the value in the Identification Number field to be written into the MI NOPID register. If disabled, that register is unmodified, making this command an effective "no operation" function.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 70%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Do not write the NOP_ID register.</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Write the NOP_ID register.</td> </tr> </tbody> </table>	Format:	Enable	Value	Name	Description	0h	Disable	Do not write the NOP_ID register.	1h	Enable	Write the NOP_ID register.
Format:	Enable											
Value	Name	Description										
0h	Disable	Do not write the NOP_ID register.										
1h	Enable	Write the NOP_ID register.										
21:0	Identification Number <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td style="width: 50%;">U22</td> </tr> </table> <p>This field contains a 22-bit number which can be written to the MI NOPID register.</p>	Format:	U22									
Format:	U22											



MI_NOOP

MI_NOOP				
Project:	BDW			
Source:	VideoCS			
Length Bias:	1			
<p>The MI_NOOP command basically performs a "no operation" in the command stream and is typically used to pad the command stream (e.g., in order to pad out a batch buffer to a QWord boundary). However, there is one minor (optional) function this command can perform - a 22-bit value can be loaded into the MI NOPID register. This provides a general-purpose command stream tagging ("breadcrumb") mechanism (e.g., to provide sequencing information for a subsequent breakpoint interrupt).</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	00h MI_NOOP	
		Format:	OpCode	
	22	Identification Number Register Write Enable		
		Format:	Enable	
		This field enables the value in the Identification Number field to be written into the MI NOPID register. If disabled, that register is unmodified - making this command an effective "no operation" function.		
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>Write the NOP_ID register.</td> </tr> </tbody> </table>	Value	Name
Value	Name			
1	Write the NOP_ID register.			
21:0	Identification Number			
	Format:	U22		
	This field contains a 22-bit number which can be written to the MI NOPID register.			

MI_PREDICATE

MI_PREDICATE				
Project:	BDW			
Source:	RenderCS			
Length Bias:	1			
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	0Ch MI_PREDICATE	
		Format:	OpCode	
	22:8	Reserved		
		Format:	MBZ	
	7:6	Load Operation		
		This field controls if/how the Predicate state bit is modified.		
		Value	Name	Description
		0h	KEEP	The Predicate state bit is unmodified.
1h		Reserved		
2h		LOAD	The Predicate state bit is loaded with the combine operation result.	
3h	LOADINV	The Predicate state bit is loaded with the inverted combine operation result.		
5	Reserved			
	Format:	MBZ		
4:3	Combine Operation			
	This field controls if/how the result of the compare operation is combined with the current Predicate state bit.			
	Value	Name	Description	
	0h	SET	The combine operation output the compare result unmodified.	
	1h	AND	The combine operation outputs the AND of the compare result and the current Predicate state bit.	
	2h	OR	The combine operation outputs the OR of the compare result and the current Predicate state bit.	
3h	XOR	The combine operation outputs the XOR of the compare result and the current Predicate state bit.		
2	Reserved			
	Format:	MBZ		

MI_PREDICATE		
1:0	<p>Compare Operation This field controls how Data DWord 0 and Data DWord 1 fields are used to generate a compare operation result and possibly modify the PredicateData register.</p>	
	Value	Name
	Description	
0h	TRUE	The compare operation outputs TRUE. The PredicateData register is unmodified.
1h	FALSE	The compare operation outputs FALSE. The PredicateData register is unmodified.
2h	SRCS_EQUAL	(Mltemp0 - Mltemp1) is computed and loaded into the PredicateData register. The compare operation outputs (Mltemp0 == Mltemp1).
3h	DELTAS_EQUAL	(Mltemp0 - Mltemp1) is computed and compared to the PredicateData register. If the values are equal, the compare operation outputs TRUE, otherwise it outputs FALSE. The PredicateData register is unmodified.

MI_REPORT_HEAD

MI_REPORT_HEAD			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	1		
<p>The MI_REPORT_HEAD command causes the Head Pointer value of the ring buffer to be written to a cacheable (snooped) system memory location.</p> <p>When the Per-Process Virtual Address Space and Execlist Enable bit is reset: The location written is relative to the address programmed in the Hardware Status Page Address Register. When the Execlist Enable is set, the head pointer will be reported to the PP HW Status Page.</p>			
Programming Notes			
This command must not be executed from a Batch Buffer (Refer to the description of the HWS_PGA register).			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	07h MI_REPORT_HEAD
		Format:	OpCode
	22:0	Reserved	
		Project:	All
		Format:	MBZ



MI_REPORT_HEAD

MI_REPORT_HEAD		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	1	
<p>The MI_REPORT_HEAD command causes the Head Pointer value of the active ring buffer to be written to a cacheable (snooped) system memory location.</p> <p>When the Execlist Enable bit is reset: The location written is relative to the address programmed in the Hardware Status Page Address Register.</p>		
Programming Notes		
<p>This command must not be executed from a Batch Buffer (Refer to the description of the HWS_PGA register). When the Execlist Disable is clear, the head pointer will be reported to the PP HW Status Page.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode
		Default Value: 07h MI_REPORT_HEAD
	22:0	Reserved
		Project:
Format:		MBZ

MI_REPORT_HEAD

MI_REPORT_HEAD		
Project:	BDW	
Source:	RenderCS	
Length Bias:	1	
<p>The MI_REPORT_HEAD command causes the Head Pointer value of the active ring buffer to be written to a cacheable (snooped) system memory location. When Execlist Enable is set, the head pointer will be reported to the PP HW Status Page. The location written is relative to the address programmed in the Hardware Status Page Address Register.</p>		
Programming Notes		
This command must not be executed from a Batch Buffer. (Refer to the description of the HWS_PGA register.)		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 07h MI_REPORT_HEAD
		Format: OpCode
	22:0	Reserved
		Format: MBZ



MI_REPORT_HEAD

MI_REPORT_HEAD		
Project:	BDW	
Source:	VideoCS	
Length Bias:	1	
<p>The MI_REPORT_HEAD command causes the Head Pointer value of the ring buffer to be written to a cacheable (snooped) system memory location. When the Per-Process Virtual Address Space and Execlist Enable bits reset: The location written is relative to the address programmed in the Hardware Status Page Address Register. When the Execlist Enable is set, the head pointer will be reported to the PP HW Status Page.</p>		
Programming Notes		
This command must not be executed from a Batch Buffer (Refer to the description of the HWS_PGA register).		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 07h MI_REPORT_HEAD
		Format: OpCode
	22:0	Reserved
		Format: MBZ

MI_REPORT_PERF_COUNT

MI_REPORT_PERF_COUNT			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
<p>The MI_REPORT_PERF_COUNT command causes the GFX hardware to write out a snap-shot of performance counters to the address specified in this command along with constant ID field supplied and the time-stamp counter. This write is required to be treated as a cacheable write irrespective of GTT entry memory type. This command is specific to the render engine.</p>			
Programming Notes			
<p>This command is to be used for performance debug mode and can be inserted after events of interest (frequently before and after a 3DPRIMITIVE command). SW is entirely responsible for managing the ID field and addresses used by such a series of commands.</p>			
<p>GTT_SELECT must not be set to 1 (i.e. GGTT) when MI_REPORT_PERF_COUNT command is programmed in a non-privileged batch buffer. Refer to the "User Mode Privileged commands" Table in MI_BATCH_BUFFER_START command section for more details.</p>			
<p>All batch buffers in PPGTT are considered as Non-privileged.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	28h MI_REPORT_PERF_COUNT
22:6	Reserved		
	Format:	MBZ	
5:0	5:0	DWord Length	
		Format:	=n
	Total Length - 2		
	Value	Name	
	2h	Excludes DWord (0,1) [Default]	
1..2	63:6	Memory Address	
		Format:	GraphicsAddress[63:6]
		<p>This field specifies 64B aligned GFX MEM address where the chap counter values are reported. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47]</p>	
	Programming Notes		
	<p>This field is ignored if "Report to OABUFFER" bit is set.</p>		
	5	Reserved	

MI_REPORT_PERF_COUNT		
		Format: MBZ
	4	<p>Core Mode Enable</p> <p>Format: U1</p> <p>This bit is set then the address will be offset by the Core ID: If Core ID 0, then there is no offset. If Core ID 1, then the Memory is offset by the size of the data (64b).</p>
	3:1	<p>Reserved</p> <p>Format: MBZ</p>
	0	<p>Use Global GTT</p> <p>Format: Boolean</p> <p>This field when set (i.e. bit = 1) selects the GGTT for address translation. When this bit is 0 (default value), HW should use PGTT for address translation.</p>
3	31:0	<p>Report ID</p> <p>Format: U32</p> <p>This field specifies the ID provided by SW for a given report command. It can be tracked to use different flavors of these reports based on where in command-stream they are inserted. This field is reported only when Counter Select Field is 0.</p> <p style="text-align: center;">Programming Notes</p> <p>If a privilege access violation occurs, the REPORT ID field in the report generated by the next legitimate MI_REPORT_PERF_COUNT will be corrupted.</p>

MI_RS_CONTEXT

MI_RS_CONTEXT			
Project:	BDW		
Source:	RenderCS		
Length Bias:	1		
The MI_RS_CONTEXT command is used to force a resource streamer context save or restore.			
Programming Notes		Project	
This command must not be used/programmed in Execution List mode of scheduling.		BDW	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	0Fh MI_RS_CONTEXT
		Format:	OpCode
	22:1	Reserved	
		Format:	MBZ
	0	Resource Streamer Save	
		Format:	U1
This bit specifies whether the MI_RS_CONTEXT command will cause the resource streamer context to be saved or restored.			
Value		Name	Description
0h		Restore	Resource Streamer context is restored
1h	Save	Resource Streamer context is saved	

MI_RS_CONTROL

MI_RS_CONTROL	
Project:	BDW
Source:	RenderCS
Length Bias:	1
The MI_RS_CONTROL command is used to start or stop the Resource Streamer.	
Programming Notes	
<ul style="list-style-type: none"> • This command must be programmed only inside a Resource Streamer enabled batch buffer. • This command provides means to selectively disable or enable Resource Streamer for set of commands in a Resource Streamer enabled batch buffer • On re-enabling the Resource Streamer through this command, command streamer will start Resource Streamer on the next non-sync command of the batch buffer. • This command status is render context save/restored during context switching. • The scope of MI_RS_CONTROL is within the batch buffer it is programmed, it doesn't get carried to the following chained batch buffer or second level batch buffer. RS control status goes back to default mode of Resource Streamer Enabled on all batch buffer arbitration boundaries. Batch buffer arbitration boundaries includes calling a chained or a second level batch buffer through MI_BATCH_BUFFER_START command or terminating a batch buffer through MI_BATCH_BUFFER_END command. • Example: <ol style="list-style-type: none"> 1. MI_BATCH_START (Primary batch buffer with RS enable) 2. Command 1 --> CS starts RS 3. Command 2 4. : 5. MI_RS_CONTROL (stop option) -> RS will stop on this command, CS sets RS control status to STOP. 6. Command 3 7. MI_BATCH_START (2nd level batch with RS enable not set, RS control status gets reset to default status of START) 8. : 9. MI_BATCH_END (Second Level Batch End) 10. Command 4 --> CS starts RS here as RS control flag gets reset to START at step-7 11. MI_BATCH_BUFFER_END 	
Workaround	
<p>Workaround :</p> <p>Due to known HW issue "Resource Streamer Control" status of MI_RS_CONTROL command is not context save/restored across context switches. SW must ensure all pool allocations (3DSTATE_BINDING_TABLE_POOL_ALLOC, 3DSTATE_GATHER_POOL_ALLOC, 3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC) are disabled and no Resource Streamer specific commands are programmed when the "Resource Streamer Control" is programmed to "Stop".</p>	

MI_RS_CONTROL

Example:

```

.....
MI_RS_CONTROL (Stop Resource Streamer)
3DSTATE_BINDING_TABLE_POOL_ALLOC (Binding Table Pool Disable)
3DSTATE_GATHER_POOL_ALLOC (Gather Pool Disable)
3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC (Constant Buffer Pool Disable)
//Following Commands must not be programmed
//3DSTATE_BINDING_TABLE_EDIT_*
//3DSTATE_GATHER_CONSTANT_*
//3DSTATE_DX9_CONSTANTF_*
.....

```

```

.....
MI_RS_CONTROL (Start Resource Streamer)
3DSTATE_BINDING_TABLE_POOL_ALLOC (Binding Table Pool Enable)
3DSTATE_GATHER_POOL_ALLOC (Gather Pool Enable)
3DSTATE_DX9_CONSTANT_BUFFER_POOL_ALLOC (Constant Buffer Pool Enable)
.....

```

DWord	Bit	Description		
0	31:29	Command Type		
		Default Value: 0h MI_COMMAND		
		Format: OpCode		
	28:23	MI Command Opcode		
		Default Value: 06h MI_RS_CONTROL		
		Format: OpCode		
	22:1	Reserved		
		Format: MBZ		
	0	0	Resource Streamer Control	
			Format: U1	
This bit specifies whether the command is starting or stopping the Resource Streamer.				
Value			Name	Description
0h			Stop	Stop and disable the Resource Streamer
1h	Start	Start and enable the Resource Streamer		



MI_RS_STORE_DATA_IMM

MI_RS_STORE_DATA_IMM			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The MI_RS_STORE_DATA_IMM command requests a write of the DWord constant supplied in the packet to the specified Memory Address.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	2Bh
		Format:	OpCode
			MI_RS_STORE_DATA_IMM
	22	Reserved	
		Project:	BDW
			Format:
21	Reserved		
20:8	Reserved		
	Format:		MBZ
7:0	DWord Length		
	Default Value:	2h	
	Format:	=n Total Length - 2. Excludes DWord (0,1).	
1..2 Project: BDW	63:2	Destination Address	
		Project:	BDW
		Format:	GraphicsAddress[63:2]
	<p>This field specifies Bits 47:2 of the Address where the DWord will be stored. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p> <p>When render engine is PPGTT enabled this Address is translated using PPGTT, else GGTT is used for translation.</p>		
1	Reserved		
	Project:	BDW	
		Format:	MBZ

MI_RS_STORE_DATA_IMM				
0	0	<p>Core Mode Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td style="width: 30%;">BDW</td> </tr> </table> <p>If this bit is set then the address will be offset by the Core ID: If Core ID 0, then there is no offset If Core ID 1, then the Memory is offset by the size of the data.</p>	Project:	BDW
Project:	BDW			
3	31:0	<p>Data DWord 0</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location.</p>	Format:	U32
Format:	U32			



MI_SEMAPHORE_SIGNAL

MI_SEMAPHORE_SIGNAL			
Project:	BDW		
Source:	CommandStreamer		
Length Bias:	2		
Description		Project	
This command is used to signal the target engine stating the memory semaphore update occurrence to one of its contexts with Target Context ID . MI_SEMAPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command on BDW. MI_ATOMIC (non-posted) command will be programmed prior to this command to update the semaphore data in memory.		BDW	
Workaround		Project	
Workaround : Post-Sync operation bit must not be set when Target Engine Select is set to RCS.		BDW	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	1Bh MI_SEMAPHORE_SIGNAL
		Format:	OpCode
22	Reserved		
	Format:	MBZ	

MI_SEMAPHORE_SIGNAL			
21	Post-Sync Operation		
	Project:	BDW	
	Source:	RenderCS	
	Value	Name	Description
	0h	No Post Sync Operation	Command is executed as usual.
	1h	Post Sync Operation	MI_SEMAPHORE_SIGNAL command is executed as a pipelined PIPE_CONTROL flush command with Semaphore Signal as post sync operation. Flush completion only guarantees the workload prior to this command is pushed till Windower unit and completion of any outstanding flushes issued prior to this command.
	Programming Notes		
	<p>Any desired pipeline flush operation can be achieved by programming PIPE_CONTROL command prior to this command.</p> <p>When this bit is set Command Streamer sends a flush down the pipe and the atomic operation is saved as post sync operation. Command streamer goes on executing the following commands. Atomic operation saved as post sync operation is executed at some point later on completion of corresponding flush issued.</p> <p>When this bit is set atomic semaphore signal operation will be out of order with rest of the MI commands programmed in the ring buffer or batch buffer, it will be in order with respect to the post sync operations resulting due to PIPE_CONTROL command.</p>		
	Workaround		
	<p>Workaround :</p> <p>"Post Sync Operation" bit must not be set when MI_SEMAPHORE_SIGNAL command is programmed by GPGPU and MEDIA workloads (i.e when PIPELINE_SELECT command is set to GPGPU or MEDIA). This is to WA FFDOP CG issue, this WA need not be implemented when FF_DOP_CG is disabled via "Fixed Function DOP Clock Gate Disable" bit in RC_PSMI_CTRL register.</p> <p>Workaround :</p> <p>Post-Sync operation bit must not be set when Target Engine Select is set to RCS.</p>		
21	Reserved		
	Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS	
	Format:	MBZ	
20:19	Reserved		
	Format:	MBZ	

MI_SEMAPHORE_SIGNAL																												
	18	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ																						
	Project:	BDW																										
	Format:	MBZ																										
	17:15	Target Engine Select <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>This field selects the target engine to which SIGNAL will be send to.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 33%;">Value</th> <th style="width: 33%;">Name</th> <th style="width: 33%;">Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>RCS</td> <td></td> </tr> <tr> <td>1h</td> <td>VCS0</td> <td></td> </tr> <tr> <td>2h</td> <td>BCS</td> <td></td> </tr> <tr> <td>3h</td> <td>VECS</td> <td></td> </tr> <tr> <td>4h</td> <td>VCS1</td> <td></td> </tr> <tr> <td>5h</td> <td>Reserved</td> <td>BDW</td> </tr> <tr> <td>6h,7h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Project:	BDW	Value	Name	Project	0h	RCS		1h	VCS0		2h	BCS		3h	VECS		4h	VCS1		5h	Reserved	BDW	6h,7h	Reserved	
	Project:	BDW																										
	Value	Name	Project																									
	0h	RCS																										
	1h	VCS0																										
	2h	BCS																										
	3h	VECS																										
4h	VCS1																											
5h	Reserved	BDW																										
6h,7h	Reserved																											
14:8	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																											
7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td style="width: 75%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2. Excludes DWord (0,1).</p>	Default Value:	0h	Format:	=n																							
Default Value:	0h																											
Format:	=n																											
1	31:0 Target Context ID <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td> <p>In execlist based scheduling this field contains the Context ID corresponding to the context of the target engine that this command is signaling. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will re-fetch the data from memory or comparison if its context ID is same as this signaled Context ID. When execlists are enabled, Target engine on receiving this Context ID sends message to the SHIM if it doesn't have the context with the same Context ID running. Message send to SHIM carries the Context ID which will be looked at by UC for rescheduling the signaled Context ID. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will fetch data from memory for comparison on receiving signal irrespective of the context id received.</p> </td> </tr> <tr> <td> <p>In ring buffer mode of scheduling this field doesn't have any relevance.</p> </td> </tr> </tbody> </table>	Project:	BDW	Description	<p>In execlist based scheduling this field contains the Context ID corresponding to the context of the target engine that this command is signaling. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will re-fetch the data from memory or comparison if its context ID is same as this signaled Context ID. When execlists are enabled, Target engine on receiving this Context ID sends message to the SHIM if it doesn't have the context with the same Context ID running. Message send to SHIM carries the Context ID which will be looked at by UC for rescheduling the signaled Context ID. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will fetch data from memory for comparison on receiving signal irrespective of the context id received.</p>	<p>In ring buffer mode of scheduling this field doesn't have any relevance.</p>																						
Project:	BDW																											
Description																												
<p>In execlist based scheduling this field contains the Context ID corresponding to the context of the target engine that this command is signaling. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will re-fetch the data from memory or comparison if its context ID is same as this signaled Context ID. When execlists are enabled, Target engine on receiving this Context ID sends message to the SHIM if it doesn't have the context with the same Context ID running. Message send to SHIM carries the Context ID which will be looked at by UC for rescheduling the signaled Context ID. Target engine waiting on MI_SEMAPHORE_WAIT in signal mode will fetch data from memory for comparison on receiving signal irrespective of the context id received.</p>																												
<p>In ring buffer mode of scheduling this field doesn't have any relevance.</p>																												

MI_SEMAPHORE_WAIT

MI_SEMAPHORE_WAIT	
Project:	BDW
Source:	CommandStreamer
Length Bias:	2
Description	
<p>This command supports memory based Semaphore WAIT. Memory based semaphores will be used for synchronization between the Producer and the Consumer contexts. Producer and Consumer Contexts could be running on different engines or on the same engine inside GT. Running on the same engine is only possible when execlists are enabled. Producer Context implements a Signal and Consumer context implements a Wait. Command Streamer on parsing this command fetches data from the Semaphore Address mentioned in this command and compares it with the inline Semaphore Data Dword.</p> <ul style="list-style-type: none"> • If comparison passes, the command streamer moves to the next command. • When execlists are enabled, if comparison fails Command streamer switches out the context. Context switch can be inhibited by setting "Inhibit Synchronous Context Switch" in CTXT_SR_CTL register. • In ring buffer mode of scheduling or Execlist with "Inhibit Synchronous context Switch", if comparison fails, Command Streamer evaluates the Compare Operation based on the Wait Mode until the compare operation is true or Wait is canceled by SW. • Exec-List Scheduling: CS generates semaphore wait interrupt to the scheduler 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, preempt the context waiting on semaphore wait. <p>MI_SEMPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command on DevBDW.</p>	
Programming Notes	Source
<p>Render CS Only: SW must always program PIPE_CONTROL with "CS Stall" and "Render Target Cache Flush Enable" set prior to programming MI_SEMAPHORE_WAIT command for GPGPU workloads i.e when pipeline select is GPGPU via PIPELINE_SELECT command. This is required to achieve better GPGPU preemption latencies for certain programming sequences. If programming PIPE_CONTROL has performance implications then preemption latencies can be trade off against performance by not implementing this programming note.</p>	RenderCS
<p>Render CS Only: Ring Buffer Scheduling: CS doesn't generate semaphore wait interrupt to the scheduler when MI_SEMAPHORE_WAIT command is un-successful.</p>	RenderCS
<p>[Ring Buffer Mode Of scheduling] [BlitterCS, VideoCS, VideoEnhancementCS, VideoCS2: Command Streamers Only]: HW loses Page Directory (PPGTT) information on becoming IDLE. SW must always program the PD information following MI_SEMAPHORE_WAIT command. This will ensure Page Directory information gets reprogrammed after exiting IDLE flow triggered on MI_SEMAPHORE_WAIT command. Alternatively SW can disable IDLE flows on MI_SEMAPHORE_WAIT by setting "Semaphore Wait Event IDLE Message Disable"</p>	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS

MI_SEMAPHORE_WAIT				
bit in "BCS_ECOSKPD" register.				
Workaround				
Workaround : [All Command Streamers][Ring Buffer Mode of Scheduling]: MI_SEMAPHORE_WAIT command must be always programmed with "Wait Mode" set to "Polling Mode" Or MI_SEMAPHORE_WAIT command with "Wait Mode" set to "Polling Mode" can be programmed when "Semaphore Wait Event IDLE message Disable" bit in "RC_PSMI_CTRL" register is set to disable Idle messaging on unsuccessful MI_SEMPAHORE_WAIT.				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	1Ch MI_SEMAPHORE_WAIT	
		Format:	OpCode	
	22	Memory Type		
		This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit <i>must</i> be 1 if the Per Process GTT Enable bit is clear.		
		Value	Name	Description
		0h	Per Process Graphics Address	
1h		Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.	
21:18	Reserved			
	Format:	MBZ		
17	Reserved			
	Format:	MBZ		
16	Reserved			
	Project:	BDW		

MI_SEMAPHORE_WAIT																													
15	<p>Wait Mode This bit specifies the WAIT behavior when the semaphore comparison fails and before the context is switched out.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1h</td> <td>Polling Mode</td> <td>In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.</td> </tr> <tr> <td style="text-align: center;">0h</td> <td>Signal Mode</td> <td>In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.</td> </tr> </tbody> </table>		Value	Name	Description	1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.	0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																		
Value	Name	Description																											
1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.																											
0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																											
14:12	<p>Compare Operation This field specifies the operation that will be executed to create the result that will either allow the context to continue or wait.</p> <p>SAD = Semaphore Address Data SDD = Semaphore Data Dword</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>SAD_GREATER_THAN_SDD</td> <td>If Indirect fetched data is greater than inline data then continue.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td>SAD_GREATER_THAN_OR_EQUAL_SDD</td> <td>If Indirect fetched data is greater than or equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">2h</td> <td>SAD_LESS_THAN_SDD</td> <td>If Indirect fetched data is less than inline data then continue.</td> </tr> <tr> <td style="text-align: center;">3h</td> <td>SAD_LESS_THAN_OR_EQUAL_SDD</td> <td>If Indirect fetched data is less than or equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">4h</td> <td>SAD_EQUAL_SDD</td> <td>If Indirect fetched data is equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">5h</td> <td>SAD_NOT_EQUAL_SDD</td> <td>If Indirect fetched data is not equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">6h</td> <td>Reserved</td> <td></td> </tr> <tr> <td style="text-align: center;">7h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>		Value	Name	Description	0h	SAD_GREATER_THAN_SDD	If Indirect fetched data is greater than inline data then continue.	1h	SAD_GREATER_THAN_OR_EQUAL_SDD	If Indirect fetched data is greater than or equal to inline data then continue.	2h	SAD_LESS_THAN_SDD	If Indirect fetched data is less than inline data then continue.	3h	SAD_LESS_THAN_OR_EQUAL_SDD	If Indirect fetched data is less than or equal to inline data then continue.	4h	SAD_EQUAL_SDD	If Indirect fetched data is equal to inline data then continue.	5h	SAD_NOT_EQUAL_SDD	If Indirect fetched data is not equal to inline data then continue.	6h	Reserved		7h	Reserved	
Value	Name	Description																											
0h	SAD_GREATER_THAN_SDD	If Indirect fetched data is greater than inline data then continue.																											
1h	SAD_GREATER_THAN_OR_EQUAL_SDD	If Indirect fetched data is greater than or equal to inline data then continue.																											
2h	SAD_LESS_THAN_SDD	If Indirect fetched data is less than inline data then continue.																											
3h	SAD_LESS_THAN_OR_EQUAL_SDD	If Indirect fetched data is less than or equal to inline data then continue.																											
4h	SAD_EQUAL_SDD	If Indirect fetched data is equal to inline data then continue.																											
5h	SAD_NOT_EQUAL_SDD	If Indirect fetched data is not equal to inline data then continue.																											
6h	Reserved																												
7h	Reserved																												
11:8	<p>Reserved</p> <table border="1"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>		Format:	MBZ																									
Format:	MBZ																												
7:0	<p>DWord Length</p> <table border="1"> <tr> <td style="width: 30%;">Format:</td> <td>=n Total Length - 2. Excludes DWord (0,1)</td> </tr> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">2h</td> <td>[Default]</td> <td>BDW</td> </tr> </tbody> </table>		Format:	=n Total Length - 2. Excludes DWord (0,1)	Value	Name	Project	2h	[Default]	BDW																			
Format:	=n Total Length - 2. Excludes DWord (0,1)																												
Value	Name	Project																											
2h	[Default]	BDW																											



MI_SEMAPHORE_WAIT						
1	31:0	<p>Semaphore Data Dword</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table> <p>This Data dword is supplied by software to control execution of the command buffer. This value is used as part of the comparison to result in waiting or continuing in the command parser if enabled.</p>	Format:	U32		
Format:	U32					
2..3 Project: BDW	63:2	<p>Semaphore Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:2]</td> </tr> </table> <p>This field is the Graphics Memory Address of the 32-bit value for the semaphore. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form.</p>	Project:	BDW	Format:	GraphicsAddress[63:2]
	Project:	BDW				
	Format:	GraphicsAddress[63:2]				
1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					

MI_SEMAPHORE_WAIT

MI_SEMAPHORE_WAIT		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	2	
Description		
<p>This command supports memory based Semaphore WAIT. Memory based semaphores will be used for synchronization between the Producer and the Consumer contexts. Producer and Consumer Contexts could be running on different engines or on the same engine inside GT, same engine only possible when execlists are enabled. Producer Context implements a Signal and Consumer context implements a Wait.</p> <p>Command Streamer on parsing this command fetches data from the Semaphore Address mentioned in this command and compares it with the inline Semaphore Data Dword.</p> <ul style="list-style-type: none"> • If comparison passes, the command streamer moves to the next command. • When execlists are enabled, if comparison fails Command streamer switches out the context. Context switch can be inhibited by setting "Inhibit Synchronous Context Switch" in BCS_CTXT_SR_CTL register. • In ring buffer mode of scheduling or Execlist with "Inhibit Synchronous context Switch", if comparison fails, Command Streamer evaluates the Compare Operation based on the Wait Mode until the compare operation is true or Wait is canceled by SW. • BCS always generates an interrupt to the scheduler on encountering semaphore failure. 		
MI_SEMPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command.		
Programming Notes		
<p>[Ring Buffer Mode Of scheduling][Video CS, Video Enhancement CS, Blitter CS]: HW loses Page Directory (PPGTT) information on becoming IDLE. SW must always program the PD information following MI_SEMAPHORE_WAIT command. This will ensure Page Directory information gets reprogrammed after exiting IDLE flow triggered on MI_SEMAPHORE_WAIT command. Alternatively SW can disable IDLE flows on MI_SEMAPHORE_WAIT by setting "Semaphore Wait Event IDLE Message Disable" bit in "BCS_ECOSKPD" register.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
Default Value: 1Ch MI_SEMAPHORE_WAIT		
		Format: OpCode

MI_SEMAPHORE_WAIT																												
22	<p>Memory Type</p> <p>This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit must be '1' if the Per Process GTT Enable bit is clear.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Per Process Graphics Address</td> <td></td> </tr> <tr> <td>1h</td> <td>Global Graphics Address</td> <td>This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Per Process Graphics Address		1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.																		
Value	Name	Description																										
0h	Per Process Graphics Address																											
1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.																										
21:16	<p>Reserved</p> <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																											
15	<p>Wait Mode</p> <p>This bit specifies the WAIT behavior when the semaphore comparison fails and before the context is switched out.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Polling Mode</td> <td>In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.</td> </tr> <tr> <td>0h</td> <td>Signal Mode</td> <td>In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.</td> </tr> </tbody> </table>	Value	Name	Description	1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.	0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																		
Value	Name	Description																										
1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.																										
0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																										
14:12	<p>Compare Operation</p> <p>This field specifies the operation that will be executed to create the result that will either allow the context to continue or wait. If the below operation is TRUE then</p> <p>SAD = Semaphore Address Data SDD = Semaphore Data Dword</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>SAD > SDD</td> <td>If Indirect fetched data is greater than inline data then continue</td> </tr> <tr> <td>1h</td> <td>SAD >= SDD</td> <td>If Indirect fetched data is greater than or equal to inline data then continue.</td> </tr> <tr> <td>2h</td> <td>SAD < SDD</td> <td>If Indirect fetched data is less than inline data then continue.</td> </tr> <tr> <td>3h</td> <td>SAD <= SDD</td> <td>If Indirect fetched data is less than or equal to inline data then continue.</td> </tr> <tr> <td>4h</td> <td>SAD == SDD</td> <td>If Indirect fetched data is equal to inline data then continue.</td> </tr> <tr> <td>5h</td> <td>SAD != SDD</td> <td>If Indirect fetched data is not equal to inline data then continue.</td> </tr> <tr> <td>6h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>7h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	0h	SAD > SDD	If Indirect fetched data is greater than inline data then continue	1h	SAD >= SDD	If Indirect fetched data is greater than or equal to inline data then continue.	2h	SAD < SDD	If Indirect fetched data is less than inline data then continue.	3h	SAD <= SDD	If Indirect fetched data is less than or equal to inline data then continue.	4h	SAD == SDD	If Indirect fetched data is equal to inline data then continue.	5h	SAD != SDD	If Indirect fetched data is not equal to inline data then continue.	6h	Reserved		7h	Reserved	
Value	Name	Description																										
0h	SAD > SDD	If Indirect fetched data is greater than inline data then continue																										
1h	SAD >= SDD	If Indirect fetched data is greater than or equal to inline data then continue.																										
2h	SAD < SDD	If Indirect fetched data is less than inline data then continue.																										
3h	SAD <= SDD	If Indirect fetched data is less than or equal to inline data then continue.																										
4h	SAD == SDD	If Indirect fetched data is equal to inline data then continue.																										
5h	SAD != SDD	If Indirect fetched data is not equal to inline data then continue.																										
6h	Reserved																											
7h	Reserved																											

MI_SEMAPHORE_WAIT		
	11:8	Reserved Format: _____ MBZ
	7:0	DWord Length Default Value: _____ 2h Excludes DWord (0,1) Format: _____ =n Total Length - 2
	1	Semaphore Data Dword Format: _____ U32 Data dword to compare. The Data dword is supplied by software to control execution of the command buffer. If the data at Semaphore Address is greater than this dword, the execution of the command buffer continues.
2..3	63:48	Reserved Format: _____ MBZ
	47:2	Semaphore Address Format: _____ GraphicsAddress[47:2] This field is the Graphics Memory Address of the 32 bit value for the semaphore.
	1:0	Reserved Format: _____ MBZ



MI_SEMAPHORE_WAIT

MI_SEMAPHORE_WAIT		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
<p>This command supports memory based Semaphore WAIT. Memory based semaphores will be used for synchronization between the Producer and the Consumer contexts. Producer and Consumer Contexts could be running on different engines or on the same engine inside GT. Running on the same engine is only possible when execlists are enabled. Producer Context implements a Signal and Consumer context implements a Wait. Command Streamer on parsing this command fetches data from the Semaphore Address mentioned in this command and compares it with the inline Semaphore Data Dword.</p> <ul style="list-style-type: none"> • If comparison passes, the command streamer moves to the next command. • When execlists are enabled, if comparison fails Command streamer switches out the context. Context switch can be inhibited by setting "Inhibit Synchronous Context Switch" in CTXT_SR_CTL register • . In ring buffer mode of scheduling or Execlist with "Inhibit Synchronous context Switch", if comparison fails, Command Streamer evaluates the Compare Operation based on the Wait Mode until the compare operation is true or Wait is canceled by SW. • Exec-List Scheduling: CS generates semaphore wait interrupt to the scheduler 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. <p>MI_SEMPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command on DevBDW.</p>		
Programming Notes		
<p>Render CS Only: SW must always program PIPE_CONTROL with "CS Stall" and "Render Target Cache Flush Enable" set prior to programming MI_SEMAPHORE_WAIT command for GPGPU workloads i.e when pipeline select is GPGPU via PIPELINE_SELECT command. This is required to achieve better GPGPU preemption latencies for certain programming sequences. If programming PIPE_CONTROL has performance implications then preemption latencies can be trade off against performance by not implementing this programming note.</p>		
<p>Render CS Only: Ring Buffer Scheduling: CS doesn't generate semaphore wait interrupt to the scheduler when MI_SEMAPHORE_WAIT command is un-successful.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
Default Value: 1Ch MI_SEMAPHORE_WAIT		
Format: OpCode		

MI_SEMAPHORE_WAIT										
22	<p>Memory Type</p> <p>This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit <i>must</i> be 1 if the Per Process GTT Enable bit is clear.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>Per Process Graphics Address</td> <td></td> </tr> <tr> <td style="text-align: center;">1h</td> <td>Global Graphics Address</td> <td>This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Per Process Graphics Address		1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.
Value	Name	Description								
0h	Per Process Graphics Address									
1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.								
21:16	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
15	<p>Wait Mode</p> <p>This bit specifies the WAIT behavior when the semaphore comparison fails and before the context is switched out.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1h</td> <td>Polling Mode</td> <td>In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.</td> </tr> <tr> <td style="text-align: center;">0h</td> <td>Signal Mode</td> <td>In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.</td> </tr> </tbody> </table>	Value	Name	Description	1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.	0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.
Value	Name	Description								
1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.								
0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.								

MI_SEMAPHORE_WAIT																													
	14:12	<p>Compare Operation</p> <p>This field specifies the operation that will be executed to create the result that will either allow the context to continue or wait.</p> <p>SAD = Semaphore Address Data SDD = Semaphore Data Dword</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>SAD_GREATER_THAN_SDD</td> <td>If Indirect fetched data is greater than inline data then continue.</td> </tr> <tr> <td style="text-align: center;">1h</td> <td>SAD_GREATER_THAN_OR_EQUAL_SDD</td> <td>If Indirect fetched data is greater than or equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">2h</td> <td>SAD_LESS_THAN_SDD</td> <td>If Indirect fetched data is less than inline data then continue.</td> </tr> <tr> <td style="text-align: center;">3h</td> <td>SAD_LESS_THAN_OR_EQUAL_SDD</td> <td>If Indirect fetched data is less than or equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">4h</td> <td>SAD_EQUAL_SDD</td> <td>If Indirect fetched data is equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">5h</td> <td>SAD_NOT_EQUAL_SDD</td> <td>If Indirect fetched data is not equal to inline data then continue.</td> </tr> <tr> <td style="text-align: center;">6h</td> <td>Reserved</td> <td></td> </tr> <tr> <td style="text-align: center;">7h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Name	Description	0h	SAD_GREATER_THAN_SDD	If Indirect fetched data is greater than inline data then continue.	1h	SAD_GREATER_THAN_OR_EQUAL_SDD	If Indirect fetched data is greater than or equal to inline data then continue.	2h	SAD_LESS_THAN_SDD	If Indirect fetched data is less than inline data then continue.	3h	SAD_LESS_THAN_OR_EQUAL_SDD	If Indirect fetched data is less than or equal to inline data then continue.	4h	SAD_EQUAL_SDD	If Indirect fetched data is equal to inline data then continue.	5h	SAD_NOT_EQUAL_SDD	If Indirect fetched data is not equal to inline data then continue.	6h	Reserved		7h	Reserved	
	Value	Name	Description																										
	0h	SAD_GREATER_THAN_SDD	If Indirect fetched data is greater than inline data then continue.																										
	1h	SAD_GREATER_THAN_OR_EQUAL_SDD	If Indirect fetched data is greater than or equal to inline data then continue.																										
	2h	SAD_LESS_THAN_SDD	If Indirect fetched data is less than inline data then continue.																										
	3h	SAD_LESS_THAN_OR_EQUAL_SDD	If Indirect fetched data is less than or equal to inline data then continue.																										
	4h	SAD_EQUAL_SDD	If Indirect fetched data is equal to inline data then continue.																										
	5h	SAD_NOT_EQUAL_SDD	If Indirect fetched data is not equal to inline data then continue.																										
	6h	Reserved																											
	7h	Reserved																											
	11:8	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																												
	7:0	<p>DWord Length</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Default Value:</td> <td style="text-align: center;">2h</td> </tr> <tr> <td>Format:</td> <td style="text-align: center;">=n Total Length - 2. Excludes DWord (0,1)</td> </tr> </table>	Default Value:	2h	Format:	=n Total Length - 2. Excludes DWord (0,1)																							
Default Value:	2h																												
Format:	=n Total Length - 2. Excludes DWord (0,1)																												
1	31:0	<p>Semaphore Data Dword</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">Format:</td> <td style="text-align: center;">U32</td> </tr> </table> <p>Data dword to compare. The Data dword is supplied by software to control execution of the command buffer. If the data at Semaphore Address is greater than this dword, the execution of the command buffer continues.</p>	Format:	U32																									
Format:	U32																												
2	31:2	<p>Semaphore Address</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">GraphicsAddress[31:2]</td> </tr> </table> <p>This field is the Graphics Memory Address of the 32-bit value for the semaphore.</p>	Format:	GraphicsAddress[31:2]																									
	Format:	GraphicsAddress[31:2]																											
1:0	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ																										
Format:	MBZ																												
3	31:16	<p>Reserved</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																												

MI_SEMAPHORE_WAIT					
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center;">15:0</td> <td> <p>Semaphore Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p> </td> </tr> </table>	15:0	<p>Semaphore Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]
15:0	<p>Semaphore Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]		
Format:	GraphicsAddress[47:32]				



MI_SEMAPHORE_WAIT

MI_SEMAPHORE_WAIT

Project:	BDW
Source:	VideoCS
Length Bias:	2

This command supports memory based Semaphore WAIT. Memory based semaphores will be used for synchronization between the Producer and the Consumer contexts. Producer and Consumer Contexts could be running on different engines or on the same engine inside GT, same engine only possible when execlists are enabled. Producer Context implements a Signal and Consumer context implements a Wait.

Command Streamer on parsing this command fetches data from the Semaphore Address mentioned in this command and compares it with the inline Semaphore Data Dword.

- If comparison passes, the command streamer moves to the next command.
- When execlists are enabled, if comparison fails Command streamer switches out the context. Context switch can be inhibited by setting "Inhibit Synchronous Context Switch" in VCS_CTXT_SR_CTL register.
- In ring buffer mode of scheduling or Execlist with "Inhibit Synchronous context Switch", if comparison fails, Command Streamer evaluates the Compare Operation based on the Wait Mode until the compare operation is true or Wait is canceled by SW.
- VCS always generates an interrupt to the scheduler on encountering semaphore failure.

MI_SEMPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command on BDW.

Programming Notes

[Ring Buffer Mode Of scheduling][Video CS]: HW loses Page Directory (PPGTT) information on becoming IDLE. SW must always program the PD information following MI_SEMAPHORE_WAIT command. This will ensure Page Directory information gets reprogrammed after exiting IDLE flow triggered on MI_SEMAPHORE_WAIT command. Alternatively SW can disable IDLE flows on MI_SEMAPHORE_WAIT command by setting "Semaphore Wait Event IDLE Message Disable" bit in "VCS_ECOSKPD" register.

DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
		Default Value: 1Ch MI_SEMAPHORE_WAIT
	Format: OpCode	
22	Memory Type	
	This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit <i>must</i> be '1' if the Per Process GTT Enable bit is clear.	
	Value	Name
0h	Per Process Graphics Address	

MI_SEMAPHORE_WAIT																												
21:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																											
15	<p>Wait Mode</p> <p>This bit specifies the WAIT behavior when the semaphore comparison fails and before the context is switched out.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td>1h</td> <td>Polling Mode</td> <td>In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.</td> </tr> <tr> <td>0h</td> <td>Signal Mode</td> <td>In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.</td> </tr> </tbody> </table>	Value	Name	Description	1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.	0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																		
Value	Name	Description																										
1h	Polling Mode	In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.																										
0h	Signal Mode	In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.																										
14:12	<p>Compare Operation</p> <p>This field specifies the operation that will be executed to create the result that will either allow the context to continue or wait. If the below operation is TRUE then</p> <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 15%;">Name</th> <th style="width: 75%;">Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>SAD > SDD</td> <td>If Indirect fetched data is greater than inline data then continue.</td> </tr> <tr> <td>1h</td> <td>SAD >= SDD</td> <td>If Indirect fetched data is greater than or equal to inline data then continue.</td> </tr> <tr> <td>2h</td> <td>SAD < SDD</td> <td>If Indirect fetched data is less than inline data then continue.</td> </tr> <tr> <td>3h</td> <td>SAD <= SDD</td> <td>If Indirect fetched data is less than or equal to inline data then continue.</td> </tr> <tr> <td>4h</td> <td>SAD == SDD</td> <td>If Indirect fetched data is equal to inline data then continue.</td> </tr> <tr> <td>5h</td> <td>SAD != SDD</td> <td>If Indirect fetched data is not equal to inline data then continue.</td> </tr> <tr> <td>6h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>7h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>SAD = Semaphore Address Data SDD = Semaphore Data Dword</p>	Value	Name	Description	0h	SAD > SDD	If Indirect fetched data is greater than inline data then continue.	1h	SAD >= SDD	If Indirect fetched data is greater than or equal to inline data then continue.	2h	SAD < SDD	If Indirect fetched data is less than inline data then continue.	3h	SAD <= SDD	If Indirect fetched data is less than or equal to inline data then continue.	4h	SAD == SDD	If Indirect fetched data is equal to inline data then continue.	5h	SAD != SDD	If Indirect fetched data is not equal to inline data then continue.	6h	Reserved		7h	Reserved	
Value	Name	Description																										
0h	SAD > SDD	If Indirect fetched data is greater than inline data then continue.																										
1h	SAD >= SDD	If Indirect fetched data is greater than or equal to inline data then continue.																										
2h	SAD < SDD	If Indirect fetched data is less than inline data then continue.																										
3h	SAD <= SDD	If Indirect fetched data is less than or equal to inline data then continue.																										
4h	SAD == SDD	If Indirect fetched data is equal to inline data then continue.																										
5h	SAD != SDD	If Indirect fetched data is not equal to inline data then continue.																										
6h	Reserved																											
7h	Reserved																											
11:8	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ																									
Format:	MBZ																											
7:0	<p>DWord Length</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>2h Excludes DWord (0,1)</td> </tr> <tr> <td>Format:</td> <td>=n</td> </tr> </table> <p>Total Length - 2</p>	Default Value:	2h Excludes DWord (0,1)	Format:	=n																							
Default Value:	2h Excludes DWord (0,1)																											
Format:	=n																											

MI_SEMAPHORE_WAIT				
1	31:0	<p>Semaphore Data Dword</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">U32</td> </tr> </table> <p>Data dword to compare. The Data dword is supplied by software to control execution of the command buffer. If the data at Semaphore Address is greater than this dword, the execution of the command buffer continues.</p>	Format:	U32
		Format:	U32	
<p>Semaphore Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">GraphicsVirtualAddress[31:2]</td> </tr> </table> <p>This field is the Graphics Memory Address of the 32 bit value for the semaphore.</p>	Format:	GraphicsVirtualAddress[31:2]		
Format:	GraphicsVirtualAddress[31:2]			
2	1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
3	15:0	<p>Semaphore Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td style="width: 30%;">MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			

MI_SEMAPHORE_WAIT

MI_SEMAPHORE_WAIT			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	2		
<p>This command supports memory based Semaphore WAIT. Memory based semaphores will be used for synchronization between the Producer and the Consumer contexts. Producer and Consumer Contexts could be running on different engines or on the same engine inside GT, same engine only possible when execlists are enabled. Producer Context implements a Signal and Consumer context implements a Wait. Command Streamer on parsing this command fetches data from the Semaphore Address mentioned in this command and compares it with the inline Semaphore Data Dword.</p> <ul style="list-style-type: none"> • If comparison passes, the command streamer moves to the next command. • When execlists are enabled, if comparison fails Command streamer switches out the context. Context switch can be inhibited by setting "Inhibit Synchronous Context Switch" in VECS_CTXT_SR_CTL register. • In ring buffer mode of scheduling or execlist with "Inhibit Synchronous context Switch", if comparison fails, Command Streamer evaluates the Compare Operation based on the Wait Mode until the compare operation is true or Wait is canceled by SW. • VECS always generates an interrupt to the scheduler on encountering semaphore failure. <p>MI_SEMPHORE_SIGNAL and MI_SEMAPHORE_WAIT together replace the MI_SEMAPHORE_MBOX command on BDW.</p>			
Programming Notes			
<p>[Ring Buffer Mode Of scheduling][Video Enhancement CS]: HW loses Page Directory (PPGTT) information on becoming IDLE. SW must always program the PD information following MI_SEMAPHORE_WAIT command. This will ensure Page Directory information gets reprogrammed after exiting IDLE flow triggered on MI_SEMAPHORE_WAIT command. Alternatively SW can disable IDLE flows on MI_SEMAPHORE_WAIT by setting "Semaphore Wait Event IDLE Message Disable" bit in "VECS_ECOSKPD" register.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	1Ch MI_SEMAPHORE_WAIT
		Format:	OpCode
22	Memory Type		
	<p>This bit will be ignored and treated as if clear when executing from a non-privileged batch buffer. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit <i>must</i> be '1' if the Per Process GTT Enable bit is clear.</p>		
	Value	Name	
0h	Per Process Graphics Address		

MI_SEMAPHORE_WAIT		
21:16	Reserved	
	Format:	MBZ
15	Wait Mode	
	This bit specifies the WAIT behavior when the semaphore comparison fails and before the context is switched out.	
	Value	Name Description
	1h	Polling Mode In this mode HW periodically reads the semaphore data from memory for comparison until it is context switched out. Periodicity will be mentioned in a SEMA_WAIT_POLL register.
0h	Signal Mode In this mode HW will reacquire the semaphore data from memory on receiving SIGNAL with the same Context ID. In ring buffer mode of scheduling Context ID associated with SIGNAL is ignored and always treated as a match.	
14:12	Compare Operation	
	This field specifies the operation that will be executed to create the result that will either allow the context to continue or wait. If the below operation is TRUE then	
	Value	Name Description
	0h	SAD > SDD If Indirect fetched data is greater than inline data then continue.
	1h	SAD >= SDD If Indirect fetched data is greater than or equal to inline data then continue.
	2h	SAD < SDD If Indirect fetched data is less than inline data then continue.
	3h	SAD <= SDD If Indirect fetched data is less than or equal to inline data then continue.
	4h	SAD == SDD If Indirect fetched data is equal to inline data then continue.
	5h	SAD != SDD If Indirect fetched data is not equal to inline data then continue.
	6h	Reserved
	7h	Reserved
	Programming Notes	
SAD = Semaphore Address Data SDD = Semaphore Data Dword		
11:8	Reserved	
	Format:	MBZ
7:0	DWord Length	
	Default Value:	2h Excludes DWord (0,1)
	Format:	=n
Total Length - 2		

MI_SEMAPHORE_WAIT				
1	31:0	<p>Semaphore Data Dword</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table> <p>Data dword to compare. The Data dword is supplied by software to control execution of the command buffer. If the data at Semaphore Address is greater than this dword, the execution of the command buffer continues.</p>	Format:	U32
		Format:	U32	
<p>Semaphore Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsVirtualAddress[31:2]</td> </tr> </table> <p>This field is the Graphics Memory Address of the 32 bit value for the semaphore.</p>	Format:	GraphicsVirtualAddress[31:2]		
Format:	GraphicsVirtualAddress[31:2]			
	1:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
3	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Semaphore Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			



MI_SET_CONTEXT

MI_SET_CONTEXT	
Project:	BDW
Source:	RenderCS
Length Bias:	2

The MI_SET_CONTEXT command is used to specify the logical context associated with the hardware context. A logical context is an area in memory used to store hardware context information, and the context is referenced via a 2KB-aligned pointer. If the (new) logical context is different (i.e., at a different memory address), the device saves the current HW context values to the current logical context address, and then restores (loads) the new logical context by reading the context from the new address and loading it into the hardware context state. If the logical context address specified in this command matches the current logical context address, this command is effectively treated as a NOOP. Specific to the Render command stream only. This command also includes some controls over the context save/restore process. The Force Restore bit can be used to refresh the on-chip device state from the same memory address if the indirect state buffers have been modified. The Restore Inhibit bit can be used to prevent the new context from being loaded at all. This must be used to prevent an uninitialized context from being loaded. Once software has initialized a context (by setting all state variables to initial values via commands), the context can then be stored and restored normally. When switching from a generic media context to a 3D context, the generic media state must be cleared via the Generic Media State Clear bit 16 in PIPE_CONTROL (or bit 4 in MI_FLUSH) before saving 3D context. MI_SET_CONTEXT commands are permitted only within a ring buffer (not within a batch buffer).

All context is saved and restored from a GGTT space.

This command does not initiate any interrupt due to context switch of any kind and does not support any workaround batch buffer or indirect context offset feature.

Programming Notes

This command is legal only if Execlist Enable in the GFX_MODE register is reset. Otherwise, execlists must be used to switch context in lieu of MI_SET_CONTEXT.

For ring buffer mode, the first 128B(2 cache lines) of the context image are saved as zeros.

This command needs to be always followed by a single MI_NOOP instruction to workaround a silicon issue.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	18h MI_SET_CONTEXT
		Format:	OpCode
	22:8	Reserved	
		Format:	MBZ
	7:0	DWord Length	
		Default Value:	0h
		Format:	=n Total Length - 2. Excludes DWord (0,1).

MI_SET_CONTEXT								
1	31:12	Logical Context Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[31:12]LogicalContext</td> </tr> </table> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <th style="text-align: center;">Description</th> </tr> <tr> <td> <p>This field contains the 4KB-aligned graphics memory address of the Logical Context that is to be loaded into the hardware context. If this address is equal to the CCID register associated with the current ring, no load will occur. Prior to loading this new context, the device will save the existing context as required. After the context switch operation completes, this address will be loaded into the associated CCID register.</p> <p>This field needs to be 4KB aligned virtual address.</p> </td> </tr> </table>	Project:	BDW	Format:	GraphicsAddress[31:12]LogicalContext	Description	<p>This field contains the 4KB-aligned graphics memory address of the Logical Context that is to be loaded into the hardware context. If this address is equal to the CCID register associated with the current ring, no load will occur. Prior to loading this new context, the device will save the existing context as required. After the context switch operation completes, this address will be loaded into the associated CCID register.</p> <p>This field needs to be 4KB aligned virtual address.</p>
	Project:	BDW						
	Format:	GraphicsAddress[31:12]LogicalContext						
	Description							
	<p>This field contains the 4KB-aligned graphics memory address of the Logical Context that is to be loaded into the hardware context. If this address is equal to the CCID register associated with the current ring, no load will occur. Prior to loading this new context, the device will save the existing context as required. After the context switch operation completes, this address will be loaded into the associated CCID register.</p> <p>This field needs to be 4KB aligned virtual address.</p>							
	11:10	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ						
9	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ			
Project:	BDW							
Format:	MBZ							
8	Reserved, Must be 1 <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>Must Be One</td> </tr> </table>	Format:	Must Be One					
Format:	Must Be One							
7:5	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
Format:	MBZ							
4	Core Mode Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set the Context Image will be offset based off the Core ID: If Core ID 0, no offset If Core ID 1, 36KB Offset</p>	Project:	BDW	Format:	Enable			
Project:	BDW							
Format:	Enable							
3	Resource Streamer State Save Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set, the resource streamer state identified in the Logical Context Data section of the Memory Data Formats chapter is saved as part of switching away from this logical context. This bit will be stored in the associated CCID register to control the context save operation when switching away from this context (as part of a subsequent MI_SET_CONTEXT command).</p>	Project:	BDW	Format:	Enable			
Project:	BDW							
Format:	Enable							

MI_SET_CONTEXT					
2	<p>Resource Streamer State Restore Enable</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If set, the resource streamer state identified in the Logical Context Data section of the Memory Data Formats chapter is loaded (or restored) as part of switching to this logical context. This bit affects the switch (if required) to the context specified in Logical Context Address. This bit will also be stored in the associated CCID register to control a subsequent context save operation when switching to this context (as part of a subsequent ring buffer switch).</p>	Project:	BDW	Format:	Enable
Project:	BDW				
Format:	Enable				
1	<p>Force Restore</p> <p>When switching to this logical context a comparison between Logical Context Address and the contents of the CCID register is performed. Normally, matching addresses prevent a context restore from occurring; however, when this bit is set a context restore is forced to occur. This bit cannot be set with Restore Inhibit. Note: This bit is not saved in the associated CCID register. It only affects the processing of this command.</p>				
0	<p>Restore Inhibit</p> <p>If set, the restore of the HW context from the logical context specified by Logical Context Address is inhibited (i.e., the existing HW context values are maintained). This bit must be used to prevent the loading of an uninitialized logical context. If clear, the context switch proceeds normally. This bit cannot be set with Force Restore. Note: This bit is not saved in the associated CCID register. It only affects the processing of this command.</p>				

MI_SET_PREDICATE

MI_SET_PREDICATE			
Project:	BDW		
Source:	RenderCS		
Length Bias:	1		
Description			
<p>This command sets the Predication Check for the subsequent commands in the command buffer except for MI_SET_PREDICATE itself. Render Command Streamer NOOPs the following commands based on the PREDICATE_ENABLE from MI_SET_PREDICATE, MI_SET_PREDICATE_RESULT and MI_SET_PREDICATE_RESULT_2 status. Resource Streamer doesn't take any action of parsing MI_SET_PREDICATE, this command is similar to any other command which is not meant for resource streamer.</p> <p>Executing MI_SET_PREDICATE command sets PREDICATE_ENABLE bits in MI_MODE register, MI_MODE register gets render context save restored.</p>			
Programming Notes			
<ul style="list-style-type: none"> MI_SET_PREDICATE predication scope must be confined within a Batch Buffer to set of commands. MI_SET_PREDICATE with Predicate Enable Must always have a corresponding MI_SET_PREDICATE with Predicate Disable within the same Batch Buffer. MI_ARB_CHK command must be programmed outside the Predication Scope of MI_SET_PREDICATE. MI_SET_PREDICATE Predication Scope must not involve any RC6 triggering events. 			
<p>Only the following command(s) can be programmed between the MI_SET_PREDICATE command enabled for predication: 3DSTATE_URB_VS 3DSTATE_URB_HS 3DSTATE_URB_DS 3DSTATE_URB_GS 3DSTATE_PUSH_CONSTANT_ALLOC_VS 3DSTATE_PUSH_CONSTANT_ALLOC_HS 3DSTATE_PUSH_CONSTANT_ALLOC_DS 3DSTATE_PUSH_CONSTANT_ALLOC_GS 3DSTATE_PUSH_CONSTANT_ALLOC_PS MI_LOAD_REGISTER_IMM MEDIA_VFE_STATE MEDIA_OBJECT MEDIA_OBJECT_WALKER MEDIA_INTERFACE_DESCRIPTOR_LOAD 3DSTATE_WM_HZ_OP</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	01h MI_SET_PREDICATE
		Format:	OpCode
22:4	Reserved		
	Project:	BDW	
	Format:	MBZ	

MI_SET_PREDICATE			
3:0	PREDICATE ENABLE		
	Project:	BDW	
	This field sets the predication logic in render command streamer when parsed. Predicate Disable is the default mode of operation.		
	Value	Name	Description
	0h	NOOP Never	Predication is Disabled and RCS will process commands as usual.
	1h	NOOP on Result2 clear	Following Commands will be NOOPED by RCS only if the MI_PREDICATE_RESULT_2 is clear.
	2h	NOOP on Result2 set	Following Commands will be NOOPED by RCS only if the MI_PREDICATE_RESULT_2 is set.
	3h	NOOP on Result clear	Following Commands will be NOOPED by RCS only if the MI_PREDICATE_RESULT is clear.
	4h	NOOP on Result set	Following Commands will be NOOPED by RCS only if the MI_PREDICATE_RESULT is set.
	5h	Execute when one slice enabled.	Following Commands will be Executed by RCS only when one slice is enabled.
	6h	Execute when two slices are enabled.	Following Commands will be Executed by RCS only when two slices are enabled.
	7h	Execute when three slices are enabled.	Following Commands will be Executed by RCS only when all the three slices are enabled.
	8h-Ah	Reserved	
	Bh, Ch	Reserved	
Dh, Eh	Reserved		
Fh	NOOP Always	Following Commands will be NOOPED by RCS unconditionally.	

MI_STORE_DATA_IMM

MI_STORE_DATA_IMM			
Project:	BDW		
Source:	VideoCS		
Length Bias:	2		
<p>The MI_STORE_DATA_IMM command requests a write of the QWord or DWord constant supplied in the packet to the specified Memory Address. This command also supports writing to consecutive dword or qword memory locations from the starting address. As the write targets a System Memory Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p>			
Programming Notes			
<p>This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll un-cached memory or device registers).</p>			
<p>This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations.</p>			
<p>This command should not be used within a non_privilege batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	20h MI_STORE_DATA_IMM
		Format:	OpCode
	22	Use Global GTT	
		Format:	U1
	<p>If set, this command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch or ring buffer. If clear, the PPGTT will be used. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit must be '1' if the Per Process GTT Enable bit is clear.</p>		

		MI_STORE_DATA_IMM	
	21	Store Qword	
		Format:	U1
		Value	Name
		Description	
	0h	Store Dword	If set, this command generates dword writes to memory. Number of dwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x' dword writes to memory.
	1h	Store Qword	If set, this command generates Qword writes to memory, two 'Data Dword' are paired to form a Qword. Number of qwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x/2' qword writes to memory.
	20:10	Reserved	
		Format:	MBZ
	9:0	DWord Length	
		Default Value:	0h Excludes DWord (0,1)
		Format:	=n Total Length - 2
1..2	63:2	Destination Address	
		Format:	GraphicsAddress[63:2]
		This field specifies the 4GB aligned base address within the host's 64-bit virtual address space. As the store address must be DWord-aligned, Bits 1:0 of that address MBZ. This address must be 8B aligned if "Store Qword" is enabled. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].	
	1:0	Reserved	
		Format:	MBZ
3	31:0	Data DWord 0	
		Format:	U32 FormatDesc
		This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).	
4	31:0	Data DWord 1	
		Format:	U32 FormatDesc
		This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).	

MI_STORE_DATA_IMM

MI_STORE_DATA_IMM			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	2		
<p>The MI_STORE_DATA_IMM command requests a write of the QWord or DWord constant supplied in the packet to the specified Memory Address. This command also supports writing to consecutive dword or qword memory locations from the starting address. As the write targets a System Memory Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p>			
Programming Notes			
<p>This command should not be used within a "non_privilege" batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or "privilege" batch buffers. If used within a non-privilege batch buffer, Use Global GTT must be clear. This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll un-cached memory or device registers). This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	20h MI_STORE_DATA_IMM
		Format:	OpCode
22	Use Global GTT		
	Project:	All	
	Format:	U1	
	<p>If set, this command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT will be used. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit must be '1' if the Per Process GTT Enable bit is clear.</p>		

MI_STORE_DATA_IMM											
	21	Store Qword									
		Project: BDW									
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Value</th> <th style="width: 10%;">Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Store Dword</td> <td>If set, this command generates dword writes to memory. Number of dwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x' dword writes to memory.</td> </tr> <tr> <td>1h</td> <td>Store Qword</td> <td>If set, this command generates Qword writes to memory, two 'Data Dword' are paired to form a Qword. Number of qwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x/2' qword writes to memory.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Store Dword	If set, this command generates dword writes to memory. Number of dwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x' dword writes to memory.	1h	Store Qword	If set, this command generates Qword writes to memory, two 'Data Dword' are paired to form a Qword. Number of qwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x/2' qword writes to memory.
		Value	Name	Description							
0h	Store Dword	If set, this command generates dword writes to memory. Number of dwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x' dword writes to memory.									
1h	Store Qword	If set, this command generates Qword writes to memory, two 'Data Dword' are paired to form a Qword. Number of qwords generated depends upon the number of 'Data Dword' programmed in the command. If 'x' number of data dwords are programmed in the command it results in 'x/2' qword writes to memory.									
	20:10	Reserved									
		Project: All									
		Format: MBZ									
	9:0	DWord Length									
		Default Value: 0h Excludes DWord (0,1)									
		Format: =n Total Length - 2									
1	31:2	Address									
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;">Format:</td> <td>GraphicsAddress[31:2]U32(2)</td> </tr> </table> <p>This field specifies Bits 31:2 of the Address where the DWord will be stored. As the store address must be DWord-aligned, Bits 1:0 of that address MBZ. This address must be 8B aligned for a store "QW" command.</p>	Format:	GraphicsAddress[31:2]U32(2)								
Format:	GraphicsAddress[31:2]U32(2)										
	1:0	Reserved									
	Format: MBZ										
2	31:16	Reserved									
	Format: MBZ										
	15:0	Address High									
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;">Format:</td> <td>GraphicsAddress[47:32]U16</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Format:	GraphicsAddress[47:32]U16								
Format:	GraphicsAddress[47:32]U16										
3	31:0	Data DWord 0									
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).</p>	Format:	U32								
Format:	U32										

MI_STORE_DATA_IMM				
4	31:0	<p>Data DWord 1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Format:	U32
Format:	U32			



MI_STORE_DATA_IMM

MI_STORE_DATA_IMM										
Project:	BDW									
Source:	BlitterCS									
Length Bias:	2									
Description										
<p>The MI_STORE_DATA_IMM command requests a write of the QWord constant supplied in the packet to the specified Memory Address. As the write targets a System Memory Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p> <p>This command supports writing to multiple consecutive dword or qword memory locations from the starting address.</p>										
Programming Notes										
<p>This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll un-cached memory or device registers). However, the cacheable nature of the transaction is determined by the setting of the "mapping type" in the GTT entry. This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations. All writes to memory generated using this command are expected to finish in order.</p> <p>This command should not be used within a "non_privilege" batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation.</p>										
DWord	Bit	Description								
0	31:29	Command Type Default Value: 0h MI_COMMAND								
		MI Command Opcode Default Value: 20h MI_STORE_DATA_IMM								
	22	Use Global GTT Project: All This bit must be '1' if the Per Process GTT Enable bit is clear. <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Per Process Graphics Address</td> <td></td> </tr> <tr> <td>1h</td> <td>Global Graphics Address</td> <td>This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</td> </tr> </tbody> </table>	Value	Name	Description	0h	Per Process Graphics Address		1h	Global Graphics Address
Value	Name	Description								
0h	Per Process Graphics Address									
1h	Global Graphics Address	This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.								

		MI_STORE_DATA_IMM		
	21	Store Qword		
		Project: BDW		
		Value	Name Description	
		0h	Store Dword	If set, this command generates dword writes to memory. Number of dwords generated depends upon the number of "Data Dword" programmed in the command. If 'x' number of data dwords are programmed in the command it results in "x" dword writes to memory.
		1h	Store Qword	If set, this command generates Qword writes to memory, two "Data Dword" are paired to form a Qword. Number of qwords generated depends upon the number of "Data Dword" programmed in the command. If 'x' number of data dwords are programmed in the command it results in "x/2" qword writes to memory.
	20:10	Reserved		
		Project: All		
		Format: MBZ		
	9:0	DWord Length		
		Default Value:	2h Excludes DWord (0,1) = 2 for DWord, 3 for QWord	
Format:		=n Total Length - 2		
1 Project: BDW	31:2	Address		
		Project: BDW		
		Format: GraphicsAddress[31:2]U32(2)		
	This field specifies Bits 31:2 of the Address where the DWord will be stored. As the store address must be DWord-aligned, Bits 1:0 of that address MBZ. This address must be 8B aligned for a store "QW" command.			
	1	Reserved		
Project: BDW				
Format: MBZ				
0	Core Mode Enable			
	Project: BDW			
	Format: U1			
This bit is set then the address will be offset by the Core ID: If Core ID 0, then there is no offset. If Core ID 1, then the Memory is offset by the size of the data(32b or 64b based off number of DW length).				
2 Project: BDW	31:16	Reserved		
		Project: BDW		
		Format: MBZ		



MI_STORE_DATA_IMM						
	15:0	<p>Address High</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]U32(2)</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]U32(2)
Project:	BDW					
Format:	GraphicsAddress[47:32]U32(2)					
3	31:0	<p>Data DWord 0</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).</p>	Project:	All	Format:	U32
Project:	All					
Format:	U32					
4	31:0	<p>Data DWord 1</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Project:	All	Format:	U32
Project:	All					
Format:	U32					

MI_STORE_DATA_IMM

MI_STORE_DATA_IMM		
Project:	BDW	
Source:	RenderCS	
Length Bias:	2	
Description		
<p>The MI_STORE_DATA_IMM command requests a write of the QWord constant supplied in the packet to the specified Memory Address. As the write targets a System Memory Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p> <p>This command supports writing to multiple consecutive dwords or qwords memory locations from the starting address.</p>		
Programming Notes		
<ul style="list-style-type: none"> This command should not be used within a "non-privilege" batch buffer to access global virtual space, doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or privilege batch buffers to access global virtual space. This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll un-cached memory or device registers). This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete eventually, there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations. <p>Number of consecutive dwords or qwords programmed must be restricted such that the DWord Length doesn't exceed 0x3FE, i.e single command supports updating 1021 consecutive dword locations or 510 qword locations.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
Default Value: 20h MI_STORE_DATA_IMM		
Format: OpCode		



MI_STORE_DATA_IMM																	
	22	Use Global GTT	Project: All	Format: Boolean	<p style="text-align: center;">Description</p> <p>If set, this command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer. If clear, the PPGTT will be used. It is allowed for this bit to be clear when executing this command from a privileged (secure) batch buffer. This bit must be '1' if the Per Process GTT Enable bit is clear.</p>												
	21	Store Qword	Project: BDW	Format: Boolean	<p>If set, this command generates Qword writes to memory, two "Data Dword" are paired to form a Qword. Number of qwords generated depends upon the number of "Data Dword" programmed in the command. If 'x' number of "Data Dwords" are programmed in this command it results in "x/2" qword writes to memory. If reset this command generates Dwords writes to memory. Number of dwords generated depends upon the number of "Data Dword" programmed in the command. If 'x' number of "Data Dwords" are programmed in this command it results in "x" dword writes to memory.</p>												
	20:10	Reserved	Format: MBZ														
	9:0	DWord Length	Project: BDW	Format: =n Total Length - 2. Excludes DWord (0,1)													
						<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>2h</td> <td>Store Dword [Default]</td> </tr> <tr> <td>3h</td> <td>Store Qword</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td>DWord Length programmed must not exceed 0x3FE.</td> <td></td> </tr> <tr> <td>If RS is enabled in the batch buffer, then the value of this field must not exceed 0x3F.</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	2h	Store Dword [Default]	3h	Store Qword	Programming Notes	Project	DWord Length programmed must not exceed 0x3FE.		If RS is enabled in the batch buffer, then the value of this field must not exceed 0x3F.
Value	Name																
2h	Store Dword [Default]																
3h	Store Qword																
Programming Notes	Project																
DWord Length programmed must not exceed 0x3FE.																	
If RS is enabled in the batch buffer, then the value of this field must not exceed 0x3F.	BDW																
1..2 Project: BDW	63:48	Reserved	Project: BDW	Format: MBZ													

MI_STORE_DATA_IMM						
	47:2	<p>Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:2]</td> </tr> </table> <p>This field specifies Bits 47:2 of the Address where the DWord will be stored. As the store address must be DWord-aligned, Bits 1:0 of that address MBZ. This address must be 8B aligned for a store "QW" command.</p>	Project:	BDW	Format:	GraphicsAddress[47:2]
	Project:	BDW				
	Format:	GraphicsAddress[47:2]				
1	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:	BDW					
Format:	MBZ					
0	<p>Core Mode Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>This bit is set then the address will be offset by the Core ID: If Core ID 0, then there is no offset If Core ID 1, then the Memory is offset by the size of the data(32b or 64b based off number of DW length).</p>	Project:	BDW	Format:	U1	
Project:	BDW					
Format:	U1					
3	31:0	<p>Data DWord 0</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).</p>	Format:	U32		
Format:	U32					
4	31:0	<p>Data DWord 1</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Format:	U32		
Format:	U32					



MI_STORE_DATA_INDEX

MI_STORE_DATA_INDEX			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	2		
<p>The MI_STORE_DATA_INDEX command requests a write of the data constant supplied in the packet to the specified offset from the System Address defined by the Hardware Status Page Address Register. As the write targets a System Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p>			
Programming Notes			
<ul style="list-style-type: none"> Use of this command with an invalid or uninitialized value in the Hardware Status Page Address Register is UNDEFINED. This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll uncached memory or device registers). This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations. 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	21h MI_STORE_DATA_INDEX
		Format:	OpCode
	22	Reserved	
		Project:	All
		Format:	MBZ
	21	Use Per-Process Hardware Status Page	
		Project:	BDW
	<p>If this bit is set, this command will index into the per-process hardware status page at offset 0K from the LRCA. If clear, the Global Hardware Status Page will be indexed. This bit must be '0' if the Execlist Enable bit is clear.</p>		
20:8	Reserved		
	Project:	All	
	Format:	MBZ	

MI_STORE_DATA_INDEX											
	7:0	<p>DWord Length</p> <table border="1"> <tr> <td>Default Value:</td> <td>0h Excludes DWord (0,1) = 2 for QWord</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table>	Default Value:	0h Excludes DWord (0,1) = 2 for QWord	Project:	All	Format:	=n Total Length - 2			
Default Value:	0h Excludes DWord (0,1) = 2 for QWord										
Project:	All										
Format:	=n Total Length - 2										
1	31:12	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ					
		Project:	All								
	Format:	MBZ									
	11:2	<p>Offset</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U10 Zero-based DWord offset into the HW status page</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[11:2]U32</td> </tr> </table> <p>This field specifies the offset (into the hardware status page) to which the data will be written. Note that the first few DWords of this status page are reserved for special-purpose data storage - targeting these reserved locations via this command is UNDEFINED. For a QWord write, the offset is valid down to bit 3 only.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[16, 1023]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U10 Zero-based DWord offset into the HW status page	Format:	GraphicsAddress[11:2]U32	Value	Name	[16, 1023]
Project:	All										
Format:	U10 Zero-based DWord offset into the HW status page										
Format:	GraphicsAddress[11:2]U32										
Value	Name										
[16, 1023]											
1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All										
Format:	MBZ										
2	31:0	<p>Data DWord 0</p> <table border="1"> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Format:	U32							
Format:	U32										
3	31:0	<p>Data Word 1</p> <table border="1"> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Format:	U32							
Format:	U32										



MI_STORE_DATA_INDEX

MI_STORE_DATA_INDEX		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	2	
<p>The MI_STORE_DATA_INDEX command requests a write of the data constant supplied in the packet to the specified offset from the System Address defined by the Hardware Status Page Address Register. As the write targets a System Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).</p>		
Programming Notes		
<p>Use of this command with an invalid or uninitialized value in the Hardware Status Page Address Register is UNDEFINED. This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll uncached memory or device registers). This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations.</p>		
DWord	Bit	Description
0	31:29	Command Type Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode Default Value: 21h MI_STORE_DATA_INDEX
	22	Reserved Project: All Format: MBZ
	21	Use Per-Process Hardware Status Page Project: BDW If this bit is set, this command will index into the per-process hardware status page at offset 0K from the LRCA. If clear, the Global Hardware Status Page will be indexed. This bit must be '0' if the Execlist Enable bit is clear.
	20:8	Reserved Project: All Format: MBZ
	7:0	DWord Length Default Value: 1h Excludes DWord (0,1) = 1 for DWord, 2 for QWord Format: =n Total Length - 2
1	31:12	Reserved Project: All Format: MBZ

MI_STORE_DATA_INDEX												
	11:2	<p>Offset</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U10 zero-based DWord offset into the HW status page.</td> </tr> <tr> <td>Format:</td> <td>HardwareStatusPageOffset[11:2]U32</td> </tr> </table> <p>This field specifies the offset (into the hardware status page) to which the data will be written. Note that the first few DWords of this status page are reserved for special-purpose data storage - targeting these reserved locations via this command is UNDEFINED. This address must be 8B aligned for a store "QW" command.</p> <table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>[16, 1023]</td> <td></td> </tr> </tbody> </table>	Project:	All	Format:	U10 zero-based DWord offset into the HW status page.	Format:	HardwareStatusPageOffset[11:2]U32	Value	Name	[16, 1023]	
	Project:	All										
Format:	U10 zero-based DWord offset into the HW status page.											
Format:	HardwareStatusPageOffset[11:2]U32											
Value	Name											
[16, 1023]												
	1:0	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All											
Format:	MBZ											
2	31:0	<p>Data DWord 0</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).</p>	Project:	All	Format:	U32						
Project:	All											
Format:	U32											
3	31:0	<p>Data DWord 1</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Project:	All	Format:	U32						
Project:	All											
Format:	U32											



MI_STORE_DATA_INDEX

MI_STORE_DATA_INDEX	
Project:	BDW
Source:	RenderCS
Length Bias:	2

The MI_STORE_DATA_INDEX command requests a write of the data constant supplied in the packet to the specified offset from the System Address defined by the Hardware Status Page Address Register. As the write targets a System Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).

Programming Notes

- Use of this command with an invalid or uninitialized value in the Hardware Status Page Address Register is UNDEFINED.
- This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll uncached memory or device registers).
- This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete eventually, there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations.

DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 21h MI_STORE_DATA_INDEX
		Format: OpCode
22	Reserved	Project: BDW
21	Use Per-Process Hardware Status Page	Project: BDW If this bit is set, this command will index into the per-process hardware status page at offset 0K from the LRCA. If clear, the Global Hardware Status Page will be indexed. This bit must be 0 if the Execlist Enable bit is clear.
20:8	Reserved	Format: MBZ
7:0	DWord Length	
	Default Value: 1h	
	Format: =n Total Length - 2. Excludes DWord (0,1) = 1 for DWord, 2 for QWord.	

MI_STORE_DATA_INDEX										
1	31:12	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
	Format:	MBZ								
	11:2	Offset <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">Format:</td> <td>U10 zero-based DWord offset into the HW status page.</td> </tr> <tr> <td>Format:</td> <td>HardwareStatusPageOffset[11:2]U32</td> </tr> </table> <p>This field specifies the offset (into the hardware status page) to which the data will be written. Note that the first few DWords of this status page are reserved for special-purpose data storage - targeting these reserved locations via this command is UNDEFINED. This address must be 8B aligned for a store QW command.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 60%; text-align: center;">Value</th> <th style="width: 40%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[16, 1023]</td> <td></td> </tr> </tbody> </table>	Format:	U10 zero-based DWord offset into the HW status page.	Format:	HardwareStatusPageOffset[11:2]U32	Value	Name	[16, 1023]	
	Format:	U10 zero-based DWord offset into the HW status page.								
Format:	HardwareStatusPageOffset[11:2]U32									
Value	Name									
[16, 1023]										
1:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ									
31:0	Data DWord 0 <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the DWord value to be written to the targeted location. For a QWord write this DWord is the lower DWord of the QWord to be reported (DW 0).</p>	Format:	U32							
Format:	U32									
3	31:0	Data DWord 1 <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>U32</td> </tr> </table> <p>This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).</p>	Format:	U32						
Format:	U32									



MI_STORE_DATA_INDEX

MI_STORE_DATA_INDEX	
Project:	BDW
Source:	VideoCS
Length Bias:	2

The MI_STORE_DATA_INDEX command requests a write of the data constant supplied in the packet to the specified offset from the System Address defined by the Hardware Status Page Address Register. As the write targets a System Address, the write operation is coherent with the CPU cache (i.e., the processor cache is snooped).

Programming Notes

- Use of this command with an invalid or uninitialized value in the Hardware Status Page Address Register is UNDEFINED.
- This command can be used for general software synchronization through variables in cacheable memory (i.e., where software does not need to poll uncached memory or device registers).
- This command simply initiates the write operation with command execution proceeding normally. Although the write operation is guaranteed to complete "eventually", there is no mechanism to synchronize command execution with the completion (or even initiation) of these operations.

DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	21h MI_STORE_DATA_INDEX
		Format:	OpCode
	22	Reserved	
		Format:	MBZ
	21	Use Per-Process Hardware Status Page	
		Project:	BDW
	If this bit is set, this command will index into the per-process hardware status page at offset 0K from the LRCA. If clear, the Global Hardware Status Page will be indexed. This bit must be '0' if the Execlist Enable bit is clear.		
	20:8	Reserved	
Format:		MBZ	
7:0	DWord Length		
	Default Value:	0h Excludes DWord (0,1) = 2 for QWord	
	Format:	=n Total Length - 2	

MI_STORE_DATA_INDEX						
1	31:12	Reserved Format: MBZ				
	11:2	Offset Format: U10 zero-based DWord offset into the HW status page Format: GraphicsAddress[11:2]U32 This field specifies the offset (into the hardware status page) to which the data will be written. For a QWord write, the offset is valid down to bit 3 only. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[16, 1023]</td> <td></td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> The first few DWords of this status page are reserved for special-purpose data storage - targeting these reserved locations via this command is UNDEFINED.	Value	Name	[16, 1023]	
	Value	Name				
	[16, 1023]					
	1:0	Reserved Format: MBZ				
2	31:0	Data DWord 0 Format: U32 FormatDesc This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).				
3	31:0	Data Word 1 Format: U32 FormatDesc This field specifies the upper DWord value to be written to the targeted QWord location (DW 1).				



MI_STORE_REGISTER_MEM

MI_STORE_REGISTER_MEM			
Project:	BDW		
Source:	CommandStreamer		
Length Bias:	2		
<p>The MI_STORE_REGISTER_MEM command requests a register read from a specified memory mapped register location in the device and store of that DWord to memory. The register address is specified along with the command to perform the read.</p>			
Programming Notes	Source		
<ul style="list-style-type: none"> The command temporarily halts command execution. The memory address for the write is snooped on the host bus. This command should not be used from within a "non-privilege" batch buffer to access global virtual space. doing so will be treated as privilege access violation. Refer "User Mode Privilege Command" in MI_BATCH_BUFFER_START command section to know HW behavior on encountering privilege access violation. This command can be used within ring buffers and/or "privilege" batch buffers to access global virtual space. This command will cause undefined data to be written to memory if given register addresses for the PGTBL_CTL_0 or FENCE registers. 			
<p>Source: BlitterCS, VideoCS, VideoEnhancementCS</p> <p>The source MMIO offset must be limited to any MMIO that is not replicated due to multiple slice configurations. If slice zero is disabled, then any MMIO read from this command streamer to a register replicated in the slice will cause a return value of zero.</p>	BlitterCS, VideoCS, VideoEnhancementCS		
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	24h MI_STORE_REGISTER_MEM
		Format:	OpCode
22	Use Global GTT		
	Format:	Boolean	
<p>It is allowed for this bit to be set when executing this command from a privileged (secure) batch or ring buffer. This bit must be clear when programmed from within a non-privileged batch buffer. This bit must be 1 if the Per Process GTT Enable bit is clear. This command will use the global GTT to translate the Address and this command must be executing from a privileged (secure) batch buffer.</p>			

MI_STORE_REGISTER_MEM									
	21	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Source:</td> <td>BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS	Format:	MBZ	
	Project:	BDW							
	Source:	BlitterCS, VideoCS, VideoCS2, VideoEnhancementCS							
	Format:	MBZ							
	21	Predicate Enable <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Source:</td> <td>RenderCS</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>If set, this command is executed (or not) depending on the current value of the MI Predicate internal state bit. This command is ignored only if PredicateEnable is set and the Predicate state bit is 0.</p>	Project:	BDW	Source:	RenderCS	Format:	U1	
	Project:	BDW							
Source:	RenderCS								
Format:	U1								
20	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								
19	Reserved <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> </table>	Project:	BDW						
Project:	BDW								
18:8	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								
7:0	DWord Length <table border="1"> <tr> <td>Format:</td> <td>=n Total Length - 2</td> </tr> </table> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>2h</td> <td>Excludes DWord (0,1) [Default]</td> <td>BDW</td> </tr> </tbody> </table>	Format:	=n Total Length - 2	Value	Name	Project	2h	Excludes DWord (0,1) [Default]	BDW
Format:	=n Total Length - 2								
Value	Name	Project							
2h	Excludes DWord (0,1) [Default]	BDW							
1	31:23	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ					
	Format:	MBZ							
	22:2	Register Address <table border="1"> <tr> <td>Format:</td> <td>MMIOAddress[22:2]MMIO_Register</td> </tr> </table> <p>This field specifies Bits 22:2 of the Register offset the DWord will be read from. As the register address must be DWord-aligned, Bits 1:0 of that address MBZ.</p> <table border="1"> <thead> <tr> <th colspan="2" style="text-align: center;">Programming Notes</th> </tr> </thead> <tbody> <tr> <td colspan="2"> <ul style="list-style-type: none"> Storing a VGA register is not permitted and will store an UNDEFINED value. The values of PGTBL_CTL0 or any of the FENCE registers cannot be stored to memory; UNDEFINED values will be written to memory if the addresses of these registers are specified. </td> </tr> </tbody> </table>	Format:	MMIOAddress[22:2]MMIO_Register	Programming Notes		<ul style="list-style-type: none"> Storing a VGA register is not permitted and will store an UNDEFINED value. The values of PGTBL_CTL0 or any of the FENCE registers cannot be stored to memory; UNDEFINED values will be written to memory if the addresses of these registers are specified. 		
Format:	MMIOAddress[22:2]MMIO_Register								
Programming Notes									
<ul style="list-style-type: none"> Storing a VGA register is not permitted and will store an UNDEFINED value. The values of PGTBL_CTL0 or any of the FENCE registers cannot be stored to memory; UNDEFINED values will be written to memory if the addresses of these registers are specified. 									
1:0	Reserved <table border="1"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ								



MI_STORE_REGISTER_MEM			
2..3 Project: BDW	63:2	Memory Address	
		Project:	BDW
		Format:	GraphicsAddress[63:2]MMIO
	<p>This field specifies the address of the memory location where the register value specified in the DWord above will be written. The address specifies the DWord location of the data. Range = GraphicsVirtualAddress[63:2] for a DWord register GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>		
	1:0	Reserved	
		Project:	BDW
		Format:	MBZ

MI_STORE_URB_MEM

MI_STORE_URB_MEM						
Project:	BDW					
Source:	RenderCS					
Length Bias:	2					
<p>The MI_STORE_URB_MEM command requests a URB read from a specified memory mapped URB location in the device and store of that DWord to memory. The URB address is specified along with the command to perform the read.</p>						
Programming Notes						
<ul style="list-style-type: none"> The command temporarily halts command execution 						
DWord	Bit	Description				
0	31:29	Command Type				
		Default Value:	0h MI_COMMAND			
		Format:	OpCode			
	28:23	MI Command Opcode				
		Default Value:	2Dh MI_STORE_URB_MEM			
		Format:	OpCode			
	22:8	Reserved				
Format:		MBZ				
7:0	DWord Length					
	Format:	=n				
	Total Length - 2. Excludes DWord (0,1).					
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 33%;">Value</th> <th style="width: 33%;">Name</th> <th style="width: 33%;">Project</th> </tr> </thead> <tbody> <tr> <td>2h</td> <td>[Default]</td> <td>BDW</td> </tr> </tbody> </table>	Value	Name	Project	2h	[Default]
Value	Name	Project				
2h	[Default]	BDW				
1	31:15	Reserved				
		Format:	MBZ			
	14:2	URB Address				
1:0	Reserved					
	Format:	MBZ				
2..3 Project: BDW	63:6	Memory Address				
		Project:	BDW			
		Format:	GraphicsAddress[63:6]			
<p>This field specifies the address of the location of where the value will be written to memory. The value must be in the first DW location of the cache line. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>						



MI_STORE_URB_MEM		
	5:0	Reserved
		Project: BDW
		Format: MBZ

MI_SUSPEND_FLUSH

MI_SUSPEND_FLUSH			
Project:		BDW	
Source:		VideoEnhancementCS	
Length Bias:		1	
Description			Project
Blocks PM Flush Requests.			BDW
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
	28:23	MI Command Opcode	
		Default Value:	0Bh MI_SUSPEND_FLUSH
	22:1	Reserved	
		Project:	All
		Format:	MBZ
	0	Suspend Flush	
		Project:	All
		Format:	Enable
Description			Project
This field suspends flush due to a PM flush request.			BDW



MI_SUSPEND_FLUSH

MI_SUSPEND_FLUSH			
Project:		BDW	
Source:		BlitterCS	
Length Bias:		1	
Description			Project
Blocks PM Flush Requests.			BDW
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
	28:23	MI Command Opcode	
		Default Value:	0Bh MI_SUSPEND_FLUSH
	22:1	Reserved	
		Project:	All
		Format:	MBZ
	0	Suspend Flush	
		Project:	All
		Format:	Enable
Description			Project
This field suspends flush due to a PM flush request.			BDW

MI_SUSPEND_FLUSH

MI_SUSPEND_FLUSH				
Project:		BDW		
Source:		RenderCS		
Length Bias:		1		
Description			Project	
Blocks PM Flush Requests.			BDW	
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	0Bh MI_SUSPEND_FLUSH	
		Format:	OpCode	
	22:1	Reserved		
		Format:	MBZ	
	0	Suspend Flush		
		Format:	Enable	
		Description		Project
		This field suspends flush due to a PM flush request.		BDW

MI_SUSPEND_FLUSH

MI_SUSPEND_FLUSH		
Project:	BDW	
Source:	VideoCS	
Length Bias:	1	
Description		Project
Blocks PM Flush Requests.		BDW
DWord	Bit	Description
0	31:29	Command Type Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode Default Value: 0Bh MI_SUSPEND_FLUSH
	22:1	Reserved Format: MBZ
	0	Suspend Flush Format: Enable
Description		Project
This field suspends flush due to a PM flush request.		BDW

MI_TOPOLOGY_FILTER

MI_TOPOLOGY_FILTER			
Project:	BDW		
Source:	RenderCS		
Length Bias:	1		
<p>This command is used to specify a specific 3DPrimType value, where the CS will ignore all 3DPRIMITIVE commands that do not have a matching 3DPrimType. This primitive culling is optional (turned off by using this command with a Topology Filter Value of 0). This command is specific to the Render command stream only.</p>			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 0h MI_COMMAND	
		Format: OpCode	
	28:23	MI Command Opcode	
		Default Value: 0Dh MI_TOPOLOGY_FILTER	
		Format: OpCode	
	22:6	Reserved	
		Format: MBZ	
	5:0	Topology Filter Value	
		Format: 3D_Prim_Topo_Type	
	<p>When non-zero, the CS will discard all 3DPRIMITIVE commands which do not match the specified 3DPrimTopologyType. When zero, no filtering is performed (normal operation).</p>		



MI_UPDATE_GTT

MI_UPDATE_GTT	
Project:	BDW
Source:	BlitterCS
Length Bias:	2

The MI_UPDATE_GTT command is used to update GGTT page table entries in a coherent manner and at a predictable place in the command flow. A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed prior to MI_UPDATE_GTT command, since work associated with preceding commands that are still in the pipeline may be referencing GTT entries that will be changed by its execution. The flush must also invalidate TLBs and read caches that may become invalid as a result of the changed GTT entries. A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed post MI_UPDATE_GTT command to ensure the GGTT is updated with modified page table entries before the following workload references the modified entries. MI_FLUSH_DWORD flush is not required if it can be guaranteed that the pipeline is free of any work that relies on changing GTT entries (such as MI_UPDATE_GTT contained in a paging DMA buffer that is doing only update/mapping activities and no rendering). MI_UPDATE_GTT command is privilege operation and will be converted to a no-op and an error flagged if it is executed from within a non-secure batch buffer. PPGTT updates cannot be done via **MI_UPDATE_GTT**, gfx driver will have to use MI_STORE_DATA_IMM for PPGTT inline updates.

DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND Format: OpCode
	28:23	MI Command Opcode
		Default Value: 23h MI_UPDATE_GTT Format: OpCode
22:10	Reserved	
	Format: MBZ	
9:0	DWord Length	Default Value: 0h Excludes DWord (0,1)
		Format: =n
		Total Length - 2
1	31:12	Entry Address
	Format: GraphicsAddress[31:12] This field holds the QW offset of the first table entry to be modified in GGTT.	
	11:0	Reserved
	Format: MBZ	
2..n	31:0	Entry Data
	Format: PageTableEntry	
	This Dword becomes the lower dword new page table entry. See PPGTT/Global GTT Table Entries (PTEs) in Memory Interface Registers.	

MI_UPDATE_GTT

MI_UPDATE_GTT				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
<p>The MI_UPDATE_GTT command is used to update GTT page table entries in a coherent manner and at a predictable place in the command flow.</p> <p>A PIPE_CONTROL flush command with "CS Stall" bit set must be programmed prior to MI_UPDATE_GTT command, since work associated with preceding commands that are still in the pipeline may be referencing GTT entries that will be changed by its execution. The flush must also invalidate TLBs and read caches that may become invalid as a result of the changed GTT entries. A PIPE_CONTROL flush command with "CS Stall" bit set must be programmed post MI_UPDATE_GTT command to ensure the GGTT is updated with modified page table entries before the following workload references the modified entries.</p> <p>PIPE_CONTROL flush is not required if it can be guaranteed that the pipeline is free of any work that relies on changing GTT entries (such as MI_UPDATE_GTT contained in a paging DMA buffer that is doing only update/mapping activities and no rendering).</p> <p>MI_UPDATE_GTT command is privilege operation and will be converted to a no-op and an error flagged if it is executed from within a non-secure batch buffer.</p> <p>PPGTT updates cannot be done via MI_UPDATE_GTT, gfx driver will have to use MI_STORE_DATA_IMM for PPGTT inline updates.</p>				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	23h MI_UPDATE_GTT	
		Format:	OpCode	
	22:10	Reserved		
		Format:	MBZ	
	9:0	DWord Length	Default Value:	0h
			Format:	=n Total Length - 2. Excludes DWord (0,1).
Programming Notes				
The value of this field must not exceed a value 3Fh when programmed in a batch buffer with resource streamer enabled.				
Project		BDW		
1	31:12	Entry Address		
		Format:	GraphicsAddress[31:12]	
		This field holds the QW offset of the first table entry to be modified in GGTT.		



MI_UPDATE_GTT				
	11:0	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
Format:	MBZ			
2..n	63:0	<p>Entry Data</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 40%;">Format:</td> <td>PageTableEntry</td> </tr> </table> <p>This Dword becomes the new page table entry. See PPGTT/Global GTT Table Entries (PTEs) in Memory Interface Registers.</p>	Format:	PageTableEntry
Format:	PageTableEntry			

MI_UPDATE_GTT

MI_UPDATE_GTT		
Project:	BDW	
Source:	VideoCS	
Length Bias:	2	
<p>The MI_UPDATE_GTT command is used to update GGTT page table entries in a coherent manner and at a predictable place in the command flow. A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed prior to MI_UPDATE_GTT command, since work associated with preceding commands that are still in the pipeline may be referencing GTT entries that will be changed by its execution. The flush must also invalidate TLBs and read caches that may become invalid as a result of the changed GTT entries. A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed post MI_UPDATE_GTT command to ensure the GGTT is updated with modified page table entries before the following workload references the modified entries. MI_FLUSH_DWORD flush is not required if it can be guaranteed that the pipeline is free of any work that relies on changing GTT entries (such as MI_UPDATE_GTT contained in a paging DMA buffer that is doing only update/mapping activities and no rendering). MI_UPDATE_GTT command is privilege operation and will be converted to a no-op and an error flagged if it is executed from within a non-secure batch buffer. PPGTT updates cannot be done via MI_UPDATE_GTT, gfx driver will have to use MI_STORE_DATA_IMM for PPGTT inline updates.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
Default Value: 23h MI_UPDATE_GTT		
22:10	Reserved	
	Format: MBZ	
9:0	DWord Length	Default Value: 0h Excludes DWord (0,1)
		Format: =n
		Total Length - 2
	31:12	Entry Address
Format: GraphicsAddress[31:12]		
This field holds the QW offset of the first table entry to be modified in GGTT.		
11:0	Reserved	
	Format: MBZ	
2..n	63:0	Entry Data
		Format: PageTableEntry
This Dword becomes the lower dword new page table entry. See PPGTT/Global GTT Table Entries (PTEs) in Memory Interface Registers.		



MI_UPDATE_GTT

MI_UPDATE_GTT		
Project:	BDW	
Source:	VideoEnhancementCS	
Length Bias:	2	
<p>The MI_UPDATE_GTT command is used to update GGTT page table entries in a coherent manner and at a predictable place in the command flow.</p> <p>A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed prior to MI_UPDATE_GTT command, since work associated with preceding commands that are still in the pipeline may be referencing GTT entries that will be changed by its execution. The flush must also invalidate TLBs and read caches that may become invalid as a result of the changed GTT entries. A MI_FLUSH_DWORD flush command with "CS Stall" bit set must be programmed post MI_UPDATE_GTT command to ensure the GGTT is updated with modified page table entries before the following workload references the modified entries.</p> <p>MI_FLUSH_DWORD flush is not required if it can be guaranteed that the pipeline is free of any work that relies on changing GTT entries (such as MI_UPDATE_GTT contained in a paging DMA buffer that is doing only update/mapping activities and no rendering).</p> <p>MI_UPDATE_GTT command is privilege operation and will be converted to a no-op and an error flagged if it is executed from within a non-secure batch buffer.</p> <p>PPGTT updates cannot be done via MI_UPDATE_GTT, gfx driver will have to use MI_STORE_DATA_IMM for PPGTT inline updates.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	Format: OpCode	
	28:23	MI Command Opcode
Default Value: 23h MI_UPDATE_GTT		
Format: OpCode		
22:10	Reserved	
	Format: MBZ	
9:0	DWord Length	
	Default Value: 0h Excludes DWord (0,1)	
	Format: =n	
Total Length - 2		
1	31:12	Entry Address
		Format: GraphicsAddress[31:12]
This field holds the QW offset of the first table entry to be modified in GGTT.		
11:0	Reserved	
	Format: MBZ	

MI_UPDATE_GTT

2..n	63:0	Entry Data	
		Format:	PageTableEntry
		<p>This Dword becomes the lower dword new page table entry. See PPGTT/Global GTT Table Entries (PTEs) in Memory Interface Registers.</p>	



MI_URB_ATOMIC_ALLOC

MI_URB_ATOMIC_ALLOC			
Project:	BDW		
Source:	RenderCS		
Length Bias:	1		
This command is used to specify the region in URB allocated for URB atomic value storage. This command is specific to the Render command stream only.			
Programming Notes			
This command can only be sent after a flush has occurred.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	09h MI_URB_ALLOC
		Format:	OpCode
	22:20	Reserved	
		Format:	MBZ
	19:12	URB Atomic Storage Offset	
		Format:	U8 Number of 128B Entries
This field specifies the offset of a 128B granular starting address in the URB. The value of URB Atomic Storage Offset plus the value of the URB Atomic Storage Size must not exceed 256.			
Value		Name	Description
	[0,255]		0-(32KB-128B)
11:9	Reserved		
	Format:	MBZ	
8:0	URB Atomic Storage Size		
	Format:	U9 Number of 128B Entries	
	This field specifies the size of the buffer in the URB in number of 128B entries. If this field has a value of zero then the URB Atomic allocation is disabled and will not be context save/restored.		
	Value	Name	Description
	[0,256]		0-32KB

MI_URB_CLEAR

MI_URB_CLEAR				
Project:	BDW			
Source:	RenderCS			
Length Bias:	2			
The MI_URB_CLEAR command allows SW to clear (write zero) to a section in the URB.				
Programming Notes				
<ul style="list-style-type: none"> The command temporarily halts command execution. This command is part of context save/restore. Only the last instance will be part of context. This command requires the 3D pipeline to be flushed before execution. 				
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value: 0h MI_COMMAND Format: OpCode		
	28:23	MI Command Opcode		
		Default Value: 19h MI_URB_CLEAR Format: OpCode		
22:8	Reserved Format: MBZ			
7:0	DWord Length			
	Default Value: 0h Format: =n Total Length - 2. Excludes DWord (0,1).			
1 Project: BDW	31:30	Reserved		
		Project: BDW Format: MBZ		
	29:16	URB Clear Length		
Project: BDW This field specifies the number of 256b entries in the URB to be cleared to zero. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">[0,16383]</td> <td></td> </tr> </tbody> </table>		Value	Name	[0,16383]
Value	Name			
[0,16383]				
15	Reserved			
	Project: BDW Format: MBZ			



MI_URB_CLEAR											
	<table border="1" style="width: 100%;"> <tr> <td style="width: 10%; text-align: center; vertical-align: top;">14:0</td> <td>URB Address</td> </tr> <tr> <td></td> <td> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>URBAddress[19:5] 256b aligned</td> </tr> </table> </td> </tr> <tr> <td colspan="2">This field specifies Bits 19:5 of the URB Address</td> </tr> </table>	14:0	URB Address		<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>URBAddress[19:5] 256b aligned</td> </tr> </table>	Project:	BDW	Format:	URBAddress[19:5] 256b aligned	This field specifies Bits 19:5 of the URB Address	
14:0	URB Address										
	<table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>URBAddress[19:5] 256b aligned</td> </tr> </table>	Project:	BDW	Format:	URBAddress[19:5] 256b aligned						
Project:	BDW										
Format:	URBAddress[19:5] 256b aligned										
This field specifies Bits 19:5 of the URB Address											

MI_USER_INTERRUPT

MI_USER_INTERRUPT		
Project:	BDW	
Source:	VideoEnhancementCS	
Length Bias:	1	
<p>The MI_USER_INTERRUPT command is used to generate a User Interrupt condition. The parser will continue parsing after processing this command. See User Interrupt.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 02h MI_USER_INTERRUPT
		Format: OpCode
	22:0	Reserved
		Project: All
		Format: MBZ



MI_USER_INTERRUPT

MI_USER_INTERRUPT		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	1	
<p>The MI_USER_INTERRUPT command is used to generate a User Interrupt condition. The parser will continue parsing after processing this command. See User Interrupt.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
	28:23	MI Command Opcode
		Default Value: 02h MI_USER_INTERRUPT
	22:0	Reserved
		Project:
Format:		MBZ

MI_USER_INTERRUPT

MI_USER_INTERRUPT		
Project:	BDW	
Source:	RenderCS	
Length Bias:	1	
<p>The MI_USER_INTERRUPT command is used to generate a User Interrupt condition. The parser will continue parsing after processing this command. See User Interrupt.</p>		
DWord	Bit	Description
0	31:29	Command Type
		Default Value: 0h MI_COMMAND
		Format: OpCode
	28:23	MI Command Opcode
		Default Value: 02h MI_USER_INTERRUPT
		Format: OpCode
	22:0	Reserved
		Format: MBZ



MI_USER_INTERRUPT

MI_USER_INTERRUPT			
Project:	BDW		
Source:	VideoCS		
Length Bias:	1		
The MI_USER_INTERRUPT command is used to generate a User Interrupt condition. The parser will continue parsing after processing this command. See User Interrupt.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	0h MI_COMMAND
		Format:	OpCode
	28:23	MI Command Opcode	
		Default Value:	02h MI_USER_INTERRUPT
		Format:	OpCode
	22:0	Reserved	
		Format:	MBZ

MI_WAIT_FOR_EVENT

MI_WAIT_FOR_EVENT					
Project:	BDW				
Source:	BlitterCS				
Length Bias:	1				
<p>The MI_WAIT_FOR_EVENT command is used to pause command stream processing until a specific event occurs or while a specific condition exists. Only one event/condition can be specified -- specifying multiple events is UNDEFINED. The effect of the wait operation depends on the source of the command. If executed from a batch buffer, the parser will halt (and suspend command arbitration) until the event/condition occurs. If executed from a ring buffer, further processing of that ring will be suspended, although command arbitration (from other rings) will continue. Note that if a specified condition does not exist (the condition code is inactive) at the time the parser executes this command, the parser proceeds, treating this command as a no-operation. If execution of this command from a primary ring buffer causes a wait to occur, the active ring buffer will effectively give up the remainder of its time slice (required in order to enable arbitration from other primary ring buffers).</p>					
Programming Notes					
<p>[Ring Buffer Mode Of scheduling Only][Blitter CS]: HW loses Page Directory (PPGTT) information on becoming IDLE. SW must always program the PD information following MI_WAIT_FOR_EVENT command. This will ensure Page Directory information gets reprogrammed on exiting IDLE flow triggered on MI_WAIT_FOR_EVENT. Alternatively SW can disable IDLE flows on MI_WAIT_FOR_EVENT by setting below bits in "BCS_ECOSKPD" register. Disable GT C6 Enter Due to Blitter Waiting on Vblank Disable GT C6 Enter Due to Blitter Waiting on Scanline Disable GT C6 Enter Due to Blitter Waiting on Flip Done</p>					
DWord	Bit	Description			
0	31:29	Command Type <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h MI_COMMAND</td> </tr> </table>	Default Value:	0h MI_COMMAND	
	Default Value:	0h MI_COMMAND			
	28:23	MI Command Opcode <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>03h MI_WAIT_FOR_EVENT</td> </tr> </table>	Default Value:	03h MI_WAIT_FOR_EVENT	
	Default Value:	03h MI_WAIT_FOR_EVENT			
22	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All				
Format:	MBZ				
21	Display Pipe C Vertical Blank Wait Enable <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait until the next Display Pipe C "Vertical Blank" event occurs. This event is described as the start of the next Display C vertical blank period. Note that this can cause a wait for up to an entire refresh period. See Vertical Blank Event in the Device Programming Interface chapter of MI Functions.</p>	Project:	BDW	Format:	Enable
Project:	BDW				
Format:	Enable				

MI_WAIT_FOR_EVENT				
20	Display Sprite C Flip Pending Wait Enable			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite C "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	All	Format:
Project:	All			
Format:	Enable			
19:16	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
15	Display Plane C Flip Pending Wait Enable			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Plane C "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	All	Format:
Project:	All			
Format:	Enable			
14	Display Pipe C Scan Line Wait Enable			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait while a Display Pipe C "Scan Line" condition exists. This condition is defined as the the start of the scan line specified in the Pipe C Display Scan Line Count Range Compare Register.</p>	Project:	BDW	Format:
Project:	BDW			
Format:	Enable			
13:12	Reserved			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:
Project:	All			
Format:	MBZ			
11	Display Pipe B Vertical Blank Wait Enable			
	<table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait until the next Display Pipe B "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe B vertical blank period. Note that this can cause a wait for up to an entire refresh period.</p>	Project:	BDW	Format:
Project:	BDW			
Format:	Enable			
10	Display Sprite B Flip Pending Wait Enable			
	<table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite B "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	All	Format:
Project:	All			
Format:	Enable			

MI_WAIT_FOR_EVENT		
9	Display Plane B Flip Pending Wait Enable	
	Project: All	
	Format: Enable	
	This field enables a wait for the duration of a Display Plane B "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).	
	Display Pipe B Scan Line Wait Enable	
8	Project: BDW	
	Format: Enable	
This field enables a wait while a Display Pipe B "Scan Line" condition exists. This condition is defined as the the start of the scan line specified in the Pipe B Display Scan Line Count Range Compare Register.		
7	Display Sprite B2 Flip Pending Wait Enable	
	Project: BDW	
	Format: Enable	
This field enables a wait for the duration of a Display Sprite B2 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).		
6	Display Sprite A2 Flip Pending Wait Enable	
	Project: BDW	
	Format: Enable	
This field enables a wait for the duration of a Display Sprite A2 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).		
5:4	Reserved	
	Project: All	
	Format: MBZ	
3	Display Pipe A Vertical Blank Wait Enable	
	Project: BDW	
	Format: Enable	
This field enables a wait until the next Display Pipe A "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe A vertical blank period. Note that this can cause a wait for up to an entire refresh period.		



MI_WAIT_FOR_EVENT					
2	<p>Display Sprite A Flip Pending Wait Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
1	<p>Display Plane A Flip Pending Wait Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Plane A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	All	Format:	Enable
Project:	All				
Format:	Enable				
0	<p>Display Pipe A Scan Line Wait Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait while a Display Pipe A "Scan Line" condition exists. This condition is defined as the the start of the scan line specified in the Pipe A Display Scan Line Count Range Compare Register.</p>	Project:	BDW	Format:	Enable
Project:	BDW				
Format:	Enable				

MI_WAIT_FOR_EVENT

MI_WAIT_FOR_EVENT		
Project:	BDW	
Source:	RenderCS	
Length Bias:	1	
Description		
<p>The MI_WAIT_FOR_EVENT command is used to pause command stream processing of this pipe only until a specific event occurs or while a specific condition exists. See Wait Events/Conditions, Device Programming Interface in <i>MI Functions</i>. Only one event/condition can be specified. Specifying multiple events is UNDEFINED. Once parsed, the parser will halt (and suspend command arbitration) until the event/condition occurs. Note that if a specified condition does not exist (the condition code is inactive) at the time the parser executes this command, the parser proceeds, treating this command as a no-operation.</p> <p>If CSunit is waiting for V-blank or flip done, HW can go into RC1/RC6 state.</p> <p>MI_NOOP setting NOP register (or any other benign command) must be set after MI_WAIT_FOR_EVENT under the following conditions:</p> <ul style="list-style-type: none"> • Back-to-back MI_WAIT_FOR_EVENT commands • MI_WAIT_FOR_EVENT is the last command before head = tail <p>Events must be unmasked in the Display Engine Render Response Mask Register (DE RRRM 0x44050) prior to waiting for them with a MI_WAIT_FOR_EVENT command, or in the case of flips or scanlines, prior to starting the flip or loading the scanline. Unmasked events will wake command streamer as they occur, so for improved power savings it is recommended to only unmask events that are required. Programming the DE RRRM register can be done through MMIO or a LOAD_REGISTER_IMMEDIATE command.</p> <p>Execution List Mode of Scheduling: CS on evaluating MI_WAIT_FOR_EVENT to be unsuccessful (has to wait for event to happen) triggers synchronous context switch stating the switch reason in Context Status Buffer. Note that synchronous context switch can be inhibited through programming "Inhibit Synchronous Context Switch" bit in CTXT_SR_CTL register or by disabling arbitration through MI_ARB_ON_OFF command.</p>		
Programming Notes		Source
<p>Ring Buffer Mode of Scheduling Only: SW must always program a dummy MI_SEMAPHORE_WAIT command in Signal Mode which is always successful prior to programming MI_WAIT_FOR_EVENT.</p> <p>If the above programming restriction is not followed, in certain order of programming sequences HW would enter IDLE_DOP instead of IDLE_C6 on encountering MI_WAIT_FOR_EVENT unsuccessful.</p>		RenderCS
<p>Render CS Only: SW must always program PIPE_CONTROL with "CS Stall" and "Render Target Cache Flush Enable" set prior to programming MI_WAIT_FOR_EVENT command for GPGPU workloads i.e when pipeline select is GPGPU via PIPELINE_SELECT command. This is required to achieve better GPGPU preemption latencies for certain programming sequences.</p> <p>If programming PIPE_CONTROL has performance implications then preemption latencies can be trade off against performance by not implementing this programming note.</p>		RenderCS
DWord	Bit	Description

		MI_WAIT_FOR_EVENT		
0	31:29	Command Type		
		Default Value:	0h MI_COMMAND	
		Format:	OpCode	
	28:23	MI Command Opcode		
		Default Value:	03h MI_WAIT_FOR_EVENT	
		Format:	OpCode	
	22	Reserved		
		Project:	BDW	
		Format:	MBZ	
	21	Display Pipe C Vertical Blank Wait Enable		
		Project:	BDW	
		Format:	Enable	
Description			Project	
This field enables a wait until the next Display Pipe C Vertical Blank event occurs. This event is described as the start of the next Display C vertical blank period. Note that this can cause a wait for up to an entire refresh period.				
	Render and Blitter Engines		BDW	
20	Display Sprite C Flip Pending Wait Enable			
	Project:	BDW		
	Format:	Enable		
	Description			Project
	This field enables a wait for the duration of a Display Sprite C Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).			
	Render and Blitter Engines		BDW	
19	Display Sprite C3 Flip Pending Wait Enable			
	Project:	BDW		
	Format:	Enable		
	This field enables a wait for the duration of a Display Sprite C3 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).			

MI_WAIT_FOR_EVENT											
18	Display Sprite B3 Flip Pending Wait Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite B3 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	BDW	Format:	Enable						
	Project:	BDW									
	Format:	Enable									
	17	Display Sprite A3 Flip Pending Wait Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite A3 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	BDW	Format:	Enable					
Project:		BDW									
Format:		Enable									
16		Display Sprite C2 Flip Pending Wait Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite C2 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	BDW	Format:	Enable					
	Project:	BDW									
	Format:	Enable									
	15	Display Plane C Flip Pending Wait Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 80%; text-align: center;">Description</th> <th style="width: 20%; text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">This field enables a wait for the duration of a Display Plane C "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</td> <td></td> </tr> <tr> <td style="padding: 5px;">Render and Blitter Engines</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Project:	BDW	Format:	Enable	Description	Project	This field enables a wait for the duration of a Display Plane C "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).		Render and Blitter Engines
Project:		BDW									
Format:		Enable									
Description		Project									
This field enables a wait for the duration of a Display Plane C "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).											
Render and Blitter Engines	BDW										
14	Display Pipe C Scan Line Wait Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 80%; text-align: center;">Description</th> <th style="width: 20%; text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">This field enables a wait while a Display Pipe C Scan Line condition exists. This condition is defined as the start of the scan line specified in the Pipe C Display Scan Line Count Range Compare Register.</td> <td></td> </tr> <tr> <td style="padding: 5px;">Render and Blitter Engines</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Project:	BDW	Format:	Enable	Description	Project	This field enables a wait while a Display Pipe C Scan Line condition exists. This condition is defined as the start of the scan line specified in the Pipe C Display Scan Line Count Range Compare Register.		Render and Blitter Engines	BDW
	Project:	BDW									
	Format:	Enable									
	Description	Project									
	This field enables a wait while a Display Pipe C Scan Line condition exists. This condition is defined as the start of the scan line specified in the Pipe C Display Scan Line Count Range Compare Register.										
Render and Blitter Engines	BDW										

MI_WAIT_FOR_EVENT				
13	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
12	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ	
Format:	MBZ			
11	Display Pipe B Vertical Blank Wait Enable			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable	
	Format:	Enable		
	Description	Project		
	This field enables a wait until the next Display Pipe B "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe B vertical blank period. Note that this can cause a wait for up to an entire refresh period.			
Render and Blitter Engines	BDW			
10	Display Sprite B Flip Pending Wait Enable			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable	
	Format:	Enable		
	Description	Project		
	This field enables a wait for the duration of a Display Sprite B "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).			
Render and Blitter Engines	BDW			
9	Display Plane B Flip Pending Wait Enable			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable	
	Format:	Enable		
	Description	Project		
	This field enables a wait for the duration of a Display Plane B Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).			
Render and Blitter Engines	BDW			
8	Display Pipe B Scan Line Wait Enable			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable	
	Format:	Enable		
	Description	Project		
	This field enables a wait while a Display Pipe B Scan Line condition exists. This condition is defined as the start of the scan line specified in the Pipe B Display Scan Line Count Range Compare Register.			
Render and Blitter Engines	BDW			

MI_WAIT_FOR_EVENT							
7	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite B2 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	BDW	Format:	Enable		
	Project:	BDW					
	Format:	Enable					
	6	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>This field enables a wait for the duration of a Display Sprite A2 Flip Pending condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</p>	Project:	BDW	Format:	Enable	
		Project:	BDW				
	Format:	Enable					
	5	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	
Project:		BDW					
Format:	MBZ						
4	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ				
	Format:	MBZ					
3	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable				
	Format:	Enable					
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 80%; text-align: center;">Description</th> <th style="width: 20%; text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">This field enables a wait until the next Display Pipe A "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe A vertical blank period. Note that this can cause a wait for up to an entire refresh period.</td> <td></td> </tr> <tr> <td style="padding: 5px;">Render and Blitter Engines</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Description	Project	This field enables a wait until the next Display Pipe A "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe A vertical blank period. Note that this can cause a wait for up to an entire refresh period.		Render and Blitter Engines	BDW
	Description	Project					
This field enables a wait until the next Display Pipe A "Vertical Blank" event occurs. This event is described as the start of the next Display Pipe A vertical blank period. Note that this can cause a wait for up to an entire refresh period.							
Render and Blitter Engines	BDW						
2	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table>	Format:	Enable				
	Format:	Enable					
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 80%; text-align: center;">Description</th> <th style="width: 20%; text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">This field enables a wait for the duration of a Display Sprite A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).</td> <td></td> </tr> <tr> <td style="padding: 5px;">Render and Blitter Engines</td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Description	Project	This field enables a wait for the duration of a Display Sprite A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).		Render and Blitter Engines	BDW
	Description	Project					
This field enables a wait for the duration of a Display Sprite A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).							
Render and Blitter Engines	BDW						

MI_WAIT_FOR_EVENT			
1	Display Plane A Flip Pending Wait Enable		
	Format:	Enable	
	Description		
	This field enables a wait for the duration of a Display Plane A "Flip Pending" condition. If a flip request is pending, the parser will wait until the flip operation has completed (i.e., the new front buffer address has now been loaded into the active front buffer registers).		
	Render and Blitter Engines		
	BDW		
	0	Display Pipe A Scan Line Wait Enable	
		Format:	Enable
		Description	
		This field enables a wait while a Display Pipe A "Scan Line" condition exists. This condition is defined as the start of the scan line specified in the Pipe A Display Scan Line Count Range Compare Register.	
Render and Blitter Engines			
BDW			
BDW			

Move

mov - Move			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The mov instruction moves the components in src0 into the channels of dst. If src0 and dst are of different types, format conversion is performed. If src0 is a scalar immediate, the immediate value is loaded into enabled channels of dst. A mov with the same source and destination type, no source modifier, and no saturation is a raw move. A packed byte destination region (B or UB type with HorzStride == 1 and ExecSize > 1) can only be written using raw move.</p>			
<p>When denorm mode is flush to zero, a raw mov instruction with saturation modifier will not flush the denorm input or output to zero (Denorm is preserved).</p>			
<p>Format: [(pred)] mov[.cmod] (exec_size) dst src0</p>			
Programming Notes			
<p>A <i>mov</i> instruction with a source modifier always copies a denorm source value to a denorm destination value (in the manner of a raw move).</p>			
<p>There is no direct conversion from B/UB to DF or DF to B/UB. Use two instructions and a word or DWord intermediate type.</p>			
<p>There is no direct conversion from B/UB to Q/UQ or Q/UQ to B/UB. Use two instructions and a word or DWord intermediate integer type.</p>			
<p>There is no direct conversion from HF to DF or DF to HF. Use two instructions and F (Float) as an intermediate type.</p>			
<p>There is no direct conversion from HF to Q/UQ or Q/UQ to HF. Use two instructions and F (Float) or a word integer type or a DWord integer type as an intermediate type.</p>			
Restriction			
<p>Raw move is not supported for Float values in ALT mode if any values are infinities or NaNs.</p>			
<p>An accumulator can be a source or destination operand but not both.</p>			
Syntax			
<p>[(pred)] mov[.cmod] (exec_size) reg reg [(pred)] mov[.cmod] (exec_size) reg imm32 [(pred)] mov[.cmod] (exec_size) reg imm64</p>			
Pseudocode			
<p>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n]; } }</p>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		

mov - Move

*B,*W,*D	F	
F	*B,*W,*D	
F	F	
*W,*D	DF	BDW
F	DF	BDW
DF	*W,*D	BDW
DF	F	BDW
DF	DF	BDW
*W,*D,*Q	*W,*D,*Q	BDW
F	*Q	BDW
DF	*Q	BDW
*Q	F	BDW
*Q	DF	BDW
*B,*W,*D	HF	BDW
F	HF	BDW
HF	*B,*W,*D	BDW
HF	F	BDW
HF	HF	BDW

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

Move Indexed

movi - Move Indexed	
Project:	BDW
Source:	Eulsa
Length Bias:	4
<p>The movi instruction performs a fast component-wise indexed move for subfields from src0 to dst. The source operand must be an indirectly-addressed register. All channels of the source operand share the same register number, which is provided by the register field of the first address subregister, with a possible immediate register offset. The register fields of the subsequent address subregisters are ignored by hardware. The subregister number of a source channel is provided by the subregister field of the corresponding address subregister, with a possible immediate subregister offset.</p> <p>The destination register may be either a directly-addressed or an indirectly-addressed register.</p> <p>This instruction effectively performs a subfield shuffling from one register to another. Up to eight subfields can be selected by an instruction.</p>	
Format: [(pred)] movi (exec_size) dst src0	
Programming Notes	
<p>HW Implementation Details:</p> <p>The source register is calculated by adding the register portion of the first index register with the register portion of the address immediate, $a0.0[11:5] + \text{addr_imm}[9:5]$</p> <p>For byte movi, byte0 of the destination is selected by $(a0.0[4:0])$, byte1 is selected by $(a0.1[4:0])$, ..., and byte7 is selected by $(a0.7[4:0])$. The rest of the bytes are undefined.</p> <p>For word movi, byte0 of the destination is selected by $(a0.0[4:1] \& 0)$, byte1 is selected by $(a0.0[4:1] \& 1)$, byte2 is selected by $(a0.1[4:1] \& 0)$, byte3 is selected by $(a0.1[4:1] \& 1)$, ..., and byte15 is selected by $(a0.7[4:1] \& 1)$. The rest of the bytes are undefined.</p> <p>For DWord or float movi, byte0 of the destination is selected by $(a0.0[4:2] \& 00b)$, byte1 is selected by $(a0.0[4:2] \& 01b)$, byte2 is selected by $(a0.0[4:2] \& 10b)$, byte3 is selected by $(a0.0[4:2] \& 11b)$, byte4 is selected by $(a0.1[4:2] \& 00b)$, byte5 is selected by $(a0.1[4:2] \& 01b)$, ..., byte31 is selected by $(a0.7[4:2] \& 11b)$.</p> <p>For all 3 conditions above, $a0.n[4:0] = a0.n[4:0] + \text{addr_imm}[4:0]$.</p>	
Restriction	
Source operand cannot be accumulators. The source operand must be a general register.	
The source and destination must have the same type.	
The execution size must be 8.	
The address register for the source must be aligned to the base (a0.0).	
The destination register (directly or indirectly addressed) must be 16-byte aligned.	
The destination region (directly or indirectly addressed) must point to the same GRF register.	
The destination stride in bytes must equal the source element size in bytes.	
The Align16 access mode is not allowed.	
All the index registers (address subregisters) used must point to the same GRF register.	
The instruction must use 1x1 indirect regioning.	
The destination offset is only used to create channel enables. Each element of the destination is directly	

movi - Move Indexed

mapped to the index registers for the movi instruction. i.e. a0.0 -> dst.0, a0.1 -> dst.1, a0.2 -> dst.2, etc.

Only 8 address subregisters are used (a0.0-a0.7). Destination element 8 will be sourced from address register zero (a0.0), dst.9 <-a0.1, etc. This is an exception to the above restriction, for example:
 movi (8) r31.8:uw r[a0.0,0]<8;8,1>:uw // r31.8:uw<-a0.0:uw, r31.9:uw<-a0.1:uw, etc.

Conditional Modifier is not allowed for this instruction.

Syntax

[(pred)] movi (exec_size) reg reg imm

Pseudocode

```
Evaluate(WrEn); srcregfile = regfile(src0); srcregbase = reg(address[0]) + reg(addr_imm); for ( n = 0; n <
RegWidth; n++ ) { if ( WrEn.chan[n] ) { srcsubreg = subreg(address[n] + addr_imm); dst.chan[n] =
srcregfile.srcreg.srcsubreg; } }
```

Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	Y	Y

Src Types	Dst Types
B	B
UB	UB
W	W
UW	UW
D	D
UD	UD
F	F

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	

Multiply

mul - Multiply

Project: BDW
 Source: Eulsa
 Length Bias: 4

The mul instruction performs component-wise multiplication of src0 and src1 and stores the results in dst. When multiplying integer datatypes, if src0 is DW and src1 is W, irrespective of the destination datatype, the accumulator maintains full 48-bit precision. This is required to handle the macro for 32x32 multiplication. The macro described in the mach instruction should be used to obtain the full precision 64-bit multiplication results. Note: A 32x32 multiply operation is handled natively, without a macro. When operating in this mode, the resulting 64-bit data is packed, unlike the macro, where the lower and upper 32 bits of the result are written to different general registers by two separate instructions. Refer to the macro description for details. When multiplying integer data types, if one of the sources is a DW, the resulting full precision data is stored in the accumulator. However, if the destination data type is either W or DW, the low bits of the result are written to the destination register and the remaining high bits are discarded. This results in undefined Overflow and Sign flags. Therefore, conditional modifiers and saturation (.sat) cannot be used in this case.

Format: [(pred)] mul[.cmod] (exec_size) dst src0 src1

Restriction

Integer source operands cannot be accumulators.

When multiplying a DW and any lower precision integer, the DW operand must on src0.

[BDW-A]: DW * W is not supported outside MACH macro.

[BDW]: When multiplying DW x DW, the dst cannot be accumulator.

Syntax

[(pred)] mul[.cmod] (exec_size) reg reg reg [(pred)] mul[.cmod] (exec_size) reg reg imm32

Pseudocode

```
Evaluate(WrEn); for ( n = 0; n < exec_size; n++ ) { if ( WrEn.chan[n] ) { dst.chan[n] = src0.chan[n] * src1.chan[n]; } }
```

Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y

Src Types	Dst Types
*B	*B
*B	*W
*B	*D
*W	*W
*W	*D
*W,*D	*D

mul - Multiply		
*D	*Q	
F	F	
DF	DF	
HF	HF	
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	

Multiply Accumulate

mac - Multiply Accumulate			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The mac instruction takes component-wise multiplication of src0 and src1, adds the results with the corresponding accumulator values, and then stores the final results in dst.</p> <p>Format: [(pred)] mac[.cmod] (exec_size) dst src0 src1</p>			
Restriction			
Accumulator is an implicit source and thus cannot be an explicit source operand.			
Syntax			
[(pred)] mac[.cmod] (exec_size) reg reg reg [(pred)] mac[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n] * src1.chan[n] + acc0.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W	*B,*W,*D		
F	F		
DF	DF	BDW	
HF	HF	BDW	
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	

Multiply Accumulate High

mach - Multiply Accumulate High	
Project:	BDW
Source:	Eulsa
Length Bias:	4
<p>The mach instruction performs DWord integer multiply-accumulate operation and outputs the high DWord (bits 63:32). For each enabled channel, this instruction multiplies the DWord in src0 with the high word of the DWord in src1, left shifts the result by 16 bits, adds it with the corresponding accumulator values, and keeps the whole 64-bit result in the accumulator. It then stores the high DWord (bits 63:32) of the results in dst. This instruction is intended to be used to emulate 32-bit DWord integer multiplication by using the large number of bits available in the accumulator. For example, the following instructions perform vector multiplication of two 32-bit signed integer sources from r2 and r3 and store the resulting vectors with the high 32 bits in r5 and the low 32 bits in r6.</p> <pre>mul (8) acc0:d r2.0<8;8,1>:d r3.0<16;8,2>:uw mach (8) r5.0<1>:d r2.0<8;8,1>:d r3.0<8;8,1>:d mov (8) r6.0<1>:d acc0:d // Low 32 bits.</pre> <p>Here is a different example including negation. An added preliminary mov is required for source modification on src1.</p> <pre>mov (8) r3.0<1>:d -r3<8;8,1>:d mul (8) acc0:d r2.0<8;8,1>:d r3.0<16;8,2>:uw mach (8) r5.0<1>:d r2.0<8;8,1>:d r3.0<8;8,1>:d // High 32 bits mov (8) r6.0<1>:d acc0:d // Low 32 bits.</pre> <p>The mach should have channel enable from the destHI of IMUL, the mov should have the channel enable from the destLO of IMUL. As mach is used to generate part of the 64-bit DWord integer results, saturation modifier should not be used. In fact, saturation modifier should not be used for any of these four instructions. Source and destination operands must be DWord integers. Source and destination must be of the same type, signed integer or unsigned integer. If dst is UD, src0 and src1 may be UD and/or D. However, if any of src0 and src1 is D, source modifier (abs) must be present to convert it to match with dst. If dst is D, src0 and src1 must also be D. They cannot be UD as it may cause unexpected overflow because the computed results are limited to 64 bits.</p>	
Format: [(pred)] mach[.cmod] (exec_size) dst src0 src1	
Restriction	
Accumulator is an implicit source and thus cannot be an explicit source operand.	
AccWrEn is required.	
Syntax	
[(pred)] mach[.cmod] (exec_size) reg reg reg [(pred)] mach[.cmod] (exec_size) reg reg imm32	
Pseudocode	
<pre>Evaluate (WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { acc.chan[n][63:0] = (src1.chan[n][31:16] * src0.chan[n][31:0]) << 16 + acc.chan[n][63:0];</pre>	

mach - Multiply Accumulate High

```
dst.chan[n][31:0] = acc.chan[n][63:32];
}
}
```

Errata	Description
	A source modifier must not be used on src1 for the macro operation. This applies to both mul and mach of the macro. If source modifier is required, an additional mov instruction may be used before the macro.

Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	Y	Y

Src Types	Dst Types
D	D
UD	UD

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	

Multiply Add

mad - Multiply Add			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The mad instruction takes component-wise multiplication of src1 and src2, adds the results with the corresponding src0 values, and then stores the final results in dst.</p> <p>The destination modifier saturation (.sat) must not be used when src1 or src2 are dwords.</p>			
Format: [(pred)] mad[.cmod] (exec_size) dst src0 src1 src2			
Restriction			
[BDW]: No explicit accumulator access because this is a three-source instruction. AccWrEn is allowed for implicitly updating the accumulator.			
[BDW]: All three-source instructions have certain restrictions, described in Instruction Formats [BDW].			
Syntax			
[(pred)] mad[.cmod] (exec_size) reg reg reg reg			
Pseudocode			
<pre> Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src1.chan[n] * src2.chan[n] + src0.chan[n]; } } </pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
F	F		
DF	DF	BDW	
HF	HF	BDW	
DWord	Bit	Description	
0..3	127:126	Reserved	
		Format:	MBZ
	125:106	Source 2	
		Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC
105		Reserved	
		Format:	MBZ

mad - Multiply Add

	104:85	Source 1	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC												
Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC																
	84	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ												
Format:	MBZ																
	83:64	Source 0	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC												
Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC																
	63:56	Destination Register Number	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>DstRegNum</td> </tr> </table>	Format:	DstRegNum												
Format:	DstRegNum																
	55:53	Destination Subregister Number	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>DstSubRegNum[2:0]</td> </tr> </table>	Format:	DstSubRegNum[2:0]												
Format:	DstSubRegNum[2:0]																
	52:49	Destination Channel Enable	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Format:</td> <td>ChanEn[4]</td> </tr> </table> <p>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</p>	Format:	ChanEn[4]												
Format:	ChanEn[4]																
	48	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW																
Format:	MBZ																
	47	NibCtrl	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>NibCtrl</td> </tr> </table>	Project:	BDW	Format:	NibCtrl										
Project:	BDW																
Format:	NibCtrl																
	46	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ										
Project:	BDW																
Format:	MBZ																
	45:44	Destination Data Type	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Project:</td> <td>BDW</td> </tr> <tr> <td colspan="2">This field contains the data type for the destination</td> </tr> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> <tr> <td>00b</td> <td>Single Precision Float</td> </tr> <tr> <td>01b</td> <td>DWord</td> </tr> <tr> <td>10b</td> <td>Unsigned DWord</td> </tr> <tr> <td>11b</td> <td>Double Precision Float</td> </tr> </table>	Project:	BDW	This field contains the data type for the destination		Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
Project:	BDW																
This field contains the data type for the destination																	
Value	Name																
00b	Single Precision Float																
01b	DWord																
10b	Unsigned DWord																
11b	Double Precision Float																

mad - Multiply Add		
48:42	Reserved	
	Project:	BDW
	Format:	MBZ
43:42	Source Data Type	
	Project:	BDW
	This field contains the data type for all three sources	
	Value	Name
	00b	Single Precision Float
	01b	DWord
	10b	Unsigned DWord
	11b	Double Precision Float
41:40	Source 2 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
39:38	Source 1 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
41:36	Reserved	
	Exists If:	/// ([Property[Source Modifier] == 'false')
	Format:	MBZ
37:36	Source 0 Modifier	
	Exists If:	/// ([Property[Source Modifier] == 'true')
	Format:	SrcMod
35	Reserved	
	Format:	MBZ
34	Reserved	
	Project:	
	Format:	MBZ
33	Flag Subregister Number This field contains the flag subregister number for instructions with a non-zero Conditional Modifier.	
32	Reserved	
	Project:	
	Format:	MBZ
31:0	Header	
	Format:	EU_INSTRUCTION_HEADER

Multiply Add for Macro

madm - Multiply Add for Macro			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The madm instruction takes component-wise multiplication of src1 and src2, adds the results with the corresponding src0 values, and then stores the final results in dst. The source and destination operands have a higher precision carried in the exponent for this operation. The madm instruction is used for macro operations, where precision is accumulated over several instructions. This accumulation requires the exponent to increase by 2 extra bits across multiple madm operations. Refer to Macros Defined in 'Math' Section for usage and restrictions of this operation.</p>			
Format: [(pred)] madm[.cmod] (exec_size) dst src0 src1 src2			
Restriction			
Accumulator access is restricted to the special accumulators (acc2-acc9). Refer to the Accumulator Section for details on the special accumulators.			
Syntax			
[(pred)] madm[.cmod] (exec_size) reg reg reg reg			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = src1.chan[n] * src2.chan[n] + src0.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
Src Types	Dst Types		
F	F		
DF	DF		
DWord	Bit	Description	
0..3	127:126	Reserved Format: MBZ	
	125:106	Source 2 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	
	105	Reserved Format: MBZ	
	104:85	Source 1 Format: EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC	

madm - Multiply Add for Macro												
84	Reserved											
	Format:	MBZ										
83:64	Source 0											
	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_THREE_SRC										
63:56	Destination Register Number											
	Format:	DstRegNum										
55:53	Destination Subregister Number											
	Format:	DstSubRegNum[2:0]										
52:49	Destination Channel Enable											
	Format:	ChanEn[4]										
<p>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</p>												
48	Reserved											
	Project:	BDW										
	Format:	MBZ										
47	NibCtrl											
	Project:	BDW										
	Format:	NibCtrl										
46	Reserved											
	Project:	BDW										
	Format:	MBZ										
45:44	Destination Data Type											
	Project:	BDW										
<p>This field contains the data type for the destination</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Single Precision Float</td> </tr> <tr> <td>01b</td> <td>DWord</td> </tr> <tr> <td>10b</td> <td>Unsigned DWord</td> </tr> <tr> <td>11b</td> <td>Double Precision Float</td> </tr> </tbody> </table>			Value	Name	00b	Single Precision Float	01b	DWord	10b	Unsigned DWord	11b	Double Precision Float
Value	Name											
00b	Single Precision Float											
01b	DWord											
10b	Unsigned DWord											
11b	Double Precision Float											

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



No Operation

nop - No Operation			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Do nothing. The nop instruction takes an instruction dispatch but performs no operation. It can be used for assembly patching in memory, or to insert a delay in the program sequence.			
Format: nop			
Restriction			
The nop instruction takes no instruction options other than Breakpoint.			
Syntax			
nop			
Pseudocode			
{ ; // The null statement, which does nothing. }			
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
DWord	Bit	Description	
0..3	127:31	Reserved	
		Format:	MBZ
	30	Reserved	
	29:7	Reserved	
		Format:	MBZ
	6:0	Opcode	
		Format:	EU_OPCODE

Oword Block Read MSD

MSDOR_OWB - Oword Block Read MSD								
Project:	BDW							
Source:	DataPort 0							
Length Bias:	1							
Family:	Block R/W							
Group:	OW Block R/W							
DWord	Bit	Description						
0	19	Header Present <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> <p>Indicates that the message requires a header.</p>	Project:	All	Format:	MDC_MHR		
	Project:	All						
	Format:	MDC_MHR						
	18	Legacy Message <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Legacy Message</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h						
	Project:	All						
	Format:	Opcode						
	17:14	Message Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>00h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Oword Block Read message</p>	Default Value:	00h	Project:	All	Format:	Opcode
	Default Value:	00h						
	Project:	All						
Format:	Opcode							
13	Invalidate After Read <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_IAR</td> </tr> </table> <p>Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs</p>	Project:	All	Format:	MDC_IAR			
Project:	All							
Format:	MDC_IAR							
12:11	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ			
Project:	All							
Format:	MBZ							
10:8	Data Elements <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DB_OW</td> </tr> </table> <p>Specifies the number of contiguous Owords to be read or written</p>	Project:	All	Format:	MDC_DB_OW			
Project:	All							
Format:	MDC_DB_OW							



MSD0R_OWB - Oword Block Read MSD

	7:0	Binding Table Index	
		Project:	All
		Format:	MDC_BTS_A32
		Specifies the Binding Table Index for the message	

Oword Block Write MSD

MSDOW_OWB - Oword Block Write MSD								
Project:	BDW							
Source:	DataPort 0							
Length Bias:	1							
Family:	Block R/W							
Group:	OW Block R/W							
DWord	Bit	Description						
0	19	<p>Header Present</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHR</td> </tr> </table> <p>Indicates that the message requires a header.</p>	Project:	All	Format:	MDC_MHR		
	Project:	All						
	Format:	MDC_MHR						
	18	<p>Legacy Message</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Legacy Message</p>	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h						
	Project:	All						
Format:	Opcode							
17:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>08h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Oword Block Write message</p>	Default Value:	08h	Project:	All	Format:	Opcode	
Default Value:	08h							
Project:	All							
Format:	Opcode							
13:11	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	All	Format:	MBZ			
Project:	All							
Format:	MBZ							
10:8	<p>Data Elements</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DB_OW</td> </tr> </table> <p>Specifies the number of contiguous Owords to be read or written</p>	Project:	All	Format:	MDC_DB_OW			
Project:	All							
Format:	MDC_DB_OW							
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_A32</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS_A32			
Project:	All							
Format:	MDC_BTS_A32							

Oword Dual Block Write MSD

MSD0W_OWDB - Oword Dual Block Write MSD								
Project:	BDW							
Source:	DataPort 0							
Length Bias:	1							
Family:	Block R/W							
Group:	OW Dual Block R/W							
DWord	Bit	Description						
0	19	Header Present <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> If set, indicates that the message includes the header.	Project:	All	Format:	Enable		
	Project:	All						
	Format:	Enable						
	18	Legacy Message <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Legacy Message	Default Value:	0h	Project:	All	Format:	Opcode
	Default Value:	0h						
	Project:	All						
Format:	Opcode							
17:14	Message Type <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Default Value:</td> <td>0Ah</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> Oword Block Read message	Default Value:	0Ah	Project:	All	Format:	Opcode	
Default Value:	0Ah							
Project:	All							
Format:	Opcode							
13:10	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> Ignored	Project:	All	Format:	MBZ			
Project:	All							
Format:	MBZ							
9:8	Data Elements <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DB_OWD</td> </tr> </table> Specifies the number of contiguous Owords to be read or written	Project:	All	Format:	MDC_DB_OWD			
Project:	All							
Format:	MDC_DB_OWD							
7:0	Binding Table Index <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS_A32</td> </tr> </table> Specifies the Binding Table Index for the message	Project:	All	Format:	MDC_BTS_A32			
Project:	All							
Format:	MDC_BTS_A32							

Oword Unaligned Block Read MSD

MSDOR_OWUB - Oword Unaligned Block Read MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Block R/W	
Group:	OW Unaligned Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18	Legacy Message
		Default Value: 0h
Project: All		
Format: Opcode		
Legacy Message		
17:14	Message Type	
	Default Value: 01h	
	Project: All	
	Format: Opcode	
Oword Unaligned Block Read message		
13:11	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
10:8	Data Elements	
	Project: All	
	Format: MDC_DB_OW	
Specifies the number of contiguous Owords to be read		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS_A32	
Specifies the Binding Table Index for the message		

PIPE_CONTROL

PIPE_CONTROL			
Project:	BDW		
Source:	RenderCS		
Length Bias:	2		
The PIPE_CONTROL command is used to effect the synchronization described above.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
		Format:	OpCode
	28:27	Command SubType	
		Default Value:	3h GFXPIPE_3D
		Format:	OpCode
	26:24	3D Command Opcode	
		Default Value:	2h PIPE_CONTROL
		Format:	OpCode
	23:16	3D Command Sub Opcode	
Default Value:		0h PIPE_CONTROL	
Format:		OpCode	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	4h DWORD_COUNT_n	
	Project:	BDW	
	Format:	=n	
	Total Length - 2. Excludes DWord (0,1).		
1	31:29	Reserved	
		Project:	All
		Format:	MBZ
	28	Reserved	
		Project:	BDW
		Format:	MBZ
	27	Reserved	
		Project:	BDW

PIPE_CONTROL				
26	Reserved			
	Project:	BDW		
	Format:	MBZ		
25	Reserved			
	Project:	All		
	Format:	MBZ		
24	Destination Address Type			
	Project:	BDW		
	Defines address space of Destination Address			
	Value	Name	Description	Project
	0h	PPGTT	Use PPGTT address space for DW write	All
	1h	GGTT	Use GGTT address space for DW write	All
	Programming Notes			
Ignored if ""No Write" is selected in Operation.				
23	LRI Post Sync Operation			
	Project:	BDW		
	Value	Name	Description	Project
	0h	No LRI Operation	No LRI operation occurs as a result of this instruction. The Post-Sync Operation field is valid and may be used to specify an operation.	All
	1h	MMIO Write Immediate Data	Write the DWord contained in Immediate Data Low (DW3) to the MMIO offset specified in the Address field.	All
	Programming Notes			Project
	This bit causes a post sync operation with an LRI (Load Register Immediate) operation. If this bit is set then the Post-Sync Operation field must be cleared.			
22	Reserved			
	Project:	All		

PIPE_CONTROL											
21	<p>Store Data Index</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>U1</td> </tr> </table> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td> <p>Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT).</p> <p>Execlist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).</p> </td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Format:	U1	Description	Project	<p>Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT).</p> <p>Execlist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).</p>	BDW				
	Format:	U1									
Description	Project										
<p>Ring Buffer Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is actually an index into the global hardware status page. This bit only applies to the Global HW status page. If this field is 1, the Destination Address Type in this command must be set to 1 (GGTT).</p> <p>Execlist Mode Scheduling: This field is valid only if the post-sync operation is not 0. If this bit is set, the store data address is index into the global hardware status page when destination address type in the command is set to 1 (GGTT). The store data address is index into the per-process hardware status page when destination address type in the command is set to 0 (PPGTT).</p>	BDW										
20	<p>Command Streamer Stall Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U1</td> </tr> </table> <p>If ENABLED, the sync operation will not occur until all previous flush operations pending a completion of those previous flushes will complete, including the flush produced from this command. This enables the command to act similar to the legacy MI_FLUSH command.</p> <table border="1" style="width: 100%; margin-top: 5px;"> <thead> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td> <p>One of the following must also be set:</p> <ul style="list-style-type: none"> • Render Target Cache Flush Enable ([12] of DW1) • Depth Cache Flush Enable ([0] of DW1) • Stall at Pixel Scoreboard ([1] of DW1) • Depth Stall ([13] of DW1) • Post-Sync Operation ([13] of DW1) • DC Flush Enable([5] of DW1) </td> <td style="text-align: center;">BDW</td> </tr> <tr> <td> <p>This bit must be always set when PIPE_CONTROL command is programmed by GPGPU and MEDIA workloads, except for the cases when only Read Only Cache Invalidation bits are set (State Cache Invalidation Enable, Instruction cache Invalidation Enable, Texture Cache Invalidation Enable, Constant Cache Invalidation Enable). This is to WA FFDOP CG issue, this WA need not implemented when FF_DOP_CG is disable via "Fixed Function DOP Clock Gate Disable" bit in RC_PSMI_CTRL register.</p> </td> <td style="text-align: center;">BDW</td> </tr> </tbody> </table>	Project:	All	Format:	U1	Programming Notes	Project	<p>One of the following must also be set:</p> <ul style="list-style-type: none"> • Render Target Cache Flush Enable ([12] of DW1) • Depth Cache Flush Enable ([0] of DW1) • Stall at Pixel Scoreboard ([1] of DW1) • Depth Stall ([13] of DW1) • Post-Sync Operation ([13] of DW1) • DC Flush Enable([5] of DW1) 	BDW	<p>This bit must be always set when PIPE_CONTROL command is programmed by GPGPU and MEDIA workloads, except for the cases when only Read Only Cache Invalidation bits are set (State Cache Invalidation Enable, Instruction cache Invalidation Enable, Texture Cache Invalidation Enable, Constant Cache Invalidation Enable). This is to WA FFDOP CG issue, this WA need not implemented when FF_DOP_CG is disable via "Fixed Function DOP Clock Gate Disable" bit in RC_PSMI_CTRL register.</p>	BDW
	Project:	All									
	Format:	U1									
	Programming Notes	Project									
<p>One of the following must also be set:</p> <ul style="list-style-type: none"> • Render Target Cache Flush Enable ([12] of DW1) • Depth Cache Flush Enable ([0] of DW1) • Stall at Pixel Scoreboard ([1] of DW1) • Depth Stall ([13] of DW1) • Post-Sync Operation ([13] of DW1) • DC Flush Enable([5] of DW1) 	BDW										
<p>This bit must be always set when PIPE_CONTROL command is programmed by GPGPU and MEDIA workloads, except for the cases when only Read Only Cache Invalidation bits are set (State Cache Invalidation Enable, Instruction cache Invalidation Enable, Texture Cache Invalidation Enable, Constant Cache Invalidation Enable). This is to WA FFDOP CG issue, this WA need not implemented when FF_DOP_CG is disable via "Fixed Function DOP Clock Gate Disable" bit in RC_PSMI_CTRL register.</p>	BDW										
19	<p>Reserved</p>										

PIPE_CONTROL			
18	TLB Invalidate		
	Project:	All	
	Format:	U1	
	<p>If ENABLED, all TLBs belonging to Render Engine will be invalidated once the flush operation is complete. Note that if the flush TLB invalidation mode is clear, a TLB invalidate will occur irrespective of this bit setting</p> <p>If ENABLED, PIPE_CONTROL command will flush the in flight data written out by render engine to Global Observation point on flush done. Also Requires stall bit ([20] of DW1) set.</p>		
	Programming Notes		
<p>If ENABLED, all TLBs belonging to Render Engine will be invalidated once the flush operation is complete. Note that if the flush TLB invalidation mode is clear, a TLB invalidate will occur irrespective of this bit setting.</p>		Project	
		BDW	
17	Reserved		
	Project:	BDW	
	Format:	MBZ	
16	Generic Media State Clear		
	Project:	BDW	
	Format:	Disable	
	<p>If set, all generic media state context information will be invalidated. Any state invalidated will not be saved as part of the render engine context image. The state only become valid once it is parsed by the command streamer.</p>		
15:14	Post Sync Operation		
	Project:	All	
	Description		
	<p>This field specifies an optional action to be taken upon completion of the synchronization operation.</p> <p>This field must be cleared if the LRI Post-Sync Operation bit is set.</p>		
	Value	Name	Description
	0h	No Write	No write occurs as a result of this instruction. This can be used to implement a "trap" operation, etc.
	1h	Write Immediate Data	Write the QWord containing Immediate Data Low, High DWs to the Destination Address
	2h	Write PS Depth Count	Write the 64-bit PS_DEPTH_COUNT register to the Destination Address
	3h	Write Timestamp	Write the 64-bit TIMESTAMP register to the Destination Address

PIPE_CONTROL				
Programming Notes				
If executed in non-secure batch buffer, the address given will be in a PPGTT address space. If in a secure ring or batch, address given will be in GGTT space				
Workaround				
Workaround : PIPECONTROL command with "Command Streamer Stall Enable" must be programmed prior to programming a PIPECONTROL command with Post Sync Op in GPGPU mode of operation (i.e when PIPELINE_SELECT command is set to GPGPU mode of operation).				
13	Depth Stall Enable			
	Project:	All		
	Format:	Enable		
	This bit must be set when obtaining a "visible pixel" count to preclude the possible inclusion in the PS_DEPTH_COUNT value written to memory of some fraction of pixels from objects initiated after the PIPE_CONTROL command.			
	Value	Name	Description	Project
	0h	Disable	3D pipeline will not stall subsequent primitives at the Depth Test stage.	All
	1h	Enable	3D pipeline will stall any subsequent primitives at the Depth Test stage until the Sync and Post-Sync operations complete.	All
	Programming Notes			Project
	This bit must be DISABLED for operations other than writing PS_DEPTH_COUNT.			BDW
	This bit will have no effect (besides preventing write cache flush) if set in a PIPE_CONTROL command issued to the Media pipe.			
12	Render Target Cache Flush Enable			
	Project:	All		
	Format:	Enable		
	Setting this bit will force Render Cache to be flushed to memory prior to this synchronization point completing. This bit must be set for all write fence sync operations to assure that results from operations initiated prior to this command are visible in memory once software observes this synchronization.			
	Value	Name	Description	Project
	0h	Disable Flush	Render Target Cache is NOT flushed.	All
	1h	Enable Flush	Render Target Cache is flushed.	All
	Programming Notes			
	This bit must be DISABLED for End-of-pipe (Read) fences, PS_DEPTH_COUNT or TIMESTAMP queries.			
	This bit must not be set when Depth Stall Enable bit is set in this packet.			

PIPE_CONTROL							
11	Instruction Cache Invalidate Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Setting this bit is independent of any other bit in this packet. This bit controls the invalidation of the L1 and L2 at the top of the pipe i.e. at the parsing time.</p>	Project:	All	Format:	Enable		
	Project:	All					
	Format:	Enable					
	10	Texture Cache Invalidation Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Setting this bit is independent of any other bit in this packet. This bit controls the invalidation of the texture caches at the top of the pipe i.e. at the parsing time.</p>	Project:	All	Format:	Enable	
Project:		All					
Format:	Enable						
9	Indirect State Pointers Disable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table>	Project:	All	Format:	Enable		
	Project:	All					
	Format:	Enable					
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 80%;">Description</th> <th style="width: 20%;">Project</th> </tr> </thead> <tbody> <tr> <td>At the completion of the post-sync operation associated with this pipe control packet, the indirect state pointers in the hardware are considered invalid; the indirect pointers are not saved in the context. If any new indirect state commands are executed in the command stream while the pipe control is pending, the new indirect state commands are preserved.</td> <td></td> </tr> <tr> <td>[BDW]: Using Invalidate State Pointer (ISP) only inhibits context restoring of Push Constant (3DSTATE_CONSTANT_*) commands. Push Constant commands are only considered as Indirect State Pointers. Once ISP is issued in a context, SW must initialize by programming push constant commands for all the shaders (at least to zero length) before attempting any rendering operation for the same context.</td> <td>BDW</td> </tr> </tbody> </table>		Description	Project	At the completion of the post-sync operation associated with this pipe control packet, the indirect state pointers in the hardware are considered invalid; the indirect pointers are not saved in the context. If any new indirect state commands are executed in the command stream while the pipe control is pending, the new indirect state commands are preserved.		[BDW]: Using Invalidate State Pointer (ISP) only inhibits context restoring of Push Constant (3DSTATE_CONSTANT_*) commands. Push Constant commands are only considered as Indirect State Pointers. Once ISP is issued in a context, SW must initialize by programming push constant commands for all the shaders (at least to zero length) before attempting any rendering operation for the same context.
Description	Project						
At the completion of the post-sync operation associated with this pipe control packet, the indirect state pointers in the hardware are considered invalid; the indirect pointers are not saved in the context. If any new indirect state commands are executed in the command stream while the pipe control is pending, the new indirect state commands are preserved.							
[BDW]: Using Invalidate State Pointer (ISP) only inhibits context restoring of Push Constant (3DSTATE_CONSTANT_*) commands. Push Constant commands are only considered as Indirect State Pointers. Once ISP is issued in a context, SW must initialize by programming push constant commands for all the shaders (at least to zero length) before attempting any rendering operation for the same context.	BDW						
8	Notify Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>If ENABLED, a Sync Completion Interrupt will be generated (if enabled by the MI Interrupt Control registers) once the sync operation is complete. See Interrupt Control Registers in Memory Interface Registers for details.</p>	Project:	All	Format:	Enable		
	Project:	All					
Format:	Enable						
7	Pipe Control Flush Enable <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>Hardware on parsing PIPECONTROL command with Pipe Control Flush Enable set will wait for all the outstanding post sync operations corresponding to previously executed PIPECONTROL commands are complete before making forward progress.</p>	Project:	BDW	Format:	Enable		
	Project:	BDW					
Format:	Enable						

PIPE_CONTROL		
6	Reserved	
	Project: BDW	
	DC Flush Enable	
	Project: BDW	
	Format: Enable	
Setting this bit enables flushing of the L3\$ portions that caches DC writes.		
Programming Notes		
DC Flush (L3 Flush) by default doesn't result in flushing/invalidating the IA Coherent lines from L3\$, however this can be achieved by setting control bit " Pipe line flush Coherent lines " in "L3SQCREG4" register.		
4	VF Cache Invalidation Enable	
	Project: All	
	Format: Enable	
	Setting this bit is independent of any other bit in this packet. This bit controls the invalidation of VF address based cache at the top of the pipe i.e. at the parsing time.	
	Workaround	
Workaround : When VF Cache Invalidate is set "Post Sync Operation" must be enabled to "Write Immediate Data" or "Write PS Depth Count" or "Write Timestamp".		
Project		
BDW		
3	Constant Cache Invalidation Enable	
	Project: All	
	Format: Enable	
Setting this bit is independent of any other bit in this packet. This bit controls the invalidation of the constant cache at the top of the pipe i.e. at the parsing time.		
2	State Cache Invalidation Enable	
	Project: All	
	Format: Enable	
Setting this bit is independent of any other bit in this packet. This bit controls the invalidation of the L1 and L2 state caches at the top of the pipe i.e. at the parsing time.		

PIPE_CONTROL			
	1	Stall At Pixel Scoreboard	
	Project:		All
Format:		Enable	
Defines the behavior of PIPE_CONTROL command at the pixel scoreboard.			
		Value	Name
		Description	Project
	0h	Disable	Stall at the pixel scoreboard is disabled.
	1h	Enable	Stall at the pixel scoreboard is enabled.
Programming Notes			
This bit must be DISABLED for End-of-pipe (Read) fences, PS_DEPTH_COUNT or TIMESTAMP queries. This bit is ignored if Depth Stall Enable is set. Further the render cache is not flushed even if Write Cache Flush Enable bit is set.			
	0	Depth Cache Flush Enable	
	Project:		All
Format:		Enable	
Setting this bit enables flushing (i.e. writing back the dirty lines to memory and invalidating the tags) of depth related caches. This bit applies to HiZ cache, Stencil cache and depth cache.			
		Value	Name
		Description	Project
	0h	Flush Disabled	Depth relates caches (HiZ, Stencil and Depth) are NOT flushed.
	1h	Flush Enabled	Depth relates caches (HiZ, Stencil and Depth) are flushed.
Programming Notes			
Ideally depth caches need to be flushed only when depth is required to be coherent in memory for later use as a texture, source or honoring CPU lock. This bit must be DISABLED for End-of-pipe (Read) fences, PS_DEPTH_COUNT or TIMESTAMP queries.			
2	31:2	Address	
	Project:		BDW
Format:		GraphicsAddress[31:2]U32	
If Post Sync Operation is set to 1h ([BDW]: LRI Post-Sync Operation must be clear): Bits 31:3 specify the QW address of where the Immediate Data following this DW in the packet to be stored. Bit 2 MBZ Ignored if "No Write" is the selected in Post-Sync Operation [BDW]: If LRI Post-Sync Operation is set: Bits 22:2 (Bits 31:23 are reserved MBZ) specify the MMIO offset destination for the data in the Immediate Data Low (DW3) field. Only DW writes are valid.			
	1:0	Reserved	
	Format:		MBZ
3 Project: BDW	31:16	Reserved	
		Project:	
Format:		MBZ	



PIPE_CONTROL										
	15:0	<p>Address High</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]U32</td> </tr> </table> <p>This field specifies the 4GB aligned base address of gfx 4GB virtual address space within the host's 64-bit virtual address space. This field is valid only if the post-sync operation is not 0 and the LRI Post-Sync Operation is clear.</p>	Project:	All	Format:	GraphicsAddress[47:32]U32				
	Project:	All								
Format:	GraphicsAddress[47:32]U32									
<p>4..5 Project: BDW</p>	63:0	<p>Immediate Data</p> <table border="1"> <tr> <td>Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>U64</td> </tr> </table> <p>This field specifies the QWord value to be written to the targeted location. Only valid when Post-Sync Operation is 1h (Write Immediate Data) or LRI Post-Sync Operation is set. Ignored if Post-Sync Operation is "No write", "Write PS_DEPTH_COUNT" or "Write TIMESTAMP".</p> <table border="1"> <tr> <th style="text-align: center;">Programming Notes</th> <th style="text-align: center;">Project</th> </tr> <tr> <td>This field must be programmed to 0 when Post-Sync Operation is set to Write PS Depth Count or Write Timestamp.</td> <td>BDW</td> </tr> </table>	Project:	BDW	Format:	U64	Programming Notes	Project	This field must be programmed to 0 when Post-Sync Operation is set to Write PS Depth Count or Write Timestamp.	BDW
Project:	BDW									
Format:	U64									
Programming Notes	Project									
This field must be programmed to 0 when Post-Sync Operation is set to Write PS Depth Count or Write Timestamp.	BDW									

PIPELINE_SELECT

PIPELINE_SELECT								
Project:	DevBWR+							
Length Bias:	1							
Description								
<p>The PIPELINE_SELECT command is used to specify which GPE pipeline is to be considered the 'current' active pipeline. Issuing 3D-pipeline-specific commands when the Media pipeline is selected, or vice versa, is UNDEFINED.</p>								
<p>Issuing 3D-pipeline-specific commands when the GPGPU pipeline is selected, or vice versa, is UNDEFINED.</p>								
<p>Programming common non pipeline commands (e.g., STATE_BASE_ADDRESS) is allowed in all pipeline modes.</p>								
Programming Notes								
<p>Software must ensure all the write caches are flushed through a stalling PIPE_CONTROL command followed by another PIPE_CONTROL command to invalidate read only caches prior to programming MI_PIPELINE_SELECT command to change the Pipeline Select Mode. Example: ... Workload-3Dmode PIPE_CONTROL (CS Stall, Depth Cache Flush Enable, Render Target Cache Flush Enable, DC Flush Enable) PIPE_CONTROL (Constant Cache Invalidate, Texture Cache Invalidate, Instruction Cache Invalidate, State Cache invalidate) PIPELINE_SELECT (GPGPU)</p>								
<p>Software must clear the COLOR_CALC_STATE Valid field in 3DSTATE_CC_STATE_POINTERS command prior to send a PIPELINE_SELECT with Pipeline Select set to GPGPU.</p>								
<p>Render CS Only: SW must always program PIPE_CONTROL with CS Stall and Render Target Cache Flush Enable set prior to programming PIPELINE_SELECT command for GPGPU workloads i.e when pipeline mode is set to GPGPU. This is required to achieve better GPGPU preemption latencies for certain programming sequences. If programming PIPE_CONTROL has performance implications then preemption latencies can be trade off against performance by not implementing this programming note.</p>								
<p>Hardware Binding Tables are only supported for 3D workloads. Resource streamer must be enabled only for 3D workloads. Resource streamer must be disabled for Media and GPGPU workloads. Batch buffer containing both 3D and GPGPU workloads must take care of disabling and enabling Resource Streamer appropriately while changing the PIPELINE_SELECT mode from 3D to GPGPU and vice versa. Resource streamer must be disabled using MI_RS_CONTROL command and Hardware Binding Tables must be disabled by programming 3DSTATE_BINDING_TABLE_POOL_ALLOC with "Binding Table Pool Enable" set to disable (i.e. value '0'). Example below shows disabling and enabling of resource streamer in a batch buffer for 3D and GPGPU workloads. MI_BATCH_BUFFER_START (Resource Streamer Enabled) PIPELINE_SELECT (3D) 3DSTATE_BINDING_TABLE_POOL_ALLOC (Binding Table Pool Enabled) 3D WORKLOAD MI_RS_CONTROL (Disable Resource Streamer) 3DSTATE_BINDING_TABLE_POOL_ALLOC (Binding Table Pool Disabled) PIPELINE_SELECT (GPGPU) GPGPU Workload 3DSTATE_BINDING_TABLE_POOL_ALLOC (Binding Table Pool Enabled) MI_RS_CONTROL (Enable Resource Streamer) 3D WORKLOAD MI_BATCH_BUFFER_END</p>								
DWord	Bit	Description						
0	31:29	<table border="1"> <thead> <tr> <th colspan="2">Command Type</th> </tr> </thead> <tbody> <tr> <td>Default Value:</td> <td>3h GFXPIPE</td> </tr> <tr> <td>Format:</td> <td>OpCode</td> </tr> </tbody> </table>	Command Type		Default Value:	3h GFXPIPE	Format:	OpCode
Command Type								
Default Value:	3h GFXPIPE							
Format:	OpCode							

PIPELINE_SELECT				
	28:27	Command SubType		
		Default Value:	1h GFXPIPE_SINGLE_DW	
		Format:	OpCode	
	26:24	3D Command Opcode		
		Format:	OpCode	
		Value	Name	Project
		1h	GFXPIPE_NONPIPELINED [Default]	BDW
	23:16	3D Command Sub Opcode		
		Default Value:	04h PIPELINE_SELECT	
		Format:	OpCode	
	6	Reserved		
		Project:	BDW	
15:2	Reserved			
	Project:	BDW		
1:0	Pipeline Selection			
	Value	Name	Description	
	0	3D	3D pipeline is selected	
	1	Media	Media pipeline is selected (Includes HD optical disc playback, HD video playback, and generic media workloads)	
	2	GPGPU	GPGPU pipeline is selected	
			BDW	

Plane

pln - Plane			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The pln instruction computes a component-wise plane equation ($w = p*u+q*v+r$ where $u/v/w$ are vectors and $p/q/r$ are scalars) of src0 and src1 and stores the results in dst. src1 is the input vector u. src0 provides input scalars p, q, and r, where p is the scalar value based on the region description of src0 and q and r are the scalar values implied from the src0 region. Specifically, q is the second component and r is the fourth component of the 4-tuple (128-bit aligned) that p belongs to.</p>			
Format: [(pred)] pln[.cmod] (exec_size) dst src0 src1			
Restriction			
This is a specialized instruction that only supports an execution size (ExecSize) of 8 or 16.			
The src0 region must be a replicated scalar (with HorzStride == VertStride == 0).			
src0 must specify .0 or .4 as the subregister number, corresponding to a subregister byte offset of 0 or 16.			
Source operands cannot be accumulators.			
Syntax			
[(pred)] pln[.cmod] (exec_size) reg reg reg			
Pseudocode			
<pre> Evaluate(WrEn); for (n = 0; n < exec_size; n++) { float dwP = src0.RegNum.SubRegNum[bits4:2]; // A DWord-aligned scalar. float dwQ = src0.RegNum.(SubRegNum[bit4:2] 0x1); // Second component. float dwR = src0.RegNum.(SubRegNum[bit4:2] 0x3); // Fourth component. if (ExecSize == 8) { u = src1.RegNum v = src1.(RegNum + 1) } else { if (n < 8) { u = src1.RegNum v = src1.(RegNum + 1) } else { u = src1.(RegNum + 2) v = src1.(RegNum + 3) } } if (WrEn.chan[n]) { dst.chan[n] = dwP * u.chan[n] + dwQ * v.chan[n] + dwR; } } </pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	N

pln - Plane		
Src Types	Dst Types	
F	F	
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	

REP16 Render Target Write MSD

MSD_RTW_REP16 - REP16 Render Target Write MSD			
Project:	BDW		
Source:	Render Cache DataPort		
Length Bias:	1		
Family:	Other		
Group:	Render Target R/W		
DWord	Bit	Description	
0	31	Reserved	
		Project:	All
		Format:	MBZ
			Ignored
	30	Message Precision Subtype	
		Default Value:	0h
		Project:	All
		Format:	Opcode
			Full precision data message
	29	Reserved	
Project:		All	
Format:		MBZ	
		Ignored	
28:25	Message Length		
	Project:	All	
	Format:	U4	
		Specifies the number of 256-bit GRF registers sent as the message payload (including the header). Valid value ranges are 1 to 15.	
24:20	Response Length		
	Project:	All	
	Format:	U5	
		Specifies the number of 256-bit GRF registers expected as the message response payload. Valid value ranges are 0 to 16.	
19	Header Present		
	Project:	All	
	Format:	MDC_MHP	
		If set, indicates that the message includes the 2-register header.	

MSD_RTW_REP16 - REP16 Render Target Write MSD

18	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW							
Format:	MBZ							
17:14	Message Type	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td style="width: 50%;">0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Render Target Write message</p>	Default Value:	0Ch	Project:	All	Format:	Opcode
Default Value:	0Ch							
Project:	All							
Format:	Opcode							
13	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW							
Format:	MBZ							
12	Last Render Target Select	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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. In general, when threads are not launched by 3D FF, this bit must be zero.</p> <div style="background-color: #e6f2ff; padding: 5px; text-align: center;">Programming Notes</div> <p>When a pixel shader has render target writes at finer granularity than the dispatch rate, last render target write to a null surface must be present at the dispatch rate with this bit set. In particular, if a kernel is dispatched at pixel rate and it only writes to render targets at sample-rate, it must include a pixel-rate render target write to a null surface with Last Render Target Select bit enabled.</p>	Project:	All	Format:	Enable		
Project:	All							
Format:	Enable							
11	Slot Group Select	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td style="width: 50%;">All</td> </tr> <tr> <td>Format:</td> <td>MDC_RT_SGS</td> </tr> </table> <p>This field selects whether slots 15:0 or slots 31:16 are used for bypassed data.</p>	Project:	All	Format:	MDC_RT_SGS		
Project:	All							
Format:	MDC_RT_SGS							

MSD_RTW_REP16 - REP16 Render Target Write MSD

10:8	Render Target Message Subtype	
	Default Value:	1h
	Project:	All
	Format:	Opcode
	SIMD16 Single source message with replicated data. Use slots [15:0] for pixel enables, X/Y addresses, and oMask.	
	Programming Notes	
	The above slots indicated are within the 16 slots selected by Slot Group Select. If SLOTGRP_HI is selected, slots [31:16] are referenced instead of [15:0].	
7:0	Binding Table Index	
	Project:	All
	Format:	MDC_BTS
	Specifies the Binding Table Index for the message	

Return

ret - Return			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
Return execution to the code sequence that called a subroutine. The ret instruction can be predicated or non-predicated. If non-predicated, all channels jump to the return IP in the first channel of src0 and restore CallMask from the second channel of src0. If predicated, the enabled channels jump to the return IP from the first channel of src0 and the corresponding bits in the CallMask are cleared to zero; if all CallMask bits are zero after the ret instruction, then execution jumps to the return IP from the first channel of src0. When SPF is on, the predication control must be scalar.			BDW
Format: [(pred)] ret (exec_size) null src0			
Restriction		Project	
This instruction cannot take accumulator as source.			
The src0 regioning control must be <2;2,1>			
Syntax			
[(pred)] ret (exec_size) null reg			
Pseudocode			Project
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { PcIP[n] = src0.chan[0]; CallMask[n] = 0; } else { PcIP[n] = IP + 1; } } for (n = exec_size; n < 32; n++) { PcIP[n] = IP + 1; } if (CallMask[n:0] == 0) { // all channels are zero Jump(src0.chan[0]); CallMask = src0.chan[1]; }			BDW
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types			
D, UD			
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	([(Operand Controls][Src0.RegFile]!='IMM')
	Format:	EU_INSTRUCTION_SOURCES_REG	
0..3	127:64	ImmSource	
		Exists If:	([(Operand Controls][Src0.RegFile]=='IMM')
	Format:	EU_INSTRUCTION_SOURCES_IMM32	
0..3	63:32	Operand Controls	
		Format:	EU_INSTRUCTION_OPERAND_CONTROLS

ret - Return			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER

Round Down

rndd - Round Down			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The <code>rndd</code> instruction takes component-wise floating point downward rounding (to the integral float number closer to negative infinity) of <code>src0</code> and storing the rounded integral float results in <code>dst</code>. This is commonly referred to as the <code>floor()</code> function. Each result follows the rules in the following tables based on the floating-point mode.</p>			
Format: <code>[(pred)] rndd[.cmod] (exec_size) dst src0</code>			
Syntax			
<code>[(pred)] rndd[.cmod] (exec_size) reg reg [(pred)] rndd[.cmod] (exec_size) reg imm32</code>			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = floor(src0.chan[n]); } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	<code>[[Operand Controls][Src0.RegFile]!='IMM']</code>
	Format:	EU_INSTRUCTION_SOURCES_REG	
	127:64	ImmSource	
Exists If:		<code>[[Operand Controls][Src0.RegFile]='IMM']</code>	
Format:	EU_INSTRUCTION_SOURCES_IMM32		
63:32	Operand Controls		
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS	
31:0	Header		
	Format:	EU_INSTRUCTION_HEADER	

Round to Nearest or Even

rnde - Round to Nearest or Even			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The rnde instruction takes component-wise floating point round-to-even operation of src0 with results in two pieces - a downward rounded integral float results stored in dst and the round-to-even increments stored in the rounding increment bits. The round-to-even increment must be added to the results in dst to create the final round-to-even values to emulate the round-to-even operation, commonly known as the round() function. The final results are the one of the two integral float values that is nearer to the input values. If the neither possibility is nearer, the even alternative is chosen. Each result follows the rules in the following tables based on the floating-point mode.</p>			
Format: [(pred)] rnde[.cmod] (exec_size) dst src0			
Syntax			
[(pred)] rnde[.cmod] (exec_size) reg reg [(pred)] rnde[.cmod] (exec_size) reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { if (src0.chan[n] - floor(src0.chan[n]) > 0.5f) { dst.chan[n] = floor(src0.chan[n]) + 1; } else if (src0.chan[n] - floor(src0.chan[n]) < 0.5f) { dst.chan[n] = floor(src0.chan[n]); } else { if (floor(src0.chan[n]) is odd) { dst.chan[n] = floor(src0.chan[n]) + 1; } else { dst.chan[n] = floor(src0.chan[n]); } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
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		



Round to Zero

rndz - Round to Zero			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The rndz instruction takes component-wise floating point round-to-zero operation of src0 with results in two pieces - a downward rounded integral float results stored in dst and the round-to-zero increments stored in the rounding increment bits. The round-to-zero increment must be added to the results in dst to create the final round-to-zero values to emulate the round-to-zero operation, commonly known as the truncate() function. The final results are the one of the two closest integral float values to the input values that is nearer to zero.</p>			
Format: [(pred)] rndz[.cmod] (exec_size) dst src0			
Syntax			
[(pred)] rndz[.cmod] (exec_size) reg reg [(pred)] rndz[.cmod] (exec_size) reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { dst.chan[n] = floor(src0.chan[n]); if (abs(src0.chan[n]) < abs(dst.chan[n])) { dst.chan[n] = floor(src0.chan[n]) + 1; } else { dst.chan[n] = floor(src0.chan[n]); } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
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	

Round Up

rndu - Round Up			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The rndu instruction takes component-wise floating point upward rounding (to the integral float number closer to positive infinity) of src0, commonly known as the ceiling() function. Each result follows the rules in the following tables based on the floating-point mode.</p>			
Format: [(pred)] rndu[.cmod] (exec_size) dst src0			
Syntax			
[(pred)] rndu[.cmod] (exec_size) reg reg [(pred)] rndu[.cmod] (exec_size) reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { if (src0.chan[n] - floor(src0.chan[n]) > 0.0f) { dst.chan[n] = floor(src0.chan[n]) + 1; } else { dst.chan[n] = src0.chan[n]; } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
F	F		
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	



Scattered Move

smov - Scattered Move			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The smov instruction moves the components in src0 into dst. For each enabled channel, copy src0 to dst. The immediate is used to selectively enable channels without using flags. When predication is enabled, the predicate mask is not generated from the flags. Instead, the immediate is used to mask the execution mask. If any channel is enabled as a result of this masking, the instruction is executed. When predication is not enabled, the immediate masks the execution mask. This provides flexibility to mask out any channel with an immediate.</p>			
Format: [(pred)] smov[.cmod] (exec_size) dst src0 src1			
Programming Notes			
When predication is disabled, the immediate provides the flexibility to perform a select operation without the use of flags.			
When predication is enabled, the usage model provides flexibility to select any bit in the flag registers for predication for execution size of 1.			
Syntax			
[(pred)] smov[.cmod] (exec_size) reg reg imm32			
Pseudocode			
if pred emask = OR (emask AND imm32) Else pmask = imm32. Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn.chan[n]) { dst.chan[n] = src0.chan[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
Src Types	Dst Types	Project	
*W,*D,*Q, HF, F, DF	*W,*D,*Q, HF, F, DF	BDW	
DWord	Bit	Description	
0..3	127:64	RegSource	
		Exists If:	([RegSource][Src1.RegFile]!='IMM')
	Format:	EU_INSTRUCTION_SOURCES_REG_REG	
127:64	127:64	ImmSource	
		Exists If:	([ImmSource][Src1.RegFile]='IMM')
	Format:	EU_INSTRUCTION_SOURCES_REG_IMM	
63:32	63:32	Operand Controls	
		Format:	EU_INSTRUCTION_OPERAND_CONTROLS

smov - Scattered Move			
	31:0	Header	
		Format:	EU_INSTRUCTION_HEADER



Scratch Block Read MSD

MSDOR_HWB - Scratch Block Read MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Block R/W	
Group:	HW Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18	Scratch Block Message
Default Value: 1h		
Project: All		
Format: Opcode		
Scratch Block Message		
17	Operation Type	
	Default Value: 0h	
	Project: All	
	Format: Opcode	
Scratch Block Read message		
16	Channel Mode	
	Project: BDW	
	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.		
15	Invalidate After Read	
	Project: All	
	Format: MDC_IAR	
Specifies if L3 cache lines accessed by the message should be invalidated after the read occurs		

MSD0R_HWB - Scratch Block Read MSD	
14	Reserved
	Project: All
	Format: MBZ
	Ignored
	13:12 Data Elements
	Project: All
	Format: MDC_DB_HW
	Specifies the number of registers to be read or written
	11:0 Address Offset
	Project: All
Format: GeneralStateOffset[17:6]	
HWORD (32 byte) based address offset to the BufferAddress in the Message Header.	



Scratch Block Write MSD

MSD0W_HWB - Scratch Block Write MSD		
Project:	BDW	
Source:	DataPort 0	
Length Bias:	1	
Family:	Block R/W	
Group:	HW Block R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHR
	Indicates that the message requires a header.	
	18	Scratch Block Message
Default Value: 1h		
Project: All		
Format: Opcode		
Scratch Block Message		
17	Operation Type	
	Default Value: 1h	
	Project: All	
	Format: Opcode	
Scratch Block Write message		
16	Channel Mode	
	Project: BDW	
	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.		
15:14	Reserved	
	Project: All	
	Format: MBZ	
Ignored		

MSD0W_HWB - Scratch Block Write MSD					
13:12	Data Elements <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_DB_HW</td> </tr> </table> <p>Specifies the number of registers to be read or written</p>	Project:	All	Format:	MDC_DB_HW
	Project:	All			
Format:	MDC_DB_HW				
11:0	Address Offset <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GeneralStateOffset[17:6]</td> </tr> </table> <p>HWORD (32 byte) based address offset to the BufferAddress in the Message Header.</p>	Project:	All	Format:	GeneralStateOffset[17:6]
	Project:	All			
	Format:	GeneralStateOffset[17:6]			



Select

sel - Select			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
<p>The sel instruction selectively moves the components in src0 or src1 into the channels of dst based on the predication. On a channel by channel basis, if the channel condition is true, data in src0 is moved into dst. Otherwise, data in src1 is moved into dst.</p> <p>As the predication is used to select the two sources, it is not included in the evaluation of WrEn. The predicate clause is mandatory if cmod is omitted/0000b. If both predication and the conditional modifier are omitted, the results are undefined.</p> <p>If the conditional modifier is specified (not 0000b, a compare is performed and the resulting condition flag is used for the sel instruction. Conditional modifiers .ge and .l follow the cmpn rules, and all other conditional modifiers follow the cmp rules. Predication is not allowed in this mode.</p> <p>A sel instruction with cmod .l is used to emulate a MIN instruction.</p> <p>A sel instruction with cmod .ge is used to emulate a MAX instruction.</p> <p>For a sel instruction with a .l or .ge conditional modifier, if one source is NaN and the other not NaN, the non-NaN source is the result. If both sources are NaNs, the result is NaN. For all other conditional modifiers, if either source is NaN then src1 is selected.</p> <p>A sel instruction without a conditional modifier always copies a denorm source value to a denorm destination value (in the manner of a raw move). This applies even if the source modifies are set on the sel instruction sources.</p> <p>The sel instruction uses any conditional modifier internally and does not update the flag register if a conditional modifier is used.</p>			
A sel instruction with cmod or source modifier will flush denorm to zero, depending on the denorm mode bit; a sel instruction without cmod and source modifier will retain denorm.			BDW
Format: (pred) sel[.cmod] (exec_size) dst src0 src1			
Syntax			
(pred) sel[.cmod] (exec_size) reg reg reg (pred) sel[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn, NoPMask); if (cmod == "0000") { // no CMod Evaluate(PMask); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { if (PMask.channel[n]) { dst.chan[n] = src0.chan[n]; } else { dst.chan[n] = src1.chan[n]; } } } else { // with CMod Evaluate(CMod); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { if (CMod.chan[n]) { dst.chan[n] = src0.chan[n]; } else { dst.chan[n] = src1.chan[n]; } } } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W*D	*B,*W,*D		
F	F		

sel - Select

DF	DF	BDW
*W,*D,*Q	*W,*D,*Q	BDW
HF	HF	BDW
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	

Send Message

send - Send Message	
Project:	BDW
Source:	Eulsa
Length Bias:	4
Description	
<p>Send a message stored in GRF starting at <src> to a shared function identified by <ex_desc> along with control from <desc> with a GRF writeback location at <dest>.</p>	
<p>The send instruction performs data communication between a thread and external function units, including shared functions (Sampler, Data Port Read, Data Port Write, URB, and Message Gateway) and some fixed functions (e.g. Thread Spawner, who also have an unique Shared Function ID). The send instruction adds an entry to the EU's message request queue. The request message is stored in a block of contiguous GRF registers. The response message, if present, will be returned to a block of contiguous GRF registers. The return GRF writes may be in any order depending on the external function units. <src> is the lead GRF register for request. <dest> is the lead GRF register for response. The message descriptor field <desc> contains the Message Length (the number of consecutive GRF registers) and the Response Length (the number of consecutive GRF registers). It also contains the header present bit, and the function control signals. The extend message descriptor field <ex_desc> contains the target function ID. WrEn is forwarded to the target function in the message sideband.</p>	
<p>The send instruction is the only way to terminate a thread. When the EOT (End of Thread) bit of <ex_desc> is set, it indicates the end of thread to the EU, the Thread Dispatcher and, in most cases, the parent fixed function. Message descriptor field <desc> can be a 32-bit immediate, imm32, or a 32-bit scalar register, <reg32a>. GEN restricts that the 32-bit scalar register <reg32a> must be the leading dword of the address register. It should be in the form of a0.0<0;1,0>.ud. When <desc> is a register operand, only the lower 29 bits of <reg32a> are used.</p>	
<p><ex_desc> is a 6-bit immediate, imm6. The lower 4bits of the <ex_desc> specifies the SFID for the message. The MSb of the message descriptor, the EOT field, always comes from bit 127 of the instruction word, which is the MSb of imm6. A thread must terminate with a send instruction with EOT turned on.</p>	
<p><src> is a 256-bit aligned GRF register. It serves as the leading GRF register of the request.</p>	
<p><dest> serves for two purposes: to provide the leading GRF register location for the response message if present, and to provide parameters to form the channel enable sideband signals. <dest> signals whether there is a response to the message request. It can be either a null register, a direct-addressed GRF register or a register-indirect GRF register. Otherwise, hardware behavior is undefined. If <dest> is null, there is no response to the request. Meanwhile, the Response Length field in <desc> must be 0. Certain types of message requests, such as memory write (store) through the Data Port, do not want response data from the function unit. If so, the posted destination operand can be null. If <dest> is a GRF register, the register number is forwarded to the shared function. In this case, the target function unit must send one or more response message phases back to the requesting thread. The number of response message phases must match the Response Length field in <desc>, which of course cannot be zero. For some cases, it could be an empty return message. An empty return message is defined as a single phase message with all channel enables turned off. The subregister number, horizontal stride, destination mask and type fields of <dest> are always valid and are used in part to generate on the WrEn. This is true even if <dest> is a null register (this is an exception for null as for most cases these fields are ignored by hardware). The 16-bit channel enables of the message sideband are formed based</p>	

send - Send Message

on the WrEn. Interpretation of the channel enable sideband signals is subject to the target external function. In general for a 'send' instruction with return messages, they are used as the destination dword write mask for the GRF registers starting at <dest>. For a message that has multiple return phases, the same set of channel enable signals applies to all the return phases. Thread managed memory coherency: A special usage of using non-null <dest> is to support write-commit signaling for memory write service by the Data Port Write unit. If <post_dest> is not null for a memory write request, the Data Port along with the Data Cache or Render Cache will wait until all the posted writes for the request have reached the coherent domain before sending back to the requesting thread an empty message to <dest> register. A memory write reaching the coherent domain, also referred to as reaching the global observable state, means that subsequent read to the same memory location, no matter which thread issues the read, must return the data of the write. The destination dependency control, {NoDDClr}, can be used in this instruction. This allows software to control the destination dependencies for multiple 'read'-type messages similar to that for multiple instructions using EU execution pipeline. As send does not check register dependencies for the post destination, {NoDDChk} should not be used for this instruction.

Restriction

Software must obey the following rules in signaling the end of thread using the send instruction: The posted destination operand must be null. No acknowledgement is allowed for the send instruction that signifies the end of thread. This is to avoid deadlock as the EU is expecting to free up the terminated thread's resource. A thread must terminate with a send instruction with message to a shared function on the output message bus; therefore, it cannot terminate with a send instruction with message to the following shared functions: Sampler unit, NULL function For example, a thread may terminate with a URB write message or a render cache write message. A root thread originated from the media (generic) pipeline must terminate with a send instruction with message to the Thread Spawner unit. A child thread should also terminate with a send to TS. Please refer to the Media Chapter for more detailed description. The send instruction can not update accumulator registers. Saturate is not supported for send instruction. ThreadCtrl are not supported for send instruction. The send with EOT should use register space R112-R127 for <src>. This is to enable loading of a new thread into the same slot while the message with EOT for current thread is pending dispatch. Any instruction updating the ARF must use a {Switch} if the ARF is not used before EOT. DepCtrl Must not be used with Send Instruction. When pagefault is enabled, the source and destination operands must not overlap. This is required to ensure the messages can be replayed.</src>

Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N

DWord	Bit	Description
0..3	127:96	Message Format: EU_INSTRUCTION_OPERAND_SEND_MSG
	95:89	Flags Format: EU_INSTRUCTION_FLAGS
	88:64	Source 0 Exists If: (Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]='Align1') Format: EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN1

send - Send Message		
88:64	Source 0	
	Exists If:	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]='Align16')
	Format:	EU_INSTRUCTION_OPERAND_SRC_REG_ALIGN16
63:32	Operand Control	
	Format:	EU_INSTRUCTION_OPERAND_CONTROLS
31:28	Controls B	
	Format:	EU_INSTRUCTION_CONTROLS_B
27:24	Shared Function ID (SFID)	
	Format:	SFID
23:8	Controls A	
	Format:	EU_INSTRUCTION_CONTROLS_A
7	Reserved	
	Format:	MBZ
6:0	Opcode	
	Format:	EU_OPCODE

Shift Left

shl - Shift Left			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			Project
<p>Perform component-wise logical left shift of the bits in src0 by the shift count indicated in src1, storing the results in dst, inserting zero bits in the number of LSBs indicated by the shift count. Hardware detects overflow properly and uses it to perform any saturation operation on the result, as long as the shifted result is within 33 bits. Otherwise, the result is undefined. Note: For word and DWord operands, the accumulators have 33 bits.</p>			
<p>In QWord mode, the shift count is taken from the low six bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 63. Otherwise the shift count is taken from the low five bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 31. The operation uses QWord mode if src0 or dst has the Q or UQ type but not if src1 is the only operand with the Q or UQ type.</p>			BDW
Format: [(pred)] shl[.cmod] (exec_size) dst src0 src1			
Restriction			
Accumulator cannot be destination, implicit or explicit.			
Syntax			
[(pred)] shl[.cmod] (exec_size) reg reg reg [(pred)] shl[.cmod] (exec_size) reg reg imm32			
Pseudocode			Project
Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { shiftCnt = src0 or dst has Q or UQ type ? src1.chan[n] & 0x3F : src1.chan[n] & 0x1F dst.chan[n] = src0.chan[n] << shiftCnt; } }			BDW
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
*B,*W,*D	*B,*W,*D		
*W,*D,*Q	*W,*D,*Q	BDW	
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		



shl - Shift Left		
	63:32	Operand Controls Format: EU_INSTRUCTION_OPERAND_CONTROLS
	31:0	Header Format: EU_INSTRUCTION_HEADER

Shift Right

shr - Shift Right			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>Perform component-wise logical right shift with zero insertion of the bits in src0 by the shift count indicated in src1, storing the results in dst. Insert zero bits in the number of MSBs indicated by the shift count. src0 and dst can have different types and can be signed or unsigned. Note: For word and DWord operands, the accumulators have 33 bits. Note: For unsigned src0 types, shr and asr produce the same result.</p> <p>In QWord mode, the shift count is taken from the low six bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 63. Otherwise the shift count is taken from the low five bits of src1 regardless of the src1 type and treated as an unsigned integer in the range 0 to 31. The operation uses QWord mode if src0 or dst has the Q or UQ type but not if src1 is the only operand with the Q or UQ type.</p> <p>Format: [(pred)] shr[.cmod] (exec_size) dst src0 src1</p>			
Syntax			
[(pred)] shr[.cmod] (exec_size) reg reg reg [(pred)] shr[.cmod] (exec_size) reg reg imm32			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < exec_size; n++) { if (WrEn.chan[n]) { shiftCnt = src0 or dst has Q or UQ type ? src1.chan[n] & 0x3F : src1.chan[n] & 0x1F dst.chan[n] = src0.chan[n] » shiftCnt; } }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types	Project	
UB, UW, UD	UB, UW, UD		
UW, UD, UQ	UW, UD, UQ	BDW	
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	



SIMD8 Render Target Write MSD

MSD_RTW_SIMD8 - SIMD8 Render Target Write MSD			
Project:	BDW		
Source:	Render Cache DataPort		
Length Bias:	1		
Family:	Other		
Group:	Render Target R/W		
DWord	Bit	Description	
0	31	Reserved	
		Project: All	
		Format: MBZ	
			Ignored
	30	Message Precision Subtype	
		Default Value: 0h	
		Project: All	
		Format: Opcode	
			Full precision data message
	29	Reserved	
		Project: All	
		Format: MBZ	
			Ignored
	28:25	Message Length	
		Project: All	
		Format: U4	
			Specifies the number of 256-bit GRF registers sent as the message payload (including the header). Valid value ranges are 1 to 15.
	24:20	Response Length	
		Project: All	
		Format: U5	
		Specifies the number of 256-bit GRF registers expected as the message response payload. Valid value ranges are 0 to 16.	

MSD_RTW_SIMD8 - SIMD8 Render Target Write MSD							
19	<p>Header Present</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the 2-register header.</p>	Project:	All	Format:	MDC_MHP		
Project:	All						
Format:	MDC_MHP						
18	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
17:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Render Target Write message</p>	Default Value:	0Ch	Project:	All	Format:	Opcode
Default Value:	0Ch						
Project:	All						
Format:	Opcode						
13	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table> <p>Ignored</p>	Project:	BDW	Format:	MBZ		
Project:	BDW						
Format:	MBZ						
12	<p>Last Render Target Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>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. In general, when threads are not launched by 3D FF, this bit must be zero.</p> <table border="1" style="width: 100%; background-color: #e6f2ff;"> <tr> <td style="text-align: center;">Programming Notes</td> </tr> </table> <p>When a pixel shader has render target writes at finer granularity than the dispatch rate, last render target write to a null surface must be present at the dispatch rate with this bit set. In particular, if a kernel is dispatched at pixel rate and it only writes to render targets at sample-rate, it must include a pixel-rate render target write to a null surface with Last Render Target Select bit enabled.</p>	Project:	All	Format:	Enable	Programming Notes	
Project:	All						
Format:	Enable						
Programming Notes							
11	<p>Slot Group Select</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_RT_SGS</td> </tr> </table> <p>This field selects whether slots 15:0 or slots 31:16 are used for bypassed data.</p>	Project:	All	Format:	MDC_RT_SGS		
Project:	All						
Format:	MDC_RT_SGS						



MSD_RTW_SIMD8 - SIMD8 Render Target Write MSD	
10:8	Render Target Message Subtype
	Default Value: 4h
	Project: All
	Format: Opcode
	SIMD8 single source message. Use slots [7:0] for pixel enables, X/Y addresses, and oMask.
Programming Notes	
The above slots indicated are within the 16 slots selected by Slot Group Select. If SLOTGRP_HI is selected, slots [23:16] are referenced instead of [7:0].	
7:0	Binding Table Index
	Project: All
	Format: MDC_BTS
Specifies the Binding Table Index for the message	

SIMD16 Render Target Write MSD

MSD_RTW_SIMD16 - SIMD16 Render Target Write MSD		
Project:	BDW	
Source:	Render Cache DataPort	
Length Bias:	1	
Family:	Other	
Group:	Render Target R/W	
DWord	Bit	Description
0	31	Reserved
		Project: All
		Format: MBZ
	Ignored	
	30	Message Precision Subtype
Default Value: 0h		
Project: All		
Format: Opcode		
Full precision data message		
29	Reserved	
	Project: All	
	Format: MBZ	
Ignored		
28:25	Message Length	
	Project: All	
	Format: U4	
Specifies the number of 256-bit GRF registers sent as the message payload (including the header). Valid value ranges are 1 to 15.		
24:20	Response Length	
	Project: All	
	Format: U5	
Specifies the number of 256-bit GRF registers expected as the message response payload. Valid value ranges are 0 to 16.		

MSD_RTW_SIMD16 - SIMD16 Render Target Write MSD

19	Header Present	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table>	Project:	All	Format:	MDC_MHP			
Project:	All								
Format:	MDC_MHP								
		If set, indicates that the message includes the 2-register header.							
18	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	Ignored		
Project:	BDW								
Format:	MBZ								
17:14	Message Type	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>0Ch</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table>	Default Value:	0Ch	Project:	All	Format:	Opcode	Render Target Write message
Default Value:	0Ch								
Project:	All								
Format:	Opcode								
13	Reserved	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:	MBZ	Ignored		
Project:	BDW								
Format:	MBZ								
12	Last Render Target Select	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table>	Project:	All	Format:	Enable	<p>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. In general, when threads are not launched by 3D FF, this bit must be zero.</p> <div style="background-color: #e6f2ff; padding: 5px; text-align: center; font-weight: bold; color: #0070c0;">Programming Notes</div> <p>When a pixel shader has render target writes at finer granularity than the dispatch rate, last render target write to a null surface must be present at the dispatch rate with this bit set. In particular, if a kernel is dispatched at pixel rate and it only writes to render targets at sample-rate, it must include a pixel-rate render target write to a null surface with Last Render Target Select bit enabled.</p>		
Project:	All								
Format:	Enable								
11	Slot Group Select	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_RT_SGS</td> </tr> </table>	Project:	All	Format:	MDC_RT_SGS	This field selects whether slots 15:0 or slots 31:16 are used for bypassed data.		
Project:	All								
Format:	MDC_RT_SGS								

MSD_RTW_SIMD16 - SIMD16 Render Target Write MSD							
10:8	<p>Render Target Message Subtype</p> <table border="1"> <tr> <td>Default Value:</td> <td>0h</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>SIMD16 Single source message. Use slots [15:0] for pixel enables, X/Y addresses, and oMask.</p> <p style="text-align: center;">Programming Notes</p> <p>The above slots indicated are within the 16 slots selected by Slot Group Select. If SLOTGRP_HI is selected, slots [31:16] are referenced instead of [15:0].</p>	Default Value:	0h	Project:	All	Format:	Opcode
Default Value:	0h						
Project:	All						
Format:	Opcode						
7:0	<p>Binding Table Index</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS		
Project:	All						
Format:	MDC_BTS						



STATE_BASE_ADDRESS

STATE_BASE_ADDRESS		
Project:	BDW	
Length Bias:	2	
<p>The STATE_BASE_ADDRESS command sets the base pointers for subsequent state, instruction, and media indirect object accesses by the GPE.</p> <p>For more information see the Base Address Utilization table in the Memory Access Indirection narrative topic.</p>		
Programming Notes		
<p>The following commands must be reissued following any change to the base addresses:</p> <ul style="list-style-type: none"> • 3DSTATE_CC_POINTERS • 3DSTATE_BINDING_TABLE_POINTERS • 3DSTATE_SAMPLER_STATE_POINTERS • 3DSTATE_VIEWPORT_STATE_POINTERS • MEDIA_STATE_POINTERS <p>Execution of this command causes a full pipeline flush, thus its use should be minimized for higher performance.</p>		
<p>SW must always program PIPE_CONTROL with "CS Stall" and "Render Target Cache Flush Enable" set before programming STATE_BASE_ADDRESS command for GPGPU workloads i.e when pipeline select is GPGPU via PIPELINE_SELECT command. This is required to achieve better GPGPU preemption latencies in certain workload programming sequences. If programming PIPE_CONTROL has performance implications then preemption latencies can be traded off against performance by not implementing this programming note.</p>		
BDW		
DWord	Bit	Description
0	31:29	Command Type Default Value: 3h GFXPIPE
	28:27	Command SubType Default Value: 0h GFXPIPE_COMMON
	26:24	3D Command Opcode Default Value: 1h GFXPIPE_NONPIPELINED
	23:16	3D Command Sub Opcode Default Value: 01h STATE_BASE_ADDRESS
	15:8	Reserved Project: All Format: MBZ

STATE_BASE_ADDRESS												
	7:0	DWord Length										
		Project: BDW										
		Format: =n Total Length - 2										
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Eh</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	Eh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)				
Value	Name	Description										
Eh	DWORD_COUNT_n [Default]	Excludes DWord (0,1)										
1..2	63:12	<p>General State Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:12]</td> </tr> </table> <p>Specifies the 4K-byte aligned base address for general state accesses. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p> <p style="text-align: center;">Programming Notes</p> <p>Bounds checking is performed on general state accesses by Data Port Shared Functions for stateless A32 messages. Bounds checking is enabled when General State Base Address [46:12] + General State Buffer Size [31:12] is $\leq 2^{47}$. This ensures that the General State Buffer does not straddle the canonical address boundary where GraphicsAddress [47] changes.</p> <table border="1"> <thead> <tr> <th>Restriction</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>General State Base Address [47:12] + General State Buffer Size [31:12] must be $< 2^{48}$. It is illegal programming for this to be $\geq 2^{48}$.</td> <td></td> </tr> <tr> <td>When using stateless (A32) Data Port messages, General State Base Address [47:12] + Buffer Base Address [31:0] must be $< 2^{48}$. It is illegal for this to be $\geq 2^{48}$.</td> <td>BDW</td> </tr> </tbody> </table>	Project:	All	Format:	GraphicsAddress[63:12]	Restriction	Project	General State Base Address [47:12] + General State Buffer Size [31:12] must be $< 2^{48}$. It is illegal programming for this to be $\geq 2^{48}$.		When using stateless (A32) Data Port messages, General State Base Address [47:12] + Buffer Base Address [31:0] must be $< 2^{48}$. It is illegal for this to be $\geq 2^{48}$.	BDW
Project:	All											
Format:	GraphicsAddress[63:12]											
Restriction	Project											
General State Base Address [47:12] + General State Buffer Size [31:12] must be $< 2^{48}$. It is illegal programming for this to be $\geq 2^{48}$.												
When using stateless (A32) Data Port messages, General State Base Address [47:12] + Buffer Base Address [31:0] must be $< 2^{48}$. It is illegal for this to be $\geq 2^{48}$.	BDW											
	11	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All											
Format:	MBZ											
	10:4	<p>General State Memory Object Control State</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state for indirect state using the General State Base Address, with the exception of the stateless data port accesses.</p>	Project:	All	Format:	MEMORY_OBJECT_CONTROL_STATE						
Project:	All											
Format:	MEMORY_OBJECT_CONTROL_STATE											
	3:1	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
Project:	All											
Format:	MBZ											

STATE_BASE_ADDRESS												
	0	General State Base Address Modify Enable										
		Project:	All									
		Format:	Enable									
		The other fields in this DWord and the following DWord are updated only when this bit is set.										
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Ignore the updated address.</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Modify the address.</td> <td>All</td> </tr> </tbody> </table>	Value	Name	Description	Project	0h	Disable	Ignore the updated address.	All	1h	Enable
Value	Name	Description	Project									
0h	Disable	Ignore the updated address.	All									
1h	Enable	Modify the address.	All									
3	31:23	Reserved										
		Project:	All									
		Format:	MBZ									
	22:16	Stateless Data Port Access Memory Object Control State										
		Project:	All									
		Format:	MEMORY_OBJECT_CONTROL_STATE									
	Specifies the memory object control state for stateless data port accesses.											
	15:0	Reserved										
		Project:	All									
		Format:	MBZ									
4..5	63:12	Surface State Base Address										
		Project:	All									
		Format:	GraphicsAddress[63:12]									
		Specifies the 4K-byte aligned base address for binding table and surface state accesses. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].										
	11	Reserved										
		Project:	All									
		Format:	MBZ									
	10:4	Surface State Memory Object Control State										
		Project:	All									
		Format:	MEMORY_OBJECT_CONTROL_STATE									
	Specifies the memory object control state for indirect state using the Surface State Base Address .											
	3:1	Reserved										
		Project:	All									
		Format:	MBZ									

STATE_BASE_ADDRESS																		
6..7	0	<p>Surface State Base Address Modify Enable</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Enable</td> </tr> </table> <p>The other fields in this DWord and the following DWord are updated only when this bit is set.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Ignore the updated address.</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Modify the address.</td> <td>All</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>Setting this bit to 1 in a batch buffer causes the resource streamer to stop; for performance reasons the SW should only place commands with this bit set in the ring buffer.</p> <p>Before programming the Surface State Base Address, the RS must be disabled. Within a batch buffer where the RS is enabled, RS may be disabled thru a MI_RS_CONTROL command with Resource Streamer Control cleared prior to the STATE_BASE_ADDRESS with Surface State Base Address Modify Enable set and then re-enabled with another MI_RS_CONTROL with Resource Streamer Control set.</p>	Project:	All	Format:	Enable	Value	Name	Description	Project	0h	Disable	Ignore the updated address.	All	1h	Enable	Modify the address.	All
	Project:	All																
	Format:	Enable																
	Value	Name	Description	Project														
	0h	Disable	Ignore the updated address.	All														
	1h	Enable	Modify the address.	All														
	63:12	<p>Dynamic State Base Address</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[63:12]</td> </tr> </table> <p>Specifies the 4K-byte aligned base address for sampler and viewport state accesses. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].</p>	Project:	All	Format:	GraphicsAddress[63:12]												
	Project:	All																
	Format:	GraphicsAddress[63:12]																
	11	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ												
Project:	All																	
Format:	MBZ																	
10:4	<p>Dynamic State Memory Object Control State</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state for indirect state using the Dynamic State Base Address. Push constants defined in 3DSTATE_CONSTANT_(VS GS PS) commands do not use this control state, although they can use the corresponding base address. The memory object control state for push constants is defined within the command.</p>	Project:	All	Format:	MEMORY_OBJECT_CONTROL_STATE													
Project:	All																	
Format:	MEMORY_OBJECT_CONTROL_STATE																	
3:1	<p>Reserved</p> <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ													
Project:	All																	
Format:	MBZ																	

		STATE_BASE_ADDRESS			
8..9	0	Dynamic State Base Address Modify Enable			
		Project:	All		
		Format:	Enable		
		The other fields in this DWord and the following DWord are updated only when this bit is set.			
		Value	Name	Description	Project
		0h	Disable	Ignore the updated address.	All
		1h	Enable	Modify the address.	All
	63:12	Indirect Object Base Address			
		Project:	All		
		Format:	GraphicsAddress[63:12]IndirectObject		
	Specifies the 4K-byte aligned base address for indirect object load in MEDIA_OBJECT command.				
11	Reserved				
	Project:	All			
	Format:	MBZ			
10:4	Indirect Object Memory Object Control State				
	Project:	All			
	Format:	MEMORY_OBJECT_CONTROL_STATE			
	Specifies the memory object control state for indirect objects using the Indirect Object Base Address .				
3:1	Reserved				
	Project:	All			
	Format:	MBZ			
0	Indirect Object Base Address Modify Enable				
	Project:	All			
	Format:	Enable			
	The other fields in this DWord and the following DWord are updated only when this bit is set.				
	Value	Name	Description	Project	
	0h	Disable	Ignore the updated address.	All	
	1h	Enable	Modify the address.	All	
10..11	63:12	Instruction Base Address			
	Project:	All			
	Format:	GraphicsAddress[63:12]			
	Specifies the 4K-byte aligned base address for all EU instruction accesses. GraphicsAddress[63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].				

STATE_BASE_ADDRESS					
	11	Reserved			
		Project:	All		
		Format:	MBZ		
	10:4	Instruction Memory Object Control State			
		Project:	All		
		Format:	MEMORY_OBJECT_CONTROL_STATE		
	Specifies the memory object control state for EU instructions using the Instruction Base Address .				
	3:1	Reserved			
		Project:	All		
		Format:	MBZ		
	0	Instruction Base Address Modify Enable			
		Project:	All		
		Format:	Enable		
	The other fields in this DWord and the following DWord are updated only when this bit is set.				
		Value	Name	Description	Project
		0h	Disable	Ignore the updated address.	All
		1h	Enable	Modify the address.	All
	12	31:12	General State Buffer Size		
			Project:	All	
			Format:	U20	
FormatDesc					
This field specifies the size of the buffer in 4K pages. Any access that straddles or goes past the end of the buffer returns 0. Note that BufferSize=0 indicates that there is no valid data in the buffer.					
Workaround			Project		
When the General State Buffer Size is programmed to 0, SLM accesses are treated as out-of-bounds (should only apply to Stateless accesses). Workaround is to program the General State Buffer Size to a value > 0.			BDW		
If Per Thread Scratch Space Size bounds checking is enabled by GT_MODE[15], then General State Buffer Size must be set larger than the maximum Per Thread Scratch Space Size.			BDW		
11:1			Reserved		
			Project:	All	
	Format:	MBZ			

STATE_BASE_ADDRESS					
	0	General State Buffer Size Modify Enable			
		Project:	All		
		Format:	Enable		
		The bound in this DWord is updated only when this bit is set.			
		Value	Name	Description	Project
		0h	Disable	Ignore the updated bound.	All
1h	Enable	Modify the updated bound.	All		
13	31:12	Dynamic State Buffer Size			
		Project:	All		
		Format:	U20		
		FormatDesc This field specifies the size of the buffer in 4K pages. Any access that straddles or goes past the end of the buffer returns 0. Note that BufferSize=0 indicates that there is no valid data in the buffer.			
	11:1	Reserved			
		Project:	All		
			Format:	MBZ	
		0	Dynamic State Buffer Size Modify Enable		
			Project:	All	
			Format:	Enable	
FormatDesc The bound in this DWord is updated only when this bit is set.					
Value			Name	Description	Project
0h			Disable	Ignore the updated bound.	All
1h	Enable	Modify the updated bound.	All		
14	31:12	Indirect Object Buffer Size			
		Project:	All		
		Format:	U20		
		FormatDesc This field specifies the size of the buffer in 4K pages. Any access that straddles or goes past the end of the buffer returns 0. Note that BufferSize=0 indicates that there is no valid data in the buffer.			

STATE_BASE_ADDRESS															
	11:1	Reserved													
		Project:	All												
	Format:	MBZ													
	0	Indirect Object Buffer Size Modify Enable													
		Project:	All												
		Format:	Enable												
		FormatDesc													
		The bound in this DWord is updated only when this bit is set.													
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Ignore the updated bound.</td> <td>All</td> </tr> <tr> <td>1h</td> <td>Enable</td> <td>Modify the updated bound.</td> <td>All</td> </tr> </tbody> </table>			Value	Name	Description	Project	0h	Disable	Ignore the updated bound.	All	1h	Enable	Modify the updated bound.
	Value	Name	Description	Project											
0h	Disable	Ignore the updated bound.	All												
1h	Enable	Modify the updated bound.	All												
15	31:12	Instruction Buffer Size													
		Project:	All												
		Format:	U20												
		FormatDesc													
	This field specifies the size of the buffer in 4K pages. Any access that straddles or goes past the end of the buffer returns 0.														
	Note that BufferSize=0 indicates that there is no valid data in the buffer.														
	11:1	Reserved													
		Project:	All												
		Format:	MBZ												
		0	Instruction Buffer size Modify Enable												
Project:			All												
Format:			Enable												
FormatDesc															
The bound in this DWord is updated only when this bit is set.															
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> <th>Project</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Disable</td> <td>Ignore the updated bound.</td> <td>All</td> </tr> </tbody> </table>			Value	Name	Description	Project	0h	Disable	Ignore the updated bound.	All					
Value	Name	Description	Project												
0h	Disable	Ignore the updated bound.	All												

STATE_PREFETCH

STATE_PREFETCH			
Project:	BDW		
Length Bias:	2		
<p>(This command is provided strictly for performance optimization opportunities, and likely requires some experimentation to evaluate the overall impact of additional prefetching.)</p> <p>The STATE_PREFETCH command causes the GPE to attempt to prefetch a sequence of 64-byte cache lines into the GPE-internal cache ("L2 ISC") used to access EU kernel instructions and fixed/shared function indirect state data. While state descriptors, surface state, and sampler state are automatically prefetched by the GPE, this command may be used to prefetch data not automatically prefetched, such as: 3D viewport state; Media pipeline Interface Descriptors; EU kernel instructions.</p>			
Restriction		Project	
Restriction BDW bug# 1910068 : Due to know HW issue this command doesn't achieve its intended purpose and must not be exercised/programmed by SW.		BDW	
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value: 3h GFXPIPE	
	28:27	Command SubType	
		Default Value: 0h GFXPIPE_COMMON	
	26:24	3D Command Opcode	
		Default Value: 0h GFXPIPE_PIPELINED	
	23:16	3D Command Sub Opcode	
Default Value: 03h STATE_PREFETCH			
15:8	Reserved		
	Project: All		
	Format: MBZ		
7:0	DWord Length	Project: All	
		Format: =n Total Length - 2	
	Value	Name	Description
	0h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
1	31:6	Prefetch Pointer	
		Project: All	
		Format: GraphicsAddress[31:6]	
<p>Specifies the 64-byte aligned address to start the prefetch from. This pointer is an absolute virtual address, it is not relative to any base pointer.</p>			

STATE_PREFETCH												
	5:3	Reserved <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ						
	Project:	All										
Format:	MBZ											
	2:0	Prefetch Count <table border="1"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>U3-1 count of cache lines</td> </tr> </table> <p>Indicates the number of contiguous 64-byte cache lines that will be prefetched.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>[0,7]</td> <td></td> <td>indicating a count of [1,8]</td> </tr> </tbody> </table>	Project:	All	Format:	U3-1 count of cache lines	Value	Name	Description	[0,7]		indicating a count of [1,8]
Project:	All											
Format:	U3-1 count of cache lines											
Value	Name	Description										
[0,7]		indicating a count of [1,8]										

STATE_SIP

STATE_SIP			
Project:	All		
Length Bias:	2		
The STATE_SIP command specifies the starting instruction location of the System Routine that is shared by all threads in execution.			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h GFXPIPE
	28:27	Command SubType	
		Default Value:	0h GFXPIPE_COMMON
	26:24	3D Command Opcode	
		Default Value:	1h GFXPIPE_NONPIPELINED
	23:16	3D Command Sub Opcode	
	Default Value:	02h STATE_SIP	
15:8	Reserved		
	Project:	All	
	Format:	MBZ	
7:0	DWord Length		
	Project:	All	
	Format:	=n Total Length - 2	
	Value	Name	Description
	1h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)
1..2 Project: BDW	63:4	System Instruction Pointer	
		Project:	All
	Format:	InstructionBaseOffset[63:4]Kernel	
	Specifies the instruction address of the system routine associated with the current context as a 128-bit granular offset from the Instruction Base Address. SIP is shared by all threads in execution. The address specifies the double quadword aligned instruction location. GraphicsAddress [63:48] are ignored by the HW and assumed to be in correct canonical form [63:48] == [47].		
	Programming Notes		
	This portion of the command is not context save/restored. The context image may restore this command as a 2 dword command rather than a 3 dword command.		
3:0	Reserved		
	Project:	All	
	Format:	MBZ	

Sum of Absolute Difference 2

sad2 - Sum of Absolute Difference 2			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The sad2 instruction takes source data channels from src0 and src1 in groups of 2-tuples. For each 2-tuple, it computes the sum-of-absolute-difference (SAD) between src0 and src1 and stores the scalar result in the first channel of the 2-tuple in dst. The results are also stored in the accumulator register. The destination operand and the accumulator maintain 16 bits per channel precision. The destination register must be aligned to even word (DWord). The even words in the destination region will contain the correct data. The odd words are also written but with undefined values.</p>			
Format: [(pred)] sad2[.cmod] (exec_size) dst src0 src1			
Restriction			
Source operands cannot be accumulators.			
The execution size cannot be 1 as the computation requires at least two data channels.			
Syntax			
[(pred)] sad2[.cmod] (exec_size) reg reg reg [(pred)] sad2[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n += 2) { if (WrEn.chan[n]) { dst.chan[n] = abs(src0.chan[n] - src1.chan[n]) + abs(src0.chan[n+1] - src1.chan[n+1]); } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
B, UB	W, UW		
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	



Sum of Absolute Difference Accumulate 2

sada2 - Sum of Absolute Difference Accumulate 2			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The sada2 instruction takes source data channels from src0 and src1 in groups of 2-tuples. For each 2-tuple, it computes the sum-of-absolute-difference (SAD) between src0 and src1, adds the intermediate result with the accumulator value corresponding to the first channel, and stores the scalar result in the first channel of the 2-tuple in dst. The destination operand and the accumulator maintain 16 bits per channel precision. Higher precision (guide bits) stored in the accumulator allows up to 64 rounds of sada2 instructions to be issued back to back without overflowing the accumulator. The destination register must be aligned to even word (DWord). The even words in the destination region will contain the correct data. The odd words are also written but with undefined values.</p>			
Format: [(pred)] sada2[.cmod] (exec_size) dst src0 src1			
Restriction			
Source operands cannot be accumulators.			
The execution size cannot be 1 as the computation requires at least two data channels.			
Syntax			
[(pred)] sada2[.cmod] (exec_size) reg reg reg [(pred)] sada2[.cmod] (exec_size) reg reg imm32			
Pseudocode			
Evaluate(WrEn); for (n = 0; n < exec_size; n += 2) { uwTmp = abs(src0.chan[n] - src1.chan[n]) + abs(src0.chan[n+1] - src1.chan[n+1]); if (WrEn.chan[n]) { dst.chan[n] = uwTmp + acc[n]; } }			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	Y	Y	Y
Src Types	Dst Types		
B, UB	W, UW		
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

sada2 - Sum of Absolute Difference Accumulate 2		
	31:0	Header
	Format:	EU_INSTRUCTION_HEADER

SWTESS_BASE_ADDRESS

SWTESS_BASE_ADDRESS								
Project:	BDW							
Length Bias:	2							
The SWTESS_BASE_ADDRESS command sets the base pointers for SW Tessellation data read access by the TE unit.								
Programming Notes								
This base address must also be comprehended in the SURFACE_STATE used by the HS kernel to write the SW tessellation data.								
Execution of this command causes a full pipeline flush, thus its use should be minimized for higher performance.								
DWord	Bit	Description						
0	31:29	Command Type Default Value: 3h GFXPIPE						
	28:27	Command SubType Default Value: 0h GFXPIPE_COMMON						
	26:24	3D Command Opcode Default Value: 1h GFXPIPE_NONPIPELINED						
	23:16	3D Command Sub Opcode Default Value: 03h SWTESS_BASE_ADDRESS						
	15:8	Reserved Project: All Format: MBZ						
	7:0	DWord Length Project: All Format: =n Total Length - 2						
			<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0h</td> <td>DWORD_COUNT_n [Default]</td> <td>Excludes DWord (0,1)</td> </tr> </tbody> </table>	Value	Name	Description	0h	DWORD_COUNT_n [Default]
Value	Name	Description						
0h	DWORD_COUNT_n [Default]	Excludes DWord (0,1)						
1	31:12	SW Tessellation Base Address Project: All Format: GraphicsAddress[31:12]						
		Specifies the 4K-byte aligned base address for TE unit SW tessellation data read accesses.						

SWTESS_BASE_ADDRESS						
	11:8	SW Tessellation Memory Object Control State <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MEMORY_OBJECT_CONTROL_STATE</td> </tr> </table> <p>Specifies the memory object control state used by the TE unit to read SW tessellation data from memory.</p>	Project:	All	Format:	MEMORY_OBJECT_CONTROL_STATE
	Project:	All				
Format:	MEMORY_OBJECT_CONTROL_STATE					
	7:0	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
Project:	All					
Format:	MBZ					
2 Project: BDW	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	All	Format:	MBZ
	Project:	All				
Format:	MBZ					
	15:0	SW Tessellation Base Address High <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Specifies most significant bits of the 4K-byte aligned base address for TE unit SW tessellation data read accesses. See SW Tessellation Base Address[31:12] in DWord 0.</p>	Project:	BDW	Format:	GraphicsAddress[47:32]
Project:	BDW					
Format:	GraphicsAddress[47:32]					

Typed Surface Read MSD

MSD1R_TS - Typed Surface Read MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Typed Surface R/W	
Group:	Scattered Typed Surface R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	If set, indicates that the message includes the header.	
	18:14	Message Type
Default Value: 05h		
Project: All		
Format: Opcode		
Typed Surface Read message		
13:12	Slot Group	
	Project: All	
	Format: MDC_SG3	
Specifies the Slot Group mode of the message (which slots are processed)		
11:8	Channel Mask	
	Project: All	
	Format: MDC_CMASK	
Specifies which RGBA channels are included in the message payload.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS	
Specifies the Binding Table Index for the message		

Typed Surface Write MSD

MSD1W_TS - Typed Surface Write MSD								
Project:	BDW							
Source:	DataPort 1							
Length Bias:	1							
Family:	Typed Surface R/W							
Group:	Scattered Typed Surface R/W							
DWord	Bit	Description						
0	19	<p>Header Present</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_MHP</td> </tr> </table> <p>If set, indicates that the message includes the header.</p>	Project:	All	Format:	MDC_MHP		
	Project:	All						
	Format:	MDC_MHP						
	18:14	<p>Message Type</p> <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>0Dh</td> </tr> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table> <p>Typed Surface Write message</p>	Default Value:	0Dh	Project:	All	Format:	Opcode
	Default Value:	0Dh						
Project:	All							
Format:	Opcode							
13:12	<p>Slot Group</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_SG3</td> </tr> </table> <p>Specifies the Slot Group mode of the message (which slots are processed)</p>	Project:	All	Format:	MDC_SG3			
Project:	All							
Format:	MDC_SG3							
11:8	<p>Channel Mask</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_CMASK</td> </tr> </table> <p>Specifies which RGBA channels are included in the message payload.</p>	Project:	All	Format:	MDC_CMASK			
Project:	All							
Format:	MDC_CMASK							
7:0	<p>Binding Table Index</p> <table border="1" style="width: 100%;"> <tr> <td>Project:</td> <td>All</td> </tr> <tr> <td>Format:</td> <td>MDC_BTS</td> </tr> </table> <p>Specifies the Binding Table Index for the message</p>	Project:	All	Format:	MDC_BTS			
Project:	All							
Format:	MDC_BTS							

Untyped Surface Write MSD

MSD1W_US - Untyped Surface Write MSD		
Project:	BDW	
Source:	DataPort 1	
Length Bias:	1	
Family:	Untyped Surface R/W	
Group:	Scattered Untyped Surface R/W	
DWord	Bit	Description
0	19	Header Present
		Project: All
		Format: MDC_MHP
	If set, indicates that the message includes the header.	
	18:14	Message Type
Default Value: 09h		
Project: All		
Format: Opcode		
Untyped Surface Write message		
13:12	SIMD Mode	
	Project: All	
	Format: MDC_SM3	
Specifies the SIMD mode of the message (number of slots processed)		
11:8	Channel Mask	
	Project: All	
	Format: MDC_UW_CMASK	
Specifies which RGBA channels are included in the message payload.		
7:0	Binding Table Index	
	Project: All	
	Format: MDC_BTS_SLM_A32	
Specifies the Binding Table Index for the message		

URB Hword Dual Block Read MSD

MSD_UR_HWDB - URB Hword Dual Block Read MSD			
Project:	BDW		
Source:	Read-Only DataPort		
Length Bias:	1		
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHR
	Indicates that the message requires a header.		
	18	Reserved	
	Project:	All	
	Format:	MBZ	
Ignored			
	17	Per Slot Offset	
		Format:	MHC_PSOP
Specifies if per-slot offsets are present and will be added to the Global Offset .			
	16	Reserved	
		Project:	All
		Format:	MBZ
Ignored			
	15	Swizzle Control	
		Project:	All
		Format:	Opcode
		Value	Name
		1	URB_INTERLEAVED [Default]
			Description
			Use two URB entries (URB Handle 0 and URB Handle 1).
			Project
			All
	14:4	Global Offset	
		Project:	All
		Format:	U11
Specifies the offset, in units of Hword elements, from the start of the URB handle for the access. If Per Slot Offset is set, the global offset is added to those offsets to form the overall offset. Range [0,1023]			



MSD_UR_HWDB - URB Hword Dual Block Read MSD

3:0	URB Opcode									
	Project:	All								
	Format:	Opcode								
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">2</td> <td>URB_READ_HWORD [Default]</td> <td>URB Hword Read message</td> <td>All</td> </tr> </tbody> </table>			Value	Name	Description	Project	2	URB_READ_HWORD [Default]	URB Hword Read message
Value	Name	Description	Project							
2	URB_READ_HWORD [Default]	URB Hword Read message	All							

URB Hword Dual Block Write MSD

MSD_UW_HWDB - URB Hword Dual Block Write MSD			
Project:	BDW		
Source:	Read-Only DataPort		
Length Bias:	1		
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHR
	Indicates that the message requires a header.		
	18	Reserved	
	Project:	All	
	Format:	MBZ	
Ignored			
	17	Per Slot Offset	
		Format:	MHC_PSOP
Specifies if per-slot offsets are present and will be added to the Global Offset .			
	16	Reserved	
		Project:	All
		Format:	MBZ
Ignored			
	15	Swizzle Control	
		Project:	All
		Format:	Opcode
		Value	Name
		1	URB_INTERLEAVED [Default]
			Description
			Use two URB entries (URB Handle 0 and URB Handle 1).
			Project
			All
	14:4	Global Offset	
		Project:	All
		Format:	U11
Specifies the offset, in units of Hword elements, from the start of the URB handle for the access. If Per Slot Offset is set, the global offset is added to those offsets to form the overall offset. Range [0,1023]			



MSD_UW_HWDB - URB Hword Dual Block Write MSD				
3:0	URB Opcode			
	Project:	All		
	Format:	Opcode		
	Value	Name	Description	Project
	0	URB_WRITE_HWORD [Default]	URB Hword Read message	All

URB Oword Block Write MSD

MSD_UW_OWB - URB Oword Block Write MSD				
Project:		BDW		
Source:		Read-Only DataPort		
Length Bias:		1		
DWord	Bit	Description		
0	19	Header Present		
		Project:	All	
		Format:	MDC_MHR	
	Indicates that the message requires a header.			
	18	Reserved		
		Project:	All	
Format:		MBZ		
Ignored				
17	Per Slot Offset			
	Format:	MHC_PSOP		
Specifies if per-slot offsets are present and will be added to the Global Offset .				
16	Reserved			
	Project:	All		
	Format:	MBZ		
Ignored				
15	Swizzle Control			
	Project:	All		
	Format:	Opcode		
	Value	Name	Description	Project
0	URB_NOSWIZZLE [Default]	Use a single URB entry (URB Handle 0).	All	
14:4	Global Offset			
	Project:	All		
	Format:	U11		
Specifies the offset, in units of Oword elements, from the start of the URB handle for the access. If Per Slot Offset is set, the global offset is added to those offsets to form the overall offset. Range [0,2047]				



MSD_UW_OWB - URB Oword Block Write MSD			
3:0	URB Opcode		
	Project:	All	
	Format:	Opcode	
	Value	Name	Description
1	URB_WRITE_OWORD [Default]	URB Oword Write message	All

URB Oword Dual Block Read MSD

MSD_UR_OWDB - URB Oword Dual Block Read MSD			
Project:		BDW	
Source:		Read-Only DataPort	
Length Bias:		1	
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHR
	Indicates that the message requires a header.		
	18	Reserved	
	Project:	All	
	Format:	MBZ	
Ignored			
	17	Per Slot Offset	
		Format:	MHC_PSOP
Specifies if per-slot offsets are present and will be added to the Global Offset .			
	16	Reserved	
		Project:	All
		Format:	MBZ
Ignored			
	15	Swizzle Control	
		Project:	All
		Format:	Opcode
		Value	Name
		Description	Project
		1	URB_INTERLEAVED [Default]
		Use two URB entries (URB Handle 0 and URB Handle 1).	All
	14:4	Global Offset	
		Project:	All
		Format:	U11
Specifies the offset, in units of Oword elements, from the start of the URB handle for the access. If Per Slot Offset is set, the global offset is added to those offsets to form the overall offset. Range [0,2047]			



MSD_UR_OWDB - URB Oword Dual Block Read MSD

3:0	URB Opcode									
	Project:	All								
	Format:	Opcode								
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> <th style="text-align: center;">Description</th> <th style="text-align: center;">Project</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">3</td> <td>URB_READ_OWORD [Default]</td> <td>URB Oword Read message</td> <td>All</td> </tr> </tbody> </table>			Value	Name	Description	Project	3	URB_READ_OWORD [Default]	URB Oword Read message
Value	Name	Description	Project							
3	URB_READ_OWORD [Default]	URB Oword Read message	All							

URB Oword Dual Block Write MSD

MSD_UW_OWDB - URB Oword Dual Block Write MSD			
Project:		BDW	
Source:		Read-Only DataPort	
Length Bias:		1	
DWord	Bit	Description	
0	19	Header Present	
		Project:	All
		Format:	MDC_MHR
	Indicates that the message requires a header.		
	18	Reserved	
		Project:	All
	Format:	MBZ	
Ignored			
17	Per Slot Offset		
	Format:	MHC_PSOP	
Specifies if per-slot offsets are present and will be added to the Global Offset .			
16	Reserved		
	Project:	All	
	Format:	MBZ	
Ignored			
15	Swizzle Control		
	Project:	All	
	Format:	Opcode	
	Value	Name	Description
	1	URB_INTERLEAVED [Default]	Use two URB entries (URB Handle 0 and URB Handle 1).
			Project
			All
14:4	Global Offset		
	Project:	All	
	Format:	U11	
Specifies the offset, in units of Oword elements, from the start of the URB handle for the access. If Per Slot Offset is set, the global offset is added to those offsets to form the overall offset. Range [0,2047]			



MSD_UW_OWDB - URB Oword Dual Block Write MSD				
3:0	URB Opcode			
	Project:	All		
	Format:	Opcode		
	Value	Name	Description	Project
	1	URB_WRITE_OWORD [Default]	URB Oword Write message	All

VEBOX_STATE

VEBOX_STATE			
Project:	BDW		
Source:	VideoEnhancementCS		
Length Bias:	2		
<p>This command controls the internal functions of the VEBOX. This command has a set of indirect state buffers:</p> <ul style="list-style-type: none"> • DN/DI state • Capture Pipe state 			
DWord	Bit	Description	
0	31:29	Command Type	
		Default Value:	3h PARALLEL_VIDEO_PIPE
		Format:	OpCode
	28:27	Pipeline	
		Default Value:	2h Media
		Format:	OpCode
	26:24	Command OpCode	
		Default Value:	4h VEBOX
		Format:	OpCode
	23:21	SubOpcode A	
Default Value:		0h	
Format:		OpCode	
20:16	SubOpcode B		
	Default Value:	2h	
	Format:	OpCode	
15:12	Reserved		
	Format:	MBZ	
11:0	DWord Length		
	Format:	=n Total Length - 2	
	Value	Name	Description
	Ah		(Excludes DWords 0, 1)
1	31:25	State Surface Control Bits	
		Project:	BDW
		See definition under "VEB_DI_IECP_COMMAND_SURFACE_CONTROL_BITS [BDW]"	

VEBOX_STATE				
24:23	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
22	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
21	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
20	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
19:15	Reserved			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Project:	BDW	Format:
Project:	BDW			
Format:	MBZ			
14	Single Slice VEBOX Enable			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Project:</td> <td>BDW</td> </tr> </table> <p>For products that have 2 entire VEBOXes that automatically split the frame, this enable emulates a 1 VEBOX product, running at 1/2 speed and only outputting a single set of per command statistics.</p>	Project:	BDW	
Project:	BDW			
13	Hot Pixel Filtering Enable Enables hot pixel detection/filtering.			
12	Alpha Plane Enable Enables the reading of an independent Alpha plane. Mutually exclusive with Vignette Enable. If Alpha from State Select is set it overrides this bit.			
11	Vignette Enable Enables Vignette Correction surface read and correction in IECP. Mutually exclusive with Alpha Plane Enable.			
	Programming Notes			
	<table border="1" style="width: 100%;"> <tr> <td style="width: 80%;">Demosaic must also be enabled if this bit is enabled.</td> <td style="text-align: center;">Project</td> </tr> <tr> <td></td> <td style="text-align: center;">BDW</td> </tr> </table>	Demosaic must also be enabled if this bit is enabled.	Project	
Demosaic must also be enabled if this bit is enabled.	Project			
	BDW			
10	Demosaic Enable The Demosaic will be used, and White balance statistics will be gathered. The Capture Pipe State Table will be read. This bit is mutually exclusive with DI Enable .			

VEBOX_STATE			
9:8	DI Output Frames		
	Indicates which frames to output in DI mode.		
	Value	Name	
	00b	Output Both Frames	
	01b	Output Previous Frame Only	
	10b	Output Current Frame Only	
	Programming Notes		
	Field is ignored if DI Enable = 0. If Previous Frame Only or Current Frame Only are selected, then the LACE Single Histogram Set must not try to collect a histogram from the disabled frame.		
	Field must be programmed to 10 (Output Current Frame Only) for DI First Frame.		
	7	444 -> 422 Downsample Method	
Project:		BDW	
Value		Name	
1		Average horizontally aligned chromas	
0		Drop right chroma of the pair [Default]	
Programming Notes			
444->422		422->420	Description
0		0	No averaging, only down sampling
0		1	Not Supported
1		0	Only Horizontal averaging
1	1	Horizontal and Vertical averaging	
6	422 -> 420 Downsample Method		
	Project:	BDW	
	Value	Name	
	1	Average vertically aligned chromas	
	0	Drop lower chroma of the pair [Default]	
	Programming Notes		
	To enable averaging in case of 420 (NV12/P016) output formats, 444->422 and 422->420 should be set.		

VEBOX_STATE										
5	<p>DN/DI First Frame</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>Indicates that this is the first frame of the stream, so previous clean is not available.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Not first field; previous clean surface state is valid</td> </tr> <tr> <td>1</td> <td>First field; previous clean surface state is invalid</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>If both DN and DI are disabled, this bit must be 0.</p>	Format:	Enable	Value	Name	0	Not first field; previous clean surface state is valid	1	First field; previous clean surface state is invalid	
	Format:	Enable								
	Value	Name								
	0	Not first field; previous clean surface state is valid								
	1	First field; previous clean surface state is invalid								
	4	<p>DI Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>Deinterlacer is bypassed if this is disabled: the output is the same as the input (same as a 2:2 cadence). FMD and STMM are not calculated and the values in the response message are 0.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Do not calculate DI</td> </tr> <tr> <td>1</td> <td>Calculate DI</td> </tr> </tbody> </table>	Format:	Enable	Value	Name	0	Do not calculate DI	1	Calculate DI
		Format:	Enable							
		Value	Name							
		0	Do not calculate DI							
	1	Calculate DI								
3	<p>DN Enable</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>Enable</td> </tr> </table> <p>Denoise is bypassed if this is low - BNE is still calculated and output, but the denoised fields are not. VDI does not read in the denoised previous frame but uses the pointer for the original previous frame.</p> <table border="1" style="width: 100%; text-align: center;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Do not denoise frame</td> </tr> <tr> <td>1</td> <td>Denoise frame</td> </tr> </tbody> </table> <p style="text-align: center;">Programming Notes</p> <p>If DN and/or Hotpixel are the only functions enabled then the only output is the Denoised Output which is the same surface format as the input.</p>	Format:	Enable	Value	Name	0	Do not denoise frame	1	Denoise frame	
	Format:	Enable								
	Value	Name								
0	Do not denoise frame									
1	Denoise frame									
2	<p>Global IECP Enable</p> <p>Indicates if any of the IECP features is enabled. If this is disabled then no state will be read from any of the state pointers. If set then the IECP state will be read.</p>									
	<p>Color Gamut Compression Enable</p> <p>Indicates if the Gamut Compression feature is enabled. If set then the Gamut State will be read. VEB_VERTEXTABLE_STATE is only needed if this bit is set.</p>									
	<p>Color Gamut Expansion Enable</p> <p>Indicates if the Gamut Expansion feature is enabled. If set then the Gamut State will be read.</p>									
2	<p>31:12 DN/DI State Pointer Low</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>GraphicAddress[31:12]</td> </tr> </table> <p>Bits 31:12 of the starting address of the DN/DI State buffer. This points to a buffer containing the 10 Dwords of the DN/DI state.</p>	Format:	GraphicAddress[31:12]							
	Format:	GraphicAddress[31:12]								

VEBOX_SURFACE_STATE

VEBOX_SURFACE_STATE	
Project:	BDW
Source:	VideoEnhancementCS
Length Bias:	2
Description	Project
The input and output data containers accessed are called "surfaces". Surface state is sent to VEBOX via an inline state command rather than using binding tables. SURFACE_STATE contains the parameters defining each surface to be accessed, including its size, format, and offsets to its subsurfaces. The surface's base address is in the execution command. Despite having multiple input and output surfaces, we limit the number of surface states to one for input surfaces and one for output surfaces. The other surfaces are derived from the input/output surface states.	
The Current Frame Input surface uses the Input SURFACE_STATE	
The Previous Denoised Input surface uses the Input SURFACE_STATE. (For 12-bit Bayer pattern inputs this will be 8-bit.)	BDW
The Current Denoised Output surface uses the Input SURFACE_STATE. (For 12-bit Bayer pattern inputs this will be 8-bit.)	BDW
The STMM/Noise History Input surface uses the Input SURFACE_STATE with Tile-Y and Width/Height a multiple of 4.	
The STMM/Noise History Output surface uses the Input SURFACE_STATE with Tile-Y and Width/Height a multiple of 4.	
The FMD per block output / per Frame Output surface uses the Linear SURFACE_STATE (see note below).	
The Alpha surface uses the Linear A8 SURFACE_STATE with Width/Height equal to Input Surface. Pitch is width rounded to next 64.	BDW
The Vignette Correction surface uses the Linear 16-bit SURFACE_STATE with: Width = $4 * ((\text{Input Width} - 3) / 4)$ Height = $((\text{Input Height} - 3) / 4)$ Pitch in bytes is (vignette width*2) rounded to next 64.	BDW
The STMM height is the same as the Input Surface height except when the input Surface Format is Bayer Pattern and the Bayer Pattern Offset is 10 or 11, in which case the height is the input height + 4. For Bayer pattern inputs when the Bayer Pattern Offset is 10 or 11, the Current Denoised Output/Previous Denoised Input will also have a height which is the input height + 4. For Bayer pattern inputs only the Current Denoised Output/Previous Denoised Input are in Tile-Y.	BDW
The linear surface for FMD statistics is linear (not tiled). The height of the per block statistics is $(\text{Input Height} + 3) / 4$ - the Input Surface height in pixels is rounded up to the next even 4 and divided by 4. The width of the per block section in bytes is equal to the width of the Input Surface in pixels rounded up to the next 16 bytes. The pitch of the per block section in bytes is equal to the width of the Input Surface in pixels rounded up to the next 64 bytes.	
The STMM surfaces must be identical to the Input surface except for the tiling mode must be Tile-Y and the pitch must be legal for Tile-Y (increased to the next larger legal pitch). If the input surface is	BDW

VEBOX_SURFACE_STATE				
packed (Surface Format from 0 to 3 for DN/DI) or 12/10-bit Bayer Pattern then the pitch for the STMM surface is 1/2 the pitch of the input surface (rounded up to the next larger legal Tile-Y pitch). The width and height must be a multiple of 4 rounded up from the input height.				
Programming Notes			Project	
VEBOX may write to memory between the surface width and the surface pitch for output surfaces.				
For 8bit Alpha input, when converting to 16bit output it is padded with 8bit zeros in the LSB.			BDW	
DWord	Bit	Description		
0	31:29	Command Type		
		Default Value:	3h PARALLEL_VIDEO_PIPE	
	Format:		OpCode	
	28:27	Media Command Pipeline		
		Default Value:	2h Media	
	Format:		OpCode	
	26:24	Media Command OpCode		
		Default Value:	4h VEBOX	
	Format:		OpCode	
	23:21	SubOpCode A		
Default Value:		0h VEBOX		
Format:		OpCode		
20:16	SubOpCode B			
	Default Value:	0h VEBOX		
Format:		OpCode		
15:12	Reserved			
	Format:		MBZ	
11:0	DWord Length			
	Format: =n Total Length - 2			
	Value	Name	Description	Project
	4h	DWORD_COUNT_n [Default]	(Excludes DWords 0, 1)	BDW
1	31:1	Reserved		
		Format:		MBZ
	0	Surface Identification		
		Specifies which set of surfaces this command refers to:		
Value		Name		
1	Output surface (all except the Denoised Current output surface)			
0	Input surface and Denoised Current Output Surface			

VEBOX_SURFACE_STATE					
2	31:18	Height			
		Format: U14			
		This field specifies the height of the surface in units of pixels. For PLANAR surface formats, this field indicates the height of the Y (luma) plane.			
		Value	Name	Description	Exists If
		[15, 16383]		representing heights [16,16384]	
		[15, 8191]			//Scalar Enabled - For Input surface only
		[63, 2047]			//Scalar - For Input surface only
		Programming Notes			Project
		Height (field value + 1) must be a multiple of 2 for PLANAR_420 surfaces. Height (field value +1) must be a multiple of 2 when the deinterlace function is enabled (field mode) or when the denoise function is enabled with Progressive DN = 0. It must be a multiple of 4 when interleaved deinterlace/denoise and PLANAR_420 are both being used. VEBOX supports a minimum height of 16.			
		Height (field value + 1) must be a multiple of 2 for Bayer surfaces.			BDW
17:4	Width	Format: U14			
		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
		[63,16383]		representing widths [64,16384]	
		[63,8191]			//Scalar Enabled - For Input surface only
		[63,2047]			//Scalar Enabled - For Input Surface only
		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_* surfaces, and must be a multiple of 4 for PLANAR_411 surfaces. VEBOX supports a minimum width of 64			
		Reserved			
		Format: MBZ			
3	31:28	Surface Format			
		Project: BDW			
		Format: U4			
		Specifies the format of the surface. All of the Y and G channels will use table 0 and all of the Cr/Cb/R/B channels will use table 1.			
		Value	Name	Description	Project
0	YCRCB_NORMAL				
1	YCRCB_SWAPUVY				

VEBOX_SURFACE_STATE				
	2	YCRCB_SWAPUV		
	3	YCRCB_SWAPY		
	4	PLANAR_420_8	NV12 with Interleave Chroma set	
	5	Reserved	Reserved	
	6	Reserved	Reserved	
	7	Reserved	Reserved	
	8	Reserved	Reserved	BDW
	9	Reserved	Reserved	
	10	Reserved	Reserved	
	11	Y8_UNORM		
	12	Reserved	Reserved	
	13	Reserved	Reserved	BDW
	14	Bayer pattern	Demosaic input only	BDW
	15	Reserved		BDW
	27	Interleave Chroma		
	Project:		BDW	
	Format:		Enable	
	This field indicates that the chroma fields are interleaved in a single plane rather than stored as two separate planes. This field is only used for PLANAR surface formats.			
26:25	Bayer Pattern Offset			
	Project:		BDW	
	Specifies the starting pixel offset for the Bayer pattern used for Capture Pipe.			
	Value	Name		
	00b	Pixel at X=0, Y=0 is Blue		
	01b	Pixel at X=0, Y=0 is Red		
	10b	Pixel at X=0, Y=0 is Green, Pixel at X=1, Y=0 is Red		
	11b	Pixel at X=0, Y=0 is Green, Pixel at X=1, Y=0 is Blue		
24	Bayer Pattern Format			
	Project:		BDW	
	Specifies the format of the Bayer Pattern:			
	Value	Name	Project	
	0b	8-bit input at a 8-bit stride		
	1b	12 or 10-bit input at a 16-bit stride. Valid data is in the MSBs		BDW
23:21	Reserved			
	Project:		BDW	
	Format:		MBZ	

VEBOX_SURFACE_STATE			
20	Reserved		
	Project:	BDW	
	Format:	MBZ	
19:3	Surface Pitch		
	Format:	U17 pitch in (Bytes - 1)	
	This field specifies the surface pitch in (#Bytes - 1):		
	Value	Name	Description
	[63, 131071]	For other linear surfaces	[64B, 128KB]
	[511, 131071]	For X-tiled surface	[512B, 128KB] = [1tile, 256 tiles]
[127, 131071]	For Y-tiled surfaces	[128B,128KB] = [1 tile, 1024 tiles]	
Programming Notes			
For tiled surfaces, the pitch must be a multiple of the tile width. For linear surfaces, the pitch must be a multiple of 64. 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.			
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 programmed to Zero always as this field is not used			
1	Tiled Surface		
	Format:	Boolean	
	This field specifies whether the surface is tiled.		
	Value	Name	Description
	1	True	Tiled
	0	False	Linear
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.			

VEBOX_SURFACE_STATE		
0	Tile Walk	
	Format:	3D_TileWalk
	This field specifies the type of memory tiling (XMajor or YMajor) employed to tile this surface. See <i>Memory Interface Functions</i> for details on memory tiling and restrictions. This field is ignored when the surface is linear.	
	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.		
4	31:29	Reserved
	Format:	MBZ
	28:16	X Offset for U
	Format:	U13 Pixel Offset
	This field must be zero for the VEBOX surface formats	
15	Reserved	
	Format:	MBZ
14:0	Y Offset for U	
	Format:	U15 Row Offset
	This field specifies the vertical offset in rows from the start (origin) or the Luma(Y) plane 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	
	This field must indicate an even number (bit 0 = 0). This field must be evenly divisible by 4 for Tile-Y surfaces (so the offset points to the start of a cache line) For Planar formats, if the surface is in YS or YF tile modes, the Y Offset for U should be an integral multiple of the Tile height of the Luma plane	
5	31:29	Reserved
	Format:	MBZ
	28:16	X Offset for V
	Format:	U13 Pixel Offset
	This field must be zero for the VEBOX surface formats.	
15	Reserved	
	Format:	MBZ

VEBOX_SURFACE_STATE			
	14:0	Y Offset for V	
		Format:	U15 Row Offset
		<p>This field specifies the vertical offset in rows from the start (origin) of the Luma(Y) plane to the start (origin) of the V(Cr) plane. This field is only used for PLANAR surface formats with Interleave Chroma disabled.</p>	
		Programming Notes	
		<p>This field must indicate an even number (bit 0 = 0). This field must be evenly divisible by 4 for Tile-Y surfaces (so the offset points to the start of a cache line). For Planar formats, if the surface is in YS or YF tile modes, the Y Offset for V should be an integral multiple of the Tile height of the Luma plane</p>	
6..7 Project: BDW	31:0	Reserved	
		Project:	BDW
		Format:	MBZ



Wait Notification

wait - Wait Notification			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
<p>The wait instruction evaluates the value of the notification count register nreg. If nreg is zero, thread execution is suspended and the thread is put in 'wait_for_notification' state. If nreg is not zero (i.e., one or more notifications have been received), nreg is decremented by one and the thread continues executing on the next instruction. If a thread is in the 'wait_for_notification' state, when a notification arrives, the notification count register is incremented by one. As the notification count register becomes nonzero, the thread wakes up to continue execution and at the same time the notification register is decremented by one. If only one notification arrived, the notification register value becomes zero. However, during the above mentioned time period, it is possible that more notifications may arrive, making the notification register nonzero again. When multiple notifications are received, software must use wait instructions to decrement notification count registers for each notification. Notification register n0.0:ud is for thread to thread communication (via the Message Gateway shared function) and n0.1:ud for host to thread communication (through MMIO registers). See the Message Gateway chapter for thread-thread communication.</p>			
Format: wait (exec_size) nreg			
Restriction			
src0 and dst must be n0.0, n0.1, or n0.2.			
Execution size must be 1 as the notification registers are scalar.			
Predication is not allowed.			
Two back-to-back wait instructions are not allowed. At minimum, a nop instruction must be inserted between two wait instructions			
Syntax			
wait (1) n#			
Pseudocode			
N/A			
Predication	Conditional Modifier	Saturation	Source Modifier
N	N	N	N
Src Types	Dst Types		
UD	UD		
DWord	Bit	Description	
0	127:64	Sources Exists If: ([Operand Control][Src1.RegFile]!='IMM') Format: EU_INSTRUCTION_SOURCES_REG	

wait - Wait Notification						
	127:64	Sources <table border="1"> <tr> <td>Exists If:</td> <td>([Operand Control][Src1.RegFile]='IMM')</td> </tr> <tr> <td>Format:</td> <td>EU_INSTRUCTION_SOURCES_IMM32</td> </tr> </table>	Exists If:	([Operand Control][Src1.RegFile]='IMM')	Format:	EU_INSTRUCTION_SOURCES_IMM32
	Exists If:	([Operand Control][Src1.RegFile]='IMM')				
	Format:	EU_INSTRUCTION_SOURCES_IMM32				
63:32	Operand Control <table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_OPERAND_CONTROLS</td> </tr> </table>	Format:	EU_INSTRUCTION_OPERAND_CONTROLS			
Format:	EU_INSTRUCTION_OPERAND_CONTROLS					
31:0	Header <table border="1"> <tr> <td>Format:</td> <td>EU_INSTRUCTION_HEADER</td> </tr> </table>	Format:	EU_INSTRUCTION_HEADER			
Format:	EU_INSTRUCTION_HEADER					

While

while - While			
Project:	BDW		
Source:	Eulsa		
Length Bias:	4		
Description			
<p>The while instruction marks the end of a do-while block. The instruction first evaluates the loop termination condition for each channel based on the current channel enables and the predication flags specified in the instruction. If any channel has not terminated, a branch is taken to a destination address specified in the instruction, and the loop continues for those channels. Otherwise, execution continues to the next instruction. It points to the first instruction with the do label of the do-while block of code. It should be a negative number for the backward referencing. In GEN binary, JIP is at location dst and must be of type W (signed word integer). If SPF is ON, none of the PclP are updated.</p>			
<p>The following table describes the 32-bit jump target offset JIP. JIP is a signed 32-bit number, added to IP pre-increment, and should point to the first instruction with the do label of the do-while block of code. It should be a negative number for the backward referencing. In GEN binary, JIP is at location src1 and must be of type D (signed dword integer).</p>			
<p>Format: [(pred)] while (exec_size) JIP</p>			
Restriction			
<p>The execution size must be the same for the while instruction and any break and cont instructions of the same code block.</p>			
Syntax			
<p>[(pred)] while (exec_size) imm32</p>			
Pseudocode			
<pre>Evaluate(WrEn); for (n = 0; n < 32; n++) { if (WrEn.chan[n]) { PclP[n] = IP + JIP; } else { PclP[n] = IP + 1; } } if (PMask == 1) { // any enabled channel true Jump(IP + JIP); }</pre>			
Predication	Conditional Modifier	Saturation	Source Modifier
Y	N	N	N
DWord	Bit	Description	
0..3	127:96	JIP	
		Project:	BDW
		Format:	S31
<p>Jump Target Offset. The relative offset in bytes if a jump is taken for the instruction.</p>			
	95	Source 0 Address Immediate [9] Sign Bit	
		Project:	BDW

while - While		
94:91	Src1.SrcType	
	Project:	BDW
	Format:	SrcType
90:89	Src1.RegFile	
	Project:	BDW
	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

XY_COLOR_BLT

XY_COLOR_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>COLOR_BLT is the simplest BLT operation. It performs a color fill to the destination (with a possible ROP). The only operand is the destination operand which is written dependent on the raster operation. The solid pattern color is stored in the pattern background register.</p> <p>This instruction is optimized to run at the maximum memory write bandwidth.</p> <p>The typical (and fastest) Raster operation code = F0 which performs a copy of the pattern background register to the destination.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value: 02h 2D Processor	
		Format: Opcode	
	28:22	Instruction Target(Opcod)	
		Default Value: 50h	
		Format: Opcode	
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		1xb	Write Alpha Channel
	x1b	Write RGB Channel	
19:12	Reserved		
	Format: MBZ		
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	
		Tile-X or Tile-Y	
10:8	Reserved		
	Format: MBZ		
7:0	DWord Length		
	Default Value: 05h		
	Format: =n		
1	31	Reserved	
		Format: MBZ	

XY_COLOR_BLT			
BR13	30	Clipping Enabled	
		Value	Name
		0b	Disabled
		1b	Enabled
	29:26	Reserved	Format: MBZ
	25:24	Color Depth	
		Value	Name
		00b	8 Bit Color
		01b	16 Bit Color(565)
		10b	16 Bit Color(1555)
	11b	32 Bit Color	
23:16	Raster Operation		
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).		
2	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.	
BR22	15:0	Destination X1 Coordinate (Left) 16 bit signed number.	
	3	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.
BR23	15:0	Destination X2 Coordinate (Right) 16 bit signed number.	
	4	31:0	Destination Base Address
Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.			
5	31:16	Reserved	
		Format: MBZ Should be programmed all 0's for 48bit addressing.	
	15:0	Destination Base Address High	
BR27		Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.	



XY_COLOR_BLT		
6 BR16	31:0	Solid Pattern Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]

XY_FULL_BLT

XY_FULL_BLT

Project: BDW
 Source: BlitterCS
 Length Bias: 2

The full BLT is the most comprehensive BLT instruction. It provides the ability to specify all 3 operands: destination, source, and pattern. The source and pattern operands are the same bit width as the destination operand.

The source and destination operands may overlap, which means that the X and Y directions can be either forward or backwards. The BLT Engine takes care of all situations. The base addresses plus the X and Y coordinates determine if there is an overlap between the source and destination operands. If the base addresses of the source and destination are the same and the Source X1 is less than Destination X1, then the BLT Engine performs the accesses in the X-backwards access pattern. There is no need to look for an actual overlap. If the base addresses are the same and Source Y1 is less than Destination Y1, then the scan line accesses start at Destination Y2 with the corresponding source scan line and the strides are subtracted for every scan line access. All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.

The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.

DWord	Bit	Description		
0 BR00	31:29	Client		
		Default Value:	02h 2D Processor	
		Format:	Opcode	
	28:22	Instruction Target(Opcode)		
		Default Value:	55h	
		Format:	Opcode	
	21:20	32bpp Byte Mask		
		This field is only used for 32bpp.		
		Value	Name	
		00b	[Default]	
1xb		Write Alpha Channel		
	x1b	Write RGB Channel		
19:16	Reserved			
	Format:	MBZ		
15	Src Tiling Enable			
	Value	Name	Description	
	0b	Tiling Disabled (Linear Blit)		
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.	

XY_FULL_BLT												
	14:12	Pattern Horizontal Seed Pixel of the scan line to start on corresponding to DST X=0.										
	11	Dest Tiling Enable										
		<table border="1" style="width: 100%;"><thead><tr><th style="width: 15%;">Value</th><th style="width: 55%;">Name</th><th style="width: 30%;">Description</th></tr></thead><tbody><tr><td>0b</td><td>Tiling Disabled (Linear Blit)</td><td></td></tr><tr><td>1b</td><td>Tiling Enabled</td><td>[BDW] [BDW]: Tile-X or Tile-Y.</td></tr></tbody></table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.	
		Value	Name	Description								
	0b	Tiling Disabled (Linear Blit)										
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.										
10:8	Pattern Vertical Seed Starting scan line of the 8x8 pattern corresponding to DST Y=0.											
7:0	DWord Length Default Value: <table border="1" style="width: 100%;"><tr><td style="width: 70%;"></td><td style="width: 30%;">0Ah</td></tr></table>		0Ah									
	0Ah											
1 BR13	31	Reserved Format: <table border="1" style="width: 100%;"><tr><td style="width: 60%;"></td><td style="width: 40%;">MBZ</td></tr></table>		MBZ								
			MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%;"><thead><tr><th style="width: 45%;">Value</th><th style="width: 55%;">Name</th></tr></thead><tbody><tr><td>0b</td><td>Disabled</td></tr><tr><td>1b</td><td>Enabled</td></tr></tbody></table>	Value	Name	0b	Disabled	1b	Enabled				
	Value	Name										
	0b	Disabled										
	1b	Enabled										
	29:26	Reserved Format: <table border="1" style="width: 100%;"><tr><td style="width: 60%;"></td><td style="width: 40%;">MBZ</td></tr></table>		MBZ								
	MBZ											
25:24	Color Depth											
	<table border="1" style="width: 100%;"><thead><tr><th style="width: 30%;">Value</th><th style="width: 70%;">Name</th></tr></thead><tbody><tr><td>00b</td><td>8 Bit Color</td></tr><tr><td>01b</td><td>16 Bit Color(565)</td></tr><tr><td>10b</td><td>16 Bit Color(1555)</td></tr><tr><td>11b</td><td>32 Bit Color</td></tr></tbody></table>		Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name										
	00b	8 Bit Color										
	01b	16 Bit Color(565)										
10b	16 Bit Color(1555)											
11b	32 Bit Color											
Raster Operation												
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.										
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.										
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.										
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.										

XY_FULL_BLT				
4 BR09	31:0	<p>Destination Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
	Format:	GraphicsAddress[31:0]		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
5 BR27	15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
6 BR11	15:0	<p>Source Pitch (double word aligned and signed) and in DWords</p> <p>2's complement. For Tiled Src (bit 15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).</p>		
	31:16	<p>Source Y1 Coordinate (Top)</p> <p>16 bit signed number.</p>		
7 BR26	15:0	<p>Source X1 Coordinate (Left)</p> <p>16 bit signed number.</p>		
	31:0	<p>Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_15 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL(64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
8 BR12	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
9 BR28	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			



XY_FULL_BLT				
10 BR15	31:0	<p>Pattern Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(28:06 are implemented) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. The pattern data must be located in linear memory. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
		Format:	GraphicsAddress[31:0]	
<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ		
Format:	MBZ			
11 BR29	15:0	<p>Pattern Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		

XY_FULL_IMMEDIATE_PATTERN_BLT

XY_FULL_IMMEDIATE_PATTERN_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>The full BLT is the most comprehensive BLT instruction. It provides the ability to specify all 3 operands: destination, source, and pattern. The source and immediate pattern operands are the same bit width as the destination operand. The immediate data sizes are 64 bytes (16 DWs), 128 bytes (32 DWs), or 256 (64 DWs) for 8, 16, and 32 bpp color patterns. DWL indicates the total number of Dwords of immediate data.</p> <p>The source and destination operands may overlap, which means that the X and Y directions can be either forward or backwards. The BLT Engine takes care of all situations. The base addresses plus the X and Y coordinates determine if there is an overlap between the source and destination operands. If the base addresses of the source and destination are the same and the Source X1 is less than Destination X1, then the BLT Engine performs the accesses in the X-backwards access pattern. There is no need to look for an actual overlap. If the base addresses are the same and Source Y1 is less than Destination Y1, then the scan line accesses start at Destination Y2 with the corresponding source scan line and the strides are subtracted for every scan line access. All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	74h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:16	Reserved		
	Format:	MBZ	
15	Src Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.

XY_FULL_IMMEDIATE_PATTERN_BLT			
	14:12	Pattern Horizontal Seed (pixel of the scan line to start on corresponding to DST X=0)	
	11	Dest Tiling Enable	
		Value	Name
		0b	Tiling Disabled (Linear Blit)
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed Starting scan line of the 8x8 pattern corresponding to DST Y=0.		
7:0	DWord Length	Default Value: 08h Excludes DWORD 0,1	
	08 + DWL = (Number of Immediate double words)h		
1 BR13	31	Reserved	
		Format: MBZ	
	30	Clipping Enabled	
		Value	Name
		0b	Disabled
	1b	Enabled	
	29:26	Reserved	
	Format: MBZ		
	25:24	Color Depth	
		Value	Name
00b		8 Bit Color	
01b		16 Bit Color(565)	
10b		16 Bit Color(1555)	
11b	32 Bit Color		
23:16	Raster Operation		
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).		
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.	
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.	
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.	
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.	

XY_FULL_IMMEDIATE_PATTERN_BLT				
4 BR9	31:0	Destination Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
	Format:	GraphicsAddress[31:0]		
31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
5 BR27	15:0	Destination Base Address High <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		
31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
6 BR11	15:0	Source Pitch (double word aligned and signed) and in DWords 2's complement. For Tiled Src (bit 15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).		
	31:16	Source Y1 Coordinate (Top) 16 bit signed number.		
7 BR26	15:0	Source X1 Coordinate (Left) 16 bit signed number.		
	31:0	Source Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Src Tiling is enabled (Bit_15 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
9 BR28	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	Source Address High <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
10..n	31:0	Immediate Data 0		



XY_FULL_MONO_PATTERN_BLT

XY_FULL_MONO_PATTERN_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>The full BLT is the most comprehensive BLT instruction. It provides the ability to specify all 3 operands: destination, source, and pattern. The pattern operand is monochrome and the source operand is the same bit width as the destination operand.</p> <p>The source and destination operands may overlap, which means that the X and Y directions can be either forward or backwards. The BLT Engine takes care of all situations. The base addresses plus the X and Y coordinates determine if there is an overlap between the source and destination operands. If the base addresses of the source and destination are the same and the Source X1 is less than Destination X1, then the BLT Engine performs the accesses in the X-backwards access pattern. There is no need to look for an actual overlap. If the base addresses are the same and Source Y1 is less than Destination Y1, then the scan line accesses start at Destination Y2 with the corresponding source scan line and the strides are subtracted for every scan line access. The monochrome pattern transparency mode indicates whether to use the pattern background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the pattern foreground color is used in the ROP operation.</p> <p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8. Setting both Solid Pattern Select = 1 and Mono Pattern Transparency = 1 is mutually exclusive. The device implementation results in NO PIXELS DRAWN.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	57h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:16	Reserved		

XY_FULL_MONO_PATTERN_BLT											
	15	Src Tiling Enable									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
		Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)									
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.								
	14:12	Pattern Horizontal Seed (pixel of the scan line to start on corresponding to DST X=0)									
	11	Dest Tiling Enable									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
		Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)									
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.									
10:8	Pattern Vectical Seed Starting scan line of the 8x8 pattern corresponding to DST Y=0.										
7:0	DWord Length										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0Ch</td> <td></td> </tr> </tbody> </table>	Value	Name	0Ch							
Value	Name										
0Ch											
1 BR13	31	Solid Pattern Select									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>No Solid Pattern</td> </tr> <tr> <td>1</td> <td>Solid Pattern</td> </tr> </tbody> </table>	Value	Name	0	No Solid Pattern	1	Solid Pattern			
		Value	Name								
	0	No Solid Pattern									
	1	Solid Pattern									
	30	Clipping Enabled									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
Value	Name										
0b	Disabled										
1b	Enabled										
29	Reserved										
	Format: MBZ										
28:27	Mono Source Transparency Mode										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled				
	Value	Name									
0	Use Background										
1	Transparency Enabled										
26	Reserved										
	Format: MBZ										



XY_FULL_MONO_PATTERN_BLT												
	25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name										
	00b	8 Bit Color										
	01b	16 Bit Color(565)										
	10b	16 Bit Color(1555)										
11b	32 Bit Color											
23:16	Raster Operation											
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
2	31:16 Destination Y1 Coordinate (Top) 16 bit signed number.											
BR22	15:0 Destination X1 Coordinate (Left) 16 bit signed number.											
3	31:16 Destination Y2 Coordinate (Bottom) 16 bit signed number.											
BR23	15:0 Destination X2 Coordinate (Right) 16 bit signed number.											
4	31:0 Destination Base Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	Format:	GraphicsAddress[31:0]									
Format:	GraphicsAddress[31:0]											
5	31:16	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> Should be programmed all 0's for 48bit addressing.	Format:	MBZ								
	Format:	MBZ										
BR27	15:0 Destination Base Address High <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]									
Format:	GraphicsAddress[47:32]											
6	31:16	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
	Format:	MBZ										
BR11	15:0 Source Pitch (double word aligned and signed) and in DWords 2's complement. For Tiled Src (bit 15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
7	31:16 Source Y1 Coordinate (Top) 16 bit signed number.											

XY_FULL_MONO_PATTERN_BLT				
BR26	15:0	Source X1 Coordinate (Left) 16 bit signed number.		
8 BR12	31:0	Source Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> (base address of the source surface: X=0, Y=0). Lower 32bits of the 48bit addressing. When Src Tiling is enabled (Bit 15 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
9 BR28	31:16	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> Should be programmed all 0's for 48bit addressing.	Format:	MBZ
Format:	MBZ			
	15:0	Source Address High <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
10 BR16	31:0	Pattern Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]		
11 BR17	31:0	Pattern Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]		
12 BR20	31:0	Pattern Data 0 (least significant DW)		
13 BR21	31:0	Pattern Data 1 (most significant DW)		



XY_FULL_MONO_PATTERN_MONO_SRC_BLT

XY_FULL_MONO_PATTERN_MONO_SRC_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>The full BLT provides the ability to specify all 3 operands: destination, source, and pattern. The pattern and source operands are monochrome.</p> <p>The monochrome source transparency mode indicates whether to use the source background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the source foreground color is used in the ROP operation.</p> <p>All non-text monochrome sources are word aligned. At the end of a scan line the monochrome source, the remaining bits until the next word boundary are ignored. The Monochrome source data bit position field [2:0] indicates which bit position within the first byte should be used as the first source pixel which corresponds to the destination X1 coordinate.</p> <p>The monochrome pattern transparency mode indicates whether to use the pattern background color or de-assert the write enables when the bit in the pattern is 0. When the source bit is 1, then the pattern foreground color is used in the ROP operation. The monochrome source transparency mode works identical to the pattern transparency mode.</p> <p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p> <p>Setting both Solid Pattern Select = 1 and Mono Pattern Transparency = 1 is mutually exclusive. The device implementation results in NO PIXELS DRAWN.</p> <p>Negative Stride (= Pitch) is NOT ALLOWED.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	58h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Monochrome source data bit position of the first pixel within a byte per scan line.		

XY_FULL_MONO_PATTERN_MONO_SRC_BLT											
	16:15	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	14:12	Pattern Horizontal Seed (pixel of the scan line to start on corresponding to DST X=0)									
	11	Tiling Enable <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 45%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
	Value	Name	Description								
	0b	Tiling Disabled (Linear Blit)									
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.									
10:8	Pattern Vertical Seed Starting scan line of the 8x8 pattern corresponding to DST Y = 0.										
7:0	DWord Length <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0Ch</td> <td></td> </tr> </tbody> </table>	Value	Name	0Ch							
Value	Name										
0Ch											
1 BR13	31	Solid Pattern Select <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>No Solid Pattern</td> </tr> <tr> <td>1</td> <td>Solid Pattern</td> </tr> </tbody> </table>	Value	Name	0	No Solid Pattern	1	Solid Pattern			
	Value	Name									
	0	No Solid Pattern									
	1	Solid Pattern									
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 45%;">Value</th> <th style="width: 55%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
	29	Mono Source Transparency Mode <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 75%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled			
	Value	Name									
0	Use Background										
1	Transparency Enabled										
28	Mono Pattern Transparency Mode <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 75%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled				
Value	Name										
0	Use Background										
1	Transparency Enabled										
27:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										

XY_FULL_MONO_PATTERN_MONO_SRC_BLT												
	25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name										
	00b	8 Bit Color										
	01b	16 Bit Color(565)										
	10b	16 Bit Color(1555)										
11b	32 Bit Color											
23:16	Raster Operation											
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
2	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.										
BR22	15:0	Destination X1 Coordinate (Left) 16 bit signed number.										
3	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.										
BR23	15:0	Destination X2 Coordinate (Right) 16 bit signed number.										
4	31:0	Destination Base Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	Format:	GraphicsAddress[31:0]								
Format:	GraphicsAddress[31:0]											
5	31:16	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> Should be programmed all 0's for 48bit addressing.	Format:	MBZ								
	Format:	MBZ										
BR27	15:0	Destination Base Address High <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]								
Format:	GraphicsAddress[47:32]											
6	31:0	Mono Source Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> (address corresponds to DST X1, Y1) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. This Monosource Base Address programmed, must always be Cache Line (64byte) aligned.	Format:	GraphicsAddress[31:0]								
Format:	GraphicsAddress[31:0]											
BR12												

XY_FULL_MONO_PATTERN_MONO_SRC_BLT				
7 BR28	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Mono Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
8 BR18	31:0	<p>Source Background Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
9 BR19	31:0	<p>Source Foreground Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
10 BR16	31:0	<p>Pattern Background Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
11 BR17	31:0	<p>Pattern Foreground Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
12 BR20	31:0	<p>Pattern Data 0</p> <p>(least significant DW)</p>		
13 BR21	31:0	<p>Pattern Data 1</p> <p>(most significant DW)</p>		

XY_FULL_MONO_SRC_BLT

XY_FULL_MONO_SRC_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>The full BLT is the most comprehensive BLT instruction. It provides the ability to specify all 3 operands: destination, source, and pattern. The source operand is monochrome and the pattern operand is the same bit width as the destination.</p> <p>The monochrome source transparency mode indicates whether to use the source background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the source foreground color is used in the ROP operation.</p> <p>All non-text and non-immediate monochrome sources are word aligned. At the end of a scan line the monochrome source, the remaining bits until the next word boundary are ignored. The Monochrome source data bit position field [2:0] indicates which bit position within the first byte should be used as the first source pixel which corresponds to the Destination X1 coordinate.</p> <p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p> <p>Negative Stride (= Pitch) is NOT ALLOWED</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	56h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Monochrome source data bit position of the first pixel within a byte per scan line.		
16:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed (pixel of the scan line to start on corresponding to DST X=0)		

XY_FULL_MONO_SRC_BLT											
	11	Tiling Enable									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
		Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)									
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.									
10:8	Pattern Vertical Seed Starting scan line of the 8x8 pattern corresponding to DST Y = 0.										
7:0	DWord Length										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0Ah</td> <td></td> </tr> </tbody> </table>	Value	Name	0Ah							
Value	Name										
0Ah											
1 BR13	31	Reserved									
		Format: MBZ									
	30	Clipping Enabled									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
		Value	Name								
	0b	Disabled									
	1b	Enabled									
	29	Mono Source Transparency Mode									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled			
	Value	Name									
0	Use Background										
1	Transparency Enabled										
28:26	Reserved										
	Format: MBZ										
25:24	Color Depth										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name									
	00b	8 Bit Color									
	01b	16 Bit Color(565)									
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									

XY_FULL_MONO_SRC_BLT				
BR23	15:0	<p>Destination X2 Coordinate (Right) 16 bit signed number.</p>		
4 BR09	31:0	<p>Destination Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
5 BR27	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
Format:	MBZ			
	15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
6 BR12	31:0	<p>Mono Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(address corresponds to DST X1, Y1) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. This Monosource Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
7 BR28	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
Format:	MBZ			
	15:0	<p>Mono Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
8 BR18	31:0	<p>Source Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
9 BR19	31:0	<p>Source Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
10 BR15	31:0	<p>Pattern Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(28:06 are implemented) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. The pattern data must be located in linear memory. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			

XY_FULL_MONO_SRC_BLT				
11 BR29	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td style="width: 40%;">MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Pattern Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td style="width: 70%;">GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			



XY_FULL_MONO_SRC_IMMEDIATE_PATTERN_BLT

DWord		Bit	Description								
Project:		BDW									
Source:		BlitterCS									
Length Bias:		2									
<p>The full BLT is the most comprehensive BLT instruction. It provides the ability to specify all 3 operands: destination, source, and pattern. The source operand is a monochrome and the immediate pattern operand is the same bit width as the destination. The immediate data sizes are 64 bytes (16 DWs), 128 bytes (32 DWs), or 256 (64DWs) for 8, 16, and 32 bpp color patterns. The monochrome source transparency mode indicates whether to use the source background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the source foreground color is used in the ROP operation. All non-text monochrome sources are word aligned. At the end of a scan line the monochrome source, the remaining bits until the next word boundary are ignored. The Monochrome source data bit position field [2:0] indicates which bit position within the first byte should be used as the first source pixel which corresponds to the destination X1 coordinate. All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation. The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8. Negative Stride (= Pitch) is NOT ALLOWED.</p>											
0	31:29		Client <table border="1"> <tr> <td>Default Value:</td> <td>02h 2D Processor</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table>	Default Value:	02h 2D Processor	Format:	Opcode				
Default Value:	02h 2D Processor										
Format:	Opcode										
BR00	28:22		Instruction Target(Opcode) <table border="1"> <tr> <td>Default Value:</td> <td>75h</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table>	Default Value:	75h	Format:	Opcode				
	Default Value:	75h									
Format:	Opcode										
21:20	32bpp Byte Mask This field is only used for 32bpp.										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <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> </tbody> </table>			Value	Name	00b	[Default]	1xb	Write Alpha Channel	x1b	Write RGB Channel
	Value	Name									
	00b	[Default]									
	1xb	Write Alpha Channel									
x1b	Write RGB Channel										
Monochrome source data bit position of the first pixel within a byte per scan line.											
19:17											
16:15	Reserved										
	Format:	MBZ									
14:12	Pattern Horizontal Seed (pixel of the scan line to start on corresponding to DST X=0)										

XY_FULL_MONO_SRC_IMMEDIATE_PATTERN_BLT											
	11	Tiling Enable <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">Value</th> <th style="width: 45%;">Name</th> <th style="width: 40%;">Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
	Value	Name	Description								
	0b	Tiling Disabled (Linear Blit)									
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.								
10:8	Pattern Vertical Seed Starting scan line of the 8x8 pattern corresponding to DST Y=0.										
7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>08h Excludes DWORD 0,1</td> </tr> </table> 08 + DWL = (Number of Immediate double words)h	Default Value:	08h Excludes DWORD 0,1								
Default Value:	08h Excludes DWORD 0,1										
1 BR13	31	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 45%;">Value</th> <th style="width: 55%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
	29	Mono Source Transparency Mode <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 75%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled			
	Value	Name									
	0	Use Background									
	1	Transparency Enabled									
28:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
25:24	Color Depth <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 25%;">Value</th> <th style="width: 75%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									

XY_FULL_MONO_SRC_IMMEDIATE_PATTERN_BLT				
BR23	15:0	Destination X2 Coordinate (Right) 16 bit signed number.		
4 BR09	31:0	Destination Base Address Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>GraphicsAddress[31:0]</td></tr></table> Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
5 BR27	31:16	Reserved Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>MBZ</td></tr></table> Should be programmed all 0's for 48bit addressing.	Format:	MBZ
Format:	MBZ			
	15:0	Destination Base Address High Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>GraphicsAddress[47:32]</td></tr></table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
6 BR12	31:0	Mono Source Address Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>GraphicsAddress[31:0]</td></tr></table> (address corresponds to DST X1, Y1) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. This Monosource Base Address programmed, must always be Cache Line (64byte) aligned.	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
7 BR28	31:16	Reserved Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>MBZ</td></tr></table> Should be programmed all 0's for 48bit addressing.	Format:	MBZ
Format:	MBZ			
	15:0	Mono Source Address High Format: <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="width: 100px;">Format:</td><td>GraphicsAddress[47:32]</td></tr></table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
8 BR18	31:0	Source Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]		
9 BR19	31:0	Source Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]		
10..n	31:0	Immediate Data		

XY_MONO_PAT_BLT

XY_MONO_PAT_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>MONO_PAT_BLT is used when we have no source and the monochrome pattern is not trivial (is not a solid color only). The monochrome pattern is loaded from the instruction stream.</p> <p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p> <p>The monochrome pattern transparency mode indicates whether to use the pattern background color or de-assert the write enables when the bit in the pattern is 0. When the pattern bit is 1, then the pattern foreground color is used in the ROP operation.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	52h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed		
		Pixel of the scan line to start on corresponding to DST X=0.	
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed		
		Scan line of the 8x8 pattern to start on corresponding to DST Y=0.	

XY_MONO_PAT_BLT											
	7:0	DWord Length <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">08h</td> <td></td> </tr> </tbody> </table>	Value	Name	08h						
Value	Name										
08h											
1 BR13	31	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0b</td> <td>Disabled</td> </tr> <tr> <td style="text-align: center;">1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
	29	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ							
Format:	MBZ										
28	Mono Pattern Transparency Mode <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%; text-align: center;">Value</th> <th style="width: 70%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Use Background</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled				
Value	Name										
0	Use Background										
1	Transparency Enabled										
27:26	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td style="text-align: center;">MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%; text-align: center;">Value</th> <th style="width: 70%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">00b</td> <td>8 Bit Color</td> </tr> <tr> <td style="text-align: center;">01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td style="text-align: center;">10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td style="text-align: center;">11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
	15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).									
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									

XY_MONO_PAT_BLT		
4 BR09	31:0	Destination Base Address Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.
		Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
5 BR27	31:16	Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
	15:0	Destination Base Address High Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.
6 BR16	31:0	Pattern Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]
7 BR17	31:0	Pattern Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]
8 BR20	31:0	Pattern Data 0
9 BR21	31:0	Pattern Data 1



XY_MONO_PAT_FIXED_BLT

XY_MONO_PAT_FIXED_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>MONO_PAT_FIXED_BLT is used when we have no source and the monochrome pattern is not trivial (is not a solid color only). The monochrome pattern is one of 10 fixed patterns described below. The pattern seeds can still be used with the fixed patterns, creating even more fixed patterns. This eliminates 2 doublewords compared to the XY_MONO_PAT_BLT command packet.</p> <p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p> <p>The monochrome pattern transparency mode indicates whether to use the pattern background color or de-assert the write enables when the bit in the pattern is 0. When the pattern bit is 1, then the pattern foreground color is used in the ROP operation.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	59h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19	Reserved		
	Format:	MBZ	

XY_MONO_PAT_FIXED_BLT

1	BR13	18:15	Fixed Pattern	
		Value	Name	
		0000b	HS_HORIZONTAL	
		0001b	HS_VERTICAL	
		0010b	HS_FDIAGONAL	
		0011b	HS_BDIAGONAL	
		0100b	HS_CROSS	
		0101b	HS_DIAGCROSS	
		0110b	Reserved	
		0111b	Reserved	
		1000b	Screen Door	
		1001b	SD Wide	
		1010b	Walking Bit (one)	
		1011b	Walking Zero	
		1100b	Reserved	
		1101b	Reserved	
		1110b	Reserved	
		1111b	Reserved	
14:12	Pattern Horizontal Seed Pixel of the scan line to start on corresponding to DST X=0.			
11	Tiling Enable			
Value	Name	Description		
0b	Tiling Disabled (Linear Blit)			
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.		
10:8	Pattern Vertical Seed Scan line of the 8x8 pattern to start on corresponding to DST Y=0.			
7:0	DWord Length			
Format:		=n		
Value	Name			
06h				
31	Reserved			
Format:		MBZ		
30	Clipping Enabled			
Value	Name			
0b	Disabled			
1b	Enabled			

XY_MONO_PAT_FIXED_BLT												
	29	Reserved Format: MBZ										
	28	Mono Pattern Transparency Mode <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled				
	Value	Name										
	0	Use Background										
	1	Transparency Enabled										
	27:26	Reserved Format: MBZ										
	25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name										
	00b	8 Bit Color										
	01b	16 Bit Color(565)										
10b	16 Bit Color(1555)											
11b	32 Bit Color											
23:16	Raster Operation											
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
2	31:16 Destination Y1 Coordinate (Top) 16 bit signed number.											
BR22	15:0 Destination X1 Coordinate (Left) 16 bit signed number.											
3	31:16 Destination Y2 Coordinate (Bottom) 16 bit signed number.											
BR23	15:0 Destination X2 Coordinate (Right) 16 bit signed number.											
4	Destination Base Address Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.											
BR09												
5	Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.											
BR27	15:0 Destination Base Address High Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.											

XY_MONO_PAT_FIXED_BLT		
6 BR16	31:0	Pattern Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]
7 BR17	31:0	Pattern Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]



XY_MONO_SRC_COPY_BLT

XY_MONO_SRC_COPY_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This BLT instruction performs a monochrome source copy where the only operands involved is a monochrome source and destination. The source and destination operands cannot overlap therefore the X and Y directions are always forward.</p> <p>All non-text monochrome sources are word aligned. At the end of a scan line of monochrome source, all bits until the next word boundary are ignored. The monochrome source data bit position field [2:0] indicates the bit position within the first byte of the scan line that should be used as the first source pixel which corresponds to the destination X1 coordinate.</p> <p>The monochrome source transparency mode indicates whether to use the source background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the source foreground color is used in the ROP operation. The ROP value chosen must involve source and no pattern data in the ROP operation. Negative Stride (= Pitch) is NOT ALLOWED.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value: 02h 2D Processor	
	Format: Opcode		
	28:22	Instruction Target(Opcode)	
		Default Value: 54h	
	Format: Opcode		
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Monochrome source data bit position of the first pixel within a byte per scan line.		
16:12	Reserved		
	Format: MBZ		
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Reserved		
	Format: MBZ		

XY_MONO_SRC_COPY_BLT											
	7:0	DWord Length <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>08h</td> <td></td> </tr> </tbody> </table>	Value	Name	08h						
Value	Name										
08h											
1 BR13	31	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
29	Mono Source Transparency Mode <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Use Background</td> </tr> <tr> <td>1</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0	Use Background	1	Transparency Enabled				
Value	Name										
0	Use Background										
1	Transparency Enabled										
28:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
25:24	Color Depth <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
	15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).									
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									
4 BR09	31:0	Destination Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table>	Format:	GraphicsAddress[31:0]							
	Format:	GraphicsAddress[31:0]									
		Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.									

XY_MONO_SRC_COPY_BLT				
5 BR27	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
6 BR12	31:0	<p>Mono Source Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(address corresponds to DST X1, Y1) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. This Monosource Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
	Format:	GraphicsAddress[31:0]		
15:0	<p>Mono Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
7 BR28	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Mono Source Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
8 BR18	31:0	<p>Source Background Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		
9 BR19	31:0	<p>Source Foreground Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]</p>		

XY_MONO_SRC_COPY_IMMEDIATE_BLT

XY_MONO_SRC_COPY_IMMEDIATE_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This instruction allows the Driver to send monochrome data through the instruction stream, eliminating the read latency of the source during command execution.</p> <p>The IMMEDIATE_BLT data MUST transfer an even number of doublewords and the exact number of quadwords. DWL indicates the total number of Dwords of immediate data.</p> <p>All non-text monochrome sources are word aligned. At the end of a scan line of monochrome source, all bits until the next word boundary are ignored. The Monochrome source data bit position field [2:0] indicates the bit position within the first byte of the scan line that should be used as the first source pixel which corresponds to the destination X1 coordinate.</p> <p>The monochrome source transparency mode indicates whether to use the source background color or de-assert the write enables when the bit in the source is 0. When the source bit is 1, then the source foreground color is used in the ROP operation. The ROP value chosen must involve source and no pattern data in the ROP operation. The monochrome source data supplied corresponds to the Destination X1 and Y1 coordinates.</p> <p>Negative Stride (= Pitch) is NOT ALLOWED.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	71h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
	x1b	Write RGB Channel	
19:17	Monochrome source data bit position of the first pixel within a byte per scan line.		
16:12	Reserved		
	Format:	MBZ	
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.

XY_MONO_SRC_COPY_IMMEDIATE_BLT											
	10:8	Reserved Format: _____ MBZ									
	7:0	DWord Length Default Value: _____ 06h Excludes DWORD 0,1 06 + DWL = (Number of Immediate double words)h									
1 BR13	31	Reserved Format: _____ MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%; text-align: center;">Value</th> <th style="width: 50%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0b</td> <td>Disabled</td> </tr> <tr> <td style="text-align: center;">1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
	29	Mono Source Transparency Mode <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%; text-align: center;">Value</th> <th style="width: 70%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0b</td> <td>Transparency Enabled</td> </tr> <tr> <td style="text-align: center;">1b</td> <td>Use Background</td> </tr> </tbody> </table>	Value	Name	0b	Transparency Enabled	1b	Use Background			
Value	Name										
0b	Transparency Enabled										
1b	Use Background										
28:26	Reserved Format: _____ MBZ										
25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%; text-align: center;">Value</th> <th style="width: 70%; text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">00b</td> <td>8 Bit Color</td> </tr> <tr> <td style="text-align: center;">01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td style="text-align: center;">10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td style="text-align: center;">11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									

XY_MONO_SRC_COPY_IMMEDIATE_BLT		
4 BR09	31:0	Destination Base Address Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.
		Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
5 BR27	31:16	Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
	15:0	Destination Base Address High Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.
6 BR18	31:0	Source Background Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]
7 BR19	31:0	Source Foreground Color 8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0]
8..n	31:0	Immediate Data

XY_PAT_BLT

XY_PAT_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>PAT_BLT is used when there is no source and the color pattern is not trivial (is not a solid color only). If clipping is enabled, all scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation. The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	51h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed	Pixel of the scan line to start on corresponding to DST X=0.	
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed	Scan line of the 8x8 pattern to start on corresponding to DST Y=0.	
7:0	DWord Length		
	Default Value:	06h	
1	31	Reserved	
		Format:	MBZ

XY_PAT_BLT			
BR13	30	Clipping Enabled	
		Value	Name
		0b	Disabled
		1b	Enabled
	29:26	Reserved	
		Format:	MBZ
	25:24	Color Depth	
		Value	Name
		00b	8 Bit Color
		01b	16 Bit Color(565)
10b		16 Bit Color(1555)	
	11b	32 Bit Color	
23:16	Raster Operation		
15:0	Destination Pitch in DWords 2's complement (Negative Pitch Not allowed for Pixel nor Text) For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).		
2	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.	
BR22	15:0	Destination X1 Coordinate (Left) 16 bit signed number.	
	3	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.
BR23	15:0	Destination X2 Coordinate (Right) 16 bit signed number.	
	4	31:0	Destination Base Address
Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.			
5	31:16	Reserved	
		Format: MBZ	
	Should be programmed all 0's for 48bit addressing.		
BR27	15:0	Destination Base Address High	
		Format: GraphicsAddress[47:32]	
Should be programmed with the upper 16bits of the 48bit addressing.			



XY_PAT_BLT				
6 BR15	31:0	<p>Pattern Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(28:06 are implemented) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. The pattern data must be located in linear memory. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
7 BR29	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
Format:	MBZ			
	15:0	<p>Pattern Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			

XY_PAT_BLT_IMMEDIATE

XY_PAT_BLT_IMMEDIATE			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>PAT_BLT_IMMEDIATE is used when there is no source and the color pattern is not trivial (is not a solid color only) and the pattern is pulled through the command stream. The immediate data sizes are 64 bytes (16 DWs), 128 bytes (32 DWs), or 256 (64DWs) for 8, 16, and 32 bpp color patterns.</p> <p>DWL indicates the total number of Dwords of immediate data. All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	72h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed		
	Pixel of the scan line to start on corresponding to DST X=0.		
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed		
	Scan line of the 8x8 pattern to start on corresponding to DST Y=0.		

XY_PAT_BLT_IMMEDIATE											
	7:0	DWord Length Default Value: 04h Excludes DWORD 0,1 04 + DWL = (Number of Immediate double)h									
1 BR13	31	Reserved Format: MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
29:26	Reserved Format: MBZ										
25:24	Color Depth <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation 15:0										
	15:0	Destination Pitch in DWords 2's complement (Negative Pitch Not allowed for Pixel nor Text) For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).									
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									
4 BR09	31:0	Destination Base Address Format: GraphicsAddress[31:0]									
		Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.									

XY_PAT_BLT_IMMEDIATE				
5 BR27	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
6..n	31:0	Immediate Data		



XY_PAT_CHROMA_BLT

XY_PAT_CHROMA_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>PAT_BLT is used when there is no source and the color pattern is not trivial (is not a solid color only). All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation. The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	76h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Transparency Range Mode		
(chroma-key) - Dst Chroma-key modes ONLY (SRC ILLEGAL)			
16:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed		
Pixel of the scan line to start on corresponding to DST X=0.			
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed		
Scan line of the 8x8 pattern to start on corresponding to DST Y=0.			
7:0	DWord Length		
	Default Value:	08h Excludes DWORD 0,1	

XY_PAT_CHROMA_BLT											
1 BR13	31	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
1b	Enabled										
29:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
25:24	Color Depth <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation 15:0 Destination Pitch in DWords 2's complement (Negative Pitch Not allowed for Pixel nor Text) For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									
4 BR09	31:0	Destination Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]							
	Format:	GraphicsAddress[31:0]									
5 BR27	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ								
Format:	MBZ										

XY_PAT_CHROMA_BLT				
	15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
6 BR15	31:0	<p>Pattern Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(28:06 are implemented) (Note no NPO2 change here). Lower 32bits of the 48bit addressing. The pattern data must be located in linear memory. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
7 BR29	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
Format:	MBZ			
	15:0	<p>Pattern Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
8 BR18	31:0	<p>Transparency Color Low (Chroma-key Low = Pixel Greater or Equal)</p>		
9 BR19	31:0	<p>Transparency Color High (Chroma-key High = Pixel Less or Equal)</p>		

XY_PAT_CHROMA_BLT_IMMEDIATE

XY_PAT_CHROMA_BLT_IMMEDIATE			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>PAT_BLT_IMMEDIATE is used when there is no source and the color pattern is not trivial (is not a solid color only) and the pattern is pulled through the command stream. The immediate data sizes are 64 bytes (16 DWs), 128 bytes (32 DWs), or 256 (64DWs) for 8, 16, and 32 bpp color patterns.</p> <p>DWL indicates the total number of Dwords of immediate data. All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation.</p> <p>The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	77h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Transparency Range Mode		
(chroma-key) - Dst Chroma-key modes ONLY (SRC ILLEGAL)			
16:15	Reserved		
	Format:	MBZ	
14:12	Pattern Horizontal Seed		
Pixel of the scan line to start on corresponding to DST X=0.			
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Pattern Vertical Seed		
Scan line of the 8x8 pattern to start on corresponding to DST Y=0.			

XY_PAT_CHROMA_BLT_IMMEDIATE											
	7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Default Value:</td> <td>06h Excludes DWORD 0,1</td> </tr> </table> <p>06 + DWL = (Number of Immediate double)h</p>	Default Value:	06h Excludes DWORD 0,1							
Default Value:	06h Excludes DWORD 0,1										
1 BR13	31	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ							
	Format:	MBZ									
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
1b	Enabled										
29:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
25:24	Color Depth <table border="1" style="width: 100%;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
Format:	MBZ										
	15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).									
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									
4 BR09	31:0	Destination Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]							
	Format:	GraphicsAddress[31:0]									
5 BR27	31:16	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 50%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ							
Format:	MBZ										

XY_PAT_CHROMA_BLT_IMMEDIATE				
	15:0	Destination Base Address High <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
6 BR18	31:0	Transparency Color Low (Chroma-key Low = Pixel Greater or Equal)		
7 BR19	31:0	Transparency Color High (Chroma-key High = Pixel Less or Equal)		
8..n	31:0	Immediate Data		

XY_PIXEL_BLT

XY_PIXEL_BLT										
Project:	BDW									
Source:	BlitterCS									
Length Bias:	2									
<p>The Destination X coordinate and Destination Y coordinate is compared with the ClipRect registers. If it is within all 4 comparisons, then the pixel supplied in the XY_SETUP_BLT instruction is written with the raster operation to (Destination Y Address + (Destination Y coordinate * Destination pitch) + (Destination X coordinate * bytes per pixel)).</p> <p>ROP field must specify pattern or fill with 0's or 1's. There is no source operand.</p> <p>Negative Stride (= Pitch) specified in the Setup command is Not Allowed</p>										
DWord	Bit	Description								
0 BR00	31:29	Client								
		Default Value: 02h 2D Processor								
	Format: Opcode									
	28:22	Instruction Target(Opcod)								
		Default Value: 24h								
	Format: Opcode									
21:12	Reserved									
	Format: MBZ									
11	Tiling Enable									
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
	Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)								
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.							
Format: MBZ										
10:8	Reserved									
	Format: MBZ									
7:0	DWord Length									
	Default Value: 00h									
1 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.								
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.								

XY_SCANLINES_BLT

XY_SCANLINES_BLT										
Project:	BDW									
Source:	BlitterCS									
Length Bias:	2									
<p>All scan lines and pixels that fall within the ClipRect Y and X coordinates are written. Only pixels within the ClipRectX coordinates and the Destination X coordinates are written using the raster operation. The Pattern Seeds correspond to Destination X = 0 (horizontal) and Y = 0 (vertical). The alignment is relative to the destination coordinates. The pixel of the pattern used / scan line is the (destination X coordinate + horizontal seed) modulo 8. The scan line of the pattern used is the (destination Y coordinate + vertical seed) modulo 8. Solid pattern should use the XY_SETUP_MONO_PATTERN_SL_BLT instruction. ROP field must specify pattern or fill with 0's or 1's. There is no source operand.</p>										
DWord	Bit	Description								
0 BR00	31:29	Client <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>02h 2D Processor</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table>	Default Value:	02h 2D Processor	Format:	Opcode				
	Default Value:	02h 2D Processor								
	Format:	Opcode								
	28:22	Instruction Target(Opcode) <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>25h</td> </tr> <tr> <td>Format:</td> <td>Opcode</td> </tr> </table>	Default Value:	25h	Format:	Opcode				
	Default Value:	25h								
	Format:	Opcode								
	21:15	Reserved <table border="1" style="width: 100%;"> <tr> <td>Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ						
Format:	MBZ									
14:12	Pattern Horizontal Seed Pixel of the scan line to start on corresponding to DST X=0.									
11	Tiling Enable <table border="1" style="width: 100%;"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
Value	Name	Description								
0b	Tiling Disabled (Linear Blit)									
1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.								
10:8	Pattern Vertical Seed Scan line of the 8x8 pattern to start on corresponding to DST Y=0.									
7:0	DWord Length <table border="1" style="width: 100%;"> <tr> <td>Default Value:</td> <td>01h</td> </tr> </table>	Default Value:	01h							
Default Value:	01h									
1 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.								
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.								
2 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.								
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.								

XY_SETUP_BLT

XY_SETUP_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This setup instruction supplies common setup information including clipping coordinates used by the XY commands: XY_PIXEL_BLT, XY_SCANLINE_BLT, XY_TEXT_BLT, and XY_TEXT_BLT_IMMEDIATE. These are the only instructions that require that state be saved between instructions other than the Clipping parameters. There are 5 dedicated registers to contain the state for the 3 setup BLT instructions (XY_SETUP_BLT, XY_SETUP_MONO_PATTERN_SL_BLT, and XY_SETUP_CLIP_BLT. All other BLTs use a temporary version of these. The 5 double word registers are: DW1 (Setup Control), DW6 (Setup Foreground color), DW5 (Setup Background color), DW7 (Setup Pattern address), and DW4 (Setup Destination Base Address).</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	01h
		Format:	Opcode
	21:20	32 bpp Byte Mask	
		Value	Name
		1xb	Write Alpha Channel
		x1b	Write RGB Channel
19:12	Reserved		
	Format:	MBZ	
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled (Tile-X or Tile-Y)	
10:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	08h	
1	31	Reserved	
		Format:	MBZ

XY_SETUP_BLT			
BR01	30	Clipping Enabled	
		Value	Name
		0b	Disabled
		1b	Enabled
	29	Mono Source Transparency Mode	
		Value	Name
		0b	Use Background
		1b	Transparency Enabled
	28:26	Reserved	
		Format:	MBZ
25:24	Color Depth		
	Value	Name	
	00b	8 Bit Color	
	01b	16 Bit Color(565)	
	10b	16 Bit Color(1555)	
	11b	32 Bit Color	
23:16	Raster Operation		
15:0	Destination Pitch in DWords 2's complement (Negative Pitch Not allowed for Pixel nor Text) For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).		
2	31:16	ClipRect Y1 Coordinate (Top) (30:16 = 15 bit positive number)	
	BR24	15:0	ClipRect X1 Coordinate (Left) (14:00 = 15 bit positive number)
3	31:16	ClipRect Y2 Coordinate (Bottom) (30:16 = 15 bit positive number)	
	BR25	15:0	ClipRect X2 Coordinate (Right) (14:00 = 15 bit positive number)
4	31:0	Setup Destination Base Address	
		Format:	GraphicsAddress[31:0]
		Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	
5	31:16	Reserved	
		Format:	MBZ
BR27		Should be programmed all 0's for 48bit addressing.	

XY_SETUP_BLT				
	15:0	<p>Setup Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			
6 BR05	31:0	<p>Setup Background Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0] All</p>		
7 BR06	31:0	<p>Setup Foreground Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0] (SLB and TB only)</p>		
8 BR07	31:0	<p>Setup Pattern Base Address for Color Pattern</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>(26:06 are implemented) (SLB only) (Note no NPO2 change here). The pattern data must be located in linear memory. Lower 32bits of the 48bit addressing. The Pattern Base Address programmed, must always be Cache Line (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
9 BR30	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Setup Pattern Base Address for Color Pattern High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			

XY_SETUP_CLIP_BLT

XY_SETUP_CLIP_BLT		
Project:	BDW	
Source:	BlitterCS	
Length Bias:	2	
<p>This command is used to only change the clip coordinate registers. These are the same clipping registers as the Setup clipping registers above.</p>		
DWord	Bit	Description
0 BR00	31:29	Client
		Default Value: 02h 2D Processor
		Format: Opcode
	28:22	Instruction Target(Opcode)
		Default Value: 03h
		Format: Opcode
	21:12	Reserved
Format: MBZ		
11	Tiling Enable	
	Value	Name
	0b	Tiling Disabled (Linear Blit)
	1b	Tiling Enabled (Tile-X or Tile-Y)
10:8	Reserved	
	Format: MBZ	
7:0	DWord Length	
	Default Value: 01h	
1 BR24	31:16	ClipRect Y1 Coordinate (Top) (30:16 = 15 bit positive number)
	15:0	ClipRect X1 Coordinate (Left) (14:00 = 15 bit positive number)
2 BR25	31:16	ClipRect Y2 Coordinate (Bottom) (30:16 = 15 bit positive number)
	15:0	ClipRect X2 Coordinate (Right) (14:00 = 15 bit positive number)

XY_SETUP_MONO_PATTERN_SL_BLT

XY_SETUP_MONO_PATTERN_SL_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This setup instruction supplies common setup information including clipping coordinates used exclusively with the following instruction: XY_SCANLINE_BLT (SLB) - 1 scan line of monochrome pattern and destination are the only operands allowed.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value: 02h 2D Processor	
			Format: Opcode
	28:22	Instruction Target(Opcode)	
		Default Value: 11h	
			Format: Opcode
	21:20	32 bpp Byte Mask	
		Value	Name
		1xb	Write Alpha Channel
		x1b	Write RGB Channel
19:12	Reserved		
	Format: MBZ		
11	Tiling Enable		
	Value	Name	
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled (Tile-X or Tile-Y)	
10:8	Reserved		
	Format: MBZ		
7:0	DWord Length		
	Default Value: 08h		
1 BR01	31	Solid Pattern Select (SLB and Pixel only)	
		Value	Name
		0	No Solid Pattern
		1	Solid Pattern

XY_SETUP_MONO_PATTERN_SL_BLT												
	30	Clipping Enabled <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 50%;">Value</th> <th style="width: 50%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled				
	Value	Name										
	0b	Disabled										
	1b	Enabled										
	29	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
	Format:	MBZ										
	28	Mono Pattern Transparency Mode <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Use Background</td> </tr> <tr> <td>1b</td> <td>Transparency Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Use Background	1b	Transparency Enabled				
	Value	Name										
	0b	Use Background										
	1b	Transparency Enabled										
	27:26	Reserved <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ								
	Format:	MBZ										
	25:24	Color Depth <table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 30%;">Value</th> <th style="width: 70%;">Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name										
	00b	8 Bit Color										
01b	16 Bit Color(565)											
10b	16 Bit Color(1555)											
11b	32 Bit Color											
23:16	Raster Operation											
15:0	Destination Pitch in DWords 2's complement (Negative Pitch Not allowed for Pixel nor Text) For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).											
2	31:16	ClipRect Y1 Coordinate (Top) (30:16 = 15 bit positive number)										
BR24	15:0	ClipRect X1 Coordinate (Left) (14:00 = 15 bit positive number)										
3	31:16	ClipRect Y2 Coordinate (Bottom) (30:16 = 15 bit positive number)										
BR25	15:0	ClipRect X2 Coordinate (Right) (14:00 = 15 bit positive number)										
4	31:0	Setup Destination Base Address <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.	Format:	GraphicsAddress[31:0]								
Format:	GraphicsAddress[31:0]											
BR09												

XY_SETUP_MONO_PATTERN_SL_BLT				
5 BR27	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Should be programmed all 0's for 48bit addressing.</p>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Setup Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>Should be programmed with the upper 16bits of the 48bit addressing.</p>	Format:	GraphicsAddress[47:32]	
Format:	GraphicsAddress[47:32]			
6 BR05	31:0	<p>Setup Background Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0] All</p>		
7 BR06	31:0	<p>Setup Foreground Color</p> <p>8 bit = [7:0], 16 bit = [15:0], 32 bit = [31:0] (SLB and TB only)</p>		
8 BR20	31:0	<p>DW0 (least significant) for a Monochrome Pattern</p>		
9 BR21	31:0	<p>DW1 (most significant) for a Monochrome Pattern</p>		

XY_SRC_COPY_BLT

XY_SRC_COPY_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This BLT instruction performs a color source copy where the only operands involved is a color source and destination of the same bit width.</p> <p>The source and destination operands may overlap, which means that the X and Y directions can be either forward or backwards. The BLT Engine takes care of all situations. The base addresses plus the X and Y coordinates determine if there is an overlap between the source and destination operands. If the base addresses of the source and destination are the same and the Source X1 is less than Destination X1, then the BLT Engine performs the accesses in the X-backwards access pattern. There is no need to look for an actual overlap. If the base addresses are the same and Source Y1 is less than Destination Y1, then the scan line accesses start at Destination Y2 with the corresponding source scan line and the strides are subtracted for every scan line access. The ROP value chosen must involve source and no pattern data in the ROP operation.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	53h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
	x1b	Write RGB Channel	
19:16	Reserved		
	Format:	MBZ	
15	Src Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
14:12	Reserved		
	Format:	MBZ	

XY_SRC_COPY_BLT											
	11	Dest Tiling Enable									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
		Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)									
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.								
10:8	Reserved										
Format:		MBZ									
	7:0	DWord Length									
		Format:	=n								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>08h</td> <td></td> </tr> </tbody> </table>		Value	Name	08h						
	Value	Name									
08h											
Format:		MBZ									
1 BR13	31	Reserved									
	Format:		MBZ								
	30	Clipping Enabled									
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>		Value	Name	0b	Disabled	1b	Enabled			
	Value	Name									
	0b	Disabled									
	1b	Enabled									
29:26	Reserved										
Format:		MBZ									
25:24	Color Depth										
<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>		Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
Value	Name										
00b	8 Bit Color										
01b	16 Bit Color(565)										
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									

XY_SRC_COPY_BLT				
4 BR09	31:0	<p>Destination Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address must be 4KB-aligned. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
	Format:	GraphicsAddress[31:0]		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Must be all 0's for 48bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
5 BR27	15:0	<p>Destination Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>The upper 16bits of the 48-bit address.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		
31:16	<p>Source Y1 Coordinate (Top) 16 bit signed number.</p>			
6 BR26	15:0	<p>Source X1 Coordinate (Left) 16 bit signed number.</p>		
7 BR11	31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table>	Format:	MBZ
	Format:	MBZ		
15:0	<p>Source Pitch (double word aligned) and in DWords 2's complement. For Tiled Src (bit 15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).</p>			
8 BR12	31:0	<p>Source Base Address</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> <p>Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Src Tiling is enabled (Bit_15 enabled), this address must be 4KB-aligned. When Tiling is not enabled, this address should be CL (64byte) aligned.</p>	Format:	GraphicsAddress[31:0]
	Format:	GraphicsAddress[31:0]		
31:16	<p>Reserved</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 70%;">Format:</td> <td>MBZ</td> </tr> </table> <p>Must be all 0's for 48-bit addressing.</p>	Format:	MBZ	
Format:	MBZ			
9 BR28	15:0	<p>Source Base Address High</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> <p>The upper 16 bits of the 48-bit address.</p>	Format:	GraphicsAddress[47:32]
	Format:	GraphicsAddress[47:32]		



XY_SRC_COPY_CHROMA_BLT

XY_SRC_COPY_CHROMA_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This BLT instruction performs a color source copy with chroma-keying where the only operands involved is a color source and destination of the same bit width.</p> <p>The source and destination operands may overlap, which means that the X and Y directions can be either forward or backwards. The BLT Engine takes care of all situations. The base addresses plus the X and Y coordinates determine if there is an overlap between the source and destination operands. If the base addresses of the source and destination are the same and the Source X1 is less than Destination X1, then the BLT Engine performs the accesses in the X-backwards access pattern. There is no need to look for an actual overlap. If the base addresses are the same and Source Y1 is less than Destination Y1, then the scan line accesses start at Destination Y2 with the corresponding source scan line and the strides are subtracted for every scan line access. The ROP value chosen must involve source and no pattern data in the ROP operation.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	73h
		Format:	Opcode
	21:20	32bpp Byte Mask	
		This field is only used for 32bpp.	
		Value	Name
		00b	[Default]
1xb		Write Alpha Channel	
x1b	Write RGB Channel		
19:17	Transparency Range Mode (chroma-key)		
16	Reserved		
	Format:	MBZ	
15	Src Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
14:12	Reserved		
	Format:	MBZ	

XY_SRC_COPY_CHROMA_BLT											
	11	Dest Tiling Enable									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Tiling Disabled (Linear Blit)</td> <td></td> </tr> <tr> <td>1b</td> <td>Tiling Enabled</td> <td>[BDW] [BDW]: Tile-X or Tile-Y.</td> </tr> </tbody> </table>	Value	Name	Description	0b	Tiling Disabled (Linear Blit)		1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
		Value	Name	Description							
	0b	Tiling Disabled (Linear Blit)									
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.								
10:8	Reserved										
Format:		MBZ									
	7:0	DWord Length									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0Ah</td> <td></td> </tr> </tbody> </table>	Value	Name	0Ah						
Value	Name										
0Ah											
1 BR13	31	Reserved									
		Format:		MBZ							
	30	Clipping Enabled									
		<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Disabled</td> </tr> <tr> <td>1b</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Name	0b	Disabled	1b	Enabled			
		Value	Name								
	0b	Disabled									
	1b	Enabled									
29:26	Reserved										
Format:		MBZ									
25:24	Color Depth										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>8 Bit Color</td> </tr> <tr> <td>01b</td> <td>16 Bit Color(565)</td> </tr> <tr> <td>10b</td> <td>16 Bit Color(1555)</td> </tr> <tr> <td>11b</td> <td>32 Bit Color</td> </tr> </tbody> </table>	Value	Name	00b	8 Bit Color	01b	16 Bit Color(565)	10b	16 Bit Color(1555)	11b	32 Bit Color
	Value	Name									
	00b	8 Bit Color									
	01b	16 Bit Color(565)									
10b	16 Bit Color(1555)										
11b	32 Bit Color										
23:16	Raster Operation										
15:0	Destination Pitch in DWords 2's complement For Tiled surfaces (bit_11 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).										
2 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.									
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.									
3 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.									
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.									

XY_SRC_COPY_CHROMA_BLT		
4 BR09	31:0	Destination Base Address Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_11 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.
		Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
5 BR27	31:16	Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
	15:0	Destination Base Address High Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.
6 BR26	31:16	Source Y1 Coordinate (Top) 16 bit signed number.
	15:0	Source X1 Coordinate (Left) 16 bit signed number.
7 BR11	31:16	Reserved Format: MBZ
	15:0	Source Pitch (double word aligned) and in DWords 2's complement. For Tiled Src (bit 15 enabled) this pitch is of 512Byte granularity for Tile-X, 128B granularity for Tile-Y and can be upto 128Kbytes (or 32KDwords).
8 BR12	31:0	Source Base Address Format: GraphicsAddress[31:0] Base address of the destination surface: X=0, Y=0. Lower 32bits of the 48bit addressing. When Tiling is enabled (Bit_15 enabled), this address is limited to 4Kbytes. When Tiling is not enabled, this address should be CL (64byte) aligned.
		Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
9 BR28	31:16	Reserved Format: MBZ Should be programmed all 0's for 48bit addressing.
	15:0	Source Base Address High Format: GraphicsAddress[47:32] Should be programmed with the upper 16bits of the 48bit addressing.
10 BR18	31:0	Transparency Color Low (Chroma-key Low = Pixel Greater or Equal)

XY_SRC_COPY_CHROMA_BLT

11	31:0	Transparency Color High (Chroma-key High = Pixel Less or Equal)
BR19		

XY_TEXT_BLT

XY_TEXT_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>All source scan lines and pixels that fall within the ClipRect Y and X coordinates are written. The source address corresponds to Destination X1 and Y1 coordinate.</p> <p>Text is either bit or byte packed. Bit packed means that the next scan line starts 1 pixel after the end of the current scan line with no bit padding. Byte packed means that the next scan line starts on the first bit of the next byte boundary after the last bit of the current line.</p> <p>Source expansion color registers are always in the SETUP_BLT.</p> <p>Negative Stride (= Pitch) is NOT ALLOWED.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	26h
		Format:	Opcode
	21:17	Reserved	
		Format:	MBZ
	16	Bit / Byte Packed	
		Byte packed is for the NT driver.	
Value		Name	
0		Bit	
1	Byte		
15:12	Reserved		
	Format:	MBZ	
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Reserved		
	Format:	MBZ	
7:0	DWord Length		
	Default Value:	03h	
1	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.	

XY_TEXT_BLT				
BR22	15:0	Destination X1 Coordinate (Left) 16 bit signed number.		
2	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.		
BR23	15:0	Destination X2 Coordinate (Right) 16 bit signed number.		
3 BR12	31:0	Source Address <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[31:0]</td> </tr> </table> (address of the first byte on scan line corresponding to Dst X1, Y1). Lower 32bits of the 48bit addressing. (Note no NPO2 change here). Since Text data is Monosource data, the Text source Base Address programmed, must always be Cache Line (64byte) aligned.	Format:	GraphicsAddress[31:0]
Format:	GraphicsAddress[31:0]			
4 BR28	31:16	Reserved <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 60%;">Format:</td> <td>MBZ</td> </tr> </table> Should be programmed with all "0"s for 48bit addressing.	Format:	MBZ
Format:	MBZ			
	15:0	Source Address High <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Format:</td> <td>GraphicsAddress[47:32]</td> </tr> </table> Should be programmed with the upper 16bits of the 48bit addressing.	Format:	GraphicsAddress[47:32]
Format:	GraphicsAddress[47:32]			



XY_TEXT_IMMEDIATE_BLT

XY_TEXT_IMMEDIATE_BLT			
Project:	BDW		
Source:	BlitterCS		
Length Bias:	2		
<p>This instruction allows the Driver to send data through the instruction stream that eliminates the read latency of reading a source from memory.</p> <p>If an operand is in system cacheable memory and either small or only accessed once, it can be copied directly to the instruction stream versus to graphics accessible memory. The IMMEDIATE_BLT data MUST transfer an even number of doublewords.</p> <p>The BLT engine will hang if it does not get an even number of doublewords. All source scan lines and pixels that fall within the ClipRect X and Y coordinates are written. The source data corresponds to Destination X1 and Y1 coordinate.</p> <p>Source expansion color registers are always in the SETUP_BLT. NEGATIVE STRIDE (= PITCH) IS NOT ALLOWED.</p>			
DWord	Bit	Description	
0 BR00	31:29	Client	
		Default Value:	02h 2D Processor
		Format:	Opcode
	28:22	Instruction Target(Opcode)	
		Default Value:	31h
		Format:	Opcode
	21:17	Reserved	
		Format:	MBZ
	16	Bit / Byte Packed	
		Byte packed is for the NT driver.	
Value		Name	
0		Bit	
1	Byte		
15:12	Reserved		
	Format:	MBZ	
11	Tiling Enable		
	Value	Name	Description
	0b	Tiling Disabled (Linear Blit)	
	1b	Tiling Enabled	[BDW] [BDW]: Tile-X or Tile-Y.
10:8	Reserved		
	Format:	MBZ	

XY_TEXT_IMMEDIATE_BLT				
	7:0	DWord Length <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Default Value:</td> <td>01h Excludes DWORD 0,1</td> </tr> </table> 01 + DWL = (Number of Immediate double words)h	Default Value:	01h Excludes DWORD 0,1
Default Value:	01h Excludes DWORD 0,1			
1 BR22	31:16	Destination Y1 Coordinate (Top) 16 bit signed number.		
	15:0	Destination X1 Coordinate (Left) 16 bit signed number.		
2 BR23	31:16	Destination Y2 Coordinate (Bottom) 16 bit signed number.		
	15:0	Destination X2 Coordinate (Right) 16 bit signed number.		
3..n	31:0	Immediate Data		