Intel® Iris® Xe and UHD Graphics Open Source

Programmer's Reference Manual

For the 2020-2021 11th Generation Intel Xeon®, Core™, Celeron®, Pentium® Gold Processors based on the "Tiger Lake" Platform

Volume 2b: Command Reference: Enumerations

December 2021, Revision 1.0

Notices and Disclaimers

Table of Contents

3D_Color_Buffer_Blend_Factor

3D_Color_Buffer_Blend_Factor	
Size (in bits):	5

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

3D_Color_Buffer_Blend_Function

3D_Color_Buffer_Blend_Function		
Size (in bits): 3		

Value	Name	Description
0	BLENDFUNCTION_ADD	BLENDFUNCTION_ADD
1	BLENDFUNCTION_SUBTRACT	BLENDFUNCTION_SUBTRACT
2	BLENDFUNCTION_REVERSE_SUBTRACT	BLENDFUNCTION_REVERSE_SUBTRACT
3	BLENDFUNCTION_MIN	BLENDFUNCTION_MIN
4	BLENDFUNCTION_MAX	BLENDFUNCTION_MAX
5 - 7	Reserved	

3D_Compare_Function

3D_Compare_Function

Size (in bits): 3

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

3D_Logic_Op_Function

3D_Logic_Op_Function	
Size (in bits):	4

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

3D_Prim_Topo_Type

3D_Prim_Topo_Type

Source:	RenderCS
Size (in bits):	6

The following table defines the encoding of the Primitive Topology Type field. See 3D Pipeline for details, programming restrictions, diagrams and a discussion of the basic primitive types.

Value	Name	Description
00h	Reserved	
01h	3DPRIM_POINTLIST	
02h	3DPRIM_LINELIST	
03h	3DPRIM_LINESTRIP	
04h	3DPRIM_TRILIST	
05h	3DPRIM_TRISTRIP	
06h	3DPRIM_TRIFAN	
07h	3DPRIM_QUADLIST	
08h	3DPRIM_QUADSTRIP	
09h	3DPRIM_LINELIST_ADJ	
0Ah	3DPRIM_LINESTRIP_ADJ	
0Bh	3DPRIM_TRILIST_ADJ	
0Ch	3DPRIM_TRISTRIP_ADJ	
0Dh	3DPRIM_TRISTRIP_REVERSE	
0Eh	3DPRIM_POLYGON	
0Fh	3DPRIM_RECTLIST	
10h	3DPRIM_LINELOOP	
11h	3DPRIM_POINTLIST _BF	
12h	3DPRIM_LINESTRIP_CONT	
13h	3DPRIM_LINESTRIP_BF	
14h	3DPRIM_LINESTRIP_CONT_BF	
15h	Reserved	
16h	3DPRIM_TRIFAN_NOSTIPPLE	
17h	Reserved	
18h	Reserved	
19h	Reserved	
1Ah	Reserved	
[1Bh-1Fh]	Reserved	
20h	3DPRIM_PATCHLIST_1	List of 1-vertex patches

3D_Prim_Topo_Type

21h	3DPRIM_PATCHLIST_2	
22h	3DPRIM_PATCHLIST_3	
23h	3DPRIM_PATCHLIST_4	
24h	3DPRIM_PATCHLIST_5	
25h	3DPRIM_PATCHLIST_6	
26h	3DPRIM_PATCHLIST_7	
27h	3DPRIM_PATCHLIST_8	
28h	3DPRIM_PATCHLIST_9	
29h	3DPRIM_PATCHLIST_10	
2ah	3DPRIM_PATCHLIST_11	
2bh	3DPRIM_PATCHLIST_12	
2ch	3DPRIM_PATCHLIST_13	
2dh	3DPRIM_PATCHLIST_14	
2eh	3DPRIM_PATCHLIST_15	
2fh	3DPRIM_PATCHLIST_16	
30h	3DPRIM_PATCHLIST_17	
31h	3DPRIM_PATCHLIST_18	
32h	3DPRIM_PATCHLIST_19	
33h	3DPRIM_PATCHLIST_20	
34h	3DPRIM_PATCHLIST_21	
35h	3DPRIM_PATCHLIST_22	
36h	3DPRIM_PATCHLIST_23	
37h	3DPRIM_PATCHLIST_24	
38h	3DPRIM_PATCHLIST_25	
39h	3DPRIM_PATCHLIST_26	
3ah	3DPRIM_PATCHLIST_27	
3bh	3DPRIM_PATCHLIST_28	
3ch	3DPRIM_PATCHLIST_29	
3dh	3DPRIM_PATCHLIST_30	
3eh	3DPRIM_PATCHLIST_31	
3Fh	3DPRIM_PATCHLIST_32	List of 32-vertex patches

3D_Stencil_Operation

3D_Stencil_Operation

Source:	RenderCS
Size (in bits):	3

Value	Name
0	STENCILOP_KEEP
1	STENCILOP_ZERO
2	STENCILOP_REPLACE
3	STENCILOP_INCRSAT
4	STENCILOP_DECRSAT
5	STENCILOP_INCR
6	STENCILOP_DECR
7	STENCILOP_INVERT

3D_Vertex_Component_Control

3D_Vertex_Component_Control	
Source:	RenderCS
Size (in bits):	3

Value	Name	Description
0	VFCOMP_NOSTORE	Don't store this component. (Not valid for Component 0, but can be used for Component 1-3). Once this setting is used for a component, all higher-numbered components (if any) MUST also use this setting. (I.e., no holes within any particular vertex element). VFCOMP_NOSTORE will not store a component if the SourceElementFormat is R64_PASSTHRU or R64G64_PASSTHRU and it is used on component 2 and 3 else 0 will be stored.
1	VFCOMP_STORE_SRC	Store corresponding component from format-converted source element. Storing a component that is not included in the Source Element Format results in an UNPREDICTABLE value being stored. VF will process Component Control fields within a VERTEX_ELEMENT_STATE structure sequentially, starting with Component 0 Control. For each Component Control field in this sequence, when VF detects (a) the Component Control field is set to STORE_SRC <u>and</u> (b) the component is <u>not</u> overwritten by an SGV, VF will store a component of the source vertex data into the destination component. The first such Component Control field satisfying this criteria will use Component 0 of the source vertex data, the second such Component Control field will use Component 1 of the source vertex data, and so on. Therefore, when a lower-numbered Component Control field (a) is set to something other than STORE_SRC (e.g., STORE_0) <u>or</u> (b) the component is overwritten with an SGV, the source vertex component used when a higher-numbered Component Control fields is set to STORE_SRC will be impacted.
2	VFCOMP_STORE_0	Store 0 (interpreted as 0.0f if accessed as a float value)
3	VFCOMP_STORE_1_FP	Store 1.0f
4	VFCOMP_STORE_1_INT	Store 0x1
5-6	-	Reserved
7	VFCOMP_STORE_PID	Store Primitive ID (as U32) Software can no longer use this encoding as PrimitiveID is passed down the FF pipeline - see explanation above.

AccWrCtrl

AccWrCtrl	
Size (in bits):	1
This field allows per instruction accumulator write control. When set, the result is written to both destination and accumulator.	

Value	Name
0	Don't write to ACC **[Default]**
1	Update ACC

AddrMode

AddrMode	
Source:	EuIsa
Size (in bits):	1

Addressing Mode This field determines the addressing method of the operand. Normally the destination operand and each source operand each have a distinct addressing mode field. When it is cleared, the register address of the operand is directly provided by bits in the instruction word. It is called a direct register addressing mode. When it is set, the register address of the operand is computed based on the address register value and an address immediate field in the instruction word. This is referred to as a register-indirect register addressing mode. This field applies to the destination operand and the first source operand, src0. Support for src1 is device dependent. See Table XX (Indirect source addressing support available in device hardware) in ISA Execution Environment for details.

Programming Notes
Instructions with 3 source operands use Direct Addressing.

Value	Name	Description
0	Direct	'Direct' register addressing
1	Indirect	'Register-Indirect' (or in short 'Indirect'). Register-indirect register addressing

AtomicCtrl

AtomicCtrl	
Source:	EuIsa
Size (in bits):	1

Enables the atomic instruction control option

Value	Name	Description
0b	No Operation **[Default]**	This value leaves thread switching up to the EU's scheduler.
1b	Atomic	Prevent any thread switch immediately following this instruction. Always execute the next instruction (which may not be next sequentially if the current instruction branches). The next instruction gets highest priority in the thread arbitration for the execution pipelines.

Attribute_Component_Format

Attribute_Component_Format	
Source:	RenderCS
Size (in bits):	2

Value	Name	Description
00b	disabled **[Default]**	All components disabled
01b	.xy	2D attribute, z and w components disabled
10b	.xyz	3D attribute, w components disabled
11b	.xyzw	4D attribute, no disabled components

ChanOff

ChanOff

Source:	EuIsa
Size (in bits):	3

Channel Offset

This enumeration (instruction field) provides offset information for ARF selection. The can be thought of as a starting channel offset for the execution mask and other ARF registers implicitly accessed. Some of the ARFs affected by this offset are:

- ce access (channel enable / the execution mask): bitwise offset
- f# access: bitwise offset (e.g. in predication or conditional modifier access)
- acc# access via AccWrEn (implicit only): in subregister units of the data type

Note: ChanOff is functionally the same as a concatenation of the previous QtrCtrl and NibCtrl and supersedes these fields.

Restriction
The execution size (ExecSize) must be a factor of the chosen offset. For instance, M28 (offset 28) can only be used with SIMD1, SIMD2, and SIMD4.

Value	Name	Description
000b	M0 **[Default]**	ARF access begins at channel 0
001b	M4	ARF access begins at channel 4 (e.g. 4 bits in on ce and f#
010b	M8	
011b	M12	
100b	M16	
101b	M20	
110b	M24	
111b	M28	

COMPONENT_ENABLES

COMPONENT_ENABLES	
Source:	RenderCS
Size (in bits):	4

If enabled, the component will be stored in the URB.

Value	Name
0000b	NONE
0001b	X
0010b	Y
0011b	XY
0100b	Z
0101b	XZ
0110b	YZ
0111b	XYZ
1000b	W
1001b	XW
1010b	YW
1011b	XYW
1100b	ZW
1101b	XZW
1110b	YZW
1111b	XYZW

DPASOperandPrecision

DPASOperandPrecision	
Source:	EuIsa
Size (in bits):	3

This operand defines the number of bits per element in the dpas instruction. E.g. s8 would indicate that a dword is chunked into 4 x 8b signed values; u2 would indicate that the operand a DWORD is broken into 16 x 2b unsigned values.

Value	Name
000b	u1
001b	u2
010b	u4
011b	u8
100b	s1
101b	s2
110b	s4
111b	s8

EU_OPCODE

EU_OPCODE	
Source:	EuIsa
Size (in bits):	7

Value	Name	
40h	add	Instruction: add
4Eh	addc	Instruction: addc
65h	and	Instruction: and
6Ch	asr	Instruction: asr
42h	avg	Instruction: avg
78h	bfe	Instruction: bfe
79h	bfi1	Instruction: bfi1
7Ah	bfi2	Instruction: bfi2
77h	bfrev	Instruction: bfrev
23h	brc	Instruction: brc
21h	brd	Instruction: brd
28h	break	Instruction: break
2Ch	call	Instruction: call
2Bh	calla	Instruction: calla
4Dh	cbit	Instruction: cbit
70h	cmp	Instruction: cmp
71h	cmpn	Instruction: cmpn
29h	cont	Instruction: cont
72h	csel	Instruction: csel
24h	else	Instruction: else
25h	endif	Instruction: endif
4Bh	fbh	Instruction: fbh
4Ch	fbl	Instruction: fbl
43h	frc	Instruction: frc

EU_OPCODE

2Eh	goto	Instruction:	goto
2Ah	halt	Instruction:	halt
22h	if	Instruction:	if
0h	illegal	Instruction:	illegal
20h	jmpi	Instruction:	jmpi
2Fh	join	Instruction:	join
4Ah	lzd	Instruction:	lzd
48h	mac	Instruction:	mac
49h	mach	Instruction:	mach
5Bh	mad	Instruction:	mad
38h	math	Instruction:	math
61h	mov	Instruction:	mov
63h	movi	Instruction:	movi
41h	mul	Instruction:	mul
60h	nop	Instruction:	nop
64h	not	Instruction:	not
66h	or	Instruction:	or
2Dh	ret	Instruction:	ret
45h	rndd	Instruction:	rndd
46h	rnde	Instruction:	rnde
44h	rndu	Instruction:	rndu
47h	rndz	Instruction:	rndz
6Fh	rol	Instruction:	rol
6Eh	ror	Instruction:	ror
62h	sel	Instruction:	sel
31h	send	Instruction:	send
32h	sendc	Instruction:	sendc
69h	shl	Instruction:	shl
68h	shr	Instruction:	shr

EU_OPCODE		
4Fh	subb	Instruction: subb
1h	sync	Instruction: sync
27h	while	Instruction: while
67h	xor	Instruction: xor

ExecSize

ExecSize	
Source:	EuIsa
Size (in bits):	3

Execution Size This field determines the number of channels operating in parallel for this instruction. The size cannot exceed the maximum number of channels allowed for the given data type.

Restriction
An operand's Width must be less-than-or-equal to ExecSize

Value	Name	Programming Notes
000b	1 Channel (Scalar operation) **[Default]**	
001b	2 Channels	
010b	4 Channels	
011b	8 Channels	
100b	16 Channels	[] 4-byte or smaller data types. Excludes DF, Q, and UQ types.
101b	32 Channels	[] 2-byte or 1-byte data types. Excludes D, DF, F, Q, UD, and UQ types.
110b-111b	Reserved	

Fixed Function ID

FFID - Fixed Function ID

Size (in bits):	4

Fixed functions are hardware units that execute complex graphics or media command on behalf of application software. Some fixed functions send work down a pipeline through a series of fixed functions. Multiple fixed functions can be running at the same time. The GPU tracks activity from the fixed function with its FFID.

Programming Notes
Software does not specify the FFID, and does not normally use the FFID value. The FFID value is available to a EU thread in an ARF.

Value	Name	Description
00h	Null	
03h	POSH Vertex Shader	
04h	Hull Shader	
05h	Domain Shader	
06h	Texel Shader	Adaptive Multi-Frequency Shader
07h	General Purpose Thread Spawner	GPGPU command queue thread dispatcher
08h	General Purpose Asynchronous Thread Spawner	GPGPU command queue's thread dispatcher for secondary queue
09h	Vertex Shader	
0Ch	Geometry Shader	

FlagModifier

FlagModifier	
Source:	EuIsa
Size (in bits):	4

Flag (Conditional) Modifier - This field sets the flag register based on the internal conditional signals output from the execution pipe such as sign, zero, overflow and NaNs, etc. If this field is set to 0000, no flag registers are updated. Flag registers are not updated for instructions with embedded compares. This field may also be referred to as the flag destination control field.

Value	Name	Description
0000b	None **[Default]**	None
0001b	(ze)	Zero
0010b	(nz)	NotZero
0011b	(gt)	Greater-than
0100b	(ge)	Greater-than-or-equal
0101b	(lt)	Less-than
0110b	(le)	Less-than-or-equal
0111b	Reserved	
1000b	(ov)	Overflow
1001b	(un)	Unordered (NaN)
1110b-1111b	Reserved	

GW_FENCE_PORTS

GW_FENCE_PORTS	
Source:	BSpec
Size (in bits):	0

Bit mask specifies the list of data ports to be fenced.

Value	Name	Description
0	None **[Default]**	No fence is performed when no bits are set.

HorzStride

HorzStride	
Source:	EuIsa
Size (in bits):	2

Horizontal Stride This field provides the distance in unit of data elements between two adjacent data elements within a row (horizontal) in the register region for the operand. This field applies to both destination and source operands. This field is not present for an immediate source operand.

A horizontal stride of 0 is used for a row that is one-element wide, useful when an instruction repeats a column value or repeats a scalar value. For example, adding a single column to every column in a 2D array or adding a scalar to every element in a 2D array uses HorzStride of 0. A horizontal stride of 1 indicates that elements are adjacent within a row. References to HorzStride in this volume normally reference the value not the encoding, so there are references to HorzStride of 4, which is encoded as 11b.

Value	Name
00b	0 elements
01b	1 elements
10b	2 elements
11b	4 elements

ImmDataType

ImmDataType	
Source:	EuIsa
Size (in bits):	4

Numeric data type of source and destination operand. Three source instructions use a 3-bit encoding that allows fewer data types.

Value	Name	Description
0000b	:uv	Packed Unsigned Half-Byte Integer Vector, 8 x 4-Bit Unsigned Integer
0001b	:uw	Unsigned Word (16-bit) Integer
0010b	:ud	Unsigned DoubleWord (32-bit) Integer
0011b	:uq	Unsigned Quadword (64-bit) Integer
0100b	:v	Packed Signed Half-Byte Integer Vector, 8 x 4-Bit Signed Integer
0101b	:w	Signed Word (16-bit) Integer
0110b	:d	Signed DoubleWord (32-bit) Integer
0111b	:q	Signed QuadWord (64-bit) Integer
1000b	:vf	Packed Restricted Float Vector, 4 x 8-Bit Restricted Precision Floating-Point Number.
1001b	:hf	Half (16-bit) Float
1010b	:f	Single-Precision (32-bit) Float
1011b	:df	Double-Precision (64-bit) Float
1100b	Reserved	
1101b	Reserved	
[1110b-1111b]	Reserved	

MathFC

MathFC	
Source:	Eulsa
Size (in bits):	4

Math Function Control

Value	Name	Description	Programming Notes	
0001b	INV	Reciprocal (Multiplicative Inverse): 1/src0	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / - Denorm | -inf | NaN | |Dest - IEEE mode | +0 | +inf | - inf | - 0 | NaN | |Dest - ALT mode | | +fmax | -fmax | | NaN | | Src | +inf | +0 | -0 | -inf | NaN | |Dest - IEEE mode | +0 | +inf | - inf | - 0 | NaN |	Syntax: MATH_UNARY_REGIMM
0010b	LOG	Natural log: ln(src0)	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / - Denorm | -inf | -F | NaN | |Dest - IEEE mode | +inf | - inf | - inf | NaN | NaN | NaN | |Dest - ALT mode | | -fmax | -fmax | | +F | NaN | |	Syntax: MATH_UNARY_REGIMM

			Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | +inf | - inf | - inf | NaN | NaN | NaN |	
0011b	EXP	Exponential (E^src0)	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / - Denorm | -inf | - F | NaN | |Dest - IEEE mode | +inf | 1 | 1 | 0 | +F | NaN | |Dest - ALT mode | | 1 | 1 | | +F | NaN | | Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | +inf | 1 | 1 | 0 | +F | NaN |	Syntax: MATH_UNARY_REGIMM
0100b	SQT	Square Root	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / - Denorm | -inf | - F | NaN | |Dest - IEEE mode | +inf | 0 | -0 | NaN | NaN | NaN | |Dest - ALT mode | | 0 | 0 | | +F | NaN | | Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | +inf | 0 | -0 |	Syntax: MATH_UNARY_REGIMM

MathFC

			NaN | NaN | NaN |	
0101b	RSQT	Reciprocal Square Root: 1/sqt(src)	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / -Denorm | -inf | -F | NaN | |Dest - IEEE mode | +0 | +inf | -inf | NaN | NaN | NaN | |Dest - ALT mode | | +fmax | +fmax | +F | NaN | | Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | +0 | +inf | -inf | NaN | NaN | NaN |	Syntax: MATH_UNARY_REGIMM
0110b	SIN	Sine function. sin(src0)	Table:special value processing | Src | +inf | +0 / +Denorm | -0 / -Denorm | -inf | -F | NaN | |Dest - IEEE mode | NaN | +0 | -0 | NaN | -1 to 1 | NaN | |Dest - ALT mode | | +0 | -0 | -1 to 1 | NaN | | Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | NaN | +0 | -0 | NaN | -1 to 1 | NaN |	Syntax: MATH_UNARY_REGIMM

MathFC

0111b	COS	Cosine function. cos(src0)	Table:special value processing `| Src | +inf | +0 / +Denorm | -0 / -Denorm | -inf | -F | NaN | |Dest - IEEE mode | NaN | +0 | -0 | NaN | -1 to 1 | NaN | |Dest - ALT mode | | +1 | +1 | | -1 to 1 | NaN | | Src | +inf | +0 | -0 | -inf | -F | NaN | |Dest - IEEE mode | NaN | +0 | -0 | NaN | -1 to 1 | NaN |`	Syntax: MATH_UNARY_REGIMM
1000b	Reserved			
[1001b-1010b]	Reserved	Previously fdiv and pow. Fdiv x/y may be emulated via mul and inv: x*inv(y). Pow can be replaced via exp and log: X^Y = E^(Y*LOG(X))		
1011b	IDIV	Integer Divide with Quotient and Remainder. The quotient goes in the destination register; the remainder goes in the following register.		Syntax: MATH_BINARY_REG_REGIMM
1100b	IQOT	Integer Quotient only		Syntax: MATH_BINARY_REG_REGIMM
1101b	IREM	Integer Remainder only		Syntax: MATH_BINARY_REG_REGIMM
1110b	INVM	Reciprocal Macro for IEEE754-compliant fdiv		Syntax: MATH_MACRO_BINARY_REG_REG
1111b	RSQTM	Reciprocal Square Root Macro for IEEE754-compliant rsqt		Syntax: MATH_MACRO_UNARY_REG

MediaCompressionFormat

CMP_FMT_MEDIA_TGL - MediaCompressionFormat _TGL

Size (in bits): 5

Media Compression Format

Value	Name	Description
00000b	CMF_0x0	Reserved
00001b	CMF_0x1	Reserved
00010b	CMF_0x2	Reserved
00011b	CMF_0x3	YUY2
00100b	CMF_0x4	Y410 (10:10:10:2)
00101b	CMF_0x5	Y216
00110b	CMF_0x6	Y416
00111b	CMF_0x7	P010
01000b	CMF_0x8	P016
01001b	CMF_0x9	AYUV
01010b	CMF_0xA	ARGB 8b
01011b	CMF_0xB	YCRCB_SwapY
01100b	CMF_0xC	YCRCB_SwapUV
01101b	CMF_0xD	YCRCB_SwapUVY
01110b	CMF_0xE	RGB 10b
01111b	CMF_0xF	NV21/NV12
10000b	CMF_0x10	RGBA_16Float

Performance Counter Report Formats

Performance Counter Report Formats	
Size (in bits):	3

Value	Name
001b	
010b	
011b	
100b	
110b	
111b	

PredCtrl

PredCtrl	
Source:	EuIsa
Size (in bits):	4

Value	Name	Exists If
0000b	No Predication (normal) **[Default]**	
0001b	Sequential Flag Channel Mapping	
0010b	Replication swizzle .x	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16')
0010b	.anyv (any from f0.0-f1.0 on the same channel)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
0011b	Replication swizzle .y	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16')
0011b	.allv (all of f0.0-f1.0 on the same channel)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
0100b	Replication swizzle .z	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16')
0100b	.any2h (any in group of 2 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
0101b	Replication swizzle .w	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16')
0101b	.all2h (all in group of 2 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
0110b	.any4h	
0111b	.all4h	
1000b-1111b	Reserved	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align16')
1000b	.any8h (any in group of 8 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1001b	.all8h (all in group of 8 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1010b	.any16h (any in group of 16 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1011b	.all16h (all in group of 16 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1100b	.any32h (any in group of 32 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1101b	.all32h (all in group of 32 channels)	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')
1110b-1111b	Reserved	(Structure[EU_INSTRUCTION_CONTROLS_A][AccessMode]=='Align1')

PREFERRED_SLM_SIZE

PREFERRED_SLM_SIZE - PREFERRED_SLM_SIZE	
Size (in bits):	0
Specifies the preferred SLM size per subslice for this kernel.	

Restriction
Selected PREFERRED_SLM_SIZE must specify a size >= the selected SLM_SIZE

Value	Name	Description
0x0	Max **[Default]**	Preferred SLM size is the largest SLM size supported in the subslice.

RegDataType

RegDataType

Source:	EuIsa
Size (in bits):	4

Destination Type Numeric data type of the destination operand dst. The bits of the destination operand are interpreted as the identified numeric data type, rather than coerced into a type implied by the operator. For a send or sendc instruction, this field applies to CurrDst, the current destination operand. Three source instructions use a 3-bit encoding that allows fewer data types.

Value	Name	Description
0000b	:ub	Unsigned Byte (8-bit) Integer
0001b	:uw	Unsigned Word (16-bit) Integer
0010b	:ud	Unsigned DoubleWord (32-bit) Integer
0011b	:uq	Unsigned Quadword (64-bit) Integer
0100b	:b	Signed Byte (8-bit) Integer
0101b	:w	Signed Word (16-bit) Integer
0110b	:d	Signed DoubleWord (32-bit) Integer
0111b	:q	Signed QuadWord (64-bit) Integer
1000b	Reserved	
1001b	:hf	Half (16-bit) Float
1010b	:f	Single-Precision (32-bit) Float
1011b	:df	Double-Precision (64-bit) Float
1100b	Reserved	
1101b	Reserved	
1110b	Reserved	
1111b	Reserved	

Registers Per Thread

REGISTERS_PER_THREAD_SIZE - Registers Per Thread		
Size (in bits):	0	
Specifies the minimum number of registers allocated for each thread dispatch.		

Value	Name	Description
0	Default **[Default]**	For all threads except Compute thread, use 128 registers. For Compute Threads, use either 128 or 256 registers, based on STATE_COMPUTE_MODE.

RENDER_BARRIER_STAGE

RENDER_BARRIER_STAGE	
Size (in bits):	7
Stages a render barrier can either signal or wait.	

Value	Name
0x1	TOP
0x2	Color
0x4	Gpgpu
0x6	GPGPU and Color
0x10	Geom
0x20	Z
0x40	PS

RENDER_BARRIER_TYPE

RENDER_BARRIER_TYPE	
Size (in bits):	2

Value	Name
0x1	Signal
0x2	Wait
0x3	Immediate

RenderCompressionFormat

CMP_FMT_RENDER_TGL - RenderCompressionFormat_TGL

Source:	BSpec
Size (in bits):	5

Compression Format

Value	Name	Description
00001b		Format: Reserved; Casting: N/A; Encoding: 0x1; IP: N/A
00010b		Format: Reserved; Casting: N/A; Encoding: 0x2; IP: N/A
01010b		(see table below)
10000b		(see table below)
10001b		(see table below)

00001b:

Format	Casting	Encoding	IP
Reserved	N/A	0x1	N/A

00010b:

Format	Casting	Encoding	IP
Reserved	N/A	0x2	N/A

01010b:

Format	Casting	Encoding	IP
B8G8R8A8_UNORM	8b	0xA	3D
B8G8R8A8_UNORM_SRGB	8b	0xA	3D
R8G8B8A8_UNORM	8b	0xA	3D
R8G8B8A8_UNORM_SRGB	8b	0xA	3D
B5G6R5_UNORM	8b	0xA	3D
B5G6R5_UNORM_SRGB	8b	0xA	3D
B5G5R5A1_UNORM	8b	0xA	3D
B5G5R5A1_UNORM_SRGB	8b	0xA	3D
B4G4R4A4_UNORM	8b	0xA	3D
B4G4R4A4_UNORM_SRGB	8b	0xA	3D
R8G8_UNORM	8b	0xA	3D
B5G5R5X1_UNORM	8b	0xA	3D
B5G5R5X1_UNORM_SRGB	8b	0xA	3D
A1B5G5R5_UNORM	8b	0xA	3D
A4B4G4R4_UNORM	8b	0xA	3D
R8_UNORM	8b	0xA	3D
A8_UNORM	8b	0xA	3D

10000b:

Format	Casting	Encoding	IP
R16G16B16A16_FLOAT	16b	0x10	3D
R16G16B16X16_FLOAT	16b	0x10	3D
R16G16_FLOAT	16b	0x10	3D
R16_FLOAT	16b	0x10	3D

10001b:

Format	Casting	Encoding	IP

CMP_FMT_RENDER_TGL - RenderCompressionFormat_TGL

		R32G32B32A32_FLOAT	8b	0x11	3D
		R32G32B32X32_FLOAT	8b	0x11	3D
		R32G32_FLOAT	8b	0x11	3D
		R32_FLOAT	8b	0x11	3D
10010b		Format	Casting	Encoding	IP
		R32G32B32A32_SINT	8b	0x12	3D
		R32G32_SINT	8b	0x12	3D
		R32_SINT	8b	0x12	3D
10011b		Format	Casting	Encoding	IP
		R32G32B32A32_UINT	8b	0x13	3D
		R32G32_UINT	8b	0x13	3D
		R32_UINT	8b	0x13	3D
		D24_UNORM_X8	8b	0x13	3D
10100b		Format	Casting	Encoding	IP
		R16G16B16A16_UNORM	16b	0x14	3D
		R16G16_UNORM	16b	0x14	3D
		R16_UNORM	16b	0x14	3D
10101b		Format	Casting	Encoding	IP
		R16G16B16A16_SNORM	16b	0x15	3D
		R16G16_SNORM	16b	0x15	3D
		R16_SNORM	16b	0x15	3D
10110b		Format	Casting	Encoding	IP
		R16G16B16A16_SINT	16b	0x16	3D
		R16G16_SINT	16b	0x16	3D
		R16_SINT	16b	0x16	3D
10111b		Format	Casting	Encoding	IP
		R16G16B16A16_UINT	16b	0x17	3D
		R16G16_UINT	16b	0x17	3D
		R16_UINT	16b	0x17	3D
11000b		Format	Casting	Encoding	IP
		R10G10B10A2_UNORM	8b*	0x18	3D
		R10G10B10A2_UNORM_SRGB	8b*	0x18	3D
		B10G10R10A2_UNORM	8b*	0x18	3D
		B10G10R10A2_UNORM_SRGB	8b*	0x18	3D

CMP_FMT_RENDER_TGL - RenderCompressionFormat_TGL

11001b		Format	Casting	Encoding	IP
		R10G10B10_FLOAT_A2_UNORM	8b*	0x19	3D
11010b		Format	Casting	Encoding	IP
		R10G10B10A2_UINT	8b*	0x1A	3D
11011b		Format	Casting	Encoding	IP
		R8G8B8A8_SNORM	8b	0x1B	3D
		R8G8_SNORM	8b	0x1B	3D
		R8_SNORM	8b	0x1B	3D
11100b		Format	Casting	Encoding	IP
		R8G8B8A8_SINT	8b	0x1C	3D
		R8G8_SINT	8b	0x1C	3D
		R8_SINT	8b	0x1C	3D
11101b		Format	Casting	Encoding	IP
		R8G8B8A8_UINT	8b	0x1D	3D
		R8G8_UINT	8b	0x1D	3D
		R8_UINT	8b	0x1D	3D
11110b		Format	Casting	Encoding	IP
		R11G11B10_FLOAT	8b*	0x1E	3D

Saturate

Saturate	
Size (in bits):	1

This field controls the destination saturation. When it is set, output data to the destination register are saturated. The saturation operation depends on the destination data type. Saturation is the operation that converts any data that is outside the saturation target range for the data type to the closest representable value with the target range. If destination type is float, saturation target range is [0, 1]. For example, any positive number greater than 1 (including +INF) is saturated to 1 and any negative number (including -INF) is saturated to 0. A NaN is saturated to 0, For integer data types, the maximum range for the given numerical data type is the saturation target range. When it is not set, output data to the destination register are not saturated. For example, a wrapped result (modular) is output to the destination for an overflowed integer data. More details can be found in the Data Types chapter.

Value	Name
0	No Destination modification **[Default]**
1	Saturate Destination

SFID

SFID	
Source:	EuIsa
Size (in bits):	4

The following table lists the assignments (encodings) of the Shared Function and Fixed Function IDs used within the GPE. A Shared Function is a valid target of a message initiated via a 'send' instruction. A Fixed Function is an identifiable unit of the 3D or Media pipeline. Note that the Thread Spawner is both a Shared Function and Fixed Function. Note: The initial intention was to combine these two ID namespaces, so that (theoretically) an agent (such as the Thread Spawner) that served both as a Shared Function and Fixed Function would have a single, unique 4-bit ID encoding. However, this combination is not a requirement of the architecture.

Programming Notes
SFID_DP_DC1 is an extension of SFID_DP_DC0 to allow for more message types. They act as a single logical entity.
SFID_DP_DC1 and SFID_DP_DC2 are extensions of SFID_DP_DC0 to allow for more message types. They act as a single logical entity.

Value	Name	Description	
0000b	SFID_NULL	Null	Syntax: SEND_BINARY
0001b	Reserved	Reserved	
0010b	SFID_SAMPLER	Sampler	Syntax: SEND_BINARY
0011b	SFID_GATEWAY	Message Gateway	Syntax: SEND_BINARY
0100b	SFID_DP_DC2	Data Cache Data Port 2	Syntax: SEND_BINARY
0101b	SFID_DP_RC	Render Cache Data Port	Syntax: SEND_BINARY
0110b	SFID_URB	URB	Syntax: SEND_BINARY
0111b	SFID_SPAWNER	Thread Spawner	Syntax: SEND_BINARY
1000b	SFID_VME	Video Motion Estimation	Syntax: SEND_BINARY
1001b	SFID_DP_DCRO	Data Cache Read Only Data Port	Syntax: SEND_BINARY
1010b	SFID_DP_DC0	Data Cache Data Port	Syntax: SEND_BINARY
1011b	SFID_PI	Pixel Interpolator	Syntax: SEND_BINARY
1100b	SFID_DP_DC1	Data Cache Data Port 1	Syntax: SEND_BINARY
1101b	SFID_CRE	Check and Refinement Engine	Syntax: SEND_BINARY
1110b-1111b	Reserved		

Shader Channel Select

Shader Channel Select	
Size (in bits):	3

Value	Name	Description
0	ZERO	
1	ONE	
2	Reserved	
3	Reserved	
4	RED	Shader channel is set to surface red channel
5	GREEN	Shader channel is set to surface green channel
6	BLUE	Shader channel is set to surface blue channel
7	ALPHA	Shader channel is set to surface alpha channel

SIMD Mode

SIMD Mode	
Size (in bits):	3

Value	Name
0	SIMD8 + Integer Return
1	SIMD8
2	SIMD16
3	Reserved2
4	SIMD16 + Integer Return
5	SIMD8H
6	SIMD16H
7	Reserved7

Slice Hash Control

Slice Hash Control	
Source:	RenderCS
Size (in bits):	2

This is a sample showing the basic structure of an explicit enumeration.

Value	Name	Description
00b	Computed	Use Computed pixelhash_id
01b	Unbalanced table[0]	Use Computed pixelhash_id when balanced, Table[0] when unbalanced
10b	Table[0]	Use Table[0]
11b	Table[1]	Use Table[1]

SrcMod

SrcMod	
Source:	EuIsa
Size (in bits):	2

Source Modifier This field specifies the numeric modification of a source operand. The value of each data element of a source operand can optionally have its absolute value taken and/or its sign inverted prior to delivery to the execution pipe. The absolute value is prior to negate such that a guaranteed negative value can be produced. This field only applies to source operand. It does not apply to destination. This field is not present for an immediate source operand.

When used with logic instructions (and, not, or, xor), this field indicates whether the source bits are inverted (bitwise NOT) before delivery to the execution pipe, regardless of the source type.

Value	Name	Description
00b	No modification	
01b	abs	Absolute value Logic instructions: No modification (This encoding cannot be selected in the assembler syntax)
10b	negate	Negate Logic instructions: Bitwise NOT, inverting the source bits
11b	negate of abs	Negate of the absolute (forced negative value) Logic instructions: No modification (This encoding cannot be selected in the assembler syntax)

SURFACE_FORMAT

SURFACE_FORMAT

Size (in bits): 9

The following table indicates the supported surface formats and the 9-bit encoding for each. Note that some of these formats are used not only by the Sampling Engine, but also by the Data Port and the Vertex Fetch unit.

Value	Name	Description
000h	R32G32B32A32_FLOAT	
001h	R32G32B32A32_SINT	
002h	R32G32B32A32_UINT	
003h	R32G32B32A32_UNORM	
004h	R32G32B32A32_SNORM	
005h	R64G64_FLOAT	
006h	R32G32B32X32_FLOAT	
007h	R32G32B32A32_SSCALED	
008h	R32G32B32A32_USCALED	
020h	R32G32B32A32_SFIXED	
021h	R64G64_PASSTHRU	
040h	R32G32B32_FLOAT	
041h	R32G32B32_SINT	
042h	R32G32B32_UINT	
043h	R32G32B32_UNORM	
044h	R32G32B32_SNORM	
045h	R32G32B32_SSCALED	
046h	R32G32B32_USCALED	
050h	R32G32B32_SFIXED	
080h	R16G16B16A16_UNORM	
081h	R16G16B16A16_SNORM	
082h	R16G16B16A16_SINT	
083h	R16G16B16A16_UINT	
084h	R16G16B16A16_FLOAT	
085h	R32G32_FLOAT	
086h	R32G32_SINT	
087h	R32G32_UINT	
088h	R32_FLOAT_X8X24_TYPELESS	
089h	X32_TYPELESS_G8X24_UINT	
08Ah	L32A32_FLOAT	

SURFACE_FORMAT		
08Bh	R32G32_UNORM	
08Ch	R32G32_SNORM	
08Dh	R64_FLOAT	
08Eh	R16G16B16X16_UNORM	
08Fh	R16G16B16X16_FLOAT	
090h	A32X32_FLOAT	
091h	L32X32_FLOAT	
092h	I32X32_FLOAT	
093h	R16G16B16A16_SSCALED	
094h	R16G16B16A16_USCALED	
095h	R32G32_SSCALED	
096h	R32G32_USCALED	
0A0h	R32G32_SFIXED	
0A1h	R64_PASSTHRU	
0C0h	B8G8R8A8_UNORM	
0C1h	B8G8R8A8_UNORM_SRGB	
0C2h	R10G10B10A2_UNORM	
0C3h	R10G10B10A2_UNORM_SRGB	
0C4h	R10G10B10A2_UINT	
0C5h	R10G10B10_SNORM_A2_UNORM	
0C7h	R8G8B8A8_UNORM	
0C8h	R8G8B8A8_UNORM_SRGB	
0C9h	R8G8B8A8_SNORM	
0CAh	R8G8B8A8_SINT	
0CBh	R8G8B8A8_UINT	
0CCh	R16G16_UNORM	
0CDh	R16G16_SNORM	
0CEh	R16G16_SINT	
0CFh	R16G16_UINT	
0D0h	R16G16_FLOAT	
0D1h	B10G10R10A2_UNORM	
0D2h	B10G10R10A2_UNORM_SRGB	
0D3h	R11G11B10_FLOAT	
0D5h	R10G10B10_FLOAT_A2_UNORM	
0D6h	R32_SINT	
0D7h	R32_UINT	

SURFACE_FORMAT

0D8h	R32_FLOAT	
0D9h	R24_UNORM_X8_TYPELESS	
0DAh	X24_TYPELESS_G8_UINT	
0DDh	L32_UNORM	
0DEh	A32_UNORM	
0DFh	L16A16_UNORM	
0E0h	I24X8_UNORM	
0E1h	L24X8_UNORM	
0E2h	A24X8_UNORM	
0E3h	I32_FLOAT	
0E4h	L32_FLOAT	
0E5h	A32_FLOAT	
0E6h	X8B8_UNORM_G8R8_SNORM	
0E7h	A8X8_UNORM_G8R8_SNORM	
0E8h	B8X8_UNORM_G8R8_SNORM	
0E9h	B8G8R8X8_UNORM	
0EAh	B8G8R8X8_UNORM_SRGB	
0EBh	R8G8B8X8_UNORM	
0ECh	R8G8B8X8_UNORM_SRGB	
0EDh	R9G9B9E5_SHAREDEXP	
0EEh	B10G10R10X2_UNORM	
0F0h	L16A16_FLOAT	
0F1h	R32_UNORM	
0F2h	R32_SNORM	
0F3h	R10G10B10X2_USCALED	
0F4h	R8G8B8A8_SSCALED	
0F5h	R8G8B8A8_USCALED	
0F6h	R16G16_SSCALED	
0F7h	R16G16_USCALED	
0F8h	R32_SSCALED	
0F9h	R32_USCALED	
100h	B5G6R5_UNORM	
101h	B5G6R5_UNORM_SRGB	
102h	B5G5R5A1_UNORM	
103h	B5G5R5A1_UNORM_SRGB	
104h	B4G4R4A4_UNORM	

SURFACE_FORMAT

105h	B4G4R4A4_UNORM_SRGB	
106h	R8G8_UNORM	
107h	R8G8_SNORM	
108h	R8G8_SINT	
109h	R8G8_UINT	
10Ah	R16_UNORM	
10Bh	R16_SNORM	
10Ch	R16_SINT	
10Dh	R16_UINT	
10Eh	R16_FLOAT	
10Fh	A8P8_UNORM_PALETTE0	
110h	A8P8_UNORM_PALETTE1	
111h	I16_UNORM	
112h	L16_UNORM	
113h	A16_UNORM	
114h	L8A8_UNORM	
115h	I16_FLOAT	
116h	L16_FLOAT	
117h	A16_FLOAT	
118h	L8A8_UNORM_SRGB	
119h	R5G5_SNORM_B6_UNORM	
11Ah	B5G5R5X1_UNORM	
11Bh	B5G5R5X1_UNORM_SRGB	
11Ch	R8G8_SSCALED	
11Dh	R8G8_USCALED	
11Eh	R16_SSCALED	
11Fh	R16_USCALED	
122h	P8A8_UNORM_PALETTE0	
123h	P8A8_UNORM_PALETTE1	
124h	A1B5G5R5_UNORM	
125h	A4B4G4R4_UNORM	
126h	L8A8_UINT	
127h	L8A8_SINT	
140h	R8_UNORM	
141h	R8_SNORM	
142h	R8_SINT	

SURFACE_FORMAT

143h	R8_UINT	
144h	A8_UNORM	
145h	I8_UNORM	
146h	L8_UNORM	
147h	P4A4_UNORM_PALETTE0	
148h	A4P4_UNORM_PALETTE0	
149h	R8_SSCALED	
14Ah	R8_USCALED	
14Bh	P8_UNORM_PALETTE0	
14Ch	L8_UNORM_SRGB	
14Dh	P8_UNORM_PALETTE1	
14Eh	P4A4_UNORM_PALETTE1	
14Fh	A4P4_UNORM_PALETTE1	
150h	Y8_UNORM	
152h	L8_UINT	
153h	L8_SINT	
154h	I8_UINT	
155h	I8_SINT	
180h	DXT1_RGB_SRGB	
181h	R1_UNORM	SET0_LEGACY: Undefined behavior if used in any feature added. See Legacy sampler feature page for details
182h	YCRCB_NORMAL	
183h	YCRCB_SWAPUVY	
184h	P2_UNORM_PALETTE0	
185h	P2_UNORM_PALETTE1	
186h	BC1_UNORM	(DXT1)
187h	BC2_UNORM	(DXT2/3)
188h	BC3_UNORM	(DXT4/5)
189h	BC4_UNORM	
18Ah	BC5_UNORM	
18Bh	BC1_UNORM_SRGB	(DXT1_SRGB)
18Ch	BC2_UNORM_SRGB	(DXT2/3_SRGB)
18Dh	BC3_UNORM_SRGB	(DXT4/5_SRGB)
18Eh	MONO8	SET0_LEGACY: Undefined behavior if used in any feature added. See Legacy sampler feature page for details
18Fh	YCRCB_SWAPUV	
190h	YCRCB_SWAPY	

SURFACE_FORMAT

191h	DXT1_RGB	
192h	RESERVED_192	This value is reserved for internal use.
193h	R8G8B8_UNORM	
194h	R8G8B8_SNORM	
195h	R8G8B8_SSCALED	
196h	R8G8B8_USCALED	
197h	R64G64B64A64_FLOAT	
198h	R64G64B64_FLOAT	
199h	BC4_SNORM	
19Ah	BC5_SNORM	
19Bh	R16G16B16_FLOAT	
19Ch	R16G16B16_UNORM	
19Dh	R16G16B16_SNORM	
19Eh	R16G16B16_SSCALED	
19Fh	R16G16B16_USCALED	
1A1h	BC6H_SF16	
1A2h	BC7_UNORM	
1A3h	BC7_UNORM_SRGB	
1A4h	BC6H_UF16	
1A5h	PLANAR_420_8	
1A6h	PLANAR_420_16	
1A8h	R8G8B8_UNORM_SRGB	
1A9h	ETC1_RGB8	
1AAh	ETC2_RGB8	
1ABh	EAC_R11	
1ACh	EAC_RG11	
1ADh	EAC_SIGNED_R11	
1AEh	EAC_SIGNED_RG11	
1AFh	ETC2_SRGB8	
1B0h	R16G16B16_UINT	
1B1h	R16G16B16_SINT	
1B2h	R32_SFIXED	
1B3h	R10G10B10A2_SNORM	
1B4h	R10G10B10A2_USCALED	
1B5h	R10G10B10A2_SSCALED	
1B6h	R10G10B10A2_SINT	

SURFACE_FORMAT

1B7h	B10G10R10A2_SNORM	
1B8h	B10G10R10A2_USCALED	
1B9h	B10G10R10A2_SSCALED	
1BAh	B10G10R10A2_UINT	
1BBh	B10G10R10A2_SINT	
1BCh	R64G64B64A64_PASSTHRU	
1BDh	R64G64B64_PASSTHRU	
1C0h	ETC2_RGB8_PTA	
1C1h	ETC2_SRGB8_PTA	
1C2h	ETC2_EAC_RGBA8	
1C3h	ETC2_EAC_SRGB8_A8	
1C8h	R8G8B8_UINT	
1C9h	R8G8B8_SINT	
1FFh	RAW	

SyncFC

SyncFC	
Source:	EuIsa
Size (in bits):	4

Subfunctions that the sync instruction supports.

Value	Name	Description	
0000b	No Operation	Performs no operation. Regular SWSB constraints are checked.	Syntax: SYNC_UNARY
0010b	SBID Read Wait	Blocks until pending out-of-order source accesses are complete. If a mask is provided as immediate value in src0 then specific SBID resources can be checked, else all SBID resources are checked for source access complete status.	Syntax: SYNC_UNARY
0011b	SBID Write Wait	Blocks until pending out-of-order writebacks are complete. If a mask is provided as immediate value in src0 then specific SBID resources can be checked, else all SBID resources are checked for writeback complete status.	Syntax: SYNC_UNARY
1110b	Wait on Barrier	[] Blocks until the notification count reaches 0. 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. This operation implicitly accesses n0 (typically n0.0 for barriers).	Syntax: SYNC_UNARY
1111b	Wait on Host Notification	Similar to .bar, but waits the host's notification register (typically n0.1). See that element for more information.	Syntax: SYNC_UNARY

TernaryDataType

TernaryDataType	
Source:	EuIsa
Size (in bits):	3

This field provide the datatype of the source and destination operands for ternary instruction.

Value	Name	Description	Exists If
000b	:ub	Unsigned Byte (8-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
001b	:uw	Unsigned Word (16-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
010b	:ud	Unsigned DoubleWord (32-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
011b	:uq	Unsigned Quadword (64-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
100b	:b	Signed Byte (8-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
101b	:w	Signed Word (16-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
110b	:d	Signed DoubleWord (32-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
111b	:q	Signed QuadWord (64-bit) Integer	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Integer)
000b	Reserved		(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
001b	:hf	Half (16-bit) Float	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
010b	:f	Single-Precision (32-bit) Float	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
011b	:df	Double-Precision (64-bit) Float	(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
100b	Reserved		(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
101b	Reserved		(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
110b	Reserved		(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)
111b	Reserved		(Structure[EU_INSTRUCTION_BASIC_THREE_SRC][ExecDataType]==Float)

TernaryVertStride

TernaryVertStride	
Source:	EuIsa
Size (in bits):	2

Source Vertical Stride is required for regioning/accessing datatypes of varied size. It is one way to obtain a vector of scalars.

Value	Name
00b	0 - elements
01b	1 - elements
10b	4 - elements
11b	8 - elements

Texture Coordinate Mode

Texture Coordinate Mode		
Size (in bits): 3		

Value	Name	Description
0h	WRAP	Map is repeated in the U direction
1h	MIRROR	Map is mirrored in the U direction
2h	CLAMP	Map is clamped to the edges of the accessed map
3h	CUBE	For cube-mapping, filtering in edges access adjacent map faces
4h	CLAMP_BORDER	Map is infinitely extended with the border color
5h	MIRROR_ONCE	Map is mirrored once about origin, then clamped
6h	HALF_BORDER	Map is infinitely extended with the average of the nearest edge texel and the border color
7h	MIRROR_101	Map is mirrored one time in each direction, but the first pixel of the reflected image is skipped, and the reflected image is effectively 1 pixel less in that direction. May only be used on 2D surfaces.

VertStride

VertStride	
Source:	EuIsa
Size (in bits):	4

Description
Vertical Stride. The field provides the vertical stride of the register region in unit of data elements for an operand. Encoding of this field provides values of 0 or powers of 2, ranging from 1 to 32 elements. Larger values are not supported due to the restriction that a source operand must reside within two adjacent 256-bit registers (64 bytes total). Special encoding 1111b (0xF) is only valid when the operand is in register-indirect addressing mode (AddrMode = 1). If this field is set to 0xF, one or more sub-registers of the address registers may be used to compute the addresses. Each address sub-register provides the origin for a row of data element. The number of address sub-registers used is determined by the division of ExecSize of the instruction by the Width fields of the operand. This field only applies to source operand. It does not apply to destination. This field is not present for an immediate source operand.

Programming Notes
Note 1: If indirect address is supported for src1, encoding 0xF is reserved for src1 - only single-index indirect addressing is supported.
Note 2: Encoding 0010 applies for QWord-size operands.

Value	Name	Programming Notes
0000b	0 elements	
0001b	1 element	Align1 mode only.
0010b	2 elements	
0011b	4 elements	
0100b	8 elements	Align1 mode only.
0101b	16 elements	Applies to byte or word operand only. Align1 mode only.
0110b	32 elements	Applies to byte operand only. Align1 mode only.
0111b-1110b	Reserved	
1111b	VxH or Vx1 mode	Only valid for register-indirect addressing in Align1 mode.

Width

Width	
Source:	EuIsa
Size (in bits):	3

This field specifies the number of elements in the horizontal dimension of the region for a source operand. This field cannot exceed the ExecSize field of the instruction. This field only applies to source operand. It does not apply to destination. This field is not present for an immediate source operand.

Programming Notes
Note that with ExecSize of 32, because the maximum Width is 16, there are at least two rows in a source region.

Value	Name
000b	1 elements
001b	2 elements
010b	4 elements
011b	8 elements
100b	16 elements
101b-111b	Reserved

WRAP_SHORTEST_ENABLE

WRAP_SHORTEST_ENABLE	
Source:	RenderCS
Size (in bits):	4

This state selects which components (if any) of Attribute [n] are to be interpolated in a "wrap shortest" fashion. Operation is UNDEFINED if any of these bits are set and the Constant Interpolation Enable bit associated with this attribute is set. Note that wrap-shortest interpolation is only supported for Attributes 0-15.

Value	Name	Description
0001b	X	Wrap Shortest X Component
0010b	Y	Wrap Shortest Y Component
0011b	XY	Wrap Shortest XY Components
0100b	Z	Wrap Shortest Z Component
0101b	XZ	Wrap Shortest XZ Components
0110b	YZ	Wrap Shortest YZ Components
0111b	XYZ	Wrap Shortest XYZ Components
1000b	W	Wrap Shortest W Component
1001b	XW	Wrap Shortest XW Components
1010b	YW	Wrap Shortest YW Components
1011b	XYW	Wrap Shortest XYW Components
1100b	ZW	Wrap Shortest ZW Components
1101b	XZW	Wrap Shortest XZW Components
1110b	YZW	Wrap Shortest YZW Components
1111b	XYZW	Wrap Shortest XYZW Components