Radeon Cayman 3D Register Reference Guide

1. Vertex Grouper and Tessellator Registers

VGT:VGT_CACHE_INVALIDATION · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x88c4			
DESCRIPTION: *VGT cache invalidation*			
Field Name	Bits	Default	Description
CACHE_INVALIDATION	1:0	none	Indicates whether VC or TC is used for cache invalidation POSSIBLE VALUES: 00 - VC_ONLY: VC_ONLY 01 - TC_ONLY: TC_ONLY 02 - VC_AND_TC: VC_AND_TC
VS_NO_EXTRA_BUFFER	5	0x0	if set to one then disable gs_on bit
AUTO_INVLD_EN	7:6	0x0	0 = no auto invalidates, 1 = ES auto invalidates, 2 = GS auto invalidates, 3 = ES and GS auto invalidate

VGT:VGT_CNTL_STATUS · [R] · 32 bits · Access: 32 · GpuF0MMReg:0x88f0			
DESCRIPTION: *Status Bits*			
Field Name	Bits	Default	Description
VGT_OUT_INDX_BUSY	0	none	If set, the Output Index block within the VGT is busy
VGT_OUT_BUSY	1	none	If set, the Output block within the VGT is busy
VGT_PT_BUSY	2	none	If set, the Pass-thru block within the VGT is busy
VGT_TE_BUSY	3	none	If set, the Tessellation Engine block within the VGT is busy
VGT_VR_BUSY	4	none	If set, the Vertex Reuse Block within the VGT is busy
VGT_PI_BUSY	5	none	
VGT_INVLD_BUSY	6	none	
VGT_GS_BUSY	7	none	If set, VGT GS is actively processing
VGT_HS_BUSY	8	none	If set, VGT HS block (this could be compute shader, local shader, or hull shader) within the VGT is busy
VGT_TE11_BUSY	9	none	If set, VGT TE11 (DX11 tessellation) block is busy
VGT_BUSY	10	none	If set, VGT is busy

VGT:VGT_DMA_BASE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x287e8			
DESCRIPTION: *VGT DMA Base Address*			
Field Name	Bits	Default	Description
BASE_ADDR	31:0	none	VGT DMA Base Address This address must be naturally aligned to a 16-bit word. Therefore, bit 0 of this register must be 0

VGT:VGT_DMA_BASE_HI · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x287e4			
DESCRIPTION: VGT DMA Base Address : upper 8-bits of 40 bit address			
Field Name	Bits	Default	Description
BASE_ADDR	7:0	none	This specfies upper 8-bits of 40-bits of DMA address

VGT:VGT_DMA_INDEX_TYPE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a7c			
DESCRIPTION: VGT DMA Index Type and Mode			
Field Name	Bits	Default	Description
INDEX_TYPE	1:0	none	VGT DMA Index Type POSSIBLE VALUES: 00 - VGT_INDEX_16: VGT_INDEX_16 16-bit index 01 - VGT_INDEX_32: VGT_INDEX_32 32-bit index
SWAP_MODE	3:2	none	DMA Swap mode POSSIBLE VALUES: 00 - VGT_DMA_SWAP_NONE: VGT_DMA_SWAP_NONE No swap 01 - VGT_DMA_SWAP_16_BIT: VGT_DMA_SWAP_16_BIT 16-bit swap 0xAABBCCDD -> 0xBBAADDCC 02 - VGT_DMA_SWAP_32_BIT: VGT_DMA_SWAP_32_BIT 32-bit swap 0xAABBCCDD -> 0xDDCCBBAA 03 - VGT_DMA_SWAP_WORD: VGT_DMA_SWAP_WORD word swap 0xAABBCCDD -> 0xCCDDAABB

VGT:VGT_DMA_MAX_SIZE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a78			
DESCRIPTION: *This is a write-only register. This register is used for handling index out of bound issue. It is expected that driver set this register to less than or equal to VGT_DMA_SIZE, specifying how many actual good data to be read from index buffer. If VGT_MAX_SIZE < VGT_DMA_SIZE, the reset of fetched indices are set to zero in VGT*			
Field Name	Bits	Default	Description
MAX_SIZE	31:0	none	VGT DMA maximum number of indices until out of bound index buffer is accessed

VGT:VGT_DMA_NUM_INSTANCES · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a88			
DESCRIPTION: VGT DMA Number of Instances			
Field Name	Bits	Default	Description
NUM_INSTANCES	31:0	none	VGT DMA Number of Instances, minimum value is 1

VGT:VGT_DMA_SIZE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a74			
DESCRIPTION: *VGT DMA Size*			
Field Name	Bits	Default	Description
NUM_INDICES	31:0	none	VGT DMA Number of indices

VGT:VGT_DRAW_INITIATOR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x287f0			
DESCRIPTION: *Draw Initiator*			
Field Name	Bits	Default	Description
SOURCE_SELECT	1:0	none	Input Source Select. If the Source Select field is set to `Auto-increment Index` mode and the Primitive Type is set to `Tri List w/Flags`, then the draw initiator is processed as just a regular `Tri List`. POSSIBLE VALUES: 00 - DI_SRC_SEL_DMA: VGT DMA Data 01 - DI_SRC_SEL_IMMEDIATE: Immediate Data 02 - DI_SRC_SEL_AUTO_INDEX: Auto-increment Index 03 - DI_SRC_SEL_RESERVED: Reserved - unused
MAJOR_MODE	3:2	none	Major Mode POSSIBLE VALUES: 00 - DI_MAJOR_MODE_0: DI_MAJOR_MODE_0 Normal (Implicit) Mode -- applies only to prim types 0-21. Some VGT state registers are ignored (their values implied) in this mode. 01 - DI_MAJOR_MODE_1: DI_MAJOR_MODE_1 Explicit Mode -- Configuration completely specified by state registers.
NOT_EOP	5	none	This bit indicates that this draw initiator should not generate an end-of-packet signal because it will be followed by one or more chained draw initiators. Care must be taken so that this draw initiator is immediately followed, at the hardware interface, by a chained draw initiator. (In other words, chained draw initiators cannot be separated over driver buffer boundaries that can be interrupted. This bit is primarily intended to be set by the CP to improve the processing parallelism of small 2D blits.) POSSIBLE VALUES: 00 - normal eop 01 - suppress eop
USE_OPAQUE	6	none	This bit indicates that this draw call is a opaque draw call

			POSSIBLE VALUES: 00 - non-opaque draw 01 - opaque draw

VGT:VGT_ENHANCE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a50			
DESCRIPTION: *Used for Late Additions of Control Bits.*			
Field Name	Bits	Default	Description
MISC	31:0	none	Misc bit

VGT:VGT_ES_PER_GS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a58			
DESCRIPTION: *Maximum ES vertices per GS thread*			
Field Name	Bits	Default	Description
ES_PER_GS	10:0	none	Maximum number of ES vertices per GS thread

VGT:VGT_EVENT_ADDRESS_REG · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x287f8			
DESCRIPTION: *This register is written only if extended_event is enabled in event_initiator. It will be stored as FIFO*			
Field Name	Bits	Default	Description
ADDRESS_LOW	27:0	none	address bit 31:4 for zpass event

VGT:VGT_EVENT_INITIATOR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a90			
DESCRIPTION: *Ring-specific: Event Inititiator*			
Field Name	Bits	Default	Description
EVENT_TYPE	5:0	none	Event Type (also called Event ID) POSSIBLE VALUES: 00 - Reserved 01 - SAMPLE_STREAMOUTSTATS1: Sample Streamout1 Statitics counters -- Inserted by the driver to request the GPU to sample counters associated with streamout. The CP will subsequently write them to memory. 02 - SAMPLE_STREAMOUTSTATS2: Sample Streamout2 Statitics counters -- Inserted by the driver to request the GPU to sample counters associated with streamout. The CP will subsequently write them to memory. 03 - SAMPLE_STREAMOUTSTATS3: Sample Streamout3 Statitics counters -- Inserted by the driver to request the GPU to sample counters associated with streamout. The CP will subsequently write them to

			memory. 04 - CACHE_FLUSH_TS: Destination Cache Flush with Timestamp -- Inserted by the driver to cause the CBs, DBs, and SX to flush all prior rendering in any destination cache, wait for write confirm, then signal the CP. 05 - CONTEXT_DONE: GFXDEC Context Done -- Inserted by the CP on the first GFXDEC state update after a draw. The 8 contexts are now shared with compute shaders and therefore, the done applies to the most recent state used for GFX, as opposed to being used for CS (compute shaders). 06 - CACHE_FLUSH: Destination Caches Flushed -- Inserted by the driver to cause the CBs, DBs, and SX to flush all prior rendering in any destination cache to memory (No Timestamp is Generated). 07 - CS_PARTIAL_FLUSH: Used to flush Compute Shader work after the CP in the VGT, SPI, SQ such that all previous CS work launched prior to this event will complete execution in the shader core and free all shader resources. 08 - VGT_STREAMOUT_SYNC: Generated and consumed by the VGT for use when syncing shader engines. The driver should not insert this event. 09 - Reserved_0x09: Reserved -- was used for SC_WAIT_WC 10 - Reserved 11 - Reserved 12 - Reserved 13 - RST_PIX_CNT: Reset SPI`s auto Pixel Counter -- Inserted by the driver. 14 - Reserved 15 - VS_PARTIAL_FLUSH: Used to flush all work between the CP and the ES, GS, VS shaders including the VGT. 16 - PS_PARTIAL_FLUSH: Used to flush all work between the CP and the ES, GS, VS, PS shaders including scan conversion, primitive assembly, and VGT. 17 - FLUSH_HS_OUTPUT: Flush Hull Shader Output -- Sent by the VGT after an HS threadgroup. Used to make sure all HS threadgroup data is processed before the corresponding DS threadgroup begins. 18 - FLUSH_LS_OUTPUT: Flush Local Shader Output -- Sent by the VGT after an LS threadgroup. Used to make sure all LS threadgroup data is processed before the corresponding HS threadgroup begins. 19 - Reserved 20 - CACHE_FLUSH_AND_INV_TS_EVENT: Destination Cache Flush and Invalidate with Timestamp -- Inserted by the driver to cause the CBs, DBs, and SX to flush and invalidate all prior rendering in any destination cache, wait for write confirm, then signal the

			CP. 21 - ZPASS_DONE: Write ZPASS counts to memory -- Inserted by the driver to instruct the DBs to write out the ZPASS counters to memory. Used to support DX10 occlusion queries. 22 - CACHE_FLUSH_AND_INV_EVENT: Destination Cache Flush and Invalidate -- Inserted by the driver to cause the CBs, DBs, and SX to flush and invalidated all prior rendering in any destination cache to memory (No Timestamp is Generated). 23 - PERFCOUNTER_START: Start enabled event based Performance counters -- Inserted by the driver. 24 - PERFCOUNTER_STOP: Stop enabled event based Performance counters that are event-enabled -- Inserted by the driver. 25 - PIPELINESTAT_START: Start pipeline/strmout stat -- Inserted by the driver. 26 - PIPELINESTAT_STOP: Stop pipeline/strmout stat -- Inserted by the driver. 27 - PERFCOUNTER_SAMPLE: Sample the performance counters of all blocks -- Inserted by the driver to read the performance counters. 28 - FLUSH_ES_OUTPUT: Flush Export Shader Output -- Inserted by the VGT to instruct the SX to flush all the ES output to memory. 29 - FLUSH_GS_OUTPUT: Flush Geometry Shader Output -- Inserted by the VGT to instruct the SX to flush all the GS output to memory. 30 - SAMPLE_PIPELINESTAT: Sample Pipeline Statistics counters -- Inserted by the driver to request the GPU to sample counters associated with pipelinestats. The CP will subsequently write them to memory. 31 - SO_VGTSTREAMOUT_FLUSH: VGT Streamout Flush -- This event will cause VGT to update the read only offsets registers and then send a VGT_CP_strmout_flushed to instruct the CP to read the offsets. 32 - SAMPLE_STREAMOUTSTATS: Sample Streamout0 Statitics counters -- Inserted by the driver to request the GPU to sample counters associated with streamout. The CP will subsequently write them to memory. 33 - RESET_VTX_CNT: Reset Vertex Count -- Inserted by the driver to reset the auto index count for vertex count. There are tow counters one for gs and non-gs and these should be reset seperately 34 - BLOCK_CONTEXT_DONE: Block Managed State (SQCONSDEC) Context Done -- Inserted by the CP on the first SQCONSDEC constant update after a draw. 35 - CS_CONTEXT_DONE: GFXDEC Context Done for CS (compute shaders) -- Converted to CONTEXT_DONE event by the VGT before it sends it

			as an event down the pipe. Therefore, for R8xx, only the CP and VGT must be aware of this event. Inserted by the CP on the first GFXDEC state update for CS after a draw that is being used to run compute shaders. This applies to the same 8 context states as the CONTEXT_DONE event, except that it applies to the most recent context that is being used for running compute shaders 36 - VGT_FLUSH: VGT Flush - Inserted by the driver to cause the VGT to be flushed. Used when GS ring buffer sizes are changed 37 - Reserved 38 - SQ_NON_EVENT: SQ Non-Event -- This event is reserved for SQ 39 - SC_SEND_DB_VPZ: SC Send Depth Block VPort Z -- Inserted by the SC when it sends the vport array Zmin and Zmax values to the DBs. 40 - BOTTOM_OF_PIPE_TS: Bottom of the Pipe Timestamp -- Inserted by the driver to request a bottom of pipe timestamp be sent to memory, no flushing required. 41 - FLUSH_SX_TS: Flush SX Timestamp - Inserted by the driver to cause the SX to flush its caches, then signal the CP. The other destination caches must also signal the CP for this event. All responses to the CP must be in the order the TS were received, regardless if the cache is required to otherwise act upon it. 42 - DB_CACHE_FLUSH_AND_INV: DB Flush and Invalidate - Inserted by the driver when the depth surface is paged out of memory. 43 - FLUSH_AND_INV_DB_DATA_TS: Flush and Invalidate DB`s Data Cache Only - Inserted by the driver to cause the DB to flush and invalidate only its data cache, wait for write confirm, then signal the CP. The other destination caches must also signal the CP for this event. All responses to the CP must be in the order the TS were received, regardless if the cache is required to otherwise act upon it. 44 - FLUSH_AND_INV_DB_META: Flush and Invalidate DB`s Meta (htile) Only - Inserted by the driver to cause the DB to flush and invalidate only its Meta cache. 45 - FLUSH_AND_INV_CB_DATA_TS: Flush and Invalidate CB`s Data Cache Only - Inserted by the driver to cause the CB to flush and invalidate only its data cache, wait for write confirm, then signal the CP. The other destination caches must also signal the CP for this event. All responses to the CP must be in the order the TS were received, regardless if the cache is required to otherwise act upon it. 46 - FLUSH_AND_INV_CB_META: Flush and Invalidate CB`s Meta (cmask/fmask) Only - Inserted by the driver to cause the CB to flush and invalidate only its Meta cache.

			47 - CS_DONE: Inserted by the driver using an EVENT_WRITE_EOS packet. The SQ, in response, will generate a signal to indicate that all CS work prior to this point has completed. 48 - PS_DONE: Inserted by the driver using an EVENT_WRITE_EOS packet. The SQ, in response, will generate a signal to indicate that all PS work prior to this point has completed. 49 - FLUSH_AND_INV_CB_PIXEL_DATA: Flush and invalidate CB`s pixel (render target) data in color cache. Does not guarantee UAV(RAT) flush-and-inv, and does not flush the cmask/fmask cache either. Typically would be inserted by the driver before resolving or expanding an MSAA buffer. No wait-idle is necessary between this flush and the subsequent resolve/expand draw command. 50 - SX_CB_RAT_ACK_REQUEST: HW generated event by the SX to signify an ack request to the CB.
ADDRESS_HI	26:18	none	address bits 39:31 for zpass event
EXTENDED_EVENT	27	none	0 for single DW event, 1 for two DW event

VGT:VGT_GROUP_DECR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a2c

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register contains the amount by which the draw initiator index count is decremented for all groups taken from the input stream except for the first group.*

Field Name	Bits	Default	Description
DECR	3:0	none	Decrement amount for groups except the first

VGT:VGT_GROUP_FIRST_DECR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a28

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register contains the amount by which the draw initiator index count is decremented for the first group taken from the input stream.*

Field Name	Bits	Default	Description
FIRST_DECR	3:0	none	Decrement amount for the first group

VGT:VGT_GROUP_PRIM_TYPE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a24

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register contains the prim type output by the grouper stage of the VGT*
Please note the following restrictions in the use of this register
1.1. The PRIM_ORDER settings of VGT_GRP_FAN, VGT_GRP_LOOP, and VGT_GRP_POLYGON are not permitted if the VGT_OUTPUT_PATH_CNTL register is set to VGT_OUTPATH_PASSTHRU. Implementing these primitive orders correctly would require the VGT Passthru Block to have storage for the worst-case compound-index.
2.2. If the VGT_OUTPUT_PATH_CNTL register is set to VGT_OUTPATH_PASSTHRU, then the PRIM_TYPE setting of VGT_GRP_3D_QUAD (with a PRIM_ORDER of either VGT_GRP_LIST or VGT_GRP_STRIP) will not

necessarily have the correct order for flat shading for either Direct3D or OpenGL. (This restriction does NOT apply to quads that are processed through the Vertex Reuse Block.) *3.3. If the VGT_OUTPUT_PATH_CNTL register is set to VGT_OUTPATH_PASSTHRU and the PRIM_TYPE field of the VGT_GROUP_PRIM_TYPE register is set to VGT_GRP_3D_QUAD, then each quad primitive will be decomposed into two triangles regardless of the setting of the RETAIN_QUADS field in the VGT_GROUP_PRIM_TYPE register.*			
Field Name	Bits	Default	Description
PRIM_TYPE	4:0	none	Prim type output by grouper stage of the VGT. POSSIBLE VALUES: 00 - VGT_GRP_3D_POINT: VGT_GRP_3D_POINT 01 - VGT_GRP_3D_LINE: VGT_GRP_3D_LINE 02 - VGT_GRP_3D_TRI: VGT_GRP_3D_TRI 03 - VGT_GRP_3D_RECT: VGT_GRP_3D_RECT 04 - VGT_GRP_3D_QUAD: VGT_GRP_3D_QUAD 05 - VGT_GRP_2D_COPY_RECT_V0: VGT_GRP_2D_COPY_RECT_V0 06 - VGT_GRP_2D_COPY_RECT_V1: VGT_GRP_2D_COPY_RECT_V1 07 - VGT_GRP_2D_COPY_RECT_V2: VGT_GRP_2D_COPY_RECT_V2 08 - VGT_GRP_2D_COPY_RECT_V3: VGT_GRP_2D_COPY_RECT_V3 09 - VGT_GRP_2D_FILL_RECT: VGT_GRP_2D_FILL_RECT 10 - VGT_GRP_2D_LINE: VGT_GRP_2D_LINE 11 - VGT_GRP_2D_TRI: VGT_GRP_2D_TRI 12 - VGT_GRP_PRIM_INDEX_LINE: VGT_GRP_PRIM_INDEX_LINE 13 - VGT_GRP_PRIM_INDEX_TRI: VGT_GRP_PRIM_INDEX_TRI 14 - VGT_GRP_PRIM_INDEX_QUAD: VGT_GRP_PRIM_INDEX_QUAD 15 - VGT_GRP_3D_LINE_ADJ: VGT_GRP_3D_LINE_ADJ 16 - VGT_GRP_3D_TRI_ADJ: VGT_GRP_3D_TRI_ADJ 17 - VGT_GRP_3D_PATCH: VGT_GRP_3D_PATCH
RETAIN_ORDER	14	none	Resetting this bit to zero causes the Grouper within the VGT to convert strips, fans, loops, and polygons into regular lists in the vgt_grouper block. It also causes the primitive indices to be re-ordered to have the provoking vertex in the correct position. This bit should be set to zero if the VGT_OUTPUT_PATH_CNTL register specifies VGT_OUTPATH_VTX_REUSE or VGT_OUTPATH_TESS_EN and the VGT_DRAW_INITIATOR prim type is between 0 and 15, inclusive, (tri list, tri strip, tri fan, etc...). This bit is implied to be zero for VGT_DRAW_INITIATOR prim

			types 0 thru 15 if the Major Mode of the VGT_DRAW_INIITIATOR is 0. If this bit is set for prim types 0 thru 15, then the primitive index order from the grouper will be retained and the indices will be incorrect for loops, fans, and polygons. Note that if the VGT_DRAW_INITIATOR.MAJOR_MODE is set to MAJOR_MODE_1 and VGT_OUTPUT_PATH_CNTL is set to VGT_OUTPATH_PASSTHRU and the VGT_GROUP_PRIM_TYPE.PRIM_TYPE is set to VGT_GRP_3D_TRI or VGT_GRP_2D_TRI and VGT_GROUP_PRIM_TYPE.PRIM_ORDER is set to VGT_GRP_STRIP, then the passthru block will perform DX/OpenGL index re-ordering for tri-strips. POSSIBLE VALUES: 00 - Reorder strip/fan/loop/polygon into lists with correct provoking vertex 01 - Retain primitive index order as they appear in the input stream
RETAIN_QUADS	15	none	This bit can only be legally set if the VGT_OUTPUT_PATH_CNTL register specifies the Tessellation Engine and the Major Mode of the VGT_DRAW_INITATOR is 1. The RETAIN_QUADS bit indicates that quads should be passed intact to the tessellation engine. If this bit is not set, then the quads will be decomposed into triangles. POSSIBLE VALUES: 00 - Decompose quads into triangles 01 - Retain quads (legal only for tessellation engine)
PRIM_ORDER	18:16	none	Prim order output by grouper stage of the VGT. POSSIBLE VALUES: 00 - VGT_GRP_LIST: VGT_GRP_LIST 01 - VGT_GRP_STRIP: VGT_GRP_STRIP 02 - VGT_GRP_FAN: VGT_GRP_FAN 03 - VGT_GRP_LOOP: VGT_GRP_LOOP 04 - VGT_GRP_POLYGON: VGT_GRP_POLYGON

VGT:VGT_GROUP_VECT_0_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a30			
DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register indicates, with bits flags, which components are relevant for vector 0 of a group. At least one component of vector 0 must be indicated. This register also contains the stride of vector 0 (in 16-bit words) in the input stream and the amount to shift the input stream (in 16-bit words) after extracting the vector.*			
Field Name	Bits	Default	Description
COMP_X_EN	0	none	Indicates that component X will be output from the grouper for vector 0

			POSSIBLE VALUES: 00 - disable 01 - enable
COMP_Y_EN	1	none	Indicates that component Y will be output from the grouper for vector 0 POSSIBLE VALUES: 00 - disable 01 - enable
COMP_Z_EN	2	none	Indicates that component Z will be output from the grouper for vector 0 POSSIBLE VALUES: 00 - disable 01 - enable
COMP_W_EN	3	none	Indicates that component W will be output from the grouper for vector 0 POSSIBLE VALUES: 00 - disable 01 - enable
STRIDE	15:8	none	The stride of vector 0 data in the input stream (in 16-bit words). Zero is NOT a legal value for an active vector. See the programming guidelines for the situation in which a vector uses no data from the shifter.
SHIFT	23:16	none	The amount to shift the input stream after extracting vector 0 (in 16-bit words). This field must be less than or equal to the STRIDE field for proper shifter operation.

VGT:VGT_GROUP_VECT_0_FMT_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a38

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register indicates how the value each component of vector 0 will be determined. If the VGT_GROUP_VECT_0_CNTL register indicates that a particular component is not selected for output from the grouper, then that component`s format control fields are ignored.*

Field Name	Bits	Default	Description
X_CONV	3:0	none	X Component Determination. POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32

			bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
X_OFFSET	7:4	none	X Component Offset. This field is the offset, in 16-bit words, of the X component in the input cycle.
Y_CONV	11:8	none	Y Component Determination. See the X component determination field for description. POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
Y_OFFSET	15:12	none	Y Component Offset. This field is the offset, in 16-bit words, of the Y component in the input cycle.
Z_CONV	19:16	none	Z Component Determination. See the X component determination field for description. POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32:

			VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
Z_OFFSET	23:20	none	Z Component Offset. This field is the offset, in 16-bit words, of the Z component in the input cycle.
W_CONV	27:24	none	W Component Determination. See the X component determination field for description. POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
W_OFFSET	31:28	none	W Component Offset. This field is the offset, in 16-bit words, of the Z component in the input cycle.

VGT:VGT_GROUP_VECT_1_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a34

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register is identical to VGT_GROUP_VECT_0_CNTL except that it applies to vector 1 of the group instead of vector 0. Also, vector 0 is required to have at least one component set; however, vector 1 may have none set.*

Field Name	Bits	Default	Description
COMP_X_EN	0	none	POSSIBLE VALUES: 00 - disable 01 - enable
COMP_Y_EN	1	none	POSSIBLE VALUES: 00 - disable 01 - enable
COMP_Z_EN	2	none	POSSIBLE VALUES: 00 - disable 01 - enable
COMP_W_EN	3	none	POSSIBLE VALUES: 00 - disable 01 - enable
STRIDE	15:8	none	
SHIFT	23:16	none	

VGT:VGT_GROUP_VECT_1_FMT_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a3c

DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register is identical to VGT_GROUP_VECT_0_FMT_CNTL except that it controls the formatting of output vector 1 instead of output vector 0.*

Field Name	Bits	Default	Description
X_CONV	3:0	none	POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
X_OFFSET	7:4	none	

Y_CONV	11:8	none	POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
Y_OFFSET	15:12	none	
Z_CONV	19:16	none	POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp 01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
Z_OFFSET	23:20	none	
W_CONV	27:24	none	POSSIBLE VALUES: 00 - VGT_GRP_INDEX_16: VGT_GRP_INDEX_16 16 bits from stream with index offset and clamp

			01 - VGT_GRP_INDEX_32: VGT_GRP_INDEX_32 32 bits from stream with index offset and clamp 02 - VGT_GRP_UINT_16: VGT_GRP_UINT_16 16 bits from stream as unsigned int 03 - VGT_GRP_UINT_32: VGT_GRP_UINT_32 32 bits from stream as unsigned int 04 - VGT_GRP_SINT_16: VGT_GRP_SINT_16 16 bits from stream as signed int 05 - VGT_GRP_SINT_32: VGT_GRP_SINT_32 32 bits from stream as signed int 06 - VGT_GRP_FLOAT_32: VGT_GRP_FLOAT_32 32 bits from stream as float 07 - VGT_GRP_AUTO_PRIM: VGT_GRP_AUTO_PRIM 24 bits from auto primitive counter 08 - VGT_GRP_FIX_1_23_TO_FLOAT: VGT_GRP_FIX_1_23_TO_FLOAT 24 bit barycentric value from tessellation engine
W_OFFSET	31:28	none	

VGT:VGT_GS_INSTANCE_CNT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b90

DESCRIPTION: *Specifies the amount of GS prim instancing*

Field Name	Bits	Default	Description
ENABLE	0	none	Enable GS instancing POSSIBLE VALUES: 00 - gs_instance_disable 01 - gs_instance_enable
CNT	8:2	none	Number of GS prim instances, if set to 0 gs instancing is treated as off, no instance id provided

VGT:VGT_GS_MAX_VERT_OUT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b38

DESCRIPTION: *VGT max verts output by the GS for each prim*

Field Name	Bits	Default	Description
MAX_VERT_OUT	10:0	none	The number of max verts that can be emitted from a geometry shader instance, legal values are 1 - 1024

VGT:VGT_GS_MODE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a40

DESCRIPTION: *VGT GS Enable Mode*

Field Name	Bits	Default	Description
MODE	1:0	none	Lower two bits of MODE, This value combined with MODE_HI indicates which of GS scenerios are enabled

			POSSIBLE VALUES: 00 - GS_OFF: GS_OFF 01 - GS_SCENARIO_A: GS_SCENARIO_A 02 - GS_SCENARIO_B: GS_SCENARIO_B 03 - GS_SCENARIO_G: GS_SCENARIO_G 04 - GS_SCENARIO_C: GS_SCENARIO_C 05 - SPRITE_EN: SPRITE_EN
ES_PASSTHRU	2	none	sets to one if VS shader is passthru when GS scenario G is used POSSIBLE VALUES: 00 - passthru_dis 01 - passthru_en
CUT_MODE	4:3	none	00: more than 512 gs emit vertices, 01: more than 256 and less than equal to 512 emit vertices, 10:more than 128 and less than or equal to 256 gs emit vertices, 11: less than or equal to 128 gs emit vertices POSSIBLE VALUES: 00 - GS_CUT_1024: GS_CUT_1024 01 - GS_CUT_512: GS_CUT_512 02 - GS_CUT_256: GS_CUT_256 03 - GS_CUT_128: GS_CUT_128
MODE_HI	8	none	Upper bit of MODE field, see description of MODE

VGT:VGT_GS_OUT_PRIM_TYPE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a6c			
DESCRIPTION: *VGT GS output primitive type*			
Field Name	Bits	Default	Description
OUTPRIM_TYPE	5:0	none	GS output primitive type POSSIBLE VALUES: 00 - POINTLIST: POINTLIST 01 - LINESTRIP: LINESTRIP 02 - TRISTRIP: TRISTRIP

VGT:VGT_GS_PER_ES · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a54			
DESCRIPTION: *Maximum GS prims per ES thread*			
Field Name	Bits	Default	Description
GS_PER_ES	10:0	none	Maximum number of GS prims per ES thread

VGT:VGT_GS_PER_VS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a5c			
DESCRIPTION: *Maximum GS threads per VS thread*			
Field Name	Bits	Default	Description

GS_PER_VS	3:0	none	Maximum number of GS threads per VS thread

VGT:VGT_GS_VERTEX_REUSE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x88d4

DESCRIPTION: *reuseability for GS path, it has nothing to do with number of good simd*

Field Name	Bits	Default	Description
VERT_REUSE	4:0	none	Reusability number for GS input prims. Can be set to either 0, or from 4-16 in normal GS G mode of operation, but it must be at least 4 if the tessellation output is piped to the GS path

VGT:VGT_HOS_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a14

DESCRIPTION: *This register controls the behavior of the Tessellation Engine block at the backend of the VGT. This register is relevant only if the VGT_OUTPUT_PATH_CNTL register specifies the Tessellation Engine block for the VGT backend path. Note that the tessellation engine is enabled by selecting the tessellation engine path in the VGT_OUTPUT_PATH_CNTL register as opposed to the single enable bit that was used in previous architectures.*

Field Name	Bits	Default	Description
TESS_MODE	1:0	none	Tessellation Mode 0 : Discrete 1 : Continuous 2 : Adaptive

VGT:VGT_HOS_MAX_TESS_LEVEL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a18

DESCRIPTION: *When using the tessellator: For continuous and discrete tessellation modes, this register contains the tessellation level. For adaptive tessellation, this register contains the maximum tessellation level. The adaptive tessellation levels will be clamped less-than or equal to this level by the tessellation engine. In all cases, the format of this register is 32-bit IEEE floating point. This register is relevant only when the VGT_OUT_CNTL register specifies `Tessellation Engine` in the Path Select field.*

When using the Dx11 tessellator: This register specifies a Max tessellation level clamp that the hardware will apply to fetched Tess Factors.

Field Name	Bits	Default	Description
MAX_TESS	31:0	none	For adaptive tessellation mode, this is the maximum tessellation clamp value. For continuous and discrete tessellation modes, this is the tessellation level. For discrete modes, values in the range (1.0, 14.0) are legal. For non-discrete modes, values in the range (1.0, 15.0) are legal. MAX_TESS must be greater than or equal to MIN_TESS. When using the Dx11 tessellator values in the range (0.0, 64.0) are legal. If the incoming factor is a Nan, a negative number or Zero, it is not clamped against this

			value.

VGT:VGT_HOS_MIN_TESS_LEVEL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a1c			
DESCRIPTION: *When using the tessellator: For continuous and discrete tessellation modes, this register is not applicable. For adaptive tessellation, this register contains the minimum tessellation level. The adaptive tessellation levels will be clamped greater-than or equal to this level by the tessellation engine. The format of this register is 32-bit IEEE floating point. This register is relevant only when the VGT_OUT_CNTL register specifies `Tessellation Engine` in the Path Select field and the VGT_HOS_CNTL register specifies adaptive tessellation mode.* *When using the Dx11 tessellator: This register specifies a Min tessellation level clamp that the hardware will apply to fetched Tess Factors.*			
Field Name	Bits	Default	Description
MIN_TESS	31:0	none	For adpative tessellation mode, this is the minimum tessellation clamp value. For continuous and discrete tessellartion modes, this register is not applicable. For discrete modes values in the range (1.0, 14.0) are legal. For non-discrete modes, values in the range (1.0, 15.0) are legal. MIN_TESS must be less than or equal to MAX_TESS. When using the Dx11 tessellator values in the range (0.0, 64.0) are legal. If the incoming factor is a Nan, a negative number or Zero, it is not clamped against this value.

VGT:VGT_HOS_REUSE_DEPTH · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a20			
DESCRIPTION: *This register tells the tessellation how many of most recently submitted vertices it can reuse. This register is relevant only when the VGT_OUT_CNTL register specifies `Tessellation Engine` in the Path Select field.*			
Field Name	Bits	Default	Description
REUSE_DEPTH	7:0	none	Set this register to 2 more than the desired reuse depth. Ideally this should be set to 16 and not changed

VGT:VGT_IMMED_DATA · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x287f4			
DESCRIPTION: *VGT Immediate Data*			
Field Name	Bits	Default	Description
DATA	31:0	none	Data written to this address is written into the VGT Immediate Data FIFO.

VGT:VGT_INDEX_TYPE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x895c
DESCRIPTION: *VGT Index Type*

Field Name	Bits	Default	Description
INDEX_TYPE	1:0	none	Index Type (applicable to prim types 0-28 only). If the Source Select field is set to `Auto-increment Index` mode, then this field is ignored and the index type is 32-bits per index POSSIBLE VALUES: 00 - DI_INDEX_SIZE_16_BIT: DI_INDEX_SIZE_16_BIT 16 bits per index 01 - DI_INDEX_SIZE_32_BIT: DI_INDEX_SIZE_32_BIT 32 bits per index

VGT:VGT_INDX_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28408			
DESCRIPTION: *For components that are that are specified to be indices (see the VGT_GROUP_VECT_0_FMT_CNTL register), this register is the offset value. Offsetting occurs prior to clamping and fix->flt conversion.*			
Field Name	Bits	Default	Description
INDX_OFFSET	31:0	none	Index offset value (32-bit adder), extend it to 32-bits

VGT:VGT_INSTANCE_STEP_RATE_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28aa0			
DESCRIPTION: *This register defines the first instance step rate*			
Field Name	Bits	Default	Description
STEP_RATE	31:0	none	Instance step rate

VGT:VGT_INSTANCE_STEP_RATE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28aa4			
DESCRIPTION: *This register defines the second instance step rate*			
Field Name	Bits	Default	Description
STEP_RATE	31:0	none	Instance step rate

VGT:VGT_LAST_COPY_STATE · [R] · 32 bits · Access: 32 · GpuF0MMReg:0x88c0			
DESCRIPTION: *This register retains the data from the last GFX_COPY_STATE command.*			
Field Name	Bits	Default	Description
SRC_STATE_ID	2:0	none	Source context from last GFX_COPY_STATE command.
DST_STATE_ID	18:16	none	Destination context from last GFX_COPY_STATE command.

VGT:VGT_LS_HS_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b58
DESCRIPTION: *Used to specify LS/HS control values*

Field Name	Bits	Default	Description
NUM_PATCHES	7:0	none	Indicates number of patches in a threadgroup
HS_NUM_INPUT_CP	13:8	none	Number of control points in HS input patch
HS_NUM_OUTPUT_CP	19:14	none	Number of control points in HS output patch

VGT:VGT_MAX_VTX_INDX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28400			
DESCRIPTION: *For components that are that are specified to be indices (see the VGT_GROUP_VECT_0_FMT_CNTL register), this register is the maximum clamp value. Clamping occurs after offsetting and prior to fix->flt conversion.*			
Field Name	Bits	Default	Description
MAX_INDX	31:0	none	maximum clamp value for index clamp, exten it to 32-bit

VGT:VGT_MIN_VTX_INDX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28404			
DESCRIPTION: *For components that are that are specified to be indices (see the VGT_GROUP_VECT_0_FMT_CNTL register), this register is the minimum clamp value. Clamping occurs after offsetting and prior to fix->flt conversion.*			
Field Name	Bits	Default	Description
MIN_INDX	31:0	none	minimum clamp value for index clamp, extend it to 32-bits

VGT:VGT_MULTI_PRIM_IB_RESET_EN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a94			
DESCRIPTION: *This register enabling reseting of prim based on reset index*			
Field Name	Bits	Default	Description
RESET_EN	0	none	IF SET, THEN RESET INDEX IS USED FOR RESETING A PRIM POSSIBLE VALUES: 00 - multi_prim reset off 01 - multi_prim reset on

VGT:VGT_MULTI_PRIM_IB_RESET_INDX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2840c			
DESCRIPTION: *This register specifies the 32-bit index value used to reset the primitive order (strip/fan/polygon)*			
Field Name	Bits	Default	Description
RESET_INDX	31:0	none	If this value matches an index in the IB, a new primitive set is started.

VGT:VGT_NUM_INDICES · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x8970			
DESCRIPTION: *VGT Number of Indices*			
Field Name	Bits	Default	Description

NUM_INDICES	31:0	none	This field indicates the number of indices to process for this draw initiator. Note this count is not necessarily the count of the primitives. It is also not the index buffer size in memory.When using Dx11 compute shaders, this register needs to be written by the driver to the product of x,y,z which are the 3 dimensions that define a compute shader threadgroup size.

VGT:VGT_NUM_INSTANCES · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x8974			
DESCRIPTION: *VGT Number of Instances*			
Field Name	Bits	Default	Description
NUM_INSTANCES	31:0	none	Number of instances in a draw call, if set to zero, it is interpreted as 1. The maximum value is 2^32-1

VGT:VGT_OUTPUT_PATH_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a10			
DESCRIPTION: *THIS REGISTER IS IGNORED IN MAJOR MODE 0 FOR PRIM TYPES 0 THRU 21 !! This register selects which backend path will be used by the VGT block.*			
Field Name	Bits	Default	Description
PATH_SELECT	2:0	none	This field indicates the VGT back-end path to be used. POSSIBLE VALUES: 00 - VGT_OUTPATH_VTX_REUSE: VGT_OUTPATH_VTX_REUSE 01 - VGT_OUTPATH_TESS_EN: VGT_OUTPATH_TESS_EN 02 - VGT_OUTPATH_PASSTHRU: VGT_OUTPATH_PASSTHRU 03 - VGT_OUTPATH_GS_BLOCK: VGT_OUTPATH_GS_BLOCK 04 - VGT_OUTPATH_HS_BLOCK: VGT_OUTPATH_HS_BLOCK

VGT:VGT_OUT_DEALLOC_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c5c			
DESCRIPTION: *This register controls, within a process vector, when the previous process vector is de-allocated.*			
Field Name	Bits	Default	Description
DEALLOC_DIST	6:0	none	From r7xx onwards this register should only be set to 16

VGT:VGT_PRIMITIVEID_EN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a84			
DESCRIPTION: *This register specifies the 32-bit start primitiveID value specified by user which is incremented for each new primitive*			
Field Name	Bits	Default	Description
PRIMITIVEID_EN	0	none	PrimitiveID generation is enabled

			POSSIBLE VALUES: 00 - suppress PrimitiveID output 01 - output primitiveID

VGT:VGT_PRIMITIVE_TYPE · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x8958

DESCRIPTION: *VGT Primitive Type*

Field Name	Bits	Default	Description
PRIM_TYPE	5:0	none	Primitive Type. This field is only used in Major mode 0. For Major Mode 1, the prim type specified in the VGT_GRP_PRIM_TYPE register is used POSSIBLE VALUES: 00 - DI_PT_NONE: DI_PT_NONE None (does not create draw trigger) 01 - DI_PT_POINTLIST: DI_PT_POINTLIST Point List 02 - DI_PT_LINELIST: DI_PT_LINELIST Line List 03 - DI_PT_LINESTRIP: DI_PT_LINESTRIP Line Strip 04 - DI_PT_TRILIST: DI_PT_TRILIST Tri List 05 - DI_PT_TRIFAN: DI_PT_TRIFAN Tri Fan 06 - DI_PT_TRISTRIP: DI_PT_TRISTRIP Tri Strip 07 - DI_PT_UNUSED_0: DI_PT_UNUSED_0 Reserved 1 08 - DI_PT_UNUSED_1: DI_PT_UNUSED_1 Reserved 2 09 - DI_PT_PATCH: DI_PT_PATCH Patch prim type used in conjuction with HS_NUM_INPUT_CP 10 - DI_PT_LINELIST_ADJ: DI_PT_LINELIST_ADJ Adjacent Line List 11 - DI_PT_LINESTRIP_ADJ: DI_PT_LINESTRIP_ADJ Adjacent Line Strip 12 - DI_PT_TRILIST_ADJ: DI_PT_TRILIST_ADJ Adjacent Tri List 13 - DI_PT_TRISTRIP_ADJ: DI_PT_TRISTRIP_ADJ Adjacent Tri Strip 14 - DI_PT_UNUSED_3: DI_PT_UNUSED_3 Reserved 3 15 - DI_PT_UNUSED_4: DI_PT_UNUSED_4 Reserved 4 16 - DI_PT_TRI_WITH_WFLAGS: DI_PT_TRI_WITH_WFLAGS Tri List w/Flags (legacy R128) 17 - DI_PT_RECTLIST: DI_PT_RECTLIST Rect List 18 - DI_PT_LINELOOP: DI_PT_LINELOOP Line LOOP 19 - DI_PT_QUADLIST: DI_PT_QUADLIST Quad

			List 20 - DI_PT_QUADSTRIP: DI_PT_QUADSTRIP Quad Strip 21 - DI_PT_POLYGON: DI_PT_POLYGON Polygon 22 - DI_PT_2D_COPY_RECT_LIST_V0: DI_PT_2D_COPY_RECT_LIST_V0 2D Copy Rect List V0 23 - DI_PT_2D_COPY_RECT_LIST_V1: DI_PT_2D_COPY_RECT_LIST_V1 2D Copy Rect List V1 24 - DI_PT_2D_COPY_RECT_LIST_V2: DI_PT_2D_COPY_RECT_LIST_V2 2D Copy Rect List V2 25 - DI_PT_2D_COPY_RECT_LIST_V3: DI_PT_2D_COPY_RECT_LIST_V3 2D Copy Rect List V3 26 - DI_PT_2D_FILL_RECT_LIST: DI_PT_2D_FILL_RECT_LIST 2D Fill Rect List 27 - DI_PT_2D_LINE_STRIP: DI_PT_2D_LINE_STRIP 2D Line Strip 28 - DI_PT_2D_TRI_STRIP: DI_PT_2D_TRI_STRIP 2D Triangle Strip

VGT:VGT_REUSE_OFF · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ab4			
DESCRIPTION: *This register will turn off reuse in for VS process vector generation. Note that we will never turn off reuse for ES process vector. Reuse will be turned off for streamout and viewport*			
Field Name	Bits	Default	Description
REUSE_OFF	0	none	reuse is off (set to 1) POSSIBLE VALUES: 00 - Reuse on 01 - Reuse off

VGT:VGT_SHADER_STAGES_EN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b54			
DESCRIPTION: *This is used to specify what shader stages are enabled. A VGT_FLUSH or PIPE FLUSH maybe required when changing to/from some combinations*			
Field Name	Bits	Default	Description
LS_EN	1:0	none	Controls the behavior of the LS stage POSSIBLE VALUES: 00 - LS_STAGE_OFF: LS shader stage is Off 01 - LS_STAGE_ON: LS shader stage is On 02 - CS_STAGE_ON: Dx11 Compute shader is On 03 - RESERVED_LS: RESERVED
HS_EN	2	none	Controls the behavior of the HS stage

			POSSIBLE VALUES: 00 - HS_STAGE_OFF: HS Stage is Off 01 - HS_STAGE_ON: HS Stage is On
ES_EN	4:3	none	Controls the behavior of the ES stage POSSIBLE VALUES: 00 - ES_STAGE_OFF: ES Stage is Off 01 - ES_STAGE_DS: ES Stage is On, the ES is the DS Shader for tessellation eveluation 02 - ES_STAGE_REAL: ES Stage is On, and a real ES is being used in conjuction with a GS 03 - RESERVED_ES: RESERVED
GS_EN	5	none	Controls the behavior of the GS stage POSSIBLE VALUES: 00 - GS_STAGE_OFF: GS Stage is Off 01 - GS_STAGE_ON: GS Stage is On, VGT_GS_MODE.bits.MODE must be set to SCENARIO_G
VS_EN	7:6	none	Controls the behavior of the VS stage POSSIBLE VALUES: 00 - VS_STAGE_REAL: VS Stage is On, writes to the parameter cache (Dx9 mode) 01 - VS_STAGE_DS: VS Stage is On, acts as an evaluation shader (DS) for Dx11 tessellation 02 - VS_STAGE_COPY_SHADER: VS Stage is On, the VS is a copy shader for fetching from the GS ring and writing to the parameter cache 03 - RESERVED_VS: RESERVED
DYNAMIC_HS	8	none	Indicates whether the output of the HS stages always stays on-chip (evergreen mode) or whether its dynamically decided to use off-chip memory and thus use multiple SIMDs to execute subsequent DS waves from the threadgroup POSSIBLE VALUES: 00 - hs_onchip 01 - hs_dynamic_off_chip

VGT:VGT_STRMOUT_BASE_OFFSET_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b10			
DESCRIPTION: *Stream out base_0 + offset_0. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	31:0	none	DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b14			
DESCRIPTION: *Stream out base_1 + offset_1. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	31:0	none	DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b18			
DESCRIPTION: *Stream out base_2 + offset_2. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	31:0	none	DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b1c			
DESCRIPTION: *Stream out base_3 + offset_3. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	31:0	none	DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_HI_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b44			
DESCRIPTION: *Upper 6-bits of 40-bits Stream out base_0 + offset_0. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	5:0	none	Upper 6-bits of 40-bits DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_HI_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b48			
DESCRIPTION: *Upper 6-bits of 40-bits Stream out base_1 + offset_1. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	5:0	none	Upper 6-bits of 40-bits DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_HI_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b4c			
DESCRIPTION: *Upper 6-bits of 40-bits Stream out base_2 + offset_2. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	5:0	none	Upper 6-bits of 40-bits DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BASE_OFFSET_HI_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b50

DESCRIPTION: *Upper 6-bits of 40-bits Stream out base_3 + offset_3. This register is owned by SQ.*			
Field Name	Bits	Default	Description
BASE_OFFSET	5:0	none	Upper 6-bits of 40-bits DWORD base+offset for given stream out buffer. Set by CP or driver.

VGT:VGT_STRMOUT_BUFFER_BASE_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ad8			
DESCRIPTION: *Stream-out base.*			
Field Name	Bits	Default	Description
BASE	31:0	none	256-byte aligned buffer base for given stream out buffer. Bits 31:0 corresponds to 39:8 of memory address. This data can be stored in the coherency registers. This register is snooped by CP.

VGT:VGT_STRMOUT_BUFFER_BASE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ae8			
DESCRIPTION: *Stream-out base.*			
Field Name	Bits	Default	Description
BASE	31:0	none	256-byte aligned buffer base for given stream out buffer. Bits 31:0 corresponds to 39:8 of memory address. This data can be stored in the coherency registers. This register is snooped by CP.

VGT:VGT_STRMOUT_BUFFER_BASE_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28af8			
DESCRIPTION: *Stream-out base.*			
Field Name	Bits	Default	Description
BASE	31:0	none	256-byte aligned buffer base for given stream out buffer. Bits 31:0 corresponds to 39:8 of memory address. This data can be stored in the coherency registers. This register is snooped by CP.

VGT:VGT_STRMOUT_BUFFER_BASE_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b08			
DESCRIPTION: *Stream-out base.*			
Field Name	Bits	Default	Description
BASE	31:0	none	256-byte aligned buffer base for given stream out buffer. Bits 31:0 corresponds to 39:8 of memory address. This data can be stored in the coherency registers. This register is snooped by CP.

VGT:VGT_STRMOUT_BUFFER_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b98			
DESCRIPTION: *Stream out enable bits. CP will use for SO coherency register validness.*			
Field Name	Bits	Default	Description

STREAM_0_BUFFER_EN	3:0	0x0	Bind buffers for stream 0. Bit 0 set to on indicates buffer 0 is bound, bit 1 for buffer 1, bit 2 for buffer 2 and bit 3 for buffer 3
STREAM_1_BUFFER_EN	7:4	0x0	Bind buffers for stream 1. Bit 0 set to on indicates buffer 0 is bound, bit 1 for buffer 1, bit 2 for buffer 2 and bit 3 for buffer 3
STREAM_2_BUFFER_EN	11:8	0x0	Bind buffers for stream 2. Bit 0 set to on indicates buffer 0 is bound, bit 1 for buffer 1, bit 2 for buffer 2 and bit 3 for buffer 3
STREAM_3_BUFFER_EN	15:12	0x0	Bind buffers for stream 3. Bit 0 set to on indicates buffer 0 is bound, bit 1 for buffer 1, bit 2 for buffer 2 and bit 3 for buffer 3

VGT:VGT_STRMOUT_BUFFER_FILLED_SIZE_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8960

DESCRIPTION: *Stream-out adjusted size.*

Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Sum of (SO_BufferOffset + BufDwordWritten) for given buffer. Read Only. To read this register the VGT needs to be flushed to the point BufDwordWritten counts are maintained.

VGT:VGT_STRMOUT_BUFFER_FILLED_SIZE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8964

DESCRIPTION: *Stream-out adjusted size.*

Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Sum of (SO_BufferOffset + BufDwordWritten) for given buffer. Read Only. To read this register the VGT needs to be flushed to the point BufDwordWritten counts are maintained.

VGT:VGT_STRMOUT_BUFFER_FILLED_SIZE_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8968

DESCRIPTION: *Stream-out adjusted size.*

Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Sum of (SO_BufferOffset + BufDwordWritten) for given buffer. Read Only. To read this register the VGT needs to be flushed to the point BufDwordWritten counts are maintained.

VGT:VGT_STRMOUT_BUFFER_FILLED_SIZE_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x896c

DESCRIPTION: *Stream-out adjusted size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Sum of (SO_BufferOffset + BufDwordWritten) for given buffer. Read Only. To read this register the VGT needs to be flushed to the point BufDwordWritten counts are maintained.

VGT:VGT_STRMOUT_BUFFER_OFFSET_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28adc			
DESCRIPTION: *Stream out offset.*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	DWORD offset for given stream out buffer. Writing this register will cause the VGT to load a Zero into BufDwordWritten[4] and SO_CurVertIndex.

VGT:VGT_STRMOUT_BUFFER_OFFSET_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28aec			
DESCRIPTION: *Stream out offset.*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	DWORD offset for given stream out buffer. Writing this register will cause the VGT to load a Zero into BufDwordWritten[4] and SO_CurVertIndex.

VGT:VGT_STRMOUT_BUFFER_OFFSET_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28afc			
DESCRIPTION: *Stream out offset.*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	DWORD offset for given stream out buffer. Writing this register will cause the VGT to load a Zero into BufDwordWritten[4] and SO_CurVertIndex.

VGT:VGT_STRMOUT_BUFFER_OFFSET_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b0c			
DESCRIPTION: *Stream out offset.*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	DWORD offset for given stream out buffer. Writing this register will cause the VGT to load a Zero into BufDwordWritten[4] and SO_CurVertIndex.

VGT:VGT_STRMOUT_BUFFER_SIZE_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ad0			
DESCRIPTION: *Stream-out size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Buffer size for given stream out buffer.

VGT:VGT_STRMOUT_BUFFER_SIZE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ae0			
DESCRIPTION: *Stream-out size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Buffer size for given stream out buffer.

VGT:VGT_STRMOUT_BUFFER_SIZE_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28af0			
DESCRIPTION: *Stream-out size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Buffer size for given stream out buffer.

VGT:VGT_STRMOUT_BUFFER_SIZE_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b00			
DESCRIPTION: *Stream-out size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	DWORD Buffer size for given stream out buffer.

VGT:VGT_STRMOUT_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b94			
DESCRIPTION: *This register enables streaming out*			
Field Name	Bits	Default	Description
STREAMOUT_0_EN	0	0x0	If set, stream output to stream 0 is enabled
STREAMOUT_1_EN	1	0x0	If set, stream output to stream 1 is enabled
STREAMOUT_2_EN	2	0x0	If set, stream output to stream 2 is enabled
STREAMOUT_3_EN	3	0x0	If set, stream output to stream 3 is enabled
RAST_STREAM	6:4	0x0	Streamed for which rasterization is enabled, If bit[6] is set then rasterization is not enabled for any stream

VGT:VGT_STRMOUT_DRAW_OPAQUE_BUFFER_FILLED_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b2c			
DESCRIPTION: *Draw opaque size.*			
Field Name	Bits	Default	Description
SIZE	31:0	none	This will be loaded by the CP for a DrawOpaque call by fetching a memory address containing last bufferfilledsize associated with the previous stream out buffer bound to the IA.

VGT:VGT_STRMOUT_DRAW_OPAQUE_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b28

DESCRIPTION: *Draw opaque offset.*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	pOffsets from the IASetVertexBuffers binding of a stream out buffer that is to be used as src data. The retrived BufferFilledSize minus this poffset if positive, will determine the amount of data from which primitives can be created.

VGT:VGT_STRMOUT_DRAW_OPAQUE_VERTEX_STRIDE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b30			
DESCRIPTION: *Draw opaque vertex stride.*			
Field Name	Bits	Default	Description
VERTEX_STRIDE	8:0	none	vertex stride used for draw opaque call

VGT:VGT_STRMOUT_VTX_STRIDE_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ad4			
DESCRIPTION: *Stream out stride.*			
Field Name	Bits	Default	Description
STRIDE	9:0	none	DWORD stride between vertices in given stream-out buffer. From stream output declarations details of dx10 spec, the max stride 2048 bytes or 512 words defined to be the spacing between the beginning of each vertex.

VGT:VGT_STRMOUT_VTX_STRIDE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ae4			
DESCRIPTION: *Stream out stride.*			
Field Name	Bits	Default	Description
STRIDE	9:0	none	DWORD stride between vertices in given stream-out buffer. From stream output declarations details of dx10 spec, the max stride 2048 bytes or 512 words defined to be the spacing between the beginning of each vertex.

VGT:VGT_STRMOUT_VTX_STRIDE_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28af4			
DESCRIPTION: *Stream out stride.*			
Field Name	Bits	Default	Description
STRIDE	9:0	none	DWORD stride between vertices in given stream-out buffer. From stream output declarations details of dx10 spec, the max stride 2048 bytes or 512 words defined to be the spacing between the beginning of each vertex.

VGT:VGT_STRMOUT_VTX_STRIDE_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b04
DESCRIPTION: *Stream out stride.*

Field Name	Bits	Default	Description
STRIDE	9:0	none	DWORD stride between vertices in given stream-out buffer. From stream output declarations details of dx10 spec, the max stride 2048 bytes or 512 words defined to be the spacing between the beginning of each vertex.

VGT:VGT_SYS_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x898c			
DESCRIPTION: *Used to specify system constraints*			
Field Name	Bits	Default	Description
DUAL_CORE_EN	0	0x1	Set if dual cores are present
MAX_LS_HS_THDGRP	6:1	0x8	Reserved.

VGT:VGT_VERTEX_REUSE_BLOCK_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c58			
DESCRIPTION: *This register controls the behavior of the Vertex Reuse block at the backend of the VGT. This register is relevant only if the VGT_OUTPUT_PATH_CNTL register (or the prim type in Major Mode 0) specifies the Vertex Reuse Block for the VGT backend path.*			
Field Name	Bits	Default	Description
VTX_REUSE_DEPTH	7:0	none	From r7xx onwards, the reuse depth should be set to 14. It can also be set to 15 (if prim type is line) and 16 (if prim type is points)

VGT:VGT_VTX_CNT_EN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ab8			
DESCRIPTION: *This register specifies auto-index generation in reuse mode. The y component of first vector output will have auto index value. The auto-index value is reset to zero by an event sent to VGT.*			
Field Name	Bits	Default	Description
VTX_CNT_EN	0	none	Set to one if auto index generation is enabled. This is for import by the vertex shader over the y channel. It is different than DRAW_INDEX_AUTO POSSIBLE VALUES: 00 - Auto off 01 - Auto on

VGT:VGT_VTX_VECT_EJECT_REG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x88b0			
DESCRIPTION: *This register defines the number of primitives that are allowed to pass during the assembly of a single vertex vector. After this number of primitives have passed, the vertex vector is submitted to the shaders for processing even if it is not full.*			
Field Name	Bits	Default	Description
PRIM_COUNT	9:0	0x7F	This is the count of primitives allowed to pass during the assembly of a single vertex vector.

2. Primitive Assembly Registers

PA:PA_CL_CLIP_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28810			
DESCRIPTION: *Clipper Control Bits*			
Field Name	Bits	Default	Description
UCP_ENA_0	0	none	Enable User-Clip Plane 0
UCP_ENA_1	1	none	Enable User-Clip Plane 1
UCP_ENA_2	2	none	Enable User-Clip Plane 2
UCP_ENA_3	3	none	Enable User-Clip Plane 3
UCP_ENA_4	4	none	Enable User-Clip Plane 4
UCP_ENA_5	5	none	Enable User-Clip Plane 5
PS_UCP_Y_SCALE_NEG	13	none	
PS_UCP_MODE	15:14	none	0 = Cull using distance from center of point 1 = Cull using radius-based distance from center of point 2 = Cull using radius-based distance from center of point, Expand and Clip on intersection 3 = Always expand and clip as trifan
CLIP_DISABLE	16	none	Disables clip code generation and clipping process for TCL
UCP_CULL_ONLY_ENA	17	none	Cull Primitives against UCPS, but don`t clip
BOUNDARY_EDGE_FLAG_ENA	18	none	Currently unused: Pending Delete. Left as placeholder for now.
DX_CLIP_SPACE_DEF	19	none	Clip space is defined as: 0: -W < X < W, -W < Y < W, -W < Z < W (OpenGL Definition) 1: -W < X < W, -W < Y < W, 0 < Z < W (DirectX Definition)
DIS_CLIP_ERR_DETECT	20	none	Disables culling of primitives for which the clipped detects an error. Default is 0
VTX_KILL_OR	21	none	Used if Vertex Kill flags are exported from Vertex Shader. If clear, ALL vertices for current primitive must be set to kill the primitive (AND MODE). If set, if ANY vertices for current primitive are set, the the primitive will be killed (OR MODE).
DX_RASTERIZATION_KILL	22	none	
DX_LINEAR_ATTR_CLIP_ENA	24	none	
VTE_VPORT_PROVOKE_DISABLE	25	none	
ZCLIP_NEAR_DISABLE	26	none	
ZCLIP_FAR_DISABLE	27	none	

PA:PA_CL_CNTL_STATUS · [R] · 32 bits · Access: 32 · GpuF0MMReg:0x8a10

DESCRIPTION: *Status Bits*			
Field Name	Bits	Default	Description
CL_BUSY	31	none	Busy Status Bit

PA:PA_CL_ENHANCE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8a14			
DESCRIPTION: *Used for Late Additions of Control Bits*			
Field Name	Bits	Default	Description
CLIP_VTX_REORDER_ENA	0	none	Enables vertex-order-independent clipping
NUM_CLIP_SEQ	2:1	none	Number of Clip Sequences Active (+1). Should be set to 3 (4 sequences) for best performance
CLIPPED_PRIM_SEQ_STALL	3	none	Forces a faster clip path if NUM_CLIP_SEQ is set to 0 (which should only be if 3 does not work)
VE_NAN_PROC_DISABLE	4	none	

PA:PA_CL_GB_HORZ_CLIP_ADJ · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bf0			
DESCRIPTION: *Horizontal Guard Band Clip Adjust Register*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	32-bit floating point value. Should be set to 1.0 for no guard band.

PA:PA_CL_GB_HORZ_DISC_ADJ · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bf4			
DESCRIPTION: *Horizontal Guard Band Discard Adjust Register*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	32-bit floating point value. Should be set to 1.0 for no guard band.

PA:PA_CL_GB_VERT_CLIP_ADJ · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28be8			
DESCRIPTION: *Vertical Guard Band Clip Adjust Register*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	32-bit floating point value. Should be set to 1.0 for no guard band.

PA:PA_CL_GB_VERT_DISC_ADJ · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28be8c			
DESCRIPTION: *Vertical Guard Band Discard Adjust Register*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	32-bit floating point value. Should be set to 1.0 for no guard band.

PA:PA_CL_NANINF_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28820			
Field Name	Bits	Default	Description
VTE_XY_INF_DISCARD	0	none	
VTE_Z_INF_DISCARD	1	none	
VTE_W_INF_DISCARD	2	none	
VTE_0XNANINF_IS_0	3	none	
VTE_XY_NAN_RETAIN	4	none	
VTE_Z_NAN_RETAIN	5	none	
VTE_W_NAN_RETAIN	6	none	
VTE_W_RECIP_NAN_IS_0	7	none	
VS_XY_NAN_TO_INF	8	none	
VS_XY_INF_RETAIN	9	none	
VS_Z_NAN_TO_INF	10	none	
VS_Z_INF_RETAIN	11	none	
VS_W_NAN_TO_INF	12	none	
VS_W_INF_RETAIN	13	none	
VS_CLIP_DIST_INF_DISCARD	14	none	
VTE_NO_OUTPUT_NEG_0	20	none	

PA:PA_CL_POINT_CULL_RAD · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x287e0			
DESCRIPTION: *Point Sprite Culling Radius Expansion SQRT(XRadExp^2 + YRadExp^2)*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_POINT_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x287dc			
DESCRIPTION: *Point Sprite Constant Size*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_POINT_X_RAD · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x287d4			
DESCRIPTION: *Point Sprite X Radius Expansion*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_POINT_Y_RAD · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x287d8			
DESCRIPTION: *Point Sprite Y Radius Expansion*			
Field Name	Bits	Default	Description

DATA_REGISTER	31:0	none	

PA:PA_CL_UCP_[0-5]_W · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x285c8-0x28618			
DESCRIPTION: *User Clip Plane Data*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_UCP_[0-5]_X · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x285bc-0x2860c			
DESCRIPTION: *User Clip Plane Data*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_UCP_[0-5]_Y · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x285c0-0x28610			
DESCRIPTION: *User Clip Plane Data*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_UCP_[0-5]_Z · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x285c4-0x28614			
DESCRIPTION: *User Clip Plane Data*			
Field Name	Bits	Default	Description
DATA_REGISTER	31:0	none	

PA:PA_CL_VPORT_XOFFSET_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28440-0x285a8			
DESCRIPTION: *Viewport Transform X Offset - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_XOFFSET	31:0	none	Viewport Offset for X coordinates. An IEEE float.

PA:PA_CL_VPORT_XSCALE_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2843c-0x285a4			
DESCRIPTION: *Viewport Transform X Scale Factor - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_XSCALE	31:0	none	Viewport Scale Factor for X coordinates. An IEEE float.

PA:PA_CL_VPORT_YOFFSET_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28448-0x285b0			
DESCRIPTION: *Viewport Transform Y Offset - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description

VPORT_YOFFSET	31:0	none	Viewport Offset for Y coordinates. An IEEE float.

PA:PA_CL_VPORT_YSCALE_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28444-0x285ac			
DESCRIPTION: *Viewport Transform Y Scale Factor - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_YSCALE	31:0	none	Viewport Scale Factor for Y coordinates. An IEEE float.

PA:PA_CL_VPORT_ZOFFSET_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28450-0x285b8			
DESCRIPTION: *Viewport Transform Z Offset - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_ZOFFSET	31:0	none	Viewport Offset for Z coordinates. An IEEE float.

PA:PA_CL_VPORT_ZSCALE_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2844c-0x285b4			
DESCRIPTION: *Viewport Transform Z Scale Factor - 1-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_ZSCALE	31:0	none	Viewport Scale Factor for Z coordinates. An IEEE float.

PA:PA_CL_VS_OUT_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2881c			
DESCRIPTION: *Vertex Shader Output Control*			
Field Name	Bits	Default	Description
CLIP_DIST_ENA_0	0	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_1	1	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_2	2	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_3	3	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_4	4	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_5	5	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CLIP_DIST_ENA_6	6	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.

CLIP_DIST_ENA_7	7	none	Enable ClipDistance# to be used for user-defined clipping. Requires VS_OUT_CCDIST#_ENA to be set.
CULL_DIST_ENA_0	8	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_1	9	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_2	10	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_3	11	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_4	12	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_5	13	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_6	14	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
CULL_DIST_ENA_7	15	none	Enable CullDistance# to be used for user-defined clip discard. Requires VS_OUT_CCDIST#_ENA to be set. If all verts of a primitive are outside (culldist<0), then primitive is discarded, else just let through (i.e. NOT clipped).
USE_VTX_POINT_SIZE	16	none	Use the PointSize output from the VS (in the x channel of VS_OUT_MISC_VEC).
USE_VTX_EDGE_FLAG	17	none	Use the EdgeFlag output from the VS (in the y channel of VS_OUT_MISC_VEC).
USE_VTX_RENDER_TARGET_INDX	18	none	Use the RenderTargetArrayIndx output from the VS (in the z channel of VS_OUT_MISC_VEC). Only valid

			for WGF Geometry Shader
USE_VTX_VIEWPORT_INDX	19	none	Use the ViewportArrayIndx output from the VS (in the w channel of VS_OUT_MISC_VEC). Only valid for WGF Geometry Shader
USE_VTX_KILL_FLAG	20	none	Use the KillFlag output from the VS (in the z channel of VS_OUT_MISC_VEC). Mutually exclusive from RTarrayindx
VS_OUT_MISC_VEC_ENA	21	none	Output the VS output misc vector from the VS (SX) to the PA (primitive assembler). Should be set if any of the fields are to be used
VS_OUT_CCDIST0_VEC_ENA	22	none	Output the VS output ccdist0 vector from the VS (SX) to the PA (primitive assembler). Should be set if any of the fields are to be used
VS_OUT_CCDIST1_VEC_ENA	23	none	Output the VS output ccdist1 vector from the VS (SX) to the PA (primitive assembler). Should be set if any of the fields are to be used

PA:PA_CL_VTE_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28818			
DESCRIPTION: *Viewport Transform Engine Control*			
Field Name	Bits	Default	Description
VPORT_X_SCALE_ENA	0	none	Viewport Transform Scale Enable for X component
VPORT_X_OFFSET_ENA	1	none	Viewport Transform Offset Enable for X component
VPORT_Y_SCALE_ENA	2	none	Viewport Transform Scale Enable for Y component
VPORT_Y_OFFSET_ENA	3	none	Viewport Transform Offset Enable for Y component
VPORT_Z_SCALE_ENA	4	none	Viewport Transform Scale Enable for Z component
VPORT_Z_OFFSET_ENA	5	none	Viewport Transform Offset Enable for Z component
VTX_XY_FMT	8	none	Indicates that the incoming X, Y have already been multiplied by 1/W0. If OFF, the Setup Engine will multiply the X, Y coordinates by 1/W0.,
VTX_Z_FMT	9	none	Indicates that the incoming Z has already been multiplied by 1/W0. If OFF, the Setup Engine will multiply the Z coordinate by 1/W0.
VTX_W0_FMT	10	none	Indicates that the incoming W0 is not 1/W0. If ON, the Setup Engine will perform the reciprocal to get 1/W0.

PA:PA_SC_AA_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28be0			
DESCRIPTION: *Multisample Antialiasing Control*			
Field Name	Bits	Default	Description
MSAA_NUM_SAMPLES	2:0	none	Specifies the number of samples to use for MSAA Detail Sampling. 0 = 1-sample, 1 = 2-sample, 2 = 4-sample, 3 =

			8-sample, 4 = 16-sample.
AA_MASK_CENTROID_DTMN	4	none	Specifies whether to apply the MSAA Mask before or after the centroid determination. 0 = before; 1 = after.
MAX_SAMPLE_DIST	16:13	none	Specifies the maximum distance (in subpixels) between the pixel center and the outermost subpixel sample. This value is used to optimize coarse walk and quad identity. Should be set to 0 when not anti-aliasing. Max value for R600 should be 8(16ths).
MSAA_EXPOSED_SAMPLES	22:20	none	Specifies the number of samples the pixel shader can see from the primitive`s coverage in the pixel.
DETAIL_TO_EXPOSED_MODE	25:24	none	Specifies the mode to use when converting from a higher detail sample mask to a lower exposed mask. 0 = if result of MASK off higher samples = 0, then OR, 1 = MASK off higher samples, 2 = OR.

PA:PA_SC_AA_MASK_X0Y0_X1Y0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c38			
DESCRIPTION: *Multisample AA Mask Pixel X0,Y0 (Upper Left) and X1,Y0 (Upper Right) of Quad*			
Field Name	Bits	Default	Description
AA_MASK_X0Y0	15:0	none	16-bit mask applied to pixel X0,Y0(ULC) as follows: LSB is Sample0, MSB is Sample15.
AA_MASK_X1Y0	31:16	none	16-bit mask applied to pixel X1,Y0(URC) as follows: LSB is Sample0, MSB is Sample15.

PA:PA_SC_AA_MASK_X0Y1_X1Y1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c3c			
DESCRIPTION: *Multisample AA Mask Pixel X0,Y1 (Lower Left) and X1,Y1 (Lower Right) of Quad*			
Field Name	Bits	Default	Description
AA_MASK_X0Y1	15:0	none	16-bit mask applied to pixel X0,Y1(LLC) as follows: LSB is Sample0, MSB is Sample15.
AA_MASK_X1Y1	31:16	none	16-bit mask applied to pixel X1,Y1(LRC) as follows: LSB is Sample0, MSB is Sample15.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y0_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bf8			
DESCRIPTION: *Multi-Sample Programmable Sample Locations 0-3 for Pixel X0,Y0 (Upper Left) of Quad - Used by SC, SPI, DB, CB*			
Field Name	Bits	Default	Description
S0_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S0_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.

S3_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y0_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bfc

DESCRIPTION: *Multi-Sample Programmable Sample Locations 4-7 for Pixel X0,Y0 (Upper Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S4_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S4_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y0_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c00

DESCRIPTION: *Multi-Sample Programmable Sample Locations 8-11 for Pixel X0,Y0 (Upper Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S8_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S8_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y0_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c04

DESCRIPTION: *Multi-Sample Programmable Sample Locations 12-15 for Pixel X0,Y0 (Upper Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S12_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S12_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.

S14_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y1_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c18

DESCRIPTION: *Multi-Sample Programmable Sample Locations 0-3 for Pixel X0,Y1 (Lower Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S0_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S0_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y1_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c1c

DESCRIPTION: *Multi-Sample Programmable Sample Locations 4-7 for Pixel X0,Y1 (Lower Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S4_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S4_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y1_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c20

DESCRIPTION: *Multi-Sample Programmable Sample Locations 8-11 for Pixel X0,Y1 (Lower Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S8_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S8_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.

S9_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X0Y1_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c24

DESCRIPTION: *Multi-Sample Programmable Sample Locations 12-15 for Pixel X0,Y1 (Lower Left) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S12_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S12_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y0_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c08

DESCRIPTION: *Multi-Sample Programmable Sample Locations 0-3 for Pixel X1,Y0 (Upper Right) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S0_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S0_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y0_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c0c

DESCRIPTION: *Multi-Sample Programmable Sample Locations 4-7 for Pixel X1,Y0 (Upper Right) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description

S4_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S4_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y0_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c10

DESCRIPTION: *Multi-Sample Programmable Sample Locations 8-11 for Pixel X1,Y0 (Upper Right) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S8_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S8_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y0_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c14

DESCRIPTION: *Multi-Sample Programmable Sample Locations 12-15 for Pixel X1,Y0 (Upper Right) of Quad - Used by SC, SPI, DB, CB*

Field Name	Bits	Default	Description
S12_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S12_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y1_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c28

DESCRIPTION: *Multi-Sample Programmable Sample Locations 0-3 for Pixel X1,Y1 (Lower Right) of Quad - Used by SC, SPI, DB, CB*			
Field Name	Bits	Default	Description
S0_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S0_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S1_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S2_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S3_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y1_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c2c			
DESCRIPTION: *Multi-Sample Programmable Sample Locations 4-7 for Pixel X1,Y1 (Lower Right) of Quad - Used by SC, SPI, DB, CB*			
Field Name	Bits	Default	Description
S4_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S4_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S5_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S6_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S7_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y1_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c30			
DESCRIPTION: *Multi-Sample Programmable Sample Locations 8-11 for Pixel X1,Y1 (Lower Right) of Quad - Used by SC, SPI, DB, CB*			
Field Name	Bits	Default	Description
S8_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S8_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S9_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S10_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S11_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_AA_SAMPLE_LOCS_PIXEL_X1Y1_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c34			
DESCRIPTION: *Multi-Sample Programmable Sample Locations 12-15 for Pixel X1,Y1 (Lower Right) of Quad - Used by SC, SPI, DB, CB*			
Field Name	Bits	Default	Description
S12_X	3:0	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S12_Y	7:4	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_X	11:8	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S13_Y	15:12	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_X	19:16	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S14_Y	23:20	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_X	27:24	none	4b signed offset from pixel center. Range -8/16 to 7/16.
S15_Y	31:28	none	4b signed offset from pixel center. Range -8/16 to 7/16.

PA:PA_SC_CENTROID_PRIORITY_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bd4			
DESCRIPTION: *Sample Locations Sorted in Centroid Priority; Driver must sort sample location distances from closest to furthest, puts closest sample location number in DISTANCE_0, next in DISTANCE_1, and so on*			
Field Name	Bits	Default	Description
DISTANCE_0	3:0	none	1st closest sample location to center
DISTANCE_1	7:4	none	2nd closest sample location to center
DISTANCE_2	11:8	none	3rd closest sample location to center
DISTANCE_3	15:12	none	3rd closest sample location to center
DISTANCE_4	19:16	none	4th closest sample location to center
DISTANCE_5	23:20	none	5th closest sample location to center
DISTANCE_6	27:24	none	6th closest sample location to center
DISTANCE_7	31:28	none	7th closest sample location to center

PA:PA_SC_CENTROID_PRIORITY_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bd8			
DESCRIPTION: *Sample Locations Sorted in Centroid Priority; Driver must sort sample location distances from closest to furthest, puts closest sample location number in DISTANCE_0, next in DISTANCE_1, and so on*			
Field Name	Bits	Default	Description
DISTANCE_8	3:0	none	
DISTANCE_9	7:4	none	
DISTANCE_10	11:8	none	
DISTANCE_11	15:12	none	
DISTANCE_12	19:16	none	
DISTANCE_13	23:20	none	
DISTANCE_14	27:24	none	
DISTANCE_15	31:28	none	

PA:PA_SC_CLIPRECT_[0-3]_BR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28214-0x2822c			
DESCRIPTION: *Clip Rectangle Bottom-Right Specification*			
Field Name	Bits	Default	Description
BR_X	14:0	none	Right x value of clip rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT
BR_Y	30:16	none	Bottom y value of clip rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT

PA:PA_SC_CLIPRECT_[0-3]_TL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28210-0x28228			
DESCRIPTION: *Clip Rectangle Top-Left Specification*			
Field Name	Bits	Default	Description
TL_X	14:0	none	Left x value of clip rectangle. 15 bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT
TL_Y	30:16	none	Top y value of clip rectangle. 15 bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT

PA:PA_SC_CLIPRECT_RULE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2820c			
DESCRIPTION: *OpenGL Clip boolean function*			
Field Name	Bits	Default	Description
CLIP_RULE	15:0	none	OpenGL Clip boolean function. The `inside` flags for each of the four clip rectangles form a 4-bit binary number. The corresponding bit in this 16-bit number specifies whether the pixel is visible.

PA:PA_SC_EDGERULE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28230			
DESCRIPTION: *Edge Rule Specification*			
Field Name	Bits	Default	Description
ER_TRI	3:0	none	Edge rule for triangles; L:R:T:B -> 1 = in, 0 = out
ER_POINT	7:4	none	Edge rule for points; L:R:T:B -> 1 = in, 0 = out
ER_RECT	11:8	none	Edge rule for rects; L:R:T:B -> 1 = in, 0 = out
ER_LINE_LR	17:12	none	Edge rule for left-right lines; TB_L:TB_R:BT_L:BT_R:HT:HB -> 1 = in, 0 = out. If PA_SC_LINE_CNTL.DX10_DIAMOND_TEST_ENA is set (only available on RV730/RV710/RV740, not available on RV770, this field needs to be set to a 0x1A
ER_LINE_RL	23:18	none	Edge rule for right-left lines; TB_L:TB_R:BT_L:BT_R:HT:HB -> 1 = in, 0 = out. If PA_SC_LINE_CNTL.DX10_DIAMOND_TEST_ENA is set (only available on RV730/RV710/RV740, not available on RV770, this field needs to be set to a 0x26
ER_LINE_TB	27:24	none	Edge rule for top-bottom lines;

			LR_L:LR_R:RL_L:RL_R -> 1 = in, 0 = out. If PA_SC_LINE_CNTL.DX10_DIAMOND_TEST_ENA is set (only available on RV730/RV710/RV740, not available on RV770, this field needs to be set to a 0xA
ER_LINE_BT	31:28	none	Edge rule for bottom-top lines; LR_L:LR_R:RL_L:RL_R -> 1 = in, 0 = out. If PA_SC_LINE_CNTL.DX10_DIAMOND_TEST_ENA is set (only available on RV730/RV710/RV740, not available on RV770, this field needs to be set to a 0xA

PA:PA_SC_GENERIC_SCISSOR_BR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28244

DESCRIPTION: *Generic Scissor rectangle specification. Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*

Field Name	Bits	Default	Description
BR_X	14:0	none	Right hand edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.
BR_Y	30:16	none	Lower edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.

PA:PA_SC_GENERIC_SCISSOR_TL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28240

DESCRIPTION: *Generic Scissor rectangle specification. Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*

Field Name	Bits	Default	Description
TL_X	14:0	none	Left hand edge of scissor rectangle. 15-bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT.
TL_Y	30:16	none	Upper edge of scissor rectangle. 15-bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT.
WINDOW_OFFSET_DISABLE	31	none	If set, generic scissor is not offset by the WINDOW_OFFSET register values.

PA:PA_SC_LINE_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28bdc

DESCRIPTION: *Line Drawing Control*

Field Name	Bits	Default	Description
EXPAND_LINE_WIDTH	9	none	If set, the line width will be expanded by the 1/cos(a) where a the minimum angle from horz or vertical. This bit most likely should be set whenever MSAA_ENABLE is set or Line Antialiasing is being done in pixel shader.
LAST_PIXEL	10	none	If set, the last pixel of a line will not be killed by the diamond exit rule.
PERPENDICULAR_ENDCAP_ENA	11	none	This functionality is only available on RV730/RV710/RV740, not available on RV770. If set, line endcaps will be perpendicular instead of axis-

			aligned.
DX10_DIAMOND_TEST_ENA	12	none	This functionality is only available on RV730/RV710/RV740, not available on RV770. If set, lines will follow DX10 line diamond conformance. When this bit is set the following fields in PA_SC_EDGERULE need to be programmed as follows: ER_LINE_LR = 0x1A; ER_LINE_RL = 0x26; ER_LINE_TB = 0xA; ER_LINE_BT = 0xA

PA:PA_SC_LINE_STIPPLE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a0c			
DESCRIPTION: *Line Stipple Control*			
Field Name	Bits	Default	Description
LINE_PATTERN	15:0	none	16-bit pattern
REPEAT_COUNT	23:16	none	Pattern bit repeat count (minus 1). Field has a valid range of 0-255 which maps to OGL api values of 1-256.
PATTERN_BIT_ORDER	28	none	Bit Ordering of Pattern Bits: 0 = Little Bit Order, 1 = Big Bit Order
AUTO_RESET_CNTL	30:29	none	Auto reset control of current pattern count/pointer. 0 = Never reset current pattern count/pointer. 1 = Reset current pattern count/pointer at each primitive (line list). 2 = Reset current pattern count/pointer at each packet (line strip).

PA:PA_SC_LINE_STIPPLE_STATE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8b10			
DESCRIPTION: *Current values for Line Stipple*			
Field Name	Bits	Default	Description
CURRENT_PTR	3:0	none	Indicates current state of pattern pointer (can be set w/ a register write).
CURRENT_COUNT	15:8	none	Current state of the repeat counter (can be set w/a register write).

PA:PA_SC_MODE_CNTL_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a48			
DESCRIPTION: *SC Mode Control Register for Various Enables*			
Field Name	Bits	Default	Description
MSAA_ENABLE	0	none	Enable MultiSample AA in DX10 and below. Used for lines in DX10.1 and above.
VPORT_SCISSOR_ENABLE	1	none	Enables viewport scissors
LINE_STIPPLE_ENABLE	2	none	Enable line stipple processing

PA:PA_SC_SCREEN_SCISSOR_BR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28034			
DESCRIPTION: *Screen Scissor rectangle specification. This scissor is NOT affected by WINDOW_OFFSET. Negative numbers clamped to 0, so reads will mismatch on negative values.*			
Field Name	Bits	Default	Description
BR_X	15:0	none	Right hand edge of scissor rectangle. 16 bits signed. Valid range -32K to 16384. Exclusive for BOTTOM_RIGHT.
BR_Y	31:16	none	Lower edge of scissor rectangle. 16 bits signed. Valid range -32K to 16384. Exclusive for BOTTOM_RIGHT.

PA:PA_SC_SCREEN_SCISSOR_TL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28030			
DESCRIPTION: *Screen Scissor rectangle specification. This scissor is NOT affected by WINDOW_OFFSET. Negative numbers clamped to 0, so reads will mismatch on negative values.*			
Field Name	Bits	Default	Description
TL_X	15:0	none	Left hand edge of scissor rectangle. 16 bits signed. Valid range -32K to 16383. Inclusive for UPPER_LEFT.
TL_Y	31:16	none	Upper edge of scissor rectangle. 16 bits signed. Valid range -32K to 16383. Inclusive for UPPER_LEFT.

PA:PA_SC_VPORT_SCISSOR_[0-15]_BR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28254-0x282cc			
DESCRIPTION: *WGF ViewportID Scissor rectangle specification(0-15). Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*			
Field Name	Bits	Default	Description
BR_X	14:0	none	Right hand edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.
BR_Y	30:16	none	Lower edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.

PA:PA_SC_VPORT_SCISSOR_[0-15]_TL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28250-0x282c8			
DESCRIPTION: *WGF ViewportId Scissor rectangle specification(0-15). Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*			
Field Name	Bits	Default	Description
TL_X	14:0	none	Left hand edge of scissor rectangle. 15-bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT.
TL_Y	30:16	none	Upper edge of scissor rectangle. 15-bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT.
WINDOW_OFFSET_DISABLE	31	none	If set, viewportId scissor is not offset by the WINDOW_OFFSET register values.

PA:PA_SC_VPORT_ZMAX_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x282d4-0x2834c			
DESCRIPTION: *Viewport Transform Z Max Clamp - 0-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_ZMAX	31:0	none	Maximum Z Value from Viewport Transform. Z values will be clamped by the DB to this value.

PA:PA_SC_VPORT_ZMIN_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x282d0-0x28348			
DESCRIPTION: *Viewport Transform Z Min Clamp - 0-15 For WGF ViewportId*			
Field Name	Bits	Default	Description
VPORT_ZMIN	31:0	none	Minimum Z Value from Viewport Transform. Z values will be clamped by the DB to this value.

PA:PA_SC_WINDOW_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28200			
DESCRIPTION: *Offset from screen coords to window coords. Vertices will be offset by these values if PA_SU_SC_MODE_CNTL.VTX_WINDOW_OFFSET_ENABLE is et. The WINDOW_SCISSOR will be offset by these values if the WINDOW_SCISSOR_TL.WINDOW_OFFSET_DISABLE is clear. If this value allows the window to extend beyond the Front Buffer (Surface) dimensions, it is expected that the SCREEN_SCISSOR is used to limit to FB surface.*			
Field Name	Bits	Default	Description
WINDOW_X_OFFSET	15:0	none	Offset in x-direction from screen to window coords. 16-bit 2`s comp signed value. Valid Range +/- 32K.
WINDOW_Y_OFFSET	31:16	none	Offset in y-direction from screen to window coords. 16-bit 2`s comp signed value. Valid Range +/- 32K.

PA:PA_SC_WINDOW_SCISSOR_BR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28208			
DESCRIPTION: *Window Scissor rectangle specification. Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*			
Field Name	Bits	Default	Description
BR_X	14:0	none	Right hand edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.
BR_Y	30:16	none	Lower edge of scissor rectangle. 15 bits unsigned. Valid range 0-16384. Exclusive for BOTTOM_RIGHT.

PA:PA_SC_WINDOW_SCISSOR_TL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28204			
DESCRIPTION: *Window Scissor rectangle specification. Scissor is conditionally (See WINDOW_OFFSET_ENABLE) offset by WINDOW_OFFSET.*			
Field Name	Bits	Default	Description
TL_X	14:0	none	Left hand edge of scissor rectangle. 15-bits unsigned. Valid range 0-16383. Inclusive for UPPER_LEFT.
TL_Y	30:16	none	Upper edge of scissor rectangle. 15-bits unsigned. Valid

			range 0-16383. Inclusive for UPPER_LEFT.
WINDOW_OFFSET_DISABLE	31	none	If set, window scissor is not offset by the WINDOW_OFFSET register values.

PA:PA_SU_CNTL_STATUS · [R] · 32 bits · Access: 32 · GpuF0MMReg:0x8a50			
DESCRIPTION: *Status Bits*			
Field Name	Bits	Default	Description
SU_BUSY	31	none	Busy Status Bit

PA:PA_SU_HARDWARE_SCREEN_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28234			
DESCRIPTION: *Hardware Screen Offset to Center Guardband*			
Field Name	Bits	Default	Description
HW_SCREEN_OFFSET_X	4:0	none	Hardware screen offset in X from 0 to 8K in units of 256 pixels.
HW_SCREEN_OFFSET_Y	12:8	none	Hardware screen offset in Y from 0 to 8K in units of 256 pixels.

PA:PA_SU_LINE_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a08			
DESCRIPTION: *Line control*			
Field Name	Bits	Default	Description
WIDTH	15:0	none	1/2 width of line, in subpixels; (16.0) fixed format.

PA:PA_SU_LINE_STIPPLE_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28824			
DESCRIPTION: *Set-Up Engine Line Stipple Control*			
Field Name	Bits	Default	Description
LINE_STIPPLE_RESET	1:0	none	line stipple reset mode: 0-no reset, 1-end of prim, 2-end of packet, 3-end of polymode line.
EXPAND_FULL_LENGTH	2	none	for antialiased line stipple, calculate stipple distance using true distance (not major).
FRACTIONAL_ACCUM	3	none	for antialiased line stipple, calculate stipple using travelled distance including fractional bits.
DIAMOND_ADJUST	4	none	for aliased line stipple, adjust stipple pattern to account for start vertex diamond exit .

PA:PA_SU_LINE_STIPPLE_SCALE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28828			
DESCRIPTION: *Set-Up Engine Line Stipple Scale Factor*			
Field Name	Bits	Default	Description
LINE_STIPPLE_SCALE	31:0	none	floating point scale factor used to derive stipple start and

			end point.

PA:PA_SU_LINE_STIPPLE_VALUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8a60			
DESCRIPTION: *Current value for Set-Up Engine Line Stipple*			
Field Name	Bits	Default	Description
LINE_STIPPLE_VALUE	23:0	none	Current value for line stipple with 8 fractional.

PA:PA_SU_POINT_MINMAX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a04			
DESCRIPTION: *Specifies maximum and minimum point & sprite sizes for per vertex size specification.*			
Field Name	Bits	Default	Description
MIN_SIZE	15:0	none	Minimum point & sprite radius size to allow. fixed point (12.4), 12 bits integer, 4 bits fractional pixels
MAX_SIZE	31:16	none	Maximum point & sprite radius size to allow. fixed point (12.4), 12 bits integer, 4 bits fractional pixels

PA:PA_SU_POINT_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28a00			
DESCRIPTION: *Dimensions for Points*			
Field Name	Bits	Default	Description
HEIGHT	15:0	none	1/2 Height (Vertical Radius) of point; fixed (12.4), 12 bits integer, 4 bits fractional pixels.
WIDTH	31:16	none	1/2 Width (Horizontal Radius)of point; fixed (12.4), 12 bits integer, 4 bits fractional pixels.

PA:PA_SU_POLY_OFFSET_BACK_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b8c			
DESCRIPTION: *Back-Facing Polygon Offset Offset*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	Specifies polygon offset offset for back-facing polygons; 32b IEEE float format.

PA:PA_SU_POLY_OFFSET_BACK_SCALE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b88			
DESCRIPTION: *Back-Facing Polygon Offset Scale*			
Field Name	Bits	Default	Description
SCALE	31:0	none	Specifies polygon offset scale for back-facing polygons; 32-bit IEEE float format.

PA:PA_SU_POLY_OFFSET_CLAMP · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b7c
DESCRIPTION: *Clamp Value for Polygon Offset*

Field Name	Bits	Default	Description
CLAMP	31:0	none	Specifies the maximum (if clamp is positive) or minimum (if clamp is negative) value clamp for the polygon offset result.

PA:PA_SU_POLY_OFFSET_DB_FMT_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b78			
DESCRIPTION: *Polygon Offset Depth Buffer Format Control*			
Field Name	Bits	Default	Description
POLY_OFFSET_NEG_NUM_DB_BITS	7:0	none	Specifies the number of bits in the depth buffer format. Specified as a negative value typically. For fixed point formats, should be number of bits (i.e. -16, -24), for float formats should be number of mantissa bits (i.e. -23). This is a signed 8b value, range -128,127
POLY_OFFSET_DB_IS_FLOAT_FMT	8	none	Specifies whether the depth buffer format is fixed or float. The NEG_NUM_DB_BITS is used differently (i.e. different POLY_OFFSET equation for fixed vs. float buffer formats.

PA:PA_SU_POLY_OFFSET_FRONT_OFFSET · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b84			
DESCRIPTION: *Front-Facing Polygon Offset Offset*			
Field Name	Bits	Default	Description
OFFSET	31:0	none	Specifies polygon offset offset for front-facing polygons; 32b IEEE float format.

PA:PA_SU_POLY_OFFSET_FRONT_SCALE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b80			
DESCRIPTION: *Front-Facing Polygon Offset Scale*			
Field Name	Bits	Default	Description
SCALE	31:0	none	Specifies polygon offset scale for front-facing polygons; 32-bit IEEE float format.

PA:PA_SU_PRIM_FILTER_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2882c			
DESCRIPTION: *Set-Up Engine Primitive Filter Control*			
Field Name	Bits	Default	Description
TRIANGLE_FILTER_DISABLE	0	none	for triangle primitive type, disable primitive filters.
LINE_FILTER_DISABLE	1	none	for line primitive type, disable primitive filters.
POINT_FILTER_DISABLE	2	none	for point primitive type, disable primitive filters.
RECTANGLE_FILTER_DISABLE	3	none	for rectangle primitive type, disable primitive filters.
TRIANGLE_EXPAND_ENA	4	none	for triangle primitive type, expand primitive bounding box for prim filters.
LINE_EXPAND_ENA	5	none	for line primitive type, expand primitive bounding box

			for prim filters.
POINT_EXPAND_ENA	6	none	for point primitive type, expand primitive bounding box for prim filters.
RECTANGLE_EXPAND_ENA	7	none	for rectangle primitive type, expand primitive bounding box for prim filters.
PRIM_EXPAND_CONSTANT	15:8	none	constant [4.4] to expand each edge of bounding box before prim filter test.

PA:PA_SU_SC_MODE_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28814			
DESCRIPTION: *SU/SC Controls for Facedness Culling, Polymode, Polygon Offset, and various Enables*			
Field Name	Bits	Default	Description
CULL_FRONT	0	none	Enable for front-face culling. POSSIBLE VALUES: 00 - Do not cull front-facing triangles. 01 - Cull front-facing triangles.
CULL_BACK	1	none	Enable for back-face culling. POSSIBLE VALUES: 00 - Do not cull back-facing triangles. 01 - Cull back-facing triangles.
FACE	2	none	X-Ored with cross product sign to determine positive facing POSSIBLE VALUES: 00 - Positive cross product is front (CCW). 01 - Negative cross product is front (CW).
POLY_MODE	4:3	none	Polygon mode enable. POSSIBLE VALUES: 00 - Disable poly mode (render triangles). 01 - Dual mode (send 2 sets of 3 polys with specified poly type). 02 - Reserved
POLYMODE_FRONT_PTYPE	7:5	none	Specifies how to render front-facing polygons. POSSIBLE VALUES: 00 - Draw points. 01 - Draw lines. 02 - Draw triangles. 03 - Reserved 3 - 7.
POLYMODE_BACK_PTYPE	10:8	none	Specifies how to render back-facing polygons. POSSIBLE VALUES: 00 - Draw points. 01 - Draw lines.

			02 - Draw triangles. 03 - Reserved 3 - 7.
POLY_OFFSET_FRONT_ENABLE	11	none	Enables front facing polygon`s offset. POSSIBLE VALUES: 00 - Disable front offset. 01 - Enable front offset.
POLY_OFFSET_BACK_ENABLE	12	none	Enables back facing polygon`s offset. POSSIBLE VALUES: 00 - Disable back offset. 01 - Enable back offset.
POLY_OFFSET_PARA_ENABLE	13	none	Enables polygon offset for non-triangle primitives. POSSIBLE VALUES: 00 - Disable front offset for parallelograms. 01 - Enable front offset for parallelograms.
VTX_WINDOW_OFFSET_ENABLE	16	none	Enables addition of PA_SC_WINDOW_OFFSET values to vertex data.
PROVOKING_VTX_LAST	19	none	Defines which vertex of a primitive is used for attribute components when flat shading is enabled POSSIBLE VALUES: 00 - 0 = First Vtx (D3D) 01 - 1 = Last Vtx (OGL)
PERSP_CORR_DIS	20	none	Disables perspective correction for all attributes
MULTI_PRIM_IB_ENA	21	none	Enables multiple primitive sets to be placed in a single index buffer, separated by RESET_INDX indices

PA:PA_SU_VTX_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28be4

DESCRIPTION: *Miscellaneous SU Control*

Field Name	Bits	Default	Description
PIX_CENTER	0	none	Specifies where the pixel center of the incoming vertex is. The drawing engine itself has pixel centers @ 0.5, so if this bit is `0`, 0.5 will be added to the X,Y coordinates to move the incoming vertex onto our internal grid. POSSIBLE VALUES: 00 - 0 = Pixel Center @ 0.0 (D3D) 01 - 1 = Pixel Center @ 0.5 (OGL)
ROUND_MODE	2:1	none	Controls conversion of X,Y coordinates from IEEE to fixed-point POSSIBLE VALUES: 00 - 0 = Truncate (OGL) 01 - 1 = Round

			02 - 2 = Round to Even (D3D) 03 - 3 = Round to Odd
QUANT_MODE	5:3	none	Controls conversion of X,Y coordinates from IEEE to fixed-point POSSIBLE VALUES: 00 - 0 = 1/16th (4 fractional bits) 01 - 1 = 1/8th (3 fractional bits) 02 - 2 = 1/4th (2 fractional bits) 03 - 3 = 1/2 (1 fractional bit) 04 - 4 = 1 (0 fractional bits) 05 - 5 = 1/256th (8 fractional bits) 06 - 6 = 1/1024th (10 fractional bits) 07 - 7 = 1/4096th (12 fractional bits)

3. General Shader Registers

SQ:SQ_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c00			
DESCRIPTION: *(1-state) SQ config options. The graphics pipe must be idle to change these.*			
Field Name	Bits	Default	Description
VC_ENABLE	0	0x0	Vertex Cache (VC) is present; set to zero to disable VC. When VC is disabled, all vertex fetches go through the TC rather than VC regardless of the instruction bit which selects TC/VC, and also disables global data sharing.
EXPORT_SRC_C	1	0x1	Export mode: 0=do not allow tex/vtx fetch to use src-c export; 1 = allow opportunistic src-c tex/vtx exports.
GFX_PRIO	3:2	0x0	A value between 0 and 3 indicating the relative priority of waves from the GFX ring. A higher value gives higher priority.
CS1_PRIO	5:4	0x0	A value between 0 and 3 indicating the relative priority of waves from the CS1 ring. A higher value gives higher priority.
CS2_PRIO	7:6	0x0	A value between 0 and 3 indicating the relative priority of waves from the CS2 ring. A higher value gives higher priority.

SQ:SQ_ESGS_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c40			
DESCRIPTION: *(1-state) Memory base address of the ES->GS ring buffer (256-byte aligned)*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_ESGS_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28900			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the ES->GS ring buffer (in DWORDs). Itemsize is the true count, not count-1 and represents [0..255] dwords.*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_ESGS_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c44			
DESCRIPTION: *(1-state) Memory region size address of the ES->GS ring buffer (in units of 256-bytes). For multi-SE chips, the buffer size should be a multiple of 512-bytes because it is automatically split by hardware. True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_ESTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c50			
DESCRIPTION: *(1-state) Memory base address of the ES Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base.*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_ESTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28908			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the ES Temp buffer (in DWORDs).*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_ESTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c54			
DESCRIPTION: *(1-state) Memory region size address of the ES Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_EX_ALLOC_TABLE_SLOTS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8e48			
DESCRIPTION: *Maximum number of export allocation table entries the SQ will use.*			
Field Name	Bits	Default	Description
PIX_SLOTS	6:0	0x20	Color buffer export.
POS_SLOTS	14:8	0x10	Position export.
SMX_SLOTS	22:16	0x20	Memory export and RAT.

SQ:SQ_GLOBAL_GPR_RESOURCE_MGMT_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c10			
DESCRIPTION: *Partition global gprs per shader stage. Field values are addresses, not counts and may overlap*			
Field Name	Bits	Default	Description
PS_GGPR_BASE	7:0	0x0	PS global gpr base address
VS_GGPR_BASE	15:8	0x0	VS global gpr base address
GS_GGPR_BASE	23:16	0x0	GS global gpr base address
ES_GGPR_BASE	31:24	0x0	ES global gpr base address

SQ:SQ_GLOBAL_GPR_RESOURCE_MGMT_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c14			
DESCRIPTION: *Partition global gprs per shader stage. Field values are addresses, not counts and may overlap*			
Field Name	Bits	Default	Description
HS_GGPR_BASE	7:0	0x0	HS global gpr base address
LS_GGPR_BASE	15:8	0x0	LS global gpr base address

CS_GGPR_BASE	23:16	0x0	CS global gpr base address

SQ:SQ_GPR_RESOURCE_MGMT_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c04			
DESCRIPTION: *(1-state) Defines how GPR space is divided among the shader stages. All ES, VS, and GS work must be flushed before writing this register. PS work must also be flushed prior to changing NUM_PS_GPRS or NUM_CLAUSE_TEMP_GPRS.*			
Field Name	Bits	Default	Description
NUM_PS_GPRS	7:0	0x0	Number of GPRs (per SIMD) assigned to the PS stage [0..255].
NUM_VS_GPRS	23:16	0x0	Number of GPRs (per SIMD) assigned to the VS stage [0..255].
NUM_CLAUSE_TEMP_GPRS	31:28	0x0	Number of GPRs reserved for clause temporaries [0-15]. This is the number of GPRs available to a single thread, so the hardware will reserve twice this many physical registers (for even & odd clauses).

SQ:SQ_GSTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c58			
DESCRIPTION: *(1-state) Memory base address of the GS Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base.*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_GSTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2890c			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the GS Temp buffer (in DWORDs).*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_GSTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c5c			
DESCRIPTION: *(1-state) Memory region size address of the GS Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_GSVS_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c48			
DESCRIPTION: *(1-state) Memory base address of the GS->ES ring buffer (256-byte aligned)*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_GSVS_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28904			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the GS->ES ring buffer (in DWORDs). This defines the max number of dwords a single invocation of the GS can output to the ring buffer: [0..16363] dwords.*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_GSVS_RING_OFFSET_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2892c			
DESCRIPTION: *Offset for Stream1 from SQ_GSVS_RING_BASE*			
Field Name	Bits	Default	Description
OFFSET	14:0	0x0	In Dwords. Maximum output of one prim for stream_0 1024 * 32 dwords = 32k

SQ:SQ_GSVS_RING_OFFSET_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28930			
DESCRIPTION: *Offset for Stream2 from SQ_GSVS_RING_BASE*			
Field Name	Bits	Default	Description
OFFSET	14:0	0x0	In Dwords. Maximum output of one prim for stream_0 + Maximum output of one prim for stream_1 2*1024 * 32 dwords = 64k

SQ:SQ_GSVS_RING_OFFSET_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28934			
DESCRIPTION: *Offset for Stream3 from SQ_GSVS_RING_BASE*			
Field Name	Bits	Default	Description
OFFSET	14:0	0x0	In Dwords. Maximum output of one prim for stream_0 + Maximum output of one prim for stream_1 + Maximum output of one prim for stream_2 2*1024 * 32 dwords = 96k

SQ:SQ_GSVS_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c4c			
DESCRIPTION: *(1-state) Memory region size address of the GS->ES ring buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_GS_VERT_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2891c			
DESCRIPTION: *(8-state) Space allocated to a single GS output vertex in GS Temp Buffer. This defines the size of a single vertex output by the GS. Multiple vertices can be output so long as the total output size does not exceed SQ_GSVS_RING_ITEMSIZE.*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_GS_VERT_ITEMSIZE_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28920			
DESCRIPTION: *Offset for vert itemsize*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	This defines the size in dwords of a single vertex output by the GS to stream 1. (used by SX, not SQ)

SQ:SQ_GS_VERT_ITEMSIZE_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28924			
DESCRIPTION: *Offset for vert itemsize*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	This defines the size in dwords of a single vertex output by the GS to stream 2. (used by SX, not SQ)

SQ:SQ_GS_VERT_ITEMSIZE_3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28928			
DESCRIPTION: *Offset for vert itemsize*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	This defines the size in dwords of a single vertex output by the GS to stream 3. (used by SX, not SQ)

SQ:SQ_HSTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8e18			
DESCRIPTION: *(1-state) Memory base address of the HS Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base.*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_HSTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28834			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the HS Temp buffer (in DWORDs).*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_HSTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8e1c			
DESCRIPTION: *(1-state) Memory region size address of the HS Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_LSTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8e10			
DESCRIPTION: *(1-state) Memory base address of the LS/CS Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base. A separate value is stored at this address for each CP ring.*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_LSTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28830			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the LS/CS Temp buffer (in DWORDs).*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_LSTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8e14			
DESCRIPTION: *(1-state) Memory region size address of the LS/CS Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_PSTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c68			
DESCRIPTION: *(1-state) Memory base address of the PS Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base.*			
Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_PSTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28914			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the PS Temp buffer (in DWORDs)*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_PSTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c6c			
DESCRIPTION: *(1-state) Memory region size address of the PS Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_VSTMP_RING_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c60
DESCRIPTION: *(1-state) Memory base address of the VS Temp buffer (256-byte aligned). On multi-SE chips, each SE needs to have an unique ring base.*

Field Name	Bits	Default	Description
MEM_BASE	31:0	0x0	Format is [39:8]

SQ:SQ_VSTMP_RING_ITEMSIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28910			
DESCRIPTION: *(8-state) Space allocated to a single pixel/vertex in the VS Temp buffer (in DWORDs)*			
Field Name	Bits	Default	Description
ITEMSIZE	14:0	0x0	Format is [16:2]

SQ:SQ_VSTMP_RING_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8c64			
DESCRIPTION: *(1-state) Memory region size address of the VS Temp buffer (in units of 256-bytes). True size, not size -1. Setting to zero disables.*			
Field Name	Bits	Default	Description
MEM_SIZE	31:0	0x0	Format is [39:8]

SQ:SQ_VTX_BASE_VTX_LOC · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3cff0			
DESCRIPTION: *(64-state) Vertex fetch base location. can be used as an index offset for vertex fetch. one entry per state (up to 64 states).*			
Field Name	Bits	Default	Description
OFFSET	31:0	0x0	Vertex Base location for vertex fetching

SQ:SQ_VTX_SEMANTIC_[0-31] · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x28380-0x283fc			
DESCRIPTION: *(8-state) Vertex Fetch Semantic Name. Used for semantic-based vertex fetches. 32 entries provided (8 states). The address in which the semantic occurs dictates which GPR the named element goes to in the vertex shader. Note that the hardware does not interpret this value, other than simply compare these 8 bits versus the 8-bit semantic in the vertex fetch instruction. These registers are readable/writable.*			
Field Name	Bits	Default	Description
SEMANTIC_ID	7:0	0x0	8-bit semantic id

SQ:SQ_VTX_SEMANTIC_CLEAR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x288f0			
DESCRIPTION: *(8-state) This register is used to clear the contents of the vertex semantic table. Entries can be cleared independently -- each has one bit in this register to clear or leave alone. This register is write-only (not readable).*			
Field Name	Bits	Default	Description
CLEAR	31:0	0x0	clear or preserve table entry

SQ:SQ_VTX_START_INST_LOC · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3cff4
DESCRIPTION: *(64-state) Vertex fetch instance offset. can be used as an index offset for vertex fetch. one entry per state (up to 64 states, but probably less than base_vtx_loc).*

Field Name	Bits	Default	Description
OFFSET	31:0	0x0	Instance start location for vertex fetching

4. Shader Instructions

SQ_MICRO:SQ_CF_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 0. This word is the default representation for CF instructions.*			
Field Name	Bits	Default	Description
ADDR	23:0	none	Bits [26:3] of the byte offset (producing a QUAD-word-aligned value) of the clause to execute (clause instructions only). Bits [26:3] of the byte offset (producing a QUAD-word-aligned value) of the control flow address to jump to (instructions that can jump). Offsets are relative to the byte address specified by PGM_START. Texture and Vertex clauses must start on 16-byte aligned addresses.
JUMPTABLE_SEL	26:24	none	Select the source of the offset used for CF_INST_JUMPTABLE instructions. Not meaningful for other instructions. POSSIBLE VALUES: 00 - SQ_CF_JUMPTABLE_SEL_CONST_A: use element A of jumptable constant selected by CF_CONST. 01 - SQ_CF_JUMPTABLE_SEL_CONST_B: use element B of jumptable constant selected by CF_CONST. 02 - SQ_CF_JUMPTABLE_SEL_CONST_C: use element C of jumptable constant selected by CF_CONST. 03 - SQ_CF_JUMPTABLE_SEL_CONST_D: use element D of jumptable constant selected by CF_CONST. 04 - SQ_CF_JUMPTABLE_SEL_INDEX_0: use index0. 05 - SQ_CF_JUMPTABLE_SEL_INDEX_1: use index1.

SQ_MICRO:SQ_CF_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 1. This word is the default representation for CF instructions.*			
Field Name	Bits	Default	Description
POP_COUNT	2:0	none	Specify the number of entries to pop from the stack, in [0..7]. Only used by certain CF instructions that pop the branch-loop stack. May be zero, to indicate no pop operation.
CF_CONST	7:3	none	Specify the CF constant to use for flow control statements. For LOOP/ENDLOOP, this specifies the integer constant to use for the loop counter, loop index initializer, and increment. For instructions using COND,

			this specifies the index of the boolean constant to use.
COND	9:8	none	Specifies how to evaluate the condition test for each pixel. Not used by all instructions. May reference CF_CONST. POSSIBLE VALUES: 00 - SQ_CF_COND_ACTIVE: condition test passes for active pixels. 01 - SQ_CF_COND_FALSE: condition test fails for all pixels. 02 - SQ_CF_COND_BOOL: condition test passes iff pixel is active and boolean referenced by CF_CONST is true. 03 - SQ_CF_COND_NOT_BOOL: condition test passes iff pixel is active and boolean referenced by CF_CONST is false.
COUNT	15:10	none	Number of instructions to execute in the clause, minus one (clause instructions only). This is interpreted as the number of instruction slots in the range [1,16]. For CALL instruction: Amount to increment call nesting counter by when executing a CALL statement; a CALL is skipped if the current nesting depth + call_count > 32. This field is interpreted in the range [0,31]. For EMIT,CUT,EMIT-CUT, bits[1:0] are the stream-id. For SET_PRIORITY, this is the new priority value.
VALID_PIXEL_MODE	20	none	If set, execute this instruction/clause as if invalid pixels are inactive. Caution: VALID_PIXEL_MODE is not the `default` mode; this bit should be set to 0 by default. Set this only in PS stage.
CF_INST	29:22	none	Type of instruction to evaluate in CF. For this encoding, CF_INST must be set to one of the following values. POSSIBLE VALUES: 00 - SQ_CF_INST_NOP: perform no operation. 01 - SQ_CF_INST_TC: execute fetch clause, through the texture cache. CF_COND=ACTIVE is required. 02 - SQ_CF_INST_VC: execute vertex fetch clause, through the vertex-cache (if exists; else sent to texture-cache). CF_COND=ACTIVE is required. 03 - SQ_CF_INST_GDS: execute a Global Data Share clause (GDS, TF (tesselation factor)). CF_COND=ACTIVE is required. 04 - SQ_CF_INST_LOOP_START: execute DX9 loop start instruction (push onto loop stack if loop body executes). 05 - SQ_CF_INST_LOOP_END: execute DX9 loop end instruction (pop loop stack if loop is finished). 06 - SQ_CF_INST_LOOP_START_DX10: execute DX10 loop start instruction (push onto loop stack if loop body executes). 07 - SQ_CF_INST_LOOP_START_NO_AL: same

			as LOOP_START but don`t push AL onto stack or update AL. 08 - SQ_CF_INST_LOOP_CONTINUE: execute continue statement (jump to end of loop if all pixels ready to continue). 09 - SQ_CF_INST_LOOP_BREAK: execute a break statement (pop loop stack if all pixels ready to break). 10 - SQ_CF_INST_JUMP: execute jump statement (may be conditional). 11 - SQ_CF_INST_PUSH: push current per-pixel active state onto stack OR jump and pop if no items would be active. 13 - SQ_CF_INST_ELSE: execute else statement (may be conditional) OR jump if no items would be active. 14 - SQ_CF_INST_POP: pop current per-pixel state from the stack. jump if no pixels are enabled after pop. 18 - SQ_CF_INST_CALL: execute subroutine call instruction (push onto address stack). 19 - SQ_CF_INST_CALL_FS: call fetch shader. The address to call is stored in a state register in SQ. 20 - SQ_CF_INST_RETURN: execute subroutine return instruction (pop address stack). Pair with CF_INST_CALL only. 21 - SQ_CF_INST_EMIT_VERTEX: signal that GS has finished exporting a vertex to memory. CF_COND=ACTIVE is required. Stream-ID is in the 2 lsb`s of the COUNT field. 22 - SQ_CF_INST_EMIT_CUT_VERTEX: emit a vertex and an end of primitive strip marker. The next emitted vertex will start a new primitive strip. CF_COND=ACTIVE is required. Stream-ID is in the 2 lsb`s of the COUNT field. 23 - SQ_CF_INST_CUT_VERTEX: emit an end of primitive strip marker. The next emitted vertex will start a new primitive strip. CF_COND=ACTIVE is required. Stream-ID is in the 2 lsb`s of the COUNT field. 24 - SQ_CF_INST_KILL: kill pixels that pass the condition test (may be conditional). No jump is taken. 26 - SQ_CF_INST_WAIT_ACK: wait for write-acks or fetch-read-acks to return before proceeding. wait if outstanding_acks > (value of ADDR field). When using a nonzero value, be aware that TC_ACK requests may return out of order with respect to VC_ACK requests. For optimal performance, BARRIER_BEFORE should never be set for this instruction. 27 - SQ_CF_INST_TC_ACK: execute fetch or constant fetch clause, with ack. CF_COND=ACTIVE is required. All previous TC/VC/GDS requests must have completed if this instruction is issued without BARRIER_BEFORE set. 28 - SQ_CF_INST_VC_ACK: execute vertex fetch clause through the vertex cache (if it exists; else uses

			texture cache), with ack. CF_COND=ACTIVE is required. All previous TC/VC/GDS requests must have completed if this instruction is issued without BARRIER_BEFORE set. 29 - SQ_CF_INST_JUMPTABLE: execute a jump through a jump table. This instruction is followed by a series of up to 256 jump instructions forming the jump table. The index into the table comes from either a loop-constant, or a GPR via the index registers. The instruction AFTER the last jump table entry should be indicated by the ADDR field. If no pixels are enabled after the condition test, then execution will continue at this address. 30 - SQ_CF_INST_GLOBAL_WAVE_SYNC: synchronize waves across the chip (including multiple simds and multiple shader engines. Details TBD. 31 - SQ_CF_INST_HALT: halt this thread executions. The only way to restart execution is to write this instruction using SQ_WAVE0_CF_INST[01] once the hardware is otherwise idle. 32 - SQ_CF_INST_END: End the shader. This is the only bit for which PROGRAM_END can be set (although PROGRAM_END will be deprecated shortly) 33 - SQ_CF_INST_LDS_DEALLOC: Deallocate the LDS resources. 34 - SQ_CF_INST_PUSH_WQM: push current per-pixel active state onto stack and apply whole quad mode OR jump if no items would be active. 35 - SQ_CF_INST_POP_WQM: pop per-pixel state added to the stack using PUSH_WQM. POP_COUNT must be 1. Jump if no pixels are enabled after pop. 36 - SQ_CF_INST_ELSE_WQM: execute else statement (may be conditional) OR jump if no items would be active. 37 - SQ_CF_INST_JUMP_ANY: execute jump statement (may be conditional). 38 - SQ_CF_INST_REACTIVATE: Re-activate all valid threads on the top of the current branch stack. Do not do this inside a loop or wqm frame. 39 - SQ_CF_INST_REACTIVATE_WQM: Re-activate all valid quads on the top of the current branch stack. Do not do this inside a loop or wqm frame. Pixel shaders only. 40 - SQ_CF_INST_INTERRUPT: Trigger an interrupt. The interrupt data will include CF_IDX0, CF_IDX1, the VMID, the CP RINGID and the shader engine id. 41 - SQ_CF_INST_INTERRUPT_AND_SLEEP: Trigger an interrupt and sleep. The interrupt data will include CF_IDX0, CF_IDX1, the VMID, the CP RINGID and the shader engine id 42 - SQ_CF_INST_SET_PRIORITY: Set the priority to the count value.

BARRIER	31	none	If set, all prior CF instructions/clauses must complete before this instruction/clause executes. If not set, this instruction/clause may run in parallel with prior instructions.

SQ_MICRO:SQ_CF_ENCODING_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 1. This is a READ-ONLY register used to help decode the CF instruction.*			
Field Name	Bits	Default	Description
ENCODING	29:28	none	READ-ONLY field used to determine the encoding used. POSSIBLE VALUES: 00 - SQ_CF_ENCODING_INST_CF: `b00: use SQ_CF_WORD encoding. 01 - SQ_CF_ENCODING_INST_ALLOC_EXPORT: `b01: use SQ_CF_ALLOC_EXPORT_WORD encoding. 02 - SQ_CF_ENCODING_INST_ALU0: `b10: use SQ_CF_ALU_WORD encoding. 03 - SQ_CF_ENCODING_INST_ALU1: `b11: use SQ_CF_ALU_WORD encoding.

SQ_MICRO:SQ_CF_ALU_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 0. This word is used by ALU clause instructions.*			
Field Name	Bits	Default	Description
ADDR	21:0	none	Bits [24:3] of the byte offset (producing a QUAD-word-aligned value) of the clause to execute. The offset is relative to the byte address specified by PGM_START.
KCACHE_BANK0	25:22	none	Bank (constant buffer number) for first set of locked cache lines.
KCACHE_BANK1	29:26	none	Bank (constant buffer number) for second set of locked cache lines.
KCACHE_MODE0	31:30	none	Mode for first set of locked cache lines. POSSIBLE VALUES: 00 - SQ_CF_KCACHE_NOP: do not lock any cache lines. 01 - SQ_CF_KCACHE_LOCK_1: lock cache line [bank][addr]. 02 - SQ_CF_KCACHE_LOCK_2: lock cache lines [bank][addr] and [bank][addr+1]. 03 - SQ_CF_KCACHE_LOCK_LOOP_INDEX: lock cache lines [bank][loop/16+addr] and [bank][loop/16+addr+1], where loop is current loop index.

SQ_MICRO:SQ_CF_ALU_WORD0_EXT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 0. This word is used by extended ALU clause instructions.*			
Field Name	Bits	Default	Description
KCACHE_BANK_INDEX_MODE0	5:4	none	Bank relative offset select. Add the indicated offset to the constant buffer bank number in KCACHE_BANK0. Indexed locks of banks 14 and 15 will be ignored. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
KCACHE_BANK_INDEX_MODE1	7:6	none	Bank relative offset select. Add the indicated offset to the constant buffer bank number in KCACHE_BANK1. Indexed locks of banks 14 and 15 will be ignored. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
KCACHE_BANK_INDEX_MODE2	9:8	none	Bank relative offset select. Add the indicated offset to the constant buffer bank number in KCACHE_BANK2. Indexed locks of banks 14 and 15 will be ignored. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
KCACHE_BANK_INDEX_MODE3	11:10	none	Bank relative offset select. Add the indicated offset to the constant buffer bank number in KCACHE_BANK3. Indexed locks of banks 14 and 15 will be ignored. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number

			03 - SQ_CF_INVALID: invalid
KCACHE_BANK2	25:22	none	Bank (constant buffer number) for third set of locked cache lines.
KCACHE_BANK3	29:26	none	Bank (constant buffer number) for fourth set of locked cache lines.
KCACHE_MODE2	31:30	none	Mode for third set of locked cache lines. POSSIBLE VALUES: 00 - SQ_CF_KCACHE_NOP: do not lock any cache lines. 01 - SQ_CF_KCACHE_LOCK_1: lock cache line [bank][addr]. 02 - SQ_CF_KCACHE_LOCK_2: lock cache lines [bank][addr] and [bank][addr+1]. 03 - SQ_CF_KCACHE_LOCK_LOOP_INDEX: lock cache lines [bank][loop/16+addr] and [bank][loop/16+addr+1], where loop is current loop index.

SQ_MICRO:SQ_CF_ALU_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Control flow instruction word 1. This word is used by ALU clause instructions.*			
Field Name	Bits	Default	Description
KCACHE_MODE1	1:0	none	Mode for second set of locked cache lines. POSSIBLE VALUES: 00 - SQ_CF_KCACHE_NOP: do not lock any cache lines. 01 - SQ_CF_KCACHE_LOCK_1: lock cache line [bank][addr]. 02 - SQ_CF_KCACHE_LOCK_2: lock cache lines [bank][addr] and [bank][addr+1]. 03 - SQ_CF_KCACHE_LOCK_LOOP_INDEX: lock cache lines [bank][loop/16+addr] and [bank][loop/16+addr+1], where loop is current loop index.
KCACHE_ADDR0	9:2	none	Constant buffer address for first set of locked cache lines. In units of cache lines where a line holds 16 128-bit constants (byte addr[15:8]).
KCACHE_ADDR1	17:10	none	Constant buffer address for second set of locked cache lines
COUNT	24:18	none	Number of instructions to execute in the clause, minus one. This is interpreted as the number of instruction slots (64-bit slots) in the range [1,128].
ALT_CONST	25	none	if set, uses constants from alternate thread type: ps->vs, vs->gs, gs->vs, es->gs (note that es and vs share constants). Has no effect on HS, LS, CS.
CF_INST	29:26	none	Type of ALU instruction to evaluate in CF. For this

			encoding, CF_INST must be set to one of the following values. POSSIBLE VALUES: 08 - SQ_CF_INST_ALU: each PRED_SET updates the active state but does not update the stack. 09 - SQ_CF_INST_ALU_PUSH_BEFORE: do CF_PUSH; then CF_INST_ALU 10 - SQ_CF_INST_ALU_POP_AFTER: do CF_INST_ALU; then do CF_INST_POP. 11 - SQ_CF_INST_ALU_POP2_AFTER: do CF_INST_ALU; then do CF_INST_POP twice. 12 - SQ_CF_INST_ALU_EXTENDED: ALU clause instruction extension - used for indexed constant buffers and 4 constant buffers per clause. This CF is the first half of the alu instruction pair. Defines constant buffer 2 and 3, and index-select for all 4 constant buffers 13 - SQ_CF_INST_ALU_CONTINUE: each PRED_SET causes a continue operation on the masked pixels. Equivalent to CF_INST_PUSH; CF_INST_ALU; CF_INST_ELSE; CF_INST_CONTINUE; CF_POP. 14 - SQ_CF_INST_ALU_BREAK: each PRED_SET causes a break operation on the masked pixels. Equivalent to CF_INST_PUSH; CF_INST_ALU; CF_INST_ELSE; CF_INST_CONTINUE; CF_POP. 15 - SQ_CF_INST_ALU_ELSE_AFTER: do CF_INST_ALU; then do CF_INST_ELSE.
WHOLE_QUAD_MODE	30	none	Indicates that all pixels in a quad containing an active, valid pixel should be considered active for this instruction. Set this only in PS stage.
BARRIER	31	none	If set, all prior CF instructions/clauses must complete before this instruction/clause executes. If not set, this instruction/clause may run in parallel with prior instructions.

SQ_MICRO:SQ_CF_ALU_WORD1_EXT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Control flow instruction word 1. This word is used by extended ALU clause instructions.*

Field Name	Bits	Default	Description
KCACHE_MODE3	1:0	none	Mode for fourth set of locked cache lines. POSSIBLE VALUES: 00 - SQ_CF_KCACHE_NOP: do not lock any cache lines. 01 - SQ_CF_KCACHE_LOCK_1: lock cache line [bank][addr]. 02 - SQ_CF_KCACHE_LOCK_2: lock cache lines [bank][addr] and [bank][addr+1]. 03 - SQ_CF_KCACHE_LOCK_LOOP_INDEX: lock cache lines [bank][loop/16+addr] and

			[bank][loop/16+addr+1], where loop is current loop index.
KCACHE_ADDR2	9:2	none	Constant buffer address for third set of locked cache lines
KCACHE_ADDR3	17:10	none	Constant buffer address for fourth set of locked cache lines
CF_INST	29:26	none	Type of ALU instruction to evaluate in CF. Must be SQ_CF_INST_ALU_EXTENDED POSSIBLE VALUES: 08 - SQ_CF_INST_ALU: each PRED_SET updates the active state but does not update the stack. 09 - SQ_CF_INST_ALU_PUSH_BEFORE: do CF_PUSH; then CF_INST_ALU 10 - SQ_CF_INST_ALU_POP_AFTER: do CF_INST_ALU; then do CF_INST_POP. 11 - SQ_CF_INST_ALU_POP2_AFTER: do CF_INST_ALU; then do CF_INST_POP twice. 12 - SQ_CF_INST_ALU_EXTENDED: ALU clause instruction extension - used for indexed constant buffers and 4 constant buffers per clause. This CF is the first half of the alu instruction pair. Defines constant buffer 2 and 3, and index-select for all 4 constant buffers 13 - SQ_CF_INST_ALU_CONTINUE: each PRED_SET causes a continue operation on the masked pixels. Equivalent to CF_INST_PUSH; CF_INST_ALU; CF_INST_ELSE; CF_INST_CONTINUE; CF_POP. 14 - SQ_CF_INST_ALU_BREAK: each PRED_SET causes a break operation on the masked pixels. Equivalent to CF_INST_PUSH; CF_INST_ALU; CF_INST_ELSE; CF_INST_CONTINUE; CF_POP. 15 - SQ_CF_INST_ALU_ELSE_AFTER: do CF_INST_ALU; then do CF_INST_ELSE.
BARRIER	31	none	If set, all prior CF instructions/clauses must complete before this instruction/clause executes. If not set, this instruction/clause may run in parallel with prior instructions.

SQ_MICRO:SQ_CF_ALLOC_EXPORT_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Word 0 of the control flow instruction for alloc/export.*			
Field Name	Bits	Default	Description
ARRAY_BASE	12:0	none	For scratch input/output, this is the base address of the array in multiples of 4 dwords [0,32764]. For stream/ring output, this is the base addess of the array in multiples of 1 dword [0,8191]. For pixel/z output, this is the index of the first export (framebuffer 0..7; computed Z: 61).

			For parameter output, this is the parameter index of the first export [0,31]. For position output, this is the position index of the first export [60,63].
TYPE	14:13	none	Type of allocation/export. In the table below, the first enumeration value listed (PIXEL, POS, PARAM) is used with CF_INST_EXPORT*. The second enumeration value listed (WRITE, WRITE_IND, WRITE_ACK, WRITE_IND_ACK) is used with CF_INST_MEM*. <u>POSSIBLE VALUES:</u> 00 - SQ_EXPORT_PIXEL: write pixel. SQ_EXPORT_WRITE: write to memory buffer. 01 - SQ_EXPORT_POS: write position. SQ_EXPORT_WRITE_IND: write to memory buffer, use offset in INDEX_GPR. 02 - SQ_EXPORT_PARAM: write parameter cache. SQ_EXPORT_WRITE_ACK: write to memory buffer, request an ACK when write is committed to memory. For RAT, ack guarantees return value has been written to memory. Warning: acknowledges for cached-RAT may return out-of-order with respect to other exports (including cacheless-RAT instructions). 03 - Unused for SX exports. SQ_EXPORT_WRITE_IND_ACK: write to memory buffer with offset in INDEX_GPR, get an ACK when done. For RAT, ack guarantees return value has been written to memory. Warning: acknowledges for cached-RAT instructions may return out-of-order with respect to other exports ((including cacheless-RAT instructions).
RW_GPR	21:15	none	GPR register read data from
RW_REL	22	none	Indicates whether GPR is an absolute address, or relative to the loop index. <u>POSSIBLE VALUES:</u> 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
INDEX_GPR	29:23	none	For any indexed export, this GPR contains an index that will be used in the computation for determining the address of the first export. The index is multipled by (ELEM_SIZE + 1). Only the X component is used (other components ignored, no swizzle allowed).
ELEM_SIZE	31:30	none	Number of DWORDs per element, minus one. This field is interpreted as a value in [1,2,4] (3 not supported). The value from INDEX_GPR and the loop counter are multiplied by this factor, if applicable. Also, BURST_COUNT is multiplied by this factor for CF_INST_MEM*. This field is ignored for CF_INST_EXPORT*. Normally, ELEMSIZE = 4

			DWORDs for scratch, one DWORD for other types.

SQ_MICRO:SQ_CF_ALLOC_EXPORT_WORD0_RAT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Word 0 of the control flow instruction for alloc/export for RAT mem exports. For all RAT_INSTs except STORE, the resource format must be UINT32 (STORE can take any format). For all RAT_INST the channel mask must be 0xF (all channels written).*			
Field Name	Bits	Default	Description
RAT_ID	3:0	none	RAT surface (MRT) ID
RAT_INST	9:4	none	RAT instruction POSSIBLE VALUES: 00 - SQ_EXPORT_RAT_INST_NOP: Internal use by SX only (flush+ack with no opcode). 01 - SQ_EXPORT_RAT_INST_STORE_TYPED: dst = src - replace with format conversion (any resource format allowed). This is the only cached RAT opcode that may write more than 1 dword (up to 4). 02 - SQ_EXPORT_RAT_INST_STORE_RAW: dst = src (no flushing of denorms). This is the only permitted opcode for CACHELESS RAT targets, and may write up to 2 dwords (from the X and Y channels) for CACHELESS targets, but only 1 dword for cached targets 03 - SQ_EXPORT_RAT_INST_STORE_RAW_FDENORM: dst = src (flush denorms to zero). 04 - SQ_EXPORT_RAT_INST_CMPXCHG_INT: dst = (cmp == dst) ? src : dst. simple bitwise compare. 05 - SQ_EXPORT_RAT_INST_CMPXCHG_FLT: dst = (cmp == dst) ? src : dst. floating point compare with denorms. 06 - SQ_EXPORT_RAT_INST_CMPXCHG_FDENORM: dst = (cmp == dst) ? src : dst. flush denorms to zero, float compare. 07 - SQ_EXPORT_RAT_INST_ADD: dst = src + dst. non-saturating integer add. 08 - SQ_EXPORT_RAT_INST_SUB: dst = dst - src. non-saturating integer sub. 09 - SQ_EXPORT_RAT_INST_RSUB: dst = src - dst. non-saturating reverse subtract. 10 - SQ_EXPORT_RAT_INST_MIN_INT: dst = (src < dst) ? src : dst. signed. 11 - SQ_EXPORT_RAT_INST_MIN_UINT: dst = (src < dst) ? src : dst. unsigned. 12 - SQ_EXPORT_RAT_INST_MAX_INT: dst = (src > dst) ? src : dst. signed 13 - SQ_EXPORT_RAT_INST_MAX_UINT: dst = (src > dst) ? src : dst. unsigned 14 - SQ_EXPORT_RAT_INST_AND: dst = dst & src.

			bitwise 15 - SQ_EXPORT_RAT_INST_OR: dst = dst | src. bitwise 16 - SQ_EXPORT_RAT_INST_XOR: dst = dst ^ src. bitwise 17 - SQ_EXPORT_RAT_INST_MSKOR: dst = (dst & ~mask) | src 18 - SQ_EXPORT_RAT_INST_INC_UINT: dst = (dst >= src) ? 0 : dst+1 19 - SQ_EXPORT_RAT_INST_DEC_UINT: dst = ((dst==0 | (dst > src)) ? src : dst-1 32 - SQ_EXPORT_RAT_INST_NOP_RTN: Internal use by SX only (flush+ack with no opcode). return dword. 34 - SQ_EXPORT_RAT_INST_XCHG_RTN: dst = src (no flushing of denorms). return dword. 35 - SQ_EXPORT_RAT_INST_XCHG_FDENORM_RTN: dst = src (flush denorms to zero). return float. return float. 36 - SQ_EXPORT_RAT_INST_CMPXCHG_INT_RTN: dst = (cmp == dst) ? src : dst. simple bitwise compare. return dword. 37 - SQ_EXPORT_RAT_INST_CMPXCHG_FLT_RTN: dst = (cmp == dst) ? src : dst. floating point compare with denorms. return float. 38 - SQ_EXPORT_RAT_INST_CMPXCHG_FDENORM_RTN: dst = (cmp == dst) ? src : dst. flush denorms to zero, float compare. return float. 39 - SQ_EXPORT_RAT_INST_ADD_RTN: dst = src + dst. non-saturating integer add. return dword. 40 - SQ_EXPORT_RAT_INST_SUB_RTN: dst = dst - src. non-saturating integer sub. return dword. 41 - SQ_EXPORT_RAT_INST_RSUB_RTN: dst = src - dst. non-saturating reverse subtract. return dword. 42 - SQ_EXPORT_RAT_INST_MIN_INT_RTN: dst = (src < dst) ? src : dst. signed. return dword. 43 - SQ_EXPORT_RAT_INST_MIN_UINT_RTN: dst = (src < dst) ? src : dst. unsigned. return dword. 44 - SQ_EXPORT_RAT_INST_MAX_INT_RTN: dst = (src > dst) ? src : dst. signed return dword. 45 - SQ_EXPORT_RAT_INST_MAX_UINT_RTN: dst = (src > dst) ? src : dst. unsigned return dword. 46 - SQ_EXPORT_RAT_INST_AND_RTN: dst = dst & src. bitwise return dword. 47 - SQ_EXPORT_RAT_INST_OR_RTN: dst = dst | src. bitwise return dword. 48 - SQ_EXPORT_RAT_INST_XOR_RTN: dst = dst ^ src. bitwise return dword. 49 - SQ_EXPORT_RAT_INST_MSKOR_RTN: dst = (dst & ~mask) | src return dword. 50 - SQ_EXPORT_RAT_INST_INC_UINT_RTN: dst =

			(dst >= src) ? 0 : dst+1 return uint. 51 - SQ_EXPORT_RAT_INST_DEC_UINT_RTN: dst = ((dst==0 | (dst > src)) ? src : dst-1 return uint.
RAT_INDEX_MODE	12:11	none	RAT index select: non-indexed, add idx0 or idx1 to RAT_ID POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
TYPE	14:13	none	Type of allocation/export. POSSIBLE VALUES: 00 - SQ_EXPORT_PIXEL: write pixel. SQ_EXPORT_WRITE: write to memory buffer. 01 - SQ_EXPORT_POS: write position. SQ_EXPORT_WRITE_IND: write to memory buffer, use offset in INDEX_GPR. 02 - SQ_EXPORT_PARAM: write parameter cache. SQ_EXPORT_WRITE_ACK: write to memory buffer, request an ACK when write is committed to memory. For RAT, ack guarantees return value has been written to memory. Warning: acknowledges for cached-RAT may return out-of-order with respect to other exports (including cacheless-RAT instructions). 03 - Unused for SX exports. SQ_EXPORT_WRITE_IND_ACK: write to memory buffer with offset in INDEX_GPR, get an ACK when done. For RAT, ack guarantees return value has been written to memory. Warning: acknowledges for cached-RAT instructions may return out-of-order with respect to other exports ((including cacheless-RAT instructions).
RW_GPR	21:15	none	GPR register to read data from. Depending on the RAT_INST opcode, this gpr contains either up to 4 dwords of data to write, or a dword of source data in the X channel, a dword return address in the Y channel and a dword of compare data in the W channel.
RW_REL	22	none	Indicates whether GPR is an absolute address, or relative to the loop index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
INDEX_GPR	29:23	none	Select the GPR that holds the buffer address coordinates. The index is multipled by (ELEM_SIZE + 1). The X, Y and Z components contain the address of the 1-d, 2-d, 3-d or 2-d slice address depending on the format of the RAT surface.

ELEM_SIZE	31:30	none	Number of DWORDs per element, minus one. This field is interpreted as a value in [1,2,4] (3 not supported). The value from INDEX_GPR and the loop counter are multiplied by this factor, if applicable. Also, BURST_COUNT is multiplied by this factor for CF_INST_MEM*. This field is ignored for CF_INST_EXPORT*. Normally, ELEMSIZE = 4 DWORDs for scratch, one DWORD for other types.

SQ_MICRO:SQ_CF_ALLOC_EXPORT_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Word 1 of the control flow instruction for alloc/export is the bitwise OR of WORD1 | WORD1_{BUF,SWIZ}. This part contains fields that are always defined.*			
Field Name	Bits	Default	Description
BURST_COUNT	19:16	none	Number of MRTs, positions, parameters, or logical export values to allocate and/or export, minus one. This field is interpreted as a value in [1,16].
VALID_PIXEL_MODE	20	none	If set, do not export data for invalid pixels. Caution: VALID_PIXEL_MODE is not the \`default\` mode; this bit should be set to 0 by default. Note that Pix/Pos/PC exports use the valid mask and active mask, and mem-exports use the active mask only. Set this only in PS stage.
CF_INST	29:22	none	Type of instruction to evaluate in CF. This value MUST be one of the alloc/export instructions listed below. POSSIBLE VALUES: 64 - SQ_CF_INST_MEM_STREAM0_BUF0: perform a memory write on the stream 0, buffer 0. 65 - SQ_CF_INST_MEM_STREAM0_BUF1: perform a memory write on the stream 0, buffer 1. 66 - SQ_CF_INST_MEM_STREAM0_BUF2: perform a memory write on the stream 0, buffer 2. 67 - SQ_CF_INST_MEM_STREAM0_BUF3: perform a memory write on the stream 0, buffer 3. 68 - SQ_CF_INST_MEM_STREAM1_BUF0: perform a memory write on the stream 1, buffer 0. 69 - SQ_CF_INST_MEM_STREAM1_BUF1: perform a memory write on the stream 1, buffer 1. 70 - SQ_CF_INST_MEM_STREAM1_BUF2: perform a memory write on the stream 1, buffer 2. 71 - SQ_CF_INST_MEM_STREAM1_BUF3: perform a memory write on the stream 1, buffer 3. 72 - SQ_CF_INST_MEM_STREAM2_BUF0: perform a memory write on the stream 2, buffer 0. 73 - SQ_CF_INST_MEM_STREAM2_BUF1: perform a memory write on the stream 2, buffer 1. 74 - SQ_CF_INST_MEM_STREAM2_BUF2: perform a memory write on the stream 2, buffer 2. 75 - SQ_CF_INST_MEM_STREAM2_BUF3: perform a memory write on the stream 2, buffer 3. 76 - SQ_CF_INST_MEM_STREAM3_BUF0:

			perform a memory write on the stream 3, buffer 0. 77 - SQ_CF_INST_MEM_STREAM3_BUF1: perform a memory write on the stream 3, buffer 1. 78 - SQ_CF_INST_MEM_STREAM3_BUF2: perform a memory write on the stream 3, buffer 2. 79 - SQ_CF_INST_MEM_STREAM3_BUF3: perform a memory write on the stream 3, buffer 3. 80 - SQ_CF_INST_MEM_SCRATCH: perform a memory write on the scratch buffer. 82 - SQ_CF_INST_MEM_RING: perform a memory write on the ring buffer. 83 - SQ_CF_INST_EXPORT: export only (not last). Used for PIXEL, POS, PARAM exports. 84 - SQ_CF_INST_EXPORT_DONE: export only (last export). Used for PIXEL, POS, PARAM exports. 85 - SQ_CF_INST_MEM_EXPORT: perform a memory write on the shared buffer . 86 - SQ_CF_INST_MEM_RAT: export to a Random Access Target - full functionality (via CB). 87 - SQ_CF_INST_MEM_RAT_CACHELESS: export to a Random Access Target without caching - reduced functionality (via DB). 88 - SQ_CF_INST_MEM_RING1: write to ring 1 (currently only applies to GSVS ring). 89 - SQ_CF_INST_MEM_RING2: write to ring 2 (currently only applies to GSVS ring). 90 - SQ_CF_INST_MEM_RING3: write to ring 3 (currently only applies to GSVS ring). 91 - SQ_CF_INST_MEM_EXPORT_COMBINED: Memory export (scatter), for single-dword exports only. Combined Address & Data in one export (data = x, data = y, address = w). Must be non-indexed-write, and no burst-writes. 92 - SQ_CF_INST_MEM_RAT_COMBINED_CACHELESS: export to a Random Access Target - reduced functionality (via DB). Combined Address & Data in one export (data = x, data = y; address = w). Must be non-indexed-write, and no burst-writes. 93 - SQ_CF_INST_MEM_RAT_COMBINED: export to a Random Access Target. Combined Address & Data in one export (data = x, data = y; address = w). Must be non-indexed-write, and no burst-writes. 94 - SQ_CF_INST_EXPORT_DONE_END_IS_NEXT: DO NOT USE; Use CF_INST_EXPORT_DONE instead. This opcode is for an unimplemented optimization. export only (last export). Use this iff the very next instruction is CF_END.
MARK	30	none	Mark memory write to be acknowledged with the next write-ack. Only applies to memory writes (scratch,scatter,etc), not pixel/position/parameter.

BARRIER	31	none	If set, all prior CF instructions/clauses must complete before this instruction/clause executes. If not set, this instruction/clause may run in parallel with prior instructions.

SQ_MICRO:SQ_CF_ALLOC_EXPORT_WORD1_BUF · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Word 1 of the control flow instruction. This subencoding is used by alloc/exports for all input/outputs to scratch/ring/stream/RAT buffers.*

Field Name	Bits	Default	Description
ARRAY_SIZE	11:0	none	Array size (elem-size units). SCRATCH: Represents values [1,4096] when ELEMSIZE=0, [4,16384] when ELEMSIZE=3. SCATTER: unused (no effect). RAT_CACHELESS export, array_size[7:0] carries the dword stride for burst exports. Stride is 1..256 dwords.
COMP_MASK	15:12	none	XYZW component mask (X is the LSB). Write the component iff the corresponding bit is 1. User should enable all components which contain address/data for the operation. For RAT store-raw, set to 0x1; for other RATs, set of 0xf.

SQ_MICRO:SQ_CF_ALLOC_EXPORT_WORD1_SWIZ · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Word 1 of the control flow instruction. This subencoding is used by alloc/exports for PIXEL, POS, and PARAM.*

Field Name	Bits	Default	Description
SEL_X	2:0	none	Specify source for each component of the export. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
SEL_Y	5:3	none	Specify source for each component of the export. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0

			06 - Reserved 07 - SQ_SEL_MASK: mask out this component
SEL_Z	8:6	none	Specify source for each component of the export. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
SEL_W	11:9	none	Specify source for each component of the export. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component

SQ_MICRO:SQ_CF_GWS_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Control flow instruction word 0. This word is used by Global Wave Sync instructions.*

Field Name	Bits	Default	Description
VALUE	9:0	none	Counter load value for Barrier and Init opcodes.
RESOURCE	20:16	none	Index of resource. 0-15 in 1-SE chips, 0-31 in 2-SE chips.
SIGN	25	none	If set, resource will be treated as signed -512..511 (as opposed to unsigned 0..1023).
VAL_INDEX_MODE	27:26	none	Override counter load value from instruction using index0/index1 registers. <u>POSSIBLE VALUES:</u> 00 - SQ_GWS_INDEX_NONE: use source from instruction 01 - SQ_GWS_INDEX_0: use index0 as value 02 - SQ_GWS_INDEX_1: use index1 as value 03 - SQ_GWS_INDEX_MIX: use a combination of index0 and index1 as value
RSRC_INDEX_MODE	29:28	none	Override resource index from instruction using index0/index1 registers. <u>POSSIBLE VALUES:</u> 00 - SQ_CF_INDEX_NONE: do not index the

			constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
GWS_OPCODE	31:30	none	Specify the atomic operation to execute on a resource. POSSIBLE VALUES: 00 - SQ_GWS_SEMA_V: semaphore V() 01 - SQ_GWS_SEMA_P: semaphore P() 02 - SQ_GWS_BARRIER: wavefront barrier 03 - SQ_GWS_INIT: resource initialization

SQ_MICRO:SQ_ALU_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *ALU instruction word 0.*

Field Name	Bits	Default	Description
SRC0_SEL	8:0	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. POSSIBLE VALUES: 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt). 223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter 229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int)

			232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only. 236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, `1` indicates `this quad starts a new primitive`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant 0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result. 255 - SQ_ALU_SRC_PS: previous scalar result.
SRC0_REL	9	none	If set, this operand uses relative addressing based on the INDEX_MODE. <u>POSSIBLE VALUES:</u> 00 - SQ_ABSOLUTE: no relative addressing.

			01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC0_CHAN	11:10	none	Specify which channel of the source to use for this operand. POSSIBLE VALUES: 00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
SRC0_NEG	12	none	If set, negate the input for this operand. Should only be set for floating point inputs.
SRC1_SEL	21:13	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. POSSIBLE VALUES: 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt). 223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter 229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int) 232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only.

			236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, `1` indicates `this quad starts a new primitive`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant 0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result. 255 - SQ_ALU_SRC_PS: previous scalar result.
SRC1_REL	22	none	If set, this operand uses relative addressing based on the INDEX_MODE. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC1_CHAN	24:23	none	Specify which channel of the source to use for this operand. POSSIBLE VALUES:

			00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
SRC1_NEG	25	none	If set, negate the input for this operand. Should only be set for floating point inputs.
INDEX_MODE	28:26	none	Specify what relative addressing mode to use for operands that have the REL bit set. POSSIBLE VALUES: 00 - SQ_INDEX_AR_X: constants/gpr: add AR.X. 04 - SQ_INDEX_LOOP: add current loop index value. 05 - SQ_INDEX_GLOBAL: treat gpr address as absolute, not thread-relative. 06 - SQ_INDEX_GLOBAL_AR_X: treat gpr address as absolute and add GPR-index (ar.x).
PRED_SEL	30:29	none	Predicate to apply to this instruction. POSSIBLE VALUES: 00 - SQ_PRED_SEL_OFF: execute all pixels. 01 - Reserved 02 - SQ_PRED_SEL_ZERO: execute when pred = 0. 03 - SQ_PRED_SEL_ONE: execute when pred = 1.
LAST	31	none	If set, this is the last 64-bit word for this instruction.

SQ_MICRO:SQ_ALU_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *ALU instruction word 1 is the bitwise OR of SQ_ALU_WORD1 | SQ_ALU_WORD1_OP[2,3]. SQ_ALU_WORD1 contains fields used by all encodings.*

Field Name	Bits	Default	Description
ENCODING	17:15	none	A read-only field used to determine whether OP2 or OP3 encoding is being used. If this field`s value is 0, the instruction is using OP2. Otherwise, the instruction is using OP3. Do not write to this field directly.
BANK_SWIZZLE	20:18	none	Specify how to load operands into the SP. POSSIBLE VALUES: 00 - SQ_ALU_VEC_012, SQ_ALU_SCL_210 01 - SQ_ALU_VEC_021, SQ_ALU_SCL_122 02 - SQ_ALU_VEC_120, SQ_ALU_SCL_212 03 - SQ_ALU_VEC_102, SQ_ALU_SCL_221 04 - SQ_ALU_VEC_201 05 - SQ_ALU_VEC_210
DST_GPR	27:21	none	Destination address to write result to. Always a GPR address.
DST_REL	28	none	If set, this operand uses relative addressing based on the INDEX_MODE.

			POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
DST_CHAN	30:29	none	Specify which channel of DST_GPR to write the result to. POSSIBLE VALUES: 00 - CHAN_X: write to X channel of dest. 01 - CHAN_Y: write to Y channel of dest. 02 - CHAN_Z: write to Z channel of dest. 03 - CHAN_W: write to W channel of dest.
CLAMP	31	none	If set, clamp the result to [0.0, 1.0]. Not mathematically defined for opcodes that produce integer results.

SQ_MICRO:SQ_ALU_WORD1_OP2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *ALU instruction word 1. This subencoding is used for OP2 instructions (instructions taking 0 to 2 operands).*

Field Name	Bits	Default	Description
SRC0_ABS	0	none	If set, take the absolute value of the input for this operand. Should only be set for floating point inputs; performed before negation.
SRC1_ABS	1	none	If set, take the absolute value of the input for this operand. Should only be set for floating point inputs; performed before negation.
UPDATE_EXECUTE_MASK	2	none	If set, update the execute mask in the SQ after executing this instruction based on the current predicate.
UPDATE_PRED	3	none	If set, update the predicate in the SP based on the predicate operation computed here.
WRITE_MASK	4	none	If set, write this scalar result to the destination GPR channel.
OMOD	6:5	none	Output modifier for this instruction. Must be set to ALU_OMOD_OFF for operations that produce an integer result. POSSIBLE VALUES: 00 - SQ_ALU_OMOD_OFF: identity. 01 - SQ_ALU_OMOD_M2: multiply by 2.0. 02 - SQ_ALU_OMOD_M4: multiply by 4.0. 03 - SQ_ALU_OMOD_D2: divide by 2.0.
ALU_INST	17:7	none	Instruction opcode. The top 3 bits of this must be zero. Caution: gaps in opcode values are not marked in the table below. Opcodes 0..95 can be used in either the Vector or Trans unit. Opcodes 128..159 are Trans-only. Opcodes 160..255 are Vector-only.

			POSSIBLE VALUES: 00 - SQ_OP2_INST_ADD 01 - SQ_OP2_INST_MUL 02 - SQ_OP2_INST_MUL_IEEE 03 - SQ_OP2_INST_MAX 04 - SQ_OP2_INST_MIN 05 - SQ_OP2_INST_MAX_DX10 06 - SQ_OP2_INST_MIN_DX10 08 - SQ_OP2_INST_SETE 09 - SQ_OP2_INST_SETGT 10 - SQ_OP2_INST_SETGE 11 - SQ_OP2_INST_SETNE 12 - SQ_OP2_INST_SETE_DX10 13 - SQ_OP2_INST_SETGT_DX10 14 - SQ_OP2_INST_SETGE_DX10 15 - SQ_OP2_INST_SETNE_DX10 16 - SQ_OP2_INST_FRACT 17 - SQ_OP2_INST_TRUNC 18 - SQ_OP2_INST_CEIL 19 - SQ_OP2_INST_RNDNE 20 - SQ_OP2_INST_FLOOR 21 - SQ_OP2_INST_ASHR_INT 22 - SQ_OP2_INST_LSHR_INT 23 - SQ_OP2_INST_LSHL_INT 25 - SQ_OP2_INST_MOV 26 - SQ_OP2_INST_NOP 30 - SQ_OP2_INST_PRED_SETGT_UINT 31 - SQ_OP2_INST_PRED_SETGE_UINT 32 - SQ_OP2_INST_PRED_SETE 33 - SQ_OP2_INST_PRED_SETGT 34 - SQ_OP2_INST_PRED_SETGE 35 - SQ_OP2_INST_PRED_SETNE 36 - SQ_OP2_INST_PRED_SET_INV 37 - SQ_OP2_INST_PRED_SET_POP 38 - SQ_OP2_INST_PRED_SET_CLR 39 - SQ_OP2_INST_PRED_SET_RESTORE 40 - SQ_OP2_INST_PRED_SETE_PUSH 41 - SQ_OP2_INST_PRED_SETGT_PUSH 42 - SQ_OP2_INST_PRED_SETGE_PUSH 43 - SQ_OP2_INST_PRED_SETNE_PUSH 44 - SQ_OP2_INST_KILLE 45 - SQ_OP2_INST_KILLGT 46 - SQ_OP2_INST_KILLGE 47 - SQ_OP2_INST_KILLNE 48 - SQ_OP2_INST_AND_INT 49 - SQ_OP2_INST_OR_INT 50 - SQ_OP2_INST_XOR_INT 51 - SQ_OP2_INST_NOT_INT 52 - SQ_OP2_INST_ADD_INT 53 - SQ_OP2_INST_SUB_INT 54 - SQ_OP2_INST_MAX_INT 55 - SQ_OP2_INST_MIN_INT

			56 - SQ_OP2_INST_MAX_UINT 57 - SQ_OP2_INST_MIN_UINT 58 - SQ_OP2_INST_SETE_INT 59 - SQ_OP2_INST_SETGT_INT 60 - SQ_OP2_INST_SETGE_INT 61 - SQ_OP2_INST_SETNE_INT 62 - SQ_OP2_INST_SETGT_UINT 63 - SQ_OP2_INST_SETGE_UINT 64 - SQ_OP2_INST_KILLGT_UINT 65 - SQ_OP2_INST_KILLGE_UINT 66 - SQ_OP2_INST_PRED_SETE_INT 67 - SQ_OP2_INST_PRED_SETGT_INT 68 - SQ_OP2_INST_PRED_SETGE_INT 69 - SQ_OP2_INST_PRED_SETNE_INT 70 - SQ_OP2_INST_KILLE_INT 71 - SQ_OP2_INST_KILLGT_INT 72 - SQ_OP2_INST_KILLGE_INT 73 - SQ_OP2_INST_KILLNE_INT 74 - SQ_OP2_INST_PRED_SETE_PUSH_INT 75 - SQ_OP2_INST_PRED_SETGT_PUSH_INT 76 - SQ_OP2_INST_PRED_SETGE_PUSH_INT 77 - SQ_OP2_INST_PRED_SETNE_PUSH_INT 78 - SQ_OP2_INST_PRED_SETLT_PUSH_INT 79 - SQ_OP2_INST_PRED_SETLE_PUSH_INT 80 - SQ_OP2_INST_FLT_TO_INT 81 - SQ_OP2_INST_BFREV_INT 82 - SQ_OP2_INST_ADDC_UINT 83 - SQ_OP2_INST_SUBB_UINT 84 - SQ_OP2_INST_GROUP_BARRIER 85 - SQ_OP2_INST_GROUP_SEQ_BEGIN 86 - SQ_OP2_INST_GROUP_SEQ_END 87 - SQ_OP2_INST_SET_MODE 88 - SQ_OP2_INST_SET_CF_IDX0 89 - SQ_OP2_INST_SET_CF_IDX1 90 - SQ_OP2_INST_SET_LDS_SIZE 129 - SQ_OP2_INST_EXP_IEEE 130 - SQ_OP2_INST_LOG_CLAMPED 131 - SQ_OP2_INST_LOG_IEEE 132 - SQ_OP2_INST_RECIP_CLAMPED 133 - SQ_OP2_INST_RECIP_FF 134 - SQ_OP2_INST_RECIP_IEEE 135 - SQ_OP2_INST_RECIPSQRT_CLAMPED 136 - SQ_OP2_INST_RECIPSQRT_FF 137 - SQ_OP2_INST_RECIPSQRT_IEEE 138 - SQ_OP2_INST_SQRT_IEEE 141 - SQ_OP2_INST_SIN 142 - SQ_OP2_INST_COS 143 - SQ_OP2_INST_MULLO_INT 144 - SQ_OP2_INST_MULHI_INT 145 - SQ_OP2_INST_MULLO_UINT 146 - SQ_OP2_INST_MULHI_UINT 147 - SQ_OP2_INST_RECIP_INT 148 - SQ_OP2_INST_RECIP_UINT

			149 - SQ_OP2_INST_RECIP_64 150 - SQ_OP2_INST_RECIP_CLAMPED_64 151 - SQ_OP2_INST_RECIPSQRT_64 152 - SQ_OP2_INST_RECIPSQRT_CLAMPED_64 153 - SQ_OP2_INST_SQRT_64 154 - SQ_OP2_INST_FLT_TO_UINT 155 - SQ_OP2_INST_INT_TO_FLT 156 - SQ_OP2_INST_UINT_TO_FLT 160 - SQ_OP2_INST_BFM_INT 162 - SQ_OP2_INST_FLT32_TO_FLT16 163 - SQ_OP2_INST_FLT16_TO_FLT32 164 - SQ_OP2_INST_UBYTE0_FLT 165 - SQ_OP2_INST_UBYTE1_FLT 166 - SQ_OP2_INST_UBYTE2_FLT 167 - SQ_OP2_INST_UBYTE3_FLT 170 - SQ_OP2_INST_BCNT_INT 171 - SQ_OP2_INST_FFBH_UINT 172 - SQ_OP2_INST_FFBL_INT 173 - SQ_OP2_INST_FFBH_INT 174 - SQ_OP2_INST_FLT_TO_UINT4 175 - SQ_OP2_INST_DOT_IEEE 176 - SQ_OP2_INST_FLT_TO_INT_RPI 177 - SQ_OP2_INST_FLT_TO_INT_FLOOR 178 - SQ_OP2_INST_MULHI_UINT24 179 - SQ_OP2_INST_MBCNT_32HI_INT 180 - SQ_OP2_INST_OFFSET_TO_FLT 181 - SQ_OP2_INST_MUL_UINT24 182 - SQ_OP2_INST_BCNT_ACCUM_PREV_INT 183 - SQ_OP2_INST_MBCNT_32LO_ACCUM_PREV_INT 184 - SQ_OP2_INST_SETE_64 185 - SQ_OP2_INST_SETNE_64 186 - SQ_OP2_INST_SETGT_64 187 - SQ_OP2_INST_SETGE_64 188 - SQ_OP2_INST_MIN_64 189 - SQ_OP2_INST_MAX_64 190 - SQ_OP2_INST_DOT4 191 - SQ_OP2_INST_DOT4_IEEE 192 - SQ_OP2_INST_CUBE 193 - SQ_OP2_INST_MAX4 196 - SQ_OP2_INST_FREXP_64 197 - SQ_OP2_INST_LDEXP_64 198 - SQ_OP2_INST_FRACT_64 199 - SQ_OP2_INST_PRED_SETGT_64 200 - SQ_OP2_INST_PRED_SETE_64 201 - SQ_OP2_INST_PRED_SETGE_64 202 - SQ_OP2_INST_MUL_64 203 - SQ_OP2_INST_ADD_64 204 - SQ_OP2_INST_MOVA_INT 205 - SQ_OP2_INST_FLT64_TO_FLT32 206 - SQ_OP2_INST_FLT32_TO_FLT64 207 - SQ_OP2_INST_SAD_ACCUM_PREV_UINT 208 - SQ_OP2_INST_DOT

			209 - SQ_OP2_INST_MUL_PREV 210 - SQ_OP2_INST_MUL_IEEE_PREV 211 - SQ_OP2_INST_ADD_PREV 212 - SQ_OP2_INST_MULADD_PREV 213 - SQ_OP2_INST_MULADD_IEEE_PREV 214 - SQ_OP2_INST_INTERP_XY 215 - SQ_OP2_INST_INTERP_ZW 216 - SQ_OP2_INST_INTERP_X 217 - SQ_OP2_INST_INTERP_Z 218 - SQ_OP2_INST_STORE_FLAGS 219 - SQ_OP2_INST_LOAD_STORE_FLAGS 220 - Reserved 221 - Reserved 223 - Reserved 224 - SQ_OP2_INST_INTERP_LOAD_P0 225 - SQ_OP2_INST_INTERP_LOAD_P10 226 - SQ_OP2_INST_INTERP_LOAD_P20

SQ_MICRO:SQ_ALU_WORD1_OP2_MOVA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *ALU instruction word 1. This subencoding is used for MOVA instructions.*			
Field Name	Bits	Default	Description
MOVA_DST	27:21	none	Indicates target of MOVA instruction. The output is not written to a GPR but instead to the destination indicated by this field. POSSIBLE VALUES: 00 - SQ_ALU_MOVA_DST_AR_X: store MOVA result 9 lsbs in AR.X. 01 - SQ_ALU_MOVA_DST_CF_PC: store MOVA result in CF_PC. From GPR sources, only 9 lsbs are copied. 02 - SQ_ALU_MOVA_DST_CF_IDX0: store MOVA result 9 lsbs in CF_IDX0 03 - SQ_ALU_MOVA_DST_CF_IDX1: store MOVA result 9 lsbs in CF_IDX1 04 - SQ_ALU_MOVA_DST_CLAUSE_GLOBAL_B0: store MOVA result 8 lsbs in CLAUSE_GLOBAL[7:0] 05 - SQ_ALU_MOVA_DST_CLAUSE_GLOBAL_B1: store MOVA result 8 lsbs in CLAUSE_GLOBAL[15:8] 06 - SQ_ALU_MOVA_DST_CLAUSE_GLOBAL_B2: store MOVA result 8 lsbs in CLAUSE_GLOBAL[23:16] 07 - SQ_ALU_MOVA_DST_CLAUSE_GLOBAL_B3: store MOVA result 8 lsbs in CLAUSE_GLOBAL[31:24]

SQ_MICRO:SQ_ALU_WORD1_OP2_EXECUTE_MASK · [R/W] · 32 bits · Access: 32 ·

GpuF0MMReg:0x8dfc			
DESCRIPTION: *ALU instruction word 1. This subencoding is used for OP2 instructions with UPDATE_EXECUTE_MASK set.*			
Field Name	Bits	Default	Description
EXECUTE_MASK_OP	6:5	none	Indicates how to update the execute mask after executing this instruction based on the current predicate. This field should be used instead of OMOD. POSSIBLE VALUES: 00 - SQ_ALU_EXECUTE_MASK_OP_DEACTIVATE: deactivate active thread if PRED bit would be set to 0 or PRED_SEL does not select this thread. 01 - SQ_ALU_EXECUTE_MASK_OP_BREAK: deactivate active thread if PRED bit would be set to 0 or PRED_SEL does not select this thread, and don`t reactivete it until we exit the innermost loop 02 - SQ_ALU_EXECUTE_MASK_OP_CONTINUE: deactivate active thread if PRED bit would be set to 0 or PRED_SEL does not select this thread, and don`t reactivate it until we repeat or exit the innermost loop 03 - SQ_ALU_EXECUTE_MASK_OP_KILL: kill active thread if PRED bit would be set to 0 or if PRED_SEL does not select this thread. This is independent of the KILL instruction and is for using predicate operations to kill threads.

SQ_MICRO:SQ_ALU_WORD1_OP3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *ALU instruction word 1. This subencoding is used for OP3 instructions (instructions taking 3 operands).*			
Field Name	Bits	Default	Description
SRC2_SEL	8:0	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. POSSIBLE VALUES: 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt).

			223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter 229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int) 232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only. 236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, `1` indicates `this quad starts a new primitive`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant

			0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result. 255 - SQ_ALU_SRC_PS: previous scalar result.
SRC2_REL	9	none	If set, this operand uses relative addressing based on the INDEX_MODE. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC2_CHAN	11:10	none	Specify which channel of the source to use for this operand. POSSIBLE VALUES: 00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
SRC2_NEG	12	none	If set, negate the input for this operand. Should only be set for floating point inputs.
ALU_INST	17:13	none	Instruction opcode. Caution: opcode values do not begin at zero. Opcodes 4..17 are Vector-only. Opcodes 20..31 can be used in either the Vector or Trans unit. Opcode 31 is Trans-only. POSSIBLE VALUES: 04 - SQ_OP3_INST_BFE_UINT 05 - SQ_OP3_INST_BFE_INT 06 - SQ_OP3_INST_BFI_INT 07 - SQ_OP3_INST_FMA 09 - SQ_OP3_INST_CNDNE_64 10 - SQ_OP3_INST_FMA_64 11 - SQ_OP3_INST_LERP_UINT 12 - SQ_OP3_INST_BIT_ALIGN_INT 13 - SQ_OP3_INST_BYTE_ALIGN_INT 14 - SQ_OP3_INST_SAD_ACCUM_UINT 15 - SQ_OP3_INST_SAD_ACCUM_HI_UINT 16 - SQ_OP3_INST_MULADD_UINT24 17 - SQ_OP3_INST_LDS_IDX_OP: This opcodes implies SQ_ALU_WORD*_LDS_IDX_OP encoding 20 - SQ_OP3_INST_MULADD 21 - SQ_OP3_INST_MULADD_M2 22 - SQ_OP3_INST_MULADD_M4

			23 - SQ_OP3_INST_MULADD_D2 24 - SQ_OP3_INST_MULADD_IEEE 25 - SQ_OP3_INST_CNDE 26 - SQ_OP3_INST_CNDGT 27 - SQ_OP3_INST_CNDGE 28 - SQ_OP3_INST_CNDE_INT 29 - SQ_OP3_INST_CNDGT_INT 30 - SQ_OP3_INST_CNDGE_INT 31 - SQ_OP3_INST_MUL_LIT

SQ_MICRO:SQ_ALU_WORD0_LDS_IDX_OP · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *ALU instruction word 0 for SQ_LDS_IDX_OP*

Field Name	Bits	Default	Description
SRC0_SEL	8:0	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. POSSIBLE VALUES: 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt). 223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter 229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int) 232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS

			only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only. 236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, \`1\` indicates \`this quad starts a new primitive\`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1\`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant 0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result. 255 - SQ_ALU_SRC_PS: previous scalar result.
SRC0_REL	9	none	If set, this operand uses relative addressing based on the INDEX_MODE. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC0_CHAN	11:10	none	Specify which channel of the source to use for this operand.

			POSSIBLE VALUES: 00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
IDX_OFFSET_4	12	none	Index offset bit 4
SRC1_SEL	21:13	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. POSSIBLE VALUES: 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt). 223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter 229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int) 232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only. 236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause

			executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, `1` indicates `this quad starts a new primitive`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant 0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result. 255 - SQ_ALU_SRC_PS: previous scalar result.
SRC1_REL	22	none	If set, this operand uses relative addressing based on the INDEX_MODE. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC1_CHAN	24:23	none	Specify which channel of the source to use for this operand. POSSIBLE VALUES: 00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
IDX_OFFSET_5	25	none	Index offset bit 5
INDEX_MODE	28:26	none	Specify what relative addressing mode to use for

			operands that have the REL bit set. <u>POSSIBLE VALUES:</u> 00 - SQ_INDEX_AR_X: constants/gpr: add AR.X. 04 - SQ_INDEX_LOOP: add current loop index value. 05 - SQ_INDEX_GLOBAL: treat gpr address as absolute, not thread-relative. 06 - SQ_INDEX_GLOBAL_AR_X: treat gpr address as absolute and add GPR-index (ar.x).
PRED_SEL	30:29	none	Predicate to apply to this instruction. <u>POSSIBLE VALUES:</u> 00 - SQ_PRED_SEL_OFF: execute all pixels. 01 - Reserved 02 - SQ_PRED_SEL_ZERO: execute when pred = 0. 03 - SQ_PRED_SEL_ONE: execute when pred = 1.
LAST	31	none	If set, this is the last 64-bit word for this instruction.

SQ_MICRO:SQ_ALU_WORD1_LDS_IDX_OP · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *ALU instruction word 1 for SQ_LDS_IDX_OP*

Field Name	Bits	Default	Description
SRC2_SEL	8:0	none	Source for operands src0, src1. Values [0,127] correspond to GPR[0..127]. Values [128,159] correspond to kcache constants in bank 0, [160,191] to bank 1, [256,287] to bank 2, [288,319] to bank3. [192,255] are inline constant values. <u>POSSIBLE VALUES:</u> 219 - SQ_ALU_SRC_LDS_OQ_A: Use contents of LDS Output Queue A and leave it on the queue. 220 - SQ_ALU_SRC_LDS_OQ_B: Use contents of LDS Output Queue B and leave it on the queue. 221 - SQ_ALU_SRC_LDS_OQ_A_POP: Use contents of LDS Output Queue A, and pop both the A and B queues at the end of the instruction group(xyzwt). 222 - SQ_ALU_SRC_LDS_OQ_B_POP: Use contents of LDS Output Queue B, and pop both the A and B queues at the end of the instruction group(xyzwt). 223 - SQ_ALU_SRC_LDS_DIRECT_A: Direct read of LDS on the A cycle. Address is defined in literal constant-0 (xy). 224 - SQ_ALU_SRC_LDS_DIRECT_B: Direct read of LDS on the B cycle. Address is defined in literal constant-0 (xy). 227 - SQ_ALU_SRC_TIME_HI: Upper 32 bits of 64-bit clock counter 228 - SQ_ALU_SRC_TIME_LO: Lower 32 bits of 64-bit clock counter

			229 - SQ_ALU_SRC_MASK_HI: Upper 32bits of active mask 230 - SQ_ALU_SRC_MASK_LO: Lower 32bits of active mask 231 - SQ_ALU_SRC_HW_WAVE_ID: Hardware wave ID (int) 232 - SQ_ALU_SRC_SIMD_ID: Simd id (int) 233 - SQ_ALU_SRC_SE_ID: Shader engine ID (int) 234 - SQ_ALU_SRC_HW_THREADGRP_ID: Hardware thread group ID (int) within simd. CS and HS only. 235 - SQ_ALU_SRC_WAVE_ID_IN_GRP: Wave id within thread group (int). CS and HS only. 236 - SQ_ALU_SRC_NUM_THREADGRP_WAVES: Number of waves in thread group (int). CS and HS only, must barrier before using. 237 - SQ_ALU_SRC_HW_ALU_ODD: Is this clause executing on the even(0) or odd(1) path (int) 238 - SQ_ALU_SRC_LOOP_IDX: Current value of the loop index (int) 240 - SQ_ALU_SRC_PARAM_BASE_ADDR: Parameter cache base (SQ_LDS_ALLOC_PS), (int) 241 - SQ_ALU_SRC_NEW_PRIM_MASK: Bit mask. 1 bit per quad, \`1\` indicates \`this quad starts a new primitive\`. Mask omits bit for first quad since it always begins a new primitive. (e.g in a vectorsize 64 system, this mask is {[15:1],1\`b1}). 242 - SQ_ALU_SRC_PRIM_MASK_HI: Upper 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 243 - SQ_ALU_SRC_PRIM_MASK_LO: Lower 32 bits of 64-bit expansion of NEW_PRIM_MASK. Used for general parameter interp. See SQ-arch spec for details. 244 - SQ_ALU_SRC_1_DBL_L: special constant 1.0 double-float, LSW. 245 - SQ_ALU_SRC_1_DBL_M: special constant 1.0 double-float, MSW. 246 - SQ_ALU_SRC_0_5_DBL_L: special constant 0.5 double-float, LSW. 247 - SQ_ALU_SRC_0_5_DBL_M: special constant 0.5 double-float, MSW. 248 - SQ_ALU_SRC_0: special constant 0.0. 249 - SQ_ALU_SRC_1: special constant 1.0 float. 250 - SQ_ALU_SRC_1_INT: special constant 1 integer. 251 - SQ_ALU_SRC_M_1_INT: special constant -1 integer. 252 - SQ_ALU_SRC_0_5: special constant 0.5 float. 253 - SQ_ALU_SRC_LITERAL: literal constant. 254 - SQ_ALU_SRC_PV: previous vector result.

			255 - SQ_ALU_SRC_PS: previous scalar result.
SRC2_REL	9	none	If set, this operand uses relative addressing based on the INDEX_MODE. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add index from INDEX_MODE to this address
SRC2_CHAN	11:10	none	Specify which channel of the source to use for this operand. POSSIBLE VALUES: 00 - SQ_CHAN_X: Use X component. 01 - SQ_CHAN_Y: Use Y component. 02 - SQ_CHAN_Z: Use Z component. 03 - SQ_CHAN_W: Use W component.
IDX_OFFSET_1	12	none	Index_offset bit 1
ALU_INST	17:13	none	Instruction opcode. Caution: opcode values do not begin at zero. POSSIBLE VALUES: 04 - SQ_OP3_INST_BFE_UINT 05 - SQ_OP3_INST_BFE_INT 06 - SQ_OP3_INST_BFI_INT 07 - SQ_OP3_INST_FMA 09 - SQ_OP3_INST_CNDNE_64 10 - SQ_OP3_INST_FMA_64 11 - SQ_OP3_INST_LERP_UINT 12 - SQ_OP3_INST_BIT_ALIGN_INT 13 - SQ_OP3_INST_BYTE_ALIGN_INT 14 - SQ_OP3_INST_SAD_ACCUM_UINT 15 - SQ_OP3_INST_SAD_ACCUM_HI_UINT 16 - SQ_OP3_INST_MULADD_UINT24 17 - SQ_OP3_INST_LDS_IDX_OP: This opcodes implies SQ_ALU_WORD*_LDS_IDX_OP encoding 20 - SQ_OP3_INST_MULADD 21 - SQ_OP3_INST_MULADD_M2 22 - SQ_OP3_INST_MULADD_M4 23 - SQ_OP3_INST_MULADD_D2 24 - SQ_OP3_INST_MULADD_IEEE 25 - SQ_OP3_INST_CNDE 26 - SQ_OP3_INST_CNDGT 27 - SQ_OP3_INST_CNDGE 28 - SQ_OP3_INST_CNDE_INT 29 - SQ_OP3_INST_CNDGT_INT 30 - SQ_OP3_INST_CNDGE_INT 31 - SQ_OP3_INST_MUL_LIT
BANK_SWIZZLE	20:18	none	Specify how to load operands into the SP. POSSIBLE VALUES: 00 - SQ_ALU_VEC_012, SQ_ALU_SCL_210

			01 - SQ_ALU_VEC_021, SQ_ALU_SCL_122 02 - SQ_ALU_VEC_120, SQ_ALU_SCL_212 03 - SQ_ALU_VEC_102, SQ_ALU_SCL_221 04 - SQ_ALU_VEC_201 05 - SQ_ALU_VEC_210
LDS_OP	26:21	none	Local Data share atomic opcode POSSIBLE VALUES: 00 - SQ_DS_INST_ADD: OP(dst,src, ...) dst=src0_sel, src=src1_sel. 1A1D ADD(dst,src) : DS(dst) += src. dst is src0_sel, src is src1_sel. 01 - SQ_DS_INST_SUB: 1A1D SUB(dst,src) : DS(dst) = DS(dst) - src 02 - SQ_DS_INST_RSUB: 1A1D RSUB(dst,src): DS(dst) = src - DS(dst) 03 - SQ_DS_INST_INC: 1A1D INC(dst) : (DS(dst)>=src) ? DS(dst) = 0 : DS(dst)++ 04 - SQ_DS_INST_DEC: 1A1D DEC(dst) : DS(dst) = ((DS(dst)==0) || (DS(dst)>src)) ? src : DS(dst)-1 05 - SQ_DS_INST_MIN_INT: 1A1D MIN(dst,src) : DS(dst) = min (DS(dst),src) 06 - SQ_DS_INST_MAX_INT: 1A1D MAX(dst,src) : DS(dst) = max(DS(dst),src) 07 - SQ_DS_INST_MIN_UINT: 1A1D MIN(dst,src) : DS(dst) = min (DS(dst),src) 08 - SQ_DS_INST_MAX_UINT: 1A1D MAX(dst,src) : DS(dst) = max(DS(dst),src) 09 - SQ_DS_INST_AND: 1A1D AND(dst,src) : DS(dst) &= src 10 - SQ_DS_INST_OR: 1A1D OR(dst,src) : DS(dst) |= src 11 - SQ_DS_INST_XOR: 1A1D XOR(dst,src) : DS(dst) ^= src 12 - SQ_DS_INST_MSKOR: 1A2D MKSOR(dst,mask,src) : DS(dst) = ((DS(dst) & ~msk) | src). 13 - SQ_DS_INST_WRITE: 1A1D WRITE(dst,src) : DS(dst) = src 14 - SQ_DS_INST_WRITE_REL: 1A2D WRITEREL(dst,src0,src1) : tmp = dst + sq_DS_idx_offset (offset in dwords). DS(dst) = src0, DS(tmp) = src1 15 - SQ_DS_INST_WRITE2: 1A2D WRITE2(dst,src0,src1) : tmp = dst+(sq_DS_idx_offset * 64). DS(dst) = src0, DS(tmp) = src1 16 - SQ_DS_INST_CMP_STORE: 1A2D CMP_STORE(dst, cmp, src) : DS(dst) = (DS(dst) == cmp) ? src : DS(dst) 17 - SQ_DS_INST_CMP_STORE_SPF: 1A2D CMP_STORE_SPF(dst, cmp, src) : DS(dst) = (DS(dst) == cmp) ? src : DS(dst) 18 - SQ_DS_INST_BYTE_WRITE: 1A1D

			BYTEWRITE (dst, src) : DS(dst) = src[7:0] 19 - SQ_DS_INST_SHORT_WRITE: 1A1D SHORTWRITE(dst, src) : DS(dst) = src[15:0] 32 - SQ_DS_INST_ADD_RET: 1A1D ADD(dst,src) : OQA=DS(dst), DS(dst) += src. dst is src0_sel, src is src1_sel. 33 - SQ_DS_INST_SUB_RET: 1A1D SUB(dst,src) : OQA=DS(dst), DS(dst) = DS(dst) - src 34 - SQ_DS_INST_RSUB_RET: 1A1D RSUB(dst,src) : OQA=DS(dst), DS(dst) = src - DS(dst) 35 - SQ_DS_INST_INC_RET: 1A1D INC(dst) : OQA=DS(dst), (DS(dst)>=src) ? DS(dst) = 0 : DS(dst)++ 36 - SQ_DS_INST_DEC_RET: 1A1D DEC(dst) : OQA=DS(dst), DS(dst) = ((DS(dst)==0) || (DS(dst)>src)) ? src : DS(dst)-1 37 - SQ_DS_INST_MIN_INT_RET: 1A1D MIN(dst,src) : OQA=DS(dst), DS(dst) = min (DS(dst),src) 38 - SQ_DS_INST_MAX_INT_RET: 1A1D MAX(dst,src) : OQA=DS(dst), DS(dst) = max(DS(dst),src) 39 - SQ_DS_INST_MIN_UINT_RET: 1A1D MIN(dst,src) : OQA=DS(dst), DS(dst) = min (DS(dst),src) 40 - SQ_DS_INST_MAX_UINT_RET: 1A1D MAX(dst,src) : OQA=DS(dst), DS(dst) = max(DS(dst),src) 41 - SQ_DS_INST_AND_RET: 1A1D AND(dst,src) : OQA=DS(dst), DS(dst) &= src 42 - SQ_DS_INST_OR_RET: 1A1D OR(dst,src) : OQA=DS(dst), DS(dst) |= src 43 - SQ_DS_INST_XOR_RET: 1A1D XOR(dst,src) : OQA=DS(dst), DS(dst) ^= src 44 - SQ_DS_INST_MSKOR_RET: 1A2D MSKOR(dst,msk,src) : OQA=DS(dst), DS(dst) = ((DS(dst) & ~msk) | src). 45 - SQ_DS_INST_XCHG_RET: 1A1D Exchange(dst,src) : OQA=DS(dst), DS(dst) = src 46 - SQ_DS_INST_XCHG_REL_RET: 1A2D ExchangeRel(dst,src0,src1) : tmp = dst + sq_DS_idx_offset. OQA=DS(dst), OQB=DS(tmp); DS(dst)=src0, DS(tmp)=src1 47 - SQ_DS_INST_XCHG2_RET: 1A2D Exchange2(dst,src0,src1) : tmp = dst + sq_DS_idx_offset*64. OQA=DS(dst), OQB=DS(tmp); DS(dst)=src0, DS(tmp)=src1 48 - SQ_DS_INST_CMP_XCHG_RET: 1A2D CompareExchange(dst,cmp,src) : OQA=DS(dst); (DS(dst)==cmp) ? DS(dst)=src : DS(dst)=DS(dst) 49 - SQ_DS_INST_CMP_XCHG_SPF_RET: 1A2D CompareExchangeSPF(dst,cmp,src) : OQA=DS(dst); (DS(dst)==cmp) ? DS(dst)=src : DS(dst)=DS(dst) 50 - SQ_DS_INST_READ_RET: 1A READ(dst) :

			OQA = DS(dst) 51 - SQ_DS_INST_READ_REL_RET: 1A READ_REL(dst) : tmp=dst+sq_DS_idx_offset; OQA=DS(dst), OQB=DS(tmp) 52 - SQ_DS_INST_READ2_RET: 2A READ2(dst0,dst1) : OQA=DS(dst0), OQB=DS(dst1) 53 - SQ_DS_INST_READWRITE_RET: 2A1D READWRITE(dst0,dst1,data) : OQA=DS(dst0), DS(dst1)=data 54 - SQ_DS_INST_BYTE_READ_RET: 1A BYTEREAD(dst) : OQA=SignExtend(DS(dst)[7:0]) 55 - SQ_DS_INST_UBYTE_READ_RET: 1A UBYTEREAD(dst) : OQA={24`h0, DS(dst)[7:0]} 56 - SQ_DS_INST_SHORT_READ_RET: 1A SHORTREAD(dst) : OQA=SignExtend(DS(dst)[15:0]} 57 - SQ_DS_INST_USHORT_READ_RET: 1A USHORTREAD(dst) : OQA={16`h0, DS(dst)[15:0]} 63 - SQ_DS_INST_ATOMIC_ORDERED_ALLOC_RET: 1A GDS-only (intercepted by ordered alloc unit). This will add the 7 lsb of 1a to a hidden ordered append count in wave order and return the pre-op value to the specified destination register. This opcode can only be used by GDS and with broadcast first set.
IDX_OFFSET_0	27	none	Index_offset bit 0. DWORD offset (except one lds op for which it`s a 64-dword offset)
IDX_OFFSET_2	28	none	Index_offset bit 2
DST_CHAN	30:29	none	Specify which channel of DST_GPR to write the result to. POSSIBLE VALUES: 00 - CHAN_X: write to X channel of dest. 01 - CHAN_Y: write to Y channel of dest. 02 - CHAN_Z: write to Z channel of dest. 03 - CHAN_W: write to W channel of dest.
IDX_OFFSET_3	31	none	Index_offset bit 3

SQ_MICRO:SQ_ALU_WORD1_LDS_DIRECT_LITERAL_HI · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Literal0 constant contents for direct LDS reads.*

Field Name	Bits	Default	Description
OFFSET_B	12:0	none	Dword offset for LDS-direct.
STRIDE_B	19:13	none	Dword stride. stride must not cause bank conflict in LDS rams.
THREAD_REL_B	22	none	1 = add (wave_in_threadgroup * vectorsize * stride) to offset.
DIRECT_READ_32	31	none	0 = read 16 dwords for A and B on each of 4 cycles. 1 = read 32 dwords for A in one cycle, 32 for B the next,

			repeat. Has different memory conflict pattern.

SQ_MICRO:SQ_ALU_WORD1_LDS_DIRECT_LITERAL_LO · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Literal0 constant contents for direct LDS reads.*			
Field Name	Bits	Default	Description
OFFSET_A	12:0	none	Dword offset for LDS-direct.
STRIDE_A	19:13	none	Dword stride. stride must not cause bank conflict in LDS rams.
THREAD_REL_A	22	none	1 = add (wave_in_threadgroup * vectorsize * stride) to offset.

SQ_MICRO:SQ_VTX_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Vertex fetch clause instruction word 0.*			
Field Name	Bits	Default	Description
VTX_INST	4:0	none	Opcode for this vertex fetch instruction. POSSIBLE VALUES: 00 - SQ_VTX_INST_FETCH: vertex fetch (X = uint32 index). Not for use with MEM_RD_WORD* or GDS_WORD* encodings. 01 - SQ_VTX_INST_SEMANTIC: semantic vertex fetch. Not for use with MEM_RD_WORD* or GDS_WORD* encodings. 14 - SQ_VTX_INST_GET_BUFFER_RESINFO: returns the number of elements in a buffer. This is a VERTEX-FETCH and uses a vertex constant, and can only be serviced by TC, not by VC.
FETCH_TYPE	6:5	none	Specify which index offset to send to VC. POSSIBLE VALUES: 00 - SQ_VTX_FETCH_VERTEX_DATA 01 - SQ_VTX_FETCH_INSTANCE_DATA 02 - SQ_VTX_FETCH_NO_INDEX_OFFSET
FETCH_WHOLE_QUAD	7	none	If set, texture instruction will fetch data for all pixels in any quad which as at least one pixel is both active and valid (result may be used as source coordinate of a dependent read). If cleared, texture instruction can ignore inactive pixels. Set this only in PS stage.
BUFFER_ID	15:8	none	Constant ID to use for this vertex fetch (indicates the buffer address, size, and format).
SRC_GPR	22:16	none	Source GPR address to get fetch address from.
SRC_REL	23	none	Indicate whether source address is absolute or relative to an index.

			POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
SRC_SEL_X	25:24	none	Indicate which component of src to use for the fetch address. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component
SRC_SEL_Y	27:26	none	Return data`s LDS write address (from Y), if LDS_REQ is enabled. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component
STRUCTURED_READ	29:28	none	Structured Read Mode, 0/1/2: nomal_read/struct_rd_offset_in_gpr/struct_rd_offset_in_instr. POSSIBLE VALUES: 00 - SQ_VTX_STRU_READ_OFF: Noraml vertex fetch: not from structured buffer 01 - SQ_VTX_STRU_READ_GPR_OFFSET: Vertex fetch from structured buffer with offset from GPR 02 - SQ_VTX_STRU_READ_INST_OFFSET: Vertex fetch from structured buffer with offset from Instruction
LDS_REQ	30	none	Retrun data will write to LDS instead of GPR, if it is enabled.
COALESCED_READ	31	none	This is a coalesced vfetch

SQ_MICRO:SQ_VTX_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Vertex fetch clause instruction word 1 is the bitwise OR of WORD1 | WORD1_{GPR,SEM}. This part contains fields shared by both subencodings.*			
Field Name	Bits	Default	Description
DST_SEL_X	11:9	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0

			06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Y	14:12	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Z	17:15	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_W	20:18	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
USE_CONST_FIELDS	21	none	If set, use format given in the fetch constant instead of in this instruction.
DATA_FORMAT	27:22	none	Indicate vertex data format (ignored if USE_CONST_FIELDS = 1).
NUM_FORMAT_ALL	29:28	none	Format of returning data (N is the number of bits derived from DATA_FORMAT and gamma) (ignored if USE_CONST_FIELDS = 1). POSSIBLE VALUES:

			00 - SQ_NUM_FORMAT_NORM: repeating fraction number (0.N) with range [0, 1] if unsigned, or [-1, 1] if signed. 01 - SQ_NUM_FORMAT_INT: integer number (N.0) with range [0, 2^N] if unsigned, or [-2^M, 2^M] if signed (M = N - 1). 02 - SQ_NUM_FORMAT_SCALED: integer number stored as a S23E8 floating-point representation (1 == 0x3f800000).
FORMAT_COMP_ALL	30	none	Indicate sign of source components (ignored if USE_CONST_FIELDS = 1). POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED
SRF_MODE_ALL	31	none	Mapping to use when converting from signed RF to float (ignored if USE_CONST_FIELDS = 1). POSSIBLE VALUES: 00 - SQ_SRF_MODE_ZERO_CLAMP_MINUS_ONE: representation with two -1 representations (one is slightly past -1 but clamped). 01 - SQ_SRF_MODE_NO_ZERO: OpenGL format lacking representation for 0.

SQ_MICRO:SQ_VTX_WORD1_GPR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Vertex fetch clause instruction word 1. This subencoding is used by fetch instructions that specify a destination GPR directly.*

Field Name	Bits	Default	Description
DST_GPR	6:0	none	Destination GPR address to write result to.
DST_REL	7	none	Indicate whether destination address is absolute or relative to an index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.

SQ_MICRO:SQ_VTX_WORD1_SEM · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Vertex fetch clause instruction word 1. This subencoding is used by semantic fetch instructions that specify the destination using a semantic table.*

Field Name	Bits	Default	Description
SEMANTIC_ID	7:0	none	Specify the 8-bit semantic ID used to lookup the destination GPR from the semantic table.

SQ_MICRO:SQ_VTX_WORD2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Vertex fetch clause instruction word 2.*			
Field Name	Bits	Default	Description
OFFSET	15:0	none	Offset to begin reading from. Byte-aligned.
ENDIAN_SWAP	17:16	none	Endian control (ignored if USE_CONST_FIELDS = 1). POSSIBLE VALUES: 00 - SQ_ENDIAN_NONE: no endian swap (XOR by 0) 01 - SQ_ENDIAN_8IN16: 8 bit swap in 16 bit word (XOR by 1): AABBCCDD -> BBAADDCC 02 - SQ_ENDIAN_8IN32: 8 bit swap in 32 bit word (XOR by 3): AABBCCDD -> DDCCBBAA
CONST_BUF_NO_STRIDE	18	none	If set, force stride to zero for constant buffer fetches that use absolute addresses.
ALT_CONST	20	none	if set, uses constants from alternate thread type: ps->vs, vs->gs, gs->vs, es->gs (note that es and vs share constants). Has no effect on HS, LS, CS.
BUFFER_INDEX_MODE	22:21	none	add index0 or index1 to the vertex buffer resource ID#, or not. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid

SQ_MICRO:SQ_TEX_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Texture fetch clause instruction word 0.*			
Field Name	Bits	Default	Description
TEX_INST	4:0	none	Opcode for this texture instruction. POSSIBLE VALUES: 00 - RESERVED_0: Reserved for SQ_VTX_INST_FETCH value in VTX_INST field of SQ_VTX_WORD0. 01 - RESERVED_1: Reserved for SQ_VTX_INST_SEMANTIC value in VTX_INST field of SQ_VTX_WORD0. 02 - RESERVED_2: Reserved for SQ_MEM_INST_MEM value in MEM_INST field of SQ_MEM_RD_WORD0. 03 - SQ_TEX_INST_LD: fetch texel, XYZL are uint32

			04 - SQ_TEX_INST_GET_TEXTURE_RESINFO: retrieve width, height, depth, number of mipmap levels 05 - SQ_TEX_INST_GET_NUMBER_OF_SAMPLES: retrieve width, height, depth, number of samples of an MSAA surface 06 - SQ_TEX_INST_GET_LOD: X = computed LOD for all pixels in quad 07 - SQ_TEX_INST_GET_GRADIENTS_H: slopes relative to horizontal: X = dx/dh, Y = dy/dh, Z = dz/dh, W = dw/dh 08 - SQ_TEX_INST_GET_GRADIENTS_V: slopes relative to vertical: X = dx/dv, Y = dy/dv, Z = dz/dv, W = dw/dv 09 - SQ_TEX_INST_SET_TEXTURE_OFFSETS: sets texture offsets from a gpr for use with gather4_o and gather4_c_o 10 - SQ_TEX_INST_KEEP_GRADIENTS: compute gradients from coordinates and store them 11 - SQ_TEX_INST_SET_GRADIENTS_H: XYZ set horizontal gradients 12 - SQ_TEX_INST_SET_GRADIENTS_V: XYZ set vertical gradients 13 - SQ_TEX_INST_PASS: returns the address read in memory 14 - RESERVED_14: Reserved for SQ_VTX_INST_GET_BUFFER_RESINFO. 15 - RESERVED_15 16 - SQ_TEX_INST_SAMPLE 17 - SQ_TEX_INST_SAMPLE_L 18 - SQ_TEX_INST_SAMPLE_LB 19 - SQ_TEX_INST_SAMPLE_LZ 20 - SQ_TEX_INST_SAMPLE_G 21 - SQ_TEX_INST_GATHER4: fetches unfiltered texels from a bilinear sample, packs into xyzw 22 - SQ_TEX_INST_SAMPLE_G_LB 23 - SQ_TEX_INST_GATHER4_O 24 - SQ_TEX_INST_SAMPLE_C 25 - SQ_TEX_INST_SAMPLE_C_L 26 - SQ_TEX_INST_SAMPLE_C_LB 27 - SQ_TEX_INST_SAMPLE_C_LZ 28 - SQ_TEX_INST_SAMPLE_C_G 29 - SQ_TEX_INST_GATHER4_C 30 - SQ_TEX_INST_SAMPLE_C_G_LB 31 - SQ_TEX_INST_GATHER4_C_O
INST_MOD	6:5	none	Instruction modifier. Different meaning for different tex_insts. Used for: LD, GetGradientsH/V and Gather4.
FETCH_WHOLE_QUAD	7	none	If set, texture instruction will fetch data for all pixels in any quad which as at least one pixel both active and valid (result may be used as source coordinate of a dependent read). If cleared, texture instruction can ignore inactive pixels.

RESOURCE_ID	15:8	none	Surface ID to read from (specifies the buffer address, size, and format). 160 available for GS and PS; 176 shared across FS and VS.
SRC_GPR	22:16	none	Source GPR address to get the texture lookup address from.
SRC_REL	23	none	Indicate whether source address is absolute or relative to an index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
ALT_CONST	24	none	if set, uses constants from alternate thread type: ps->vs, vs->gs, gs->vs, es->gs (note that es and vs share constants). Has no effect on HS, LS, CS.
RESOURCE_INDEX_MODE	26:25	none	add index0 or index1 to the resource ID#, or not. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
SAMPLER_INDEX_MODE	28:27	none	add index0 or index1 to the sampler ID#, or not. POSSIBLE VALUES: 00 - SQ_CF_INDEX_NONE: do not index the constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid

SQ_MICRO:SQ_TEX_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Texture fetch clause instruction word 1.*

Field Name	Bits	Default	Description
DST_GPR	6:0	none	Destination GPR address to write result to.
DST_REL	7	none	Indicate whether destination address is absolute or relative to an index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.

DST_SEL_X	11:9	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Y	14:12	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Z	17:15	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_W	20:18	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved

			07 - SQ_SEL_MASK: mask out this component
LOD_BIAS	27:21	none	Constant LOD bias to add to the computed bias for this lookup. Twos-complement S3.4 fixpoint value with range [-4, 4).
COORD_TYPE_X	28	none	Indicate the type of the src.XYZW component. POSSIBLE VALUES: 00 - SQ_TEX_UNNORMALIZED: Component is in [0, dim); repeat and mirror modes unavailable. 01 - SQ_TEX_NORMALIZED: Component is in [0, 1]; repeat and mirror modes available.
COORD_TYPE_Y	29	none	Indicate the type of the src.XYZW component. POSSIBLE VALUES: 00 - SQ_TEX_UNNORMALIZED: Component is in [0, dim); repeat and mirror modes unavailable. 01 - SQ_TEX_NORMALIZED: Component is in [0, 1]; repeat and mirror modes available.
COORD_TYPE_Z	30	none	Indicate the type of the src.XYZW component. POSSIBLE VALUES: 00 - SQ_TEX_UNNORMALIZED: Component is in [0, dim); repeat and mirror modes unavailable. 01 - SQ_TEX_NORMALIZED: Component is in [0, 1]; repeat and mirror modes available.
COORD_TYPE_W	31	none	Indicate the type of the src.XYZW component. POSSIBLE VALUES: 00 - SQ_TEX_UNNORMALIZED: Component is in [0, dim); repeat and mirror modes unavailable. 01 - SQ_TEX_NORMALIZED: Component is in [0, 1]; repeat and mirror modes available.

SQ_MICRO:SQ_TEX_WORD2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Texture fetch clause instruction word 2.*			
Field Name	Bits	Default	Description
OFFSET_X	4:0	none	Value added to X component of texel address before sampling (in texel space). S3.1 fixpoint value ranging from [-8, 8).
OFFSET_Y	9:5	none	Value added to Y component of texel address before sampling (in texel space). S3.1 fixpoint value ranging from [-8, 8).
OFFSET_Z	14:10	none	Value added to Z component of texel address before sampling (in texel space). S3.1 fixpoint value ranging from [-8, 8).
SAMPLER_ID	19:15	none	Sampler ID to use (specifies filter options, etc.). Value in the range [0, 17].

SRC_SEL_X	22:20	none	Indicate component source for src.XYZW. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
SRC_SEL_Y	25:23	none	Indicate component source for src.XYZW. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
SRC_SEL_Z	28:26	none	Indicate component source for src.XYZW. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
SRC_SEL_W	31:29	none	Indicate component source for src.XYZW. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0

SQ_MICRO:SQ_MEM_GDS_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Memory: Global data share instruction word 0.*			
Field Name	Bits	Default	Description
MEM_INST	4:0	none	Opcode, borrowed from vertex fetch instruction set. Must be SQ_MEM_INST_MEM. <u>POSSIBLE VALUES:</u> 02 - SQ_MEM_INST_MEM: memory read/write. This opcode is exlusively for MEM_RD_WORD* and GDS_WORD* encodings.
MEM_OP	10:8	none	Sub-opcode for GDS read/writes or TF-buffer writes.

			The sub-opcode MUST match the CF_INST opcode used to issue the clause, as indicated below. POSSIBLE VALUES: 00 - SQ_MEM_OP_RD_SCRATCH: Scratch (temp) buffer read. Only for use in CF_INST_VC/TC[_ACK] clause. 02 - SQ_MEM_OP_RD_SCATTER: Scatter (mem-export) buffer read. Only for use in CF_INST_VC/TC[_ACK] clause. 04 - SQ_MEM_OP_GDS: Global Data sharing read or write. Only for use in CF_INST_GDS clause. 05 - SQ_MEM_OP_TF_WRITE: Tesselation Buffer write. Only for use in CF_INST_GDS clause.
SRC_GPR	17:11	none	Source GPR (supplies data to GDS or TF-buffer). TF_write: X=(tf_idx + tf_base), Y=tf_lod, Z=unused.
SRC_REL_MODE	19:18	none	Indicate whether source-GPR is absolute or relative to an index, or global-gpr. POSSIBLE VALUES: 00 - SQ_REL_NONE: Normal mode - no offset applied to GPR address. 01 - SQ_REL_LOOP: add current loop index value. 02 - SQ_REL_GLOBAL: treat gpr address as absolute, not thread-relative.
SRC_SEL_X	22:20	none	Select source component from GPR.xzyw01. Unused components should be set to zero. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
SRC_SEL_Y	25:23	none	Select source component from GPR.xzyw01. Unused components should be set to zero. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
SRC_SEL_Z	28:26	none	Select source component from GPR.xzyw01. Unused components should be set to zero. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component

			01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0

SQ_MICRO:SQ_MEM_GDS_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Memory: Global data share instruction word 1.*			
Field Name	Bits	Default	Description
DST_GPR	6:0	none	For GDS ops which return data, indicates which GPR to return data to. Ops which return 1 value write the X channel, ops which return 2 values write to X and Y. Ignored if not return value or TF_WRITE.
DST_REL_MODE	8:7	none	Indicate whether source-GPR is absolute or relative to an index, or global-gpr. Ignored if not return value or TF_WRITE. POSSIBLE VALUES: 00 - SQ_REL_NONE: Normal mode - no offset applied to GPR address. 01 - SQ_REL_LOOP: add current loop index value. 02 - SQ_REL_GLOBAL: treat gpr address as absolute, not thread-relative.
GDS_OP	14:9	none	Global Data Share operation. Ignored for TF-write. POSSIBLE VALUES: 00 - SQ_DS_INST_ADD: OP(dst,src, ...) dst=src0_sel, src=src1_sel. 1A1D ADD(dst,src) : DS(dst) += src. dst is src0_sel, src is src1_sel. 01 - SQ_DS_INST_SUB: 1A1D SUB(dst,src) : DS(dst) = DS(dst) - src 02 - SQ_DS_INST_RSUB: 1A1D RSUB(dst,src): DS(dst) = src - DS(dst) 03 - SQ_DS_INST_INC: 1A1D INC(dst) : (DS(dst)>=src) ? DS(dst) = 0 : DS(dst)++ 04 - SQ_DS_INST_DEC: 1A1D DEC(dst) : DS(dst) = ((DS(dst)==0) || (DS(dst)>src)) ? src : DS(dst)-1 05 - SQ_DS_INST_MIN_INT: 1A1D MIN(dst,src) : DS(dst) = min (DS(dst),src) 06 - SQ_DS_INST_MAX_INT: 1A1D MAX(dst,src) : DS(dst) = max(DS(dst),src) 07 - SQ_DS_INST_MIN_UINT: 1A1D MIN(dst,src) : DS(dst) = min (DS(dst),src) 08 - SQ_DS_INST_MAX_UINT: 1A1D MAX(dst,src) : DS(dst) = max(DS(dst),src) 09 - SQ_DS_INST_AND: 1A1D AND(dst,src) : DS(dst) &= src 10 - SQ_DS_INST_OR: 1A1D OR(dst,src) : DS(dst)

			|= src 11 - SQ_DS_INST_XOR: 1A1D XOR(dst,src) : DS(dst) ^= src 12 - SQ_DS_INST_MSKOR: 1A2D MKSOR(dst,mask,src) : DS(dst) = ((DS(dst) & ~msk) | src). 13 - SQ_DS_INST_WRITE: 1A1D WRITE(dst,src) : DS(dst) = src 14 - SQ_DS_INST_WRITE_REL: 1A2D WRITEREL(dst,src0,src1) : tmp = dst + sq_DS_idx_offset (offset in dwords). DS(dst) = src0, DS(tmp) = src1 15 - SQ_DS_INST_WRITE2: 1A2D WRITE2(dst,src0,src1) : tmp = dst+(sq_DS_idx_offset * 64). DS(dst) = src0, DS(tmp) = src1 16 - SQ_DS_INST_CMP_STORE: 1A2D CMP_STORE(dst, cmp, src) : DS(dst) = (DS(dst) == cmp) ? src : DS(dst) 17 - SQ_DS_INST_CMP_STORE_SPF: 1A2D CMP_STORE_SPF(dst, cmp, src) : DS(dst) = (DS(dst) == cmp) ? src : DS(dst) 18 - SQ_DS_INST_BYTE_WRITE: 1A1D BYTEWRITE (dst, src) : DS(dst) = src[7:0] 19 - SQ_DS_INST_SHORT_WRITE: 1A1D SHORTWRITE(dst, src) : DS(dst) = src[15:0] 32 - SQ_DS_INST_ADD_RET: 1A1D ADD(dst,src) : OQA=DS(dst), DS(dst) += src. dst is src0_sel, src is src1_sel. 33 - SQ_DS_INST_SUB_RET: 1A1D SUB(dst,src) : OQA=DS(dst), DS(dst) = DS(dst) - src 34 - SQ_DS_INST_RSUB_RET: 1A1D RSUB(dst,src) : OQA=DS(dst), DS(dst) = src - DS(dst) 35 - SQ_DS_INST_INC_RET: 1A1D INC(dst) : OQA=DS(dst), (DS(dst)>=src) ? DS(dst) = 0 : DS(dst)++ 36 - SQ_DS_INST_DEC_RET: 1A1D DEC(dst) : OQA=DS(dst), DS(dst) = ((DS(dst)==0) || (DS(dst)>src)) ? src : DS(dst)-1 37 - SQ_DS_INST_MIN_INT_RET: 1A1D MIN(dst,src) : OQA=DS(dst), DS(dst) = min (DS(dst),src) 38 - SQ_DS_INST_MAX_INT_RET: 1A1D MAX(dst,src) : OQA=DS(dst), DS(dst) = max(DS(dst),src) 39 - SQ_DS_INST_MIN_UINT_RET: 1A1D MIN(dst,src) : OQA=DS(dst), DS(dst) = min (DS(dst),src) 40 - SQ_DS_INST_MAX_UINT_RET: 1A1D MAX(dst,src) : OQA=DS(dst), DS(dst) = max(DS(dst),src) 41 - SQ_DS_INST_AND_RET: 1A1D AND(dst,src) : OQA=DS(dst), DS(dst) &= src 42 - SQ_DS_INST_OR_RET: 1A1D OR(dst,src) : OQA=DS(dst), DS(dst) |= src

			43 - SQ_DS_INST_XOR_RET: 1A1D XOR(dst,src) : OQA=DS(dst), DS(dst) ^= src 44 - SQ_DS_INST_MSKOR_RET: 1A2D MSKOR(dst,msk,src) : OQA=DS(dst), DS(dst) = ((DS(dst) & ~msk) | src). 45 - SQ_DS_INST_XCHG_RET: 1A1D Exchange(dst,src) : OQA=DS(dst), DS(dst) = src 46 - SQ_DS_INST_XCHG_REL_RET: 1A2D ExchangeRel(dst,src0,src1) : tmp = dst + sq_DS_idx_offset. OQA=DS(dst), OQB=DS(tmp); DS(dst)=src0, DS(tmp)=src1 47 - SQ_DS_INST_XCHG2_RET: 1A2D Exchange2(dst,src0,src1) : tmp = dst + sq_DS_idx_offset*64. OQA=DS(dst), OQB=DS(tmp); DS(dst)=src0, DS(tmp)=src1 48 - SQ_DS_INST_CMP_XCHG_RET: 1A2D CompareExchange(dst,cmp,src) : OQA=DS(dst); (DS(dst)==cmp) ? DS(dst)=src : DS(dst)=DS(dst) 49 - SQ_DS_INST_CMP_XCHG_SPF_RET: 1A2D CompareExchangeSPF(dst,cmp,src) : OQA=DS(dst); (DS(dst)==cmp) ? DS(dst)=src : DS(dst)=DS(dst) 50 - SQ_DS_INST_READ_RET: 1A READ(dst) : OQA = DS(dst) 51 - SQ_DS_INST_READ_REL_RET: 1A READ_REL(dst) : tmp=dst+sq_DS_idx_offset; OQA=DS(dst), OQB=DS(tmp) 52 - SQ_DS_INST_READ2_RET: 2A READ2(dst0,dst1) : OQA=DS(dst0), OQB=DS(dst1) 53 - SQ_DS_INST_READWRITE_RET: 2A1D READWRITE(dst0,dst1,data) : OQA=DS(dst0), DS(dst1)=data 54 - SQ_DS_INST_BYTE_READ_RET: 1A BYTEREAD(dst) : OQA=SignExtend(DS(dst)[7:0]) 55 - SQ_DS_INST_UBYTE_READ_RET: 1A UBYTEREAD(dst) : OQA={24`h0, DS(dst)[7:0]} 56 - SQ_DS_INST_SHORT_READ_RET: 1A SHORTREAD(dst) : OQA=SignExtend(DS(dst)[15:0]} 57 - SQ_DS_INST_USHORT_READ_RET: 1A USHORTREAD(dst) : OQA={16`h0, DS(dst)[15:0]} 63 - SQ_DS_INST_ATOMIC_ORDERED_ALLOC_RET: 1A GDS-only (intercepted by ordered alloc unit). This will add the 7 lsb of 1a to a hidden ordered append count in wave order and return the pre-op value to the specified destination register. This opcode can only be used by GDS and with broadcast first set.
DS_OFFSET	22:16	none	DWORD offset for GDS read or write. Ignored for TF-write.
UAV_INDEX_MODE	25:24	none	add index0, index1, or nothing to the UAV_ID. <u>POSSIBLE VALUES:</u> 00 - SQ_CF_INDEX_NONE: do not index the

			constant buffer 01 - SQ_CF_INDEX_0: add index0 to the constant (CB#/T#/S#/UAV#) number 02 - SQ_CF_INDEX_1: add index1 to the constant (CB#/T#/S#/UAV#) number 03 - SQ_CF_INVALID: invalid
UAV_ID	29:26	none	Identifies append/consume count within group of a context. Do not use with TF.
ALLOC_CONSUME	30	none	1 = accessing append/consume counter. Ignored for TF-write and GDS with no return.
BCAST_FIRST_REQ	31	none	GDS should only process and respond to the first active pixel. Return data is broadcast to all pixels regardless of active status.

SQ_MICRO:SQ_MEM_GDS_WORD2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc

DESCRIPTION: *Memory: Global data share instruction word 2.*

Field Name	Bits	Default	Description
DST_SEL_X	2:0	none	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Y	5:3	none	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Z	8:6	none	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_W	11:9	none	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component

			03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component

SQ_MICRO:SQ_MEM_RD_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Memory-read clause instruction word 0.*			
Field Name	Bits	Default	Description
MEM_INST	4:0	none	Opcode, borrowed from vertex fetch instruction set. Must be SQ_MEM_INST_MEM. POSSIBLE VALUES: 02 - SQ_MEM_INST_MEM: memory read/write. This opcode is exlusively for MEM_RD_WORD* and GDS_WORD* encodings.
ELEM_SIZE	6:5	none	Number of DWORDs per element, minus one. This field is interpreted as a value in [1,2,4] (3 is illegal). The value from INDEX_GPR and is multiplied by this factor, if applicable. Normally, ELEMSIZE = 4 DWORDs for scratch, one DWORD for other types.
FETCH_WHOLE_QUAD	7	none	If set, texture instruction will fetch data for all pixels in any quad which as at least one pixel valid (result may be used as source coordinate of a dependent read). If cleared, texture instruction can ignore inactive pixels. Set this only in PS stage.
MEM_OP	10:8	none	Sub-opcode for memory reads: scratch & scatter. The sub-opcode MUST match the CF_INST opcode used to issue the clause, as indicated below. POSSIBLE VALUES: 00 - SQ_MEM_OP_RD_SCRATCH: Scratch (temp) buffer read. Only for use in CF_INST_VC/TC[_ACK] clause. 02 - SQ_MEM_OP_RD_SCATTER: Scatter (mem-export) buffer read. Only for use in CF_INST_VC/TC[_ACK] clause. 04 - SQ_MEM_OP_GDS: Global Data sharing read or write. Only for use in CF_INST_GDS clause. 05 - SQ_MEM_OP_TF_WRITE: Tesselation Buffer write. Only for use in CF_INST_GDS clause.
UNCACHED	11	none	Uncached (cache-bypass) read. Any time you write and read within one shader pass, it must be uncached.
INDEXED	12	none	Indexed access or not. Indexed includes source-GPR in address calculation.
SRC_SEL_Y	14:13	none	Retrun data`s LDS write address (from Y), if LDS_REQ is enable.

			POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component
SRC_GPR	22:16	none	Source GPR address to get fetch address from.
SRC_REL	23	none	Indicate whether source address is absolute or relative to an index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
SRC_SEL_X	25:24	none	Indicate which component of src to use for the fetch address. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component
BURST_CNT	29:26	none	Burst count 0=1-read, 15=16-reads. Array_base and dst_gpr are incremented for each step in the burst.
LDS_REQ	30	none	The memory read data will write to LDS instead of GPR, if this bit is enabled.
COALESCED_READ	31	none	This is a coalesced mem read.

SQ_MICRO:SQ_MEM_RD_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Memory-Read instruction word 1.*			
Field Name	Bits	Default	Description
DST_GPR	6:0	none	Destination GPR address to write result to.
DST_REL	7	none	Indicate whether destination address is absolute or relative to an index. POSSIBLE VALUES: 00 - SQ_ABSOLUTE: no relative addressing. 01 - SQ_RELATIVE: add current loop index value to this address.
DST_SEL_X	11:9	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component

			03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Y	14:12	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_Z	17:15	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DST_SEL_W	20:18	none	Indicate which component of the result to write to dst.XYZW. Can be used to mask out components when writing to destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0 06 - Reserved 07 - SQ_SEL_MASK: mask out this component
DATA_FORMAT	27:22	none	Indicate vertex data format.
NUM_FORMAT_ALL	29:28	none	Format of returning data (N is the number of bits derived from DATA_FORMAT and gamma). POSSIBLE VALUES: 00 - SQ_NUM_FORMAT_NORM: repeating fraction number (0.N) with range [0, 1] if unsigned, or [-

			1, 1] if signed. 01 - SQ_NUM_FORMAT_INT: integer number (N.0) with range [0, 2^N] if unsigned, or [-2^M, 2^M] if signed (M = N - 1). 02 - SQ_NUM_FORMAT_SCALED: integer number stored as a S23E8 floating-point representation (1 == 0x3f800000).
FORMAT_COMP_ALL	30	none	Indicate sign of source components. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED
SRF_MODE_ALL	31	none	Mapping to use when converting from signed RF to float. POSSIBLE VALUES: 00 - SQ_SRF_MODE_ZERO_CLAMP_MINUS_ONE: representation with two -1 representations (one is slightly past -1 but clamped). 01 - SQ_SRF_MODE_NO_ZERO: OpenGL format lacking representation for 0.

SQ_MICRO:SQ_MEM_RD_WORD2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x8dfc			
DESCRIPTION: *Memory-read clause instruction word 2.*			
Field Name	Bits	Default	Description
ARRAY_BASE	12:0	none	Array base (same definition as export instruction). In elem-sized units (1,2 or 4 dwords), represents values 0..8191.
ENDIAN_SWAP	17:16	none	Endian control (ignored if USE_CONST_FIELDS = 1). POSSIBLE VALUES: 00 - SQ_ENDIAN_NONE: no endian swap (XOR by 0) 01 - SQ_ENDIAN_8IN16: 8 bit swap in 16 bit word (XOR by 1): AABBCCDD -> BBAADDCC 02 - SQ_ENDIAN_8IN32: 8 bit swap in 32 bit word (XOR by 3): AABBCCDD -> DDCCBBAA
ARRAY_SIZE	31:20	none	Array size (same definition as export instruction). In elem-sized units (1,2 or 4 dwords), represents values 1..4096. Used only for SCRATCH read (no effect on scatter)

5. Shader Vertex Resource Constants

SQ:SQ_VTX_CONSTANT_WORD0_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30000			
DESCRIPTION: *Vertex fetch constant state, word 0. Offsets and sizes are defined by SQ_FETCH_RESOURCE_<wavetype>_OFFSET and SQ_FETCH_RESOURCE_<wavetype>_COUNT. PS=0..175, VS/ES=176..335, GS=336..495, HS=496..655, LS=656..815, CS=816..991, FS=992..1023. Texture resources and vertex fetch constants share the same physical registers. There is a stride of 8 dwords between resources.*			
Field Name	Bits	Default	Description
BASE_ADDRESS	31:0	0x0	Base address of buffer, aligned in bytes. Low 32 bits only; see also BASE_ADDRESS_HI.

SQ:SQ_VTX_CONSTANT_WORD1_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30004			
DESCRIPTION: *Vertex fetch constant state, word 1.*			
Field Name	Bits	Default	Description
SIZE	31:0	0x0	Size of buffer minus 1, in bytes.

SQ:SQ_VTX_CONSTANT_WORD2_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30008			
DESCRIPTION: *Vertex fetch constant state, word 2.*			
Field Name	Bits	Default	Description
BASE_ADDRESS_HI	7:0	0x0	Base address of buffer, aligned in bytes. High 8 bits.
STRIDE	19:8	0x0	Stride of the vertex element, in bytes.
DATA_FORMAT	25:20	0x0	Vertex data format.
NUM_FORMAT_ALL	27:26	0x0	Format of returning data (N is the number of bits derived from DATA_FORMAT and gamma). POSSIBLE VALUES: 00 - SQ_NUM_FORMAT_NORM: repeating fraction number (0.N) with range [0, 1] if unsigned, or [-1, 1] if signed. 01 - SQ_NUM_FORMAT_INT: integer number (N.0) with range [0, 2^N] if unsigned, or [-2^M, 2^M] if signed (M = N - 1). 02 - SQ_NUM_FORMAT_SCALED: integer number stored as a S23E8 floating-point representation (1 == 0x3f800000).
FORMAT_COMP_ALL	28	0x0	Sign of source components. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED
SRF_MODE_ALL	29	0x0	Mapping to use when converting from signed RF to

			float. POSSIBLE VALUES: 00 - SQ_SRF_MODE_ZERO_CLAMP_MINUS_ONE: representation with two -1 representations (one is slightly past -1 but clamped). 01 - SQ_SRF_MODE_NO_ZERO: OpenGL format lacking representation for 0.
ENDIAN_SWAP	31:30	0x0	Endian control. POSSIBLE VALUES: 00 - SQ_ENDIAN_NONE: no endian swap (XOR by 0) 01 - SQ_ENDIAN_8IN16: 8 bit swap in 16 bit word (XOR by 1): AABBCCDD -> BBAADDCC 02 - SQ_ENDIAN_8IN32: 8 bit swap in 32 bit word (XOR by 3): AABBCCDD -> DDCCBBAA

SQ:SQ_VTX_CONSTANT_WORD3_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3000c			
DESCRIPTION: *Vertex fetch constant state, word 3.*			
Field Name	Bits	Default	Description
CACHE_SWIZZLE	0	0x0	Cache Swizzle.
UNCACHED	2	0x0	Uncached (auto-invalidate) memory read. Typically used for scratch and mem-export reads.
DST_SEL_X	5:3	0x0	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_Y	8:6	0x0	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_Z	11:9	0x0	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_W	14:12	0x0	POSSIBLE VALUES: 00 - SQ_SEL_X: use X component

			01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0

SQ:SQ_VTX_CONSTANT_WORD4_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30010			
DESCRIPTION: *Vertex fetch constant state, word 4.*			
Field Name	Bits	Default	Description
NUM_ELEMENTS	31:0	0x0	Number of elements in vertex buffer. Used for bufinfo.

SQ:SQ_VTX_CONSTANT_WORD7_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3001c			
DESCRIPTION: *Vertex fetch constant state, word 7.*			
Field Name	Bits	Default	Description
TYPE	31:30	0x0	Constant type/state. POSSIBLE VALUES: 00 - SQ_TEX_VTX_INVALID_TEXTURE 01 - SQ_TEX_VTX_INVALID_BUFFER 02 - SQ_TEX_VTX_VALID_TEXTURE 03 - SQ_TEX_VTX_VALID_BUFFER

6. Shader Texture Resource Constants

SQ:SQ_TEX_RESOURCE_WORD0_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30000			
DESCRIPTION: *Texture resource state, word 0. Offsets and sizes are defined by SQ_FETCH_RESOURCE_<wavetype>_OFFSET and SQ_FETCH_RESOURCE_<wavetype>_COUNT. PS=0..175, VS/ES=176..335, GS=336..495, HS=496..655, LS=656..815, CS=816..991, FS=992..1023. Texture resources and vertex fetch constants share the same physical registers. There is a stride of 8 dwords between resources.*			
Field Name	Bits	Default	Description
DIM	2:0	0x0	Texture dimensionality. POSSIBLE VALUES: 00 - SQ_TEX_DIM_1D 01 - SQ_TEX_DIM_2D 02 - SQ_TEX_DIM_3D 03 - SQ_TEX_DIM_CUBEMAP 04 - SQ_TEX_DIM_1D_ARRAY 05 - SQ_TEX_DIM_2D_ARRAY 06 - SQ_TEX_DIM_2D_MSAA 07 - SQ_TEX_DIM_2D_ARRAY_MSAA
IGNORE_SHADER_ENGINE_TILING	3	0x0	When asserted TC will force NUM_SHADER_ENGINES to 0 into the addrlib.
NON_DISP_TILING_ORDER	5:4	0x0	Memory tiling type (Color vs. Depth).
PITCH	17:6	0x0	Pitch minus 1, in units of 8 texels (maximum pitch is 16384).
TEX_WIDTH	31:18	0x0	Width of the texture minus 1.

SQ:SQ_TEX_RESOURCE_WORD1_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30004			
DESCRIPTION: *Texture resource state, word 1.*			
Field Name	Bits	Default	Description
TEX_HEIGHT	13:0	0x0	Height of the texture minus 1; number of stacks minus 1, for 1D arrays.
TEX_DEPTH	26:14	0x0	Depth of the texture minus 1; number of stacks minus 1, for 2D arrays.
ARRAY_MODE	31:28	0x0	Memory tiling control.

SQ:SQ_TEX_RESOURCE_WORD2_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30008			
DESCRIPTION: *Texture resource state, word 2.*			
Field Name	Bits	Default	Description
BASE_ADDRESS	31:0	0x0	256byte-aligned address of the base map (level 0).

SQ:SQ_TEX_RESOURCE_WORD3_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3000c			
DESCRIPTION: *Texture resource state, word 3.*			
Field Name	Bits	Default	Description
MIP_ADDRESS	31:0	0x0	256byte-aligned address of the second map (level 1), indicating start of the remaining monolithic mipmap chain.

SQ:SQ_TEX_RESOURCE_WORD4_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30010			
DESCRIPTION: *Texture resource state, word 4.*			
Field Name	Bits	Default	Description
FORMAT_COMP_X	1:0	0x0	Sign control for each channel of the data in memory. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED 02 - SQ_FORMAT_COMP_UNSIGNED_BIASED
FORMAT_COMP_Y	3:2	0x0	Sign control for each channel of the data in memory. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED 02 - SQ_FORMAT_COMP_UNSIGNED_BIASED
FORMAT_COMP_Z	5:4	0x0	Sign control for each channel of the data in memory. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED 02 - SQ_FORMAT_COMP_UNSIGNED_BIASED
FORMAT_COMP_W	7:6	0x0	Sign control for each channel of the data in memory. POSSIBLE VALUES: 00 - SQ_FORMAT_COMP_UNSIGNED 01 - SQ_FORMAT_COMP_SIGNED 02 - SQ_FORMAT_COMP_UNSIGNED_BIASED
NUM_FORMAT_ALL	9:8	0x0	Format of returning data (N is the number of bits derived from DATA_FORMAT and gamma). POSSIBLE VALUES: 00 - SQ_NUM_FORMAT_NORM: repeating fraction number (0.N) with range [0, 1] if unsigned, or [-1, 1] if signed. 01 - SQ_NUM_FORMAT_INT: integer number (N.0) with range [0, 2^N] if unsigned, or [-2^M, 2^M] if signed (M = N - 1). 02 - SQ_NUM_FORMAT_SCALED: integer number stored as a S23E8 floating-point representation (1 == 0x3f800000).

SRF_MODE_ALL	10	0x0	Mapping to use when converting from signed RF/unsigned RF biased to float. POSSIBLE VALUES: 00 - SQ_SRF_MODE_ZERO_CLAMP_MINUS_ONE: representation with two -1 representations (one is slightly past -1 but clamped). 01 - SQ_SRF_MODE_NO_ZERO: OpenGL format lacking representation for 0.
FORCE_DEGAMMA	11	0x0	If set, forces degamma on XYZ if format is FMT_8_8_8_8, FMT_BC1, FMT_BC2, or FMT_BC3 (required for DX9 compliance). If cleared, normal behaviour unless FORCE_DEGAMMA in sampler state is set.
ENDIAN_SWAP	13:12	0x0	Endian control. POSSIBLE VALUES: 00 - SQ_ENDIAN_NONE: no endian swap (XOR by 0) 01 - SQ_ENDIAN_8IN16: 8 bit swap in 16 bit word (XOR by 1): AABBCCDD -> BBAADDCC 02 - SQ_ENDIAN_8IN32: 8 bit swap in 32 bit word (XOR by 3): AABBCCDD -> DDCCBBAA
LOG2_NUM_FRAGMENTS	15:14	0x0	Log2 of fragment.
DST_SEL_X	18:16	0x0	What component of returning data to write to each channel of the destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_Y	21:19	0x0	What component of returning data to write to each channel of the destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_Z	24:22	0x0	What component of returning data to write to each channel of the destination GPR. POSSIBLE VALUES: 00 - SQ_SEL_X: use X component

			01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
DST_SEL_W	27:25	0x0	What component of returning data to write to each channel of the destination GPR. <u>POSSIBLE VALUES:</u> 00 - SQ_SEL_X: use X component 01 - SQ_SEL_Y: use Y component 02 - SQ_SEL_Z: use Z component 03 - SQ_SEL_W: use W component 04 - SQ_SEL_0: use constant 0.0 05 - SQ_SEL_1: use constant 1.0
BASE_LEVEL	31:28	0x0	Absolute index of largest mipmap level we can use (0 == largest), applied post-min/mag determination. Has precedence over NUM_LEVELS. Effectively redefines the size of hte base map to the one BASE_LEVEL points at (affects min/mag switch point).

SQ:SQ_TEX_RESOURCE_WORD5_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30014			
DESCRIPTION: *Texture resource state, word 5.*			
Field Name	Bits	Default	Description
LAST_LEVEL	3:0	0x0	Absolute index of smallest mipmap level we can use (should be >= BASE_LEVEL). The value is NOT relative to BASE_LEVEL.
BASE_ARRAY	16:4	0x0	Absolute index of first valid array slice to use.
LAST_ARRAY	29:17	0x0	Absolute index of last valid array slice to use. For cubemaps and cubemap arrays, LAST_ARRAY must be programmed with BASE_ARRAY + (N*6) - 1, where N is the number of cubemaps in the array, or N=1 for a single cubemap.

SQ:SQ_TEX_RESOURCE_WORD6_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x30018			
DESCRIPTION: *Texture resource state, word 6.*			
Field Name	Bits	Default	Description
Reserved	2:0	0x0	
PERF_MODULATION	5:3	0x0	Scale factor used to scale down sampler`s PERF_MIP, PERF_Z, and LOD_BIAS_SEC values. Effectively corresponds to a 3b RF number used to scale the above-mentioned values.
INTERLACED	6	0x0	If set, the format is stored interlaced.
MIN_LOD	19:8	0x0	MIN LOD clamp, applied after sampler clamp. U4.8 format.

TILE_SPLIT	31:29	0x0	Specifies the number of bytes that will be stored contiguously for each tile. If the tile data requires more storage than this amount, it is split into multiple slices. This field must not be larger than GB_ADDR_CONFIG.DRAM_ROW_SIZE. Applies only to 2D tiling modes. POSSIBLE VALUES: 00 - SQ_ADDR_SURF_TILE_SPLIT_64B 01 - SQ_ADDR_SURF_TILE_SPLIT_128B 02 - SQ_ADDR_SURF_TILE_SPLIT_256B 03 - SQ_ADDR_SURF_TILE_SPLIT_512B 04 - SQ_ADDR_SURF_TILE_SPLIT_1KB 05 - SQ_ADDR_SURF_TILE_SPLIT_2KB 06 - SQ_ADDR_SURF_TILE_SPLIT_4KB

SQ:SQ_TEX_RESOURCE_WORD7_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3001c			
DESCRIPTION: *Texture resource state, word 7.*			
Field Name	Bits	Default	Description
DATA_FORMAT	5:0	0x0	Data format of each texel in the texture.
MACRO_TILE_ASPECT	7:6	0x0	Specifies the macro tile aspect ratio. POSSIBLE VALUES: 00 - SQ_ADDR_SURF_MACRO_ASPECT_1 01 - SQ_ADDR_SURF_MACRO_ASPECT_2 02 - SQ_ADDR_SURF_MACRO_ASPECT_4 03 - SQ_ADDR_SURF_MACRO_ASPECT_8
BANK_WIDTH	9:8	0x0	Specifies the number of tiles in the X direction to be incorporated into the same bank. POSSIBLE VALUES: 00 - SQ_ADDR_SURF_BANK_WH_1 01 - SQ_ADDR_SURF_BANK_WH_2 02 - SQ_ADDR_SURF_BANK_WH_4 03 - SQ_ADDR_SURF_BANK_WH_8
BANK_HEIGHT	11:10	0x0	Specifies the number of tiles in the Y direction to be incorporated into the same bank. POSSIBLE VALUES: 00 - SQ_ADDR_SURF_BANK_WH_1 01 - SQ_ADDR_SURF_BANK_WH_2 02 - SQ_ADDR_SURF_BANK_WH_4 03 - SQ_ADDR_SURF_BANK_WH_8
DEPTH_SAMPLE_ORDER	15	0x0	Indicates if samples are stored consecutively per element. Depth and stencil surfaces implicitly utilize this sample order.
NUM_BANKS	17:16	0x0	Specifies the number of banks for tiling purposes.

			POSSIBLE VALUES: 00 - SQ_ADDR_SURF_2_BANK 01 - SQ_ADDR_SURF_4_BANK 02 - SQ_ADDR_SURF_8_BANK 03 - SQ_ADDR_SURF_16_BANK
TYPE	31:30	0x0	Constant type/state. POSSIBLE VALUES: 00 - SQ_TEX_VTX_INVALID_TEXTURE 01 - SQ_TEX_VTX_INVALID_BUFFER 02 - SQ_TEX_VTX_VALID_TEXTURE 03 - SQ_TEX_VTX_VALID_BUFFER

SQ:SQ_TEX_RESOURCE_CLEAR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x3ff04

DESCRIPTION: *Clear all texture/vertex-buffer constants, for all shader stages. write only.*

Field Name	Bits	Default	Description
CLR	31:0	0x0	clear constants. 0xffffffff = clear all constants, other values indicate individual constant number to clear.

7. Shader Texture Sampler Constants

SQ:SQ_TEX_SAMPLER_WORD0_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3c000

DESCRIPTION: *Texture sampler state, word 0. Offsets and sizes are defined by SQ_TEX_SAMPLER_<wavetype>_OFFSET and SQ_TEX_SAMPLER_<wavetype>_COUNT. PS=0..17, VS/ES=18..35, GS=36..53, HS=54..71, LS=72..89, CS=90..107. There is a stride of 3 dwords between samplers.*

Field Name	Bits	Default	Description
CLAMP_X	2:0	0x0	Clamp mode for each texture coordinate. POSSIBLE VALUES: 00 - SQ_TEX_WRAP 01 - SQ_TEX_MIRROR 02 - SQ_TEX_CLAMP_LAST_TEXEL: [0,1] normalized, [0,dimen] unnormalized 03 - SQ_TEX_MIRROR_ONCE_LAST_TEXEL: [-1,1] 04 - SQ_TEX_CLAMP_HALF_BORDER: [0,1] normalized, [0,dimen] unnormalized 05 - SQ_TEX_MIRROR_ONCE_HALF_BORDER: [-1,1] 06 - SQ_TEX_CLAMP_BORDER: [0,1] normalized, [0,dimen] unnormalized 07 - SQ_TEX_MIRROR_ONCE_BORDER: [-1,1]
CLAMP_Y	5:3	0x0	Clamp mode for each texture coordinate. POSSIBLE VALUES: 00 - SQ_TEX_WRAP 01 - SQ_TEX_MIRROR 02 - SQ_TEX_CLAMP_LAST_TEXEL: [0,1] normalized, [0,dimen] unnormalized 03 - SQ_TEX_MIRROR_ONCE_LAST_TEXEL: [-1,1] 04 - SQ_TEX_CLAMP_HALF_BORDER: [0,1] normalized, [0,dimen] unnormalized 05 - SQ_TEX_MIRROR_ONCE_HALF_BORDER: [-1,1] 06 - SQ_TEX_CLAMP_BORDER: [0,1] normalized, [0,dimen] unnormalized 07 - SQ_TEX_MIRROR_ONCE_BORDER: [-1,1]
CLAMP_Z	8:6	0x0	Clamp mode for each texture coordinate. POSSIBLE VALUES: 00 - SQ_TEX_WRAP 01 - SQ_TEX_MIRROR 02 - SQ_TEX_CLAMP_LAST_TEXEL: [0,1] normalized, [0,dimen] unnormalized 03 - SQ_TEX_MIRROR_ONCE_LAST_TEXEL: [-1,1] 04 - SQ_TEX_CLAMP_HALF_BORDER: [0,1]

			normalized, [0,dimen] unnormalized 05 - SQ_TEX_MIRROR_ONCE_HALF_BORDER: [-1,1] 06 - SQ_TEX_CLAMP_BORDER: [0,1] normalized, [0,dimen] unnormalized 07 - SQ_TEX_MIRROR_ONCE_BORDER: [-1,1]
XY_MAG_FILTER	10:9	0x0	Defines the filter to use on X and Y when magnifying/minifying. POSSIBLE VALUES: 00 - SQ_TEX_XY_FILTER_POINT 01 - SQ_TEX_XY_FILTER_BILINEAR
XY_MIN_FILTER	12:11	0x0	Defines the filter to use on X and Y when magnifying/minifying. POSSIBLE VALUES: 00 - SQ_TEX_XY_FILTER_POINT 01 - SQ_TEX_XY_FILTER_BILINEAR
Z_FILTER	14:13	0x0	Defines the filter to use between Z layers/mip levels. POSSIBLE VALUES: 00 - SQ_TEX_Z_FILTER_NONE 01 - SQ_TEX_Z_FILTER_POINT 02 - SQ_TEX_Z_FILTER_LINEAR
MIP_FILTER	16:15	0x0	Defines the filter to use between Z layers/mip levels. POSSIBLE VALUES: 00 - SQ_TEX_Z_FILTER_NONE 01 - SQ_TEX_Z_FILTER_POINT 02 - SQ_TEX_Z_FILTER_LINEAR
Reserved	19:17	0x0	
BORDER_COLOR_TYPE	21:20	0x0	Source of border colour when a border is hit. POSSIBLE VALUES: 00 - SQ_TEX_BORDER_COLOR_TRANS_BLACK: (0.0, 0.0, 0.0, 0.0) 01 - SQ_TEX_BORDER_COLOR_OPAQUE_BLACK: (0.0, 0.0, 0.0, 1.0) 02 - SQ_TEX_BORDER_COLOR_OPAQUE_WHITE: (1.0, 1.0, 1.0, 1.0) 03 - SQ_TEX_BORDER_COLOR_REGISTER: use BORDER_COLOR_[XYZW]
DEPTH_COMPARE_FUNCTION	24:22	0x0	Depth comparison function to use for any SAMPLE_C* instruction. POSSIBLE VALUES:

			00 - SQ_TEX_DEPTH_COMPARE_NEVER: always 0 01 - SQ_TEX_DEPTH_COMPARE_LESS: 1 if incoming Z < fetched data 02 - SQ_TEX_DEPTH_COMPARE_EQUAL: 1 if incoming Z == fetched data 03 - SQ_TEX_DEPTH_COMPARE_LESSEQUAL: 1 if incoming Z <= fetched data 04 - SQ_TEX_DEPTH_COMPARE_GREATER: 1 if incoming Z > fetched data 05 - SQ_TEX_DEPTH_COMPARE_NOTEQUAL: 1 if incoming Z != fetched data 06 - SQ_TEX_DEPTH_COMPARE_GREATEREQUAL: 1 if incoming Z >= fetched data 07 - SQ_TEX_DEPTH_COMPARE_ALWAYS: always 1
FORCE_UNNORMALIZED	25	0x0	Force Un-normalized.
Reserved	29:27	0x0	

SQ:SQ_TEX_SAMPLER_WORD1_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3c004

DESCRIPTION: *Texture sampler state, word 1.*

Field Name	Bits	Default	Description
MIN_LOD	11:0	0x0	LOD lower bound applied to computed LOD before determining min/mag. Format is u4.8, range [0, 16).
MAX_LOD	23:12	0x0	LOD upper bound applied to computed LOD before determining min/mag. Format is u4.8, range [0, 16).
PERF_MIP	27:24	0x0	Performance setting for mipmap transition. Remaps fractional bit of LOD using one of 7 lookup tables (0 = fraction is untouched).
PERF_Z	31:28	0x0	Performance setting for volume layer transition. Remaps fractional bit of Z using one of 4 lookup tables (0 = fraction is untouched).

SQ:SQ_TEX_SAMPLER_WORD2_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3c008

DESCRIPTION: *Texture sampler state, word 2.*

Field Name	Bits	Default	Description
LOD_BIAS	13:0	0x0	LOD bias to apply after computing the mipmap but before applying MIN_LOD (hence before min/mag determination). Bias is 2`s complement s6.8 with range [-32, 32).
LOD_BIAS_SEC	19:14	0x0	Secondary LOD bias added after LOD bias. Used to alleviate small batch issues on OpenGL. Bias is 2`s complement S1.4 with range [-2, 2).
MC_COORD_TRUNCATE	20	0x0	If set, keep only MSB of fractional part of coordinate

			(for MPEG). If cleared, don`t truncate fractions.
FORCE_DEGAMMA	21	0x0	If set, forces degamma on XYZ if format is FMT_8_8_8_8, FMT_BC1, FMT_BC2, or FMT_BC3 (required for DX9 compliance). If cleared, normal behaviour unless FORCE_DEGAMMA in resource state is set.
Reserved	27:22	0x0	
TRUNCATE_COORD	28	0x0	1=truncate when point sampling; 0=round-nearest-even to n.6 and drop fraction when point sampling.
DISABLE_CUBE_WRAP	29	0x0	1=disable cubemap wrapping for non-point sampled cubemaps (dx9). 0=allow cubemap wrapping (dx10)
TYPE	31	0x0	Sampler type. If set, the sampler is valid.

SQ:SQ_TEX_SAMPLER_CLEAR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x3ff00

DESCRIPTION: *Clear all texture sampler constants, for all shader stages. write only.*

Field Name	Bits	Default	Description
CLR	31:0	0x0	clear constants. 0xffffffff = clear all constants, other values indicate individual constant number to clear.

8. Shader Constant Registers

SQ:SQ_ALU_CONST_BUFFER_SIZE_GS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x281c0-0x281fc			
DESCRIPTION: *(8-state). Number of elements in this constant buffer [0..4096], in units of 16 constants (cache lines). Associated with SQ_ALU_CONST_CACHE_GS_0. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*			
Field Name	Bits	Default	Description
DATA	8:0	0x0	Number of constant buffer elements

SQ:SQ_ALU_CONST_BUFFER_SIZE_HS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28f80-0x28fbc			
DESCRIPTION: *(8-state). Number of elements in this constant buffer [0..4096], in units of 16 constants (cache lines). Associated with SQ_ALU_CONST_CACHE_HS_0. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*			
Field Name	Bits	Default	Description
DATA	8:0	0x0	Number of constant buffer elements

SQ:SQ_ALU_CONST_BUFFER_SIZE_LS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28fc0-0x28ffc			
DESCRIPTION: *(8-state). Number of elements in this constant buffer [0..4096], in units of 16 constants (cache lines). Associated with SQ_ALU_CONST_CACHE_LS_0. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*			
Field Name	Bits	Default	Description
DATA	8:0	0x0	Number of constant buffer elements

SQ:SQ_ALU_CONST_BUFFER_SIZE_PS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28140-0x2817c			
DESCRIPTION: *(8-state). Number of elements in this constant buffer [0..4096], in units of 16 constants (cache lines). Associated with SQ_ALU_CONST_CACHE_PS_0. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*			
Field Name	Bits	Default	Description
DATA	8:0	0x0	Number of constant buffer elements

SQ:SQ_ALU_CONST_BUFFER_SIZE_VS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28180-0x281bc			
DESCRIPTION: *(8-state). Number of elements in this constant buffer [0..4096], in units of 16 constants (cache lines). Associated with SQ_ALU_CONST_CACHE_VS_0. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*			
Field Name	Bits	Default	Description

DATA	8:0	0x0	Number of constant buffer elements

SQ:SQ_ALU_CONST_CACHE_GS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x289c0-0x289fc

DESCRIPTION: *(8-state) Base address of constant-buffer #0 used by the constant cache, 256B aligned address [39:8]. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*

Field Name	Bits	Default	Description
DATA	31:0	0x0	TBD

SQ:SQ_ALU_CONST_CACHE_HS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28f00-0x28f3c

DESCRIPTION: *(8-state) Base address of constant-buffer #0 used by the constant cache, 256B aligned address [39:8]. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*

Field Name	Bits	Default	Description
DATA	31:0	0x0	TBD

SQ:SQ_ALU_CONST_CACHE_LS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28f40-0x28f7c

DESCRIPTION: *(8-state) Base address of constant-buffer #0 used by the constant cache, 256B aligned address [39:8]. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*

Field Name	Bits	Default	Description
DATA	31:0	0x0	TBD

SQ:SQ_ALU_CONST_CACHE_PS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28940-0x2897c

DESCRIPTION: *(8-state) Base address of constant-buffer #0 used by the constant cache, 256B aligned address [39:8]. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*

Field Name	Bits	Default	Description
DATA	31:0	0x0	TBD

SQ:SQ_ALU_CONST_CACHE_VS_[0-15] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28980-0x289bc

DESCRIPTION: *(8-state) Base address of constant-buffer #0 used by the constant cache, 256B aligned address [39:8]. Used by both VS and ES shaders. You must always write both CONST_BUFFER_SIZE and CONST_CACHE, unless size=0 in which case you may write only size.*

Field Name	Bits	Default	Description

DATA	31:0	0x0	TBD

SQ:SQ_BOOL_CONST_[0-5] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3a500-0x3a514

DESCRIPTION: *(64-state) DX9 Boolean constants - these are available as input to flow control instructions such as `IF`. Each boolean is 32 bits. PS=0, VS/ES=1, GS=2, HS=3, LS=4, CS=5. The booleans are usable in both dx9 and dx10 modes.*

Field Name	Bits	Default	Description
BOOLEANS	31:0	0x0	32 one-bit booleans for static branching

SQ:SQ_JUMPTABLE_CONST_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3a200

DESCRIPTION: *(64-state) Jumptable constants- shared with loop constants. Used with SQ_CF_INST_JUMPTABLE to select the offset when CF_JUMPTABLE_SEL field is CONST_A through CONST_D.*

Field Name	Bits	Default	Description
CONST_A	7:0	0x0	Jumptable offset A (unsigned)
CONST_B	15:8	0x0	Jumptable offset B (unsigned)
CONST_C	23:16	0x0	Jumptable offset C (unsigned)
CONST_D	31:24	0x0	Jumptable offset D (unsigned)

SQ:SQ_LOOP_BOOL_CLEAR · [W] · 32 bits · Access: 32 · GpuF0MMReg:0x3ff08

DESCRIPTION: *Clear all Loop and boolean constants, for all shader stages. write only.*

Field Name	Bits	Default	Description
CLR	31:0	0x0	clear constants. 0xffffffff = clear all constants, other values indicate individual constant number to clear. Note that loop constants are 0..191, and booleans are 192..197 (corresponding to bool0 to bool5).

SQ:SQ_LOOP_CONST_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3a200

DESCRIPTION: *(64-state) DX9 loop counter constants - these are used to define the behaviour of a programmed loop. There are 96 loop counter constants available - 32 each for the PS, VS/ES, GS, HS, LS, CS. PS=0..31, VS/ES=32..63, GS=64..95, HS=96..127, LS=128..159, CS=160..191. The loop counter is usable in both DX9 and DX10 modes. This version is used for SQ_CF_INST_LOOP and SQ_CF_INST_LOOP_NO_AL statements.*

Field Name	Bits	Default	Description
COUNT	11:0	0x0	Total number of loop iterations (unsigned)
INIT	23:12	0x0	Initial value of loop counter AL (unsigned)
INC	31:24	0x0	Amount loop counter increments after each loop iteration (signed)

SQ:SQ_LOOP_CONST_DX10_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x3a200

DESCRIPTION: *(64-state) DX9 loop counter constants - these are used to define the behaviour of a programmed loop. The loop counter is usable in both DX9 and DX10 modes. This version is used for SQ_CF_INST_LOOP_DX10 statements.*			
Field Name	Bits	Default	Description
COUNT	31:0	0x0	Total number of loop iterations (unsigned)

9. Shader Program Setup Registers

SQ:SQ_PGM_EXPORTS_PS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2884c			
DESCRIPTION: *(8-state). Defines the exports from the Pixel Shader Program.*			
Field Name	Bits	Default	Description
EXPORT_MODE	4:0	0x0	Pixel Shader export mode. bbbbz where bbbb is how many color we export (0-8) and z is export z or not. It is illegal to program this to all zeros.

SQ:SQ_PGM_RESOURCES_2_ES · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28894			
DESCRIPTION: *Additional shader resources*			
Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	
ALLOW_SINGLE_DENORM_OUT	5	0x0	
ALLOW_DOUBLE_DENORM_IN	6	0x0	
ALLOW_DOUBLE_DENORM_OUT	7	0x0	
SINGLE_IEEE_MODE	8	0x0	
DOUBLE_IEEE_MODE	9	0x0	

SQ:SQ_PGM_RESOURCES_2_GS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2887c			
DESCRIPTION: *Additional shader resources*			
Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to

			positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	
ALLOW_SINGLE_DENORM_OUT	5	0x0	
ALLOW_DOUBLE_DENORM_IN	6	0x0	
ALLOW_DOUBLE_DENORM_OUT	7	0x0	
SINGLE_IEEE_MODE	8	0x0	
DOUBLE_IEEE_MODE	9	0x0	

SQ:SQ_PGM_RESOURCES_2_HS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288c0			
DESCRIPTION: *Additional shader resources*			
Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	
ALLOW_SINGLE_DENORM_OUT	5	0x0	
ALLOW_DOUBLE_DENORM_IN	6	0x0	
ALLOW_DOUBLE_DENORM_OUT	7	0x0	
SINGLE_IEEE_MODE	8	0x0	
DOUBLE_IEEE_MODE	9	0x0	

SQ:SQ_PGM_RESOURCES_2_LS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288d8			
DESCRIPTION: *Additional shader resources*			
Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	
ALLOW_SINGLE_DENORM_OUT	5	0x0	
ALLOW_DOUBLE_DENORM_IN	6	0x0	
ALLOW_DOUBLE_DENORM_OUT	7	0x0	
SINGLE_IEEE_MODE	8	0x0	
DOUBLE_IEEE_MODE	9	0x0	

SQ:SQ_PGM_RESOURCES_2_PS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28848			
DESCRIPTION: *Additional shader resources*			
Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	Rounding mode for single-floats POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	Rounding mode for double-floats POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity

			02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	Denormal handling for single floats: 0 = flush input denormals, 1 = allow input denormals.
ALLOW_SINGLE_DENORM_OUT	5	0x0	Denormal handling for single floats: 0 = flush output denormals, 1 = allow output denormals.
ALLOW_DOUBLE_DENORM_IN	6	0x0	Denormal handling for double floats: 0 = flush input denormals, 1 = allow input denormals.
ALLOW_DOUBLE_DENORM_OUT	7	0x0	Denormal handling for double floats: 0 = flush output denormals, 1 = allow output denormals.
SINGLE_IEEE_MODE	8	0x0	IEEE mode for single-floats
DOUBLE_IEEE_MODE	9	0x0	IEEE mode for double-floats

SQ:SQ_PGM_RESOURCES_2_VS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28864

DESCRIPTION: *Additional shader resources*

Field Name	Bits	Default	Description
SINGLE_ROUND	1:0	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
DOUBLE_ROUND	3:2	0x0	POSSIBLE VALUES: 00 - SQ_ROUND_NEAREST_EVEN: Round to nearest even 01 - SQ_ROUND_PLUS_INFINITY: Round to positive infinity 02 - SQ_ROUND_MINUS_INFINITY: Round to negative infinity 03 - SQ_ROUND_TO_ZERO: Round to zero
ALLOW_SINGLE_DENORM_IN	4	0x0	
ALLOW_SINGLE_DENORM_OUT	5	0x0	
ALLOW_DOUBLE_DENORM_IN	6	0x0	
ALLOW_DOUBLE_DENORM_OUT	7	0x0	
SINGLE_IEEE_MODE	8	0x0	
DOUBLE_IEEE_MODE	9	0x0	

SQ:SQ_PGM_RESOURCES_ES · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28890

DESCRIPTION: *(8-state). Resource requirements to run the ES program. Can only read most recent version, not all 8 states.*

Field Name	Bits	Default	Description

NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
USE_LS_CONSTS	16	0x0	1=uses LS consts, 0=uses VS consts
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN. See SP doc for details.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.

SQ:SQ_PGM_RESOURCES_FS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288a8

DESCRIPTION: *(8-state). Resource requirements to run the FS program. The FS shares with either the VS (gs-off) or ES (gs-on) and performs a single allocation equal to the VS+FS or ES+FS resource requirements. The SPI allocates stack space as (VS/ES + FS_stack_size) in the same manner as GPRs. Max_call_depth and fetch_cache_lines will be inherited from the parent shader (VS or ES). Can only read most recent version, not all 8 states.*

Field Name	Bits	Default	Description
NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN.

SQ:SQ_PGM_RESOURCES_GS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28878

DESCRIPTION: *(8-state). Resource requirements to run the GS program. Can only read most recent version, not all 8 states.*

Field Name	Bits	Default	Description
NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.

SQ:SQ_PGM_RESOURCES_HS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288bc

DESCRIPTION: *(8-state). Resource requirements to run the HS program. Can only read most recent version, not all 8 states.*

Field Name	Bits	Default	Description
NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]

DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.

SQ:SQ_PGM_RESOURCES_LS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288d4			
DESCRIPTION: *(8-state). Resource requirements to run the LS program. Can only read most recent version, not all 8 states.*			
Field Name	Bits	Default	Description
NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
USE_VS_CONSTS	16	0x0	1=uses VS constants, 0=uses LS constants
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN. See SP doc for details.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.

SQ:SQ_PGM_RESOURCES_PS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28844			
DESCRIPTION: *(8-state). Resource requirements to run the PS program. Can only read most recent version, not all 8 states.*			
Field Name	Bits	Default	Description
NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.
CLAMP_CONSTS	31	0x0	Clamp ALU constants to [-1.0, 1.0]. Used for shader versions below PS2.0. Applies only to Constant-file constants (not literals) and only to const-file entries 0..7. Other entries are never clamped.

SQ:SQ_PGM_RESOURCES_VS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28860			
DESCRIPTION: *(8-state). Resource requirements to run the VS program. Can only read most recent version, not all 8 states.*			
Field Name	Bits	Default	Description

NUM_GPRS	7:0	0x0	number of GPRs required to run this program [0..127]
STACK_SIZE	15:8	0x0	number of stack entries needed [0..255]
USE_LS_CONSTS	16	0x0	1=uses LS consts, 0=uses VS consts
DX10_CLAMP	21	0x0	DX10 clamp mode. (1 = dx10 mode, 0 = dx9 mode). This applies to all shaders. This affects how the SP output clamp treats NaN. See SP doc for details.
UNCACHED_FIRST_INST	28	0x0	Ensure that the first instruction is not read from the first instruction cache. Should only be used for debugging if there is a problem with the cache.

SQ:SQ_PGM_START_ES · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2888c

DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the export shader (ES)*

Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_FS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288a4

DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the fetch shader (FS)*

Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_GS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28874

DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the geometry shader (GS)*

Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_HS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288b8

DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the hull shader (HS)*

Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_LS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x288d0

DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the Vertex-to-LDS shader (LS)*

Field Name	Bits	Default	Description

PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_PS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28840			
DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the pixel shader (PS)*			
Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

SQ:SQ_PGM_START_VS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2885c			
DESCRIPTION: *(8-state) Memory address of the (256-byte aligned) first CF instruction of the shader code for the vertex shader (VS)*			
Field Name	Bits	Default	Description
PGM_START	31:0	0x0	Format is [39:8]

10. Shader Interpolator Registers

SPI:SPI_BARYC_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286e0			
DESCRIPTION: *Determines set of BARYC values computed by the SPI.*			
Field Name	Bits	Default	Description
PERSP_CENTER_ENA	1:0	0x0	Perspective gradients @ pixel center, only processed when persp gradients are on POSSIBLE VALUES: 00 - Off 01 - On at center 02 - On at centroid
PERSP_CENTROID_ENA	5:4	0x0	Perspective gradients @ pixel centroid, only processed when persp gradients are on POSSIBLE VALUES: 00 - Off 01 - On at centroid 02 - On at center
PERSP_SAMPLE_ENA	9:8	0x0	Perspective gradients @ sample, only processed when persp gradients are on POSSIBLE VALUES: 00 - Off 01 - On
PERSP_PULL_MODEL_ENA	13:12	0x0	Provide I,J,1/W to GPR for pull model interpolation, only processed when persp gradients are on POSSIBLE VALUES: 00 - Off 01 - On
LINEAR_CENTER_ENA	17:16	0x0	Linear gradients @ pixel center, only processed when linear gradients are on POSSIBLE VALUES: 00 - Off 01 - On at center 02 - On at centroid
LINEAR_CENTROID_ENA	21:20	0x0	Linear gradients @ pixel centroid, only processed when linear gradients are on POSSIBLE VALUES: 00 - Off 01 - On at centroid 02 - On at center
LINEAR_SAMPLE_ENA	25:24	0x0	Linear gradients @ sample, only processed when linear

			gradients are on POSSIBLE VALUES: 00 - Off 01 - On

SPI:SPI_CONFIG_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9100			
Field Name	Bits	Default	Description
GPR_WRITE_PRIORITY	17:0	0x2C688	3 bits for each type to set relative priority. PS=[2:0], VS=[5:3], GS=[8:6], ES=[11:9], HS=[14:12], LS=[17:15]

SPI:SPI_CONFIG_CNTL_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x913c			
Field Name	Bits	Default	Description
VTX_DONE_DELAY	3:0	0x0	POSSIBLE VALUES: 00 - delay 14 clks (defalut, min value needed for pele config) 01 - delay 16 clks 02 - delay 18 clks 03 - delay 20 clks 04 - delay 22 clks 05 - delay 24 clks 06 - delay 26 clks 07 - delay 28 clks 08 - delay 30 clks 09 - delay 32 clks 10 - delay 34 clks 11 - delay 4 clks 12 - delay 6 clks 13 - delay 8 clks 14 - delay 10 clks 15 - delay 12 clks
INTERP_ONE_PRIM_PER_ROW	4	0x0	POSSIBLE VALUES: 00 - Interpolate two prims per clock, assuming no conflicts (default) 01 - Only interpolate one prim per clock
BC_OPTIMIZE_DISABLE	5	0x0	POSSIBLE VALUES: 00 - Write center/centroid IJ in one clk when possible (default) 01 - Always write 1 IJ per clock
PC_LIMIT_ENABLE	6	0x0	Enable artificial param cache limit based on PC_LIMIT_SIZE. Performance debug feature.
PC_LIMIT_STRICT	7	0x0	If clear, pc alloc fails if head > limit, guaranteeing at least one wave will fit. If set, pc alloc fails if head + space > limit, guaranteeing head never passes limit.
PC_LIMIT_SIZE	31:16	0x100	Artificial limit for SPI param cache allocation, should be set to at least (vs_output_count * num_good_pipes * 2) for all active VS or could cause a deadlock when using

			LIMIT_STRICT.

SPI:SPI_FOG_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286dc			
DESCRIPTION: *Fog interpolation control*			
Field Name	Bits	Default	Description
PASS_FOG_THROUGH_PS	0	0x0	Enables the passing of VS fog from param cache location VS_OUT_FOG_VEC_ADDR.X to the LDS at FOG_ADDR.X

SPI:SPI_GPR_MGMT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286f8			
DESCRIPTION: *Per-context GPR limit for each shader type. This is a soft-limit, meaning allocation only fails once currently allocated space > limit.*			
Field Name	Bits	Default	Description
NUM_PS_GPRS	4:0	0x0	PS GPR limit, format is [7:3]. A setting of 1 means 8 GPR, 31 means 248, and a setting of 0 disables the limit.
NUM_VS_GPRS	9:5	0x0	Same desc as PS
NUM_GS_GPRS	14:10	0x0	Same desc as PS
NUM_ES_GPRS	19:15	0x0	Same desc as PS
NUM_HS_GPRS	24:20	0x0	Same desc as PS
NUM_LS_GPRS	29:25	0x0	Same desc as PS, limit applies to LS and CS.

SPI:SPI_INPUT_Z · [R/W] · 8 bits · Access: 8 · GpuF0MMReg:0x286d8			
Field Name	Bits	Default	Description
PROVIDE_Z_TO_SPI	0	0x0	

SPI:SPI_INTERP_CONTROL_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286d4			
DESCRIPTION: *Interpolator control settings*			
Field Name	Bits	Default	Description
FLAT_SHADE_ENA	0	0x0	Global flat shade enable used in conjunction with per-parameter flat shade control
PNT_SPRITE_ENA	1	0x0	Enable PT_SPRITE_TEX override for point primitives
PNT_SPRITE_OVRD_X	4:2	0x0	POSSIBLE VALUES: 00 - SPI_PNT_SPRITE_SEL_0: Override component with 0.0f 01 - SPI_PNT_SPRITE_SEL_1: Override component with 1.0f 02 - SPI_PNT_SPRITE_SEL_S: Override component with S value 03 - SPI_PNT_SPRITE_SEL_T: Override component with T value 04 - SPI_PNT_SPRITE_SEL_NONE: Keep

			interpolated result
PNT_SPRITE_OVRD_Y	7:5	0x0	POSSIBLE VALUES: 00 - SPI_PNT_SPRITE_SEL_0: Override component with 0.0f 01 - SPI_PNT_SPRITE_SEL_1: Override component with 1.0f 02 - SPI_PNT_SPRITE_SEL_S: Override component with S value 03 - SPI_PNT_SPRITE_SEL_T: Override component with T value 04 - SPI_PNT_SPRITE_SEL_NONE: Keep interpolated result
PNT_SPRITE_OVRD_Z	10:8	0x0	POSSIBLE VALUES: 00 - SPI_PNT_SPRITE_SEL_0: Override component with 0.0f 01 - SPI_PNT_SPRITE_SEL_1: Override component with 1.0f 02 - SPI_PNT_SPRITE_SEL_S: Override component with S value 03 - SPI_PNT_SPRITE_SEL_T: Override component with T value 04 - SPI_PNT_SPRITE_SEL_NONE: Keep interpolated result
PNT_SPRITE_OVRD_W	13:11	0x0	POSSIBLE VALUES: 00 - SPI_PNT_SPRITE_SEL_0: Override component with 0.0f 01 - SPI_PNT_SPRITE_SEL_1: Override component with 1.0f 02 - SPI_PNT_SPRITE_SEL_S: Override component with S value 03 - SPI_PNT_SPRITE_SEL_T: Override component with T value 04 - SPI_PNT_SPRITE_SEL_NONE: Keep interpolated result
PNT_SPRITE_TOP_1	14	0x0	POSSIBLE VALUES: 00 - T is 1.0 at bottom of primitive 01 - T is 1.0 at top of primitive

SPI:SPI_PS_INPUT_CNTL_[0-31] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28644-0x286c0

DESCRIPTION: *PS interpolator settings for parameter 0*

Field Name	Bits	Default	Description
SEMANTIC	7:0	0x0	PS input semantic mapping
DEFAULT_VAL	9:8	0x0	Selects value to force into GPR if no semantic match found POSSIBLE VALUES: 00 - 0.0f, 0.0f, 0.0f, 0.0f 01 - 0.0f, 0.0f, 0.0f, 1.0f 02 - 1.0f, 1.0f, 1.0f, 0.0f

			03 - 1,0f, 1.0f, 1.0f, 1.0f
FLAT_SHADE	10	0x0	Flat shade select
CYL_WRAP	16:13	0x0	4-bit cylindrical wrap control (1 bit per component)
PT_SPRITE_TEX	17	0x0	Override this parameter with texture coordinates if global enable set and prim is a point

SPI:SPI_PS_IN_CONTROL_0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286cc

DESCRIPTION: *Interpolator control settings*

Field Name	Bits	Default	Description
NUM_INTERP	5:0	0x0	Number of parameters to interp (not minus 1). Should include VS Fog term, if enabled.
POSITION_ENA	8	0x0	Load per-pixel floating point position into the PS
POSITION_CENTROID	9	0x0	Calculate per-pixel position at pixel centroid
POSITION_ADDR	14:10	0x0	Relative GPR address where flt position is loaded (0->31)
PARAM_GEN	18:15	0x0	Generate gradients for ST coordinates, written into LDS at location (NUM_INTERP). Only bit 0 is used, bits 1-3 should be set to 0.
PERSP_GRADIENT_ENA	28	0x0	Enable perspective gradients (if linear is set to 0, persp is always enabled)
LINEAR_GRADIENT_ENA	29	0x0	Enable linear gradients
POSITION_SAMPLE	30	0x0	Calculate per-pixel position at iterated sample num

SPI:SPI_PS_IN_CONTROL_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286d0

DESCRIPTION: *Interpolator control settings*

Field Name	Bits	Default	Description
FRONT_FACE_ENA	8	0x0	Enable GPR load with x=face, y=prim_type, z=pixel_coverage, w=ps_gen_index. Term written at FRONT_FACE_ADDR
FRONT_FACE_ALL_BITS	11	0x0	POSSIBLE VALUES: 00 - Sign bit represents isFF (dx9, -1.0f == backFace, +1.0f == frontFace) 01 - Replace whole 32b val with isFF (WGF, 1 == frontFace, 0 == backFace)
FRONT_FACE_ADDR	16:12	0x0	Relative GPR address to load (0->31)
FOG_ADDR	23:17	0x0	Relative LDS address to load (0->NUM_INTERP-1)
FIXED_PT_POSITION_ENA	24	0x0	Load per-pixel fixed point position into the PS
FIXED_PT_POSITION_ADDR	29:25	0x0	Relative GPR address where fxd position is loaded (0->31)
POSITION_ULC	30	0x0	Force floating point position to upper left corner of pixel (X.0, Y.0)

SPI:SPI_PS_IN_CONTROL_2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286e4

DESCRIPTION: *Interpolator control settings*			
Field Name	Bits	Default	Description
LINE_STIPPLE_TEX_ADDR	7:0	0x0	GPR address for per pixel line stipple texture coord. (0 -> 126)
LINE_STIPPLE_TEX_ENA	8	0x0	Enable line stipple texture generation in the PA, per pixel calc and GPR load in the SPI.

SPI:SPI_LDS_MGMT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286fc			
DESCRIPTION: *Per-context LDS limit for each shader type. This is a soft-limit, meaning allocation only fails once currently allocated space > limit.*			
Field Name	Bits	Default	Description
NUM_PS_LDS	7:0	0x0	PS LDS limit, format is [12:5]. A setting of 1 means 32 dwords, 255 means 8160 dwords, 0 disables the limit.
NUM_LS_LDS	15:8	0x0	Same desc as PS, applies to LS for draws and CS for dispatch.

SPI:SPI_STACK_MGMT · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28700			
DESCRIPTION: *Per-context Stack limit for each shader type. This is a soft-limit, meaning allocation only fails once currently allocated space > limit.*			
Field Name	Bits	Default	Description
NUM_PS_STACK	4:0	0x0	PS Stack limit, format is [8:4]. A setting of 1 means 16 entries, 31 means 496 entries, and 0 disables the limit. Max stack entries is 512.
NUM_VS_STACK	9:5	0x0	Same desc as PS
NUM_GS_STACK	14:10	0x0	Same desc as PS
NUM_ES_STACK	19:15	0x0	Same desc as PS
NUM_HS_STACK	24:20	0x0	Same desc as PS
NUM_LS_STACK	29:25	0x0	Same desc as PS, limit applies to LS and CS.

SPI:SPI_VS_OUT_CONFIG · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x286c4			
DESCRIPTION: *VS output configuration*			
Field Name	Bits	Default	Description
VS_PER_COMPONENT	0	0x0	When set, each entry in SPI_VS_OUT_ID_0-9 represents one component of a vector (not valid for DX10). Otherwise each entry represents an entire vector
VS_EXPORT_COUNT	5:1	0x0	Number of vectors exported by the VS (value is minus 1)
VS_HALF_PACK	6	0x0	Setting this bit causes the VGT to only load VS wavefronts half full of verts and the SPI to alloc/dealloc half the param cache space for each wave. Required for configs with > 1 quad pipe when (((VS_EXPORT_COUNT + 1) *

			GPU__GC__QP_PER_SIMD * 2) > GPU__SX__PARAMETER_CACHE_DEPTH)
VS_EXPORTS_FOG	8	0x0	Set when VS exports fog
VS_OUT_FOG_VEC_ADDR	13:9	0x0	Vector address where VS exported fog. Fog factor will always be in the X channel

SPI:SPI_VS_OUT_ID_[0-9] · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2861c-0x28640

DESCRIPTION: *VS output semantic mapping for 4 components/vectors*

Field Name	Bits	Default	Description
SEMANTIC_0	7:0	0x0	
SEMANTIC_1	15:8	0x0	
SEMANTIC_2	23:16	0x0	
SEMANTIC_3	31:24	0x0	

SPI:SPI_WAVE_MGMT_1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28704

DESCRIPTION: *Per-context wave_buffer limit for each shader type. This is a soft-limit, meaning allocation only fails once currently allocated space > limit.*

Field Name	Bits	Default	Description
NUM_PS_WAVES	4:0	0x0	PS wave limit, format is [7:3]. A setting of 1 means 8 waves, 31 means 248, and 0 disables the limit (256).
NUM_VS_WAVES	9:5	0x0	Same desc as PS
NUM_GS_WAVES	14:10	0x0	Same desc as PS
NUM_ES_WAVES	19:15	0x0	Same desc as PS
NUM_HS_WAVES	24:20	0x0	Same desc as PS
NUM_LS_WAVES	29:25	0x0	Same desc as PS, limit applies to LS only and not CS in this case.

SPI:SPI_WAVE_MGMT_2 · [R/W] · 16 bits · Access: 16 · GpuF0MMReg:0x28708

DESCRIPTION: *Per-context wave_buffer limit for each shader type. This is a soft-limit, meaning allocation only fails once currently allocated space > limit.*

Field Name	Bits	Default	Description
NUM_CS_WAVES_ONE_RING	4:0	0x0	CS wave limit, format is [7:3]. A setting of 1 means 8 waves, 31 means 248, and 0 disables the limit (256). This field is used when a given CS ring is the only ring active in the SPI.
NUM_CS_WAVES_MULTI_RING	9:5	0x0	This field is used for a given CS ring when there are multiple rings active.

11. Shader Export Registers

SX:SX_ALPHA_REF · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28438			
Field Name	Bits	Default	Description
ALPHA_REF	31:0	none	Reference value for alpha test, which is specified in IEEE floating point.

SX:SX_ALPHA_TEST_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28410			
Field Name	Bits	Default	Description
ALPHA_FUNC	2:0	none	Specifies the function used to compare the fragment alpha value (produced by the shader pipe) to ALPHA_REF, the reference alpha value. The alpha test passes (keeping the pixel) if frag_alpha OP alpha_ref is true. POSSIBLE VALUES: 00 - REF_NEVER: never pass 01 - REF_LESS: pass if left < right 02 - REF_EQUAL: pass if left = right 03 - REF_LEQUAL: pass if left <= right 04 - REF_GREATER: pass if left > right 05 - REF_NOTEQUAL: pass if left != right 06 - REF_GEQUAL: pass if left >= right 07 - REF_ALWAYS: always pass
ALPHA_TEST_ENABLE	3	none	If alpha test is enabled, then a failed ALPHA_FUNC comparison causes the pixel to be killed. POSSIBLE VALUES: 00 - DISABLE: force ALPHA_FUNC to ALWAYS 01 - ENABLE: discard pixels that do not pass the alpha test.
ALPHA_TEST_BYPASS	8	none	Driver can st this bit to bypass the alpha test for surface types that don`t support alpha testing. POSSIBLE VALUES: 00 - DISABLE: discard pixels that do not pass the alpha test. 01 - ENABLE: force ALPHA_FUNC to ALWAYS.

SX:SX_EXPORT_BUFFER_SIZES · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x900c			
DESCRIPTION: *Register that defines export buffer ring sizes.*			
Field Name	Bits	Default	Description
COLOR_BUFFER_SIZE	7:0	0x3F	Number of 4 line buffers -1 in color buffer. Each memory buffer corresponds to 4 lines of 16*128 bits elements. Minimum acceptable value of register field is 0x8 (8

			MRT+Z).
POSITION_BUFFER_SIZE	15:8	0xF	Number of 4 line buffers -1 in position buffer. Each memory buffer corresponds to 4 lines of 16*128 bits elements. Minimum acceptable value of register field is 0x3. Not programable for Wekiva, for derivatives, max value is 0xF.
SMX_BUFFER_SIZE	23:16	0x2F	Number of 4 line buffers -1 in smx buffer. Each memory buffer corresponds to 4 lines of 16*128 bits elements

SX:SX_MEMORY_EXPORT_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9010			
DESCRIPTION: *Defines the base address of the memory export. Only available if chip supports GPU__GC__MEM_EXPORT_PRESENT.*			
Field Name	Bits	Default	Description
ADDRESS	31:0	0x0	Format is [39:8]

SX:SX_MEMORY_EXPORT_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9014			
DESCRIPTION: *Defines the aperture of the memory export. Only available if chip supports GPU__GC__MEM_EXPORT_PRESENT.*			
Field Name	Bits	Default	Description
SIZE	31:0	0x0	Format is [39:8] so SIZE is in increments of 256 bytes (64 DW), if index > size, SX will clamp to Size - 1 dword and disable the write.

SX:SX_MISC · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28350			
Field Name	Bits	Default	Description
MULTIPASS	0	none	POSSIBLE VALUES: 00 - Do not kill all primitives 01 - Kill all primitives

SX:SX_SCATTER_EXPORT_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28358			
DESCRIPTION: *Defines the base address of the scatter or LDS export. Read performed by SXM block using GRBM instancing.*			
Field Name	Bits	Default	Description
ADDRESS	31:0	0x0	Format is [39:8]

SX:SX_SCATTER_EXPORT_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2835c			
DESCRIPTION: *Defines the aperture of the scatter or LDS export. Read performed by SXM block using GRBM instancing.*			
Field Name	Bits	Default	Description
SIZE	31:0	0x0	Format is [39:8] so SIZE is in increments of 256 bytes

			(64 DW), if index > size, SX will clamp to Size - 1 dword and disable the write.

SX:SX_SURFACE_SYNC · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28354			
DESCRIPTION: *This register is used for CP surface sync purposes, controls SX related surfaces.*			
Field Name	Bits	Default	Description
SURFACE_SYNC_MASK	9:0	none	Used to tell which SX surface type to flush on a sync request from the CP. Mapping is as follows: 0:0 Stream out 0 1:1 Stream out 1 2:2 Stream out 2 3:3 Stream out 3 4:4 Scratch 5:5 Reduction Buffer 6:6 Ring Buffer 7:7 F-Buffer 8:8 Memory export 9:9 Cacheless RAT surfaces

12. Cache Control Registers

SMX:SMX_DC_CTL0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa020			
DESCRIPTION: *Write combining Cache control register. Read performed by SXS block using GRBM instancing.*			
Field Name	Bits	Default	Description
USE_HASH_FUNCTION	0	0x1	Use hash function to select set
NUMBER_OF_SETS	9:1	0x8	Cache depth in number of sets, 1,2, or 4
FLUSH_ALL_ON_EVENT	10	0x0	Flush whole cache everytime an event is seen
STALL_ON_EVENT	11	0x0	Stall all data while event is processed
DISABLE_WRITE_EVICT	12	0x0	Disable simultaneous writes and evicts in the write combiners.
FORCE_FULL_STRICT_RR	13	0x0	Force full line machine in a strict round-robin arbitration between sets
STALL_FLFLUSH	14	0x0	Stall update probes while flflush machine evicts.
ALLOW_DUPLICATES	15	0x1	Allow duplicate cachelines on memory exports (and RAT_CACHELESS), when this is set order of export operations in the thread is not guaranteed if ALLOW_MULTI_EVICT is set but performance of the cache will improve
ALLOW_RAT_DUPLICATES	16	0x1	Allow duplicate cachelines on RAT exports, when this is set order of export operations in the thread is not guaranteed if ALLOW_MULTI_EVICT is set but performance of the cache will improve
ENABLE_FLFLUSH_MACHINE	17	0x1	Allows the cache to oportunisticaly write full lines to memory ahead of time
CACHE_EVENT_OPTIMIZE	18	0x1	Allows writes to occur while events are being processed.
EVENT_LONG_LATENCY	19	0x0	Forces events to wait for update traffic, resulting in longer waits before events get sent.
NO_RAT_NON_RAT_MIX	20	0x0	RAT transactions use 1/2 the sets, non-RAT use the other half, no mixing allowed

SMX:SMX_EVENT_CTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa02c			
DESCRIPTION: *Controls the affects of events on data present in cache. Read performed by SXS block using GRBM instancing.*			
Field Name	Bits	Default	Description
ES_FLUSH_CTL	2:0	0x3	Controls ES related events <u>POSSIBLE VALUES:</u> 00 - Flush only generic ES traffic 01 - Flush generic and scratch ES traffic 02 - Flush generic and sync ES traffic 03 - Flush all ES traffic 04 - Flush whole cache

GS_FLUSH_CTL	5:3	0x3	Controls GS related events <u>POSSIBLE VALUES:</u> 00 - Flush only generic GS traffic 01 - Flush generic and scratch GS traffic 02 - Flush generic and sync GS traffic 03 - Flush all GS traffic 04 - Flush whole cache
ACK_FLUSH_CTL	7:6	0x0	Controls Ack events <u>POSSIBLE VALUES:</u> 00 - Flush ack traffic of same type 01 - Flush all ack traffic 02 - Flush all traffic of same type 03 - Flush whole cache
SYNC_FLUSH_CTL	8	0x0	Controls surface sync events <u>POSSIBLE VALUES:</u> 00 - Flush only sync marked traffic 01 - Flush whole cache
WAIT_FOR_IDLE_ON_EVENT	9	0x0	Controls how the events are allowed in the SXS tile <u>POSSIBLE VALUES:</u> 00 - Submit event while SXS is flushing the previous one 01 - Wait for full idle on the SXS event machine before next event is allowed

13. Texture Pipe Registers

TP:TD_CNTL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9494			
DESCRIPTION: *Texture Data Common Control*			
Field Name	Bits	Default	Description
SYNC_PHASE_SH	1:0	0x0	Programmable Sync Phase Offset to be used for sending data back to SH.
PAD_STALL_EN	8	0x0	When set (=1), the Texture Data pipeline will internally enforce all stall periods to be multiples of 4 clock cycles.
EXTEND_LDS_STALL	10:9	0x0	Extend LDS stall from SX by 1 to 3 cycles more than the default. The default is 4 cycles. POSSIBLE VALUES: 00 - 0: leave default extendion - total of 4 cycles 01 - 1: Extend LDS stall by 1 more cycle - total of 5 cycles 02 - 2: Extend LDS stall by 2 more cycle - total of 6 cycles 03 - 3: Extend LDS stall by 3 more cycle - total of 7 cycles
GATHER4_FLOAT_MODE	16	0x0	32-bit float handling for Gather4 opcodes POSSIBLE VALUES: 00 - 0: Pass denorm/NaN untouched like mov operation 01 - 1: Override denorm/NaN as in SAMPLE
LD_FLOAT_MODE	18	0x0	Not supported POSSIBLE VALUES: 00 - Not supported
GATHER4_DX9_MODE	19	0x0	Return data in DX9 order: W, Z, Y, X = TL, BR, BL, TR POSSIBLE VALUES: 00 - 0: Return data in DX10 order: W, Z, Y, X = TL, TR, BR, BL 01 - 1: Return data in DX9 order: W, Z, Y, X = TL, BR, BL, TR

TP:TD_CS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa474			
Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_CS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa470

Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_CS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa46c			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_CS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa464			
Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_CS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa468			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TD_GS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa438			
Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_GS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa434			
Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_GS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa430			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_GS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa428			
Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_GS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa42c			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TD_HS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa44c			
Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_HS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa448			
Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_HS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa444			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_HS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa43c			
Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_HS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa440			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TD_LS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa460			
Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_LS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa45c			
Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_LS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa458			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_LS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa450

Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_LS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa454			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TD_PS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa410			
Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_PS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa40c			
Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_PS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa408			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_PS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa400			
Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_PS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa404			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TD_STATUS · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9498			
DESCRIPTION: *Texture Data Status*			
Field Name	Bits	Default	Description
BUSY (Access: R)	31	none	

TP:TD_VS_BORDER_COLOR_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa424

Field Name	Bits	Default	Description
ALPHA	31:0	none	

TP:TD_VS_BORDER_COLOR_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa420			
Field Name	Bits	Default	Description
BLUE	31:0	none	

TP:TD_VS_BORDER_COLOR_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa41c			
Field Name	Bits	Default	Description
GREEN	31:0	none	

TP:TD_VS_BORDER_COLOR_INDEX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa414			
Field Name	Bits	Default	Description
INDEX	4:0	none	

TP:TD_VS_BORDER_COLOR_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0xa418			
Field Name	Bits	Default	Description
RED	31:0	none	

TP:TA_CNTL_AUX · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9508			
DESCRIPTION: *Texture Addresser Common Control*			
Field Name	Bits	Default	Description
DISABLE_CUBE_WRAP	0	0x0	CubeMap Clamp Policy Override POSSIBLE VALUES: 00 - Force Clamp X,Y policy to wrap for CubeMaps 01 - Allow other clamp modest
Reserved	1	0x0	
GETLOD_SELECT	3:2	0x0	GetLOD Data Return control POSSIBLE VALUES: 00 - Sampler and Resource clamped LOD in Resource View space, Zero fraction for MipPoint 01 - Sampler and Resource clamped LOD in Resource View space, Fraction kept always 02 - Sampler clamped LOD in Resource View space, Fraction kept always 03 - Sampler and Resource clamped LOD in Resource space, Zero fraction for MipPoint
DISABLE_IDLE_STALL	4	0x0	Idles in stalls override

			POSSIBLE VALUES: 00 - Idles included in stalls for power 01 - Idles excluded from stalls
VERTEX_FETCH_64_MODE	8	0x0	Control granularity of multicycle condition for 64-bit formats POSSIBLE VALUES: 00 - 16K DWORD Multicycling 01 - 32-bit Multicycling
VERTEX_FETCH_96_MODE	9	0x0	Control granularity of multicycle condition for 96-bit formats POSSIBLE VALUES: 00 - 16K DWORD Multicycling 01 - Always 3-cycle
VERTEX_FETCH_128_MODE	10	0x0	Control granularity of multicycle condition for 128-bit formats POSSIBLE VALUES: 00 - 16K DWORD Multicycling 01 - 64-/32-bit Multicycling
SYNC_GRADIENT	24	0x1	This has been deprecated for 7xx and beyond. Treat as RESERVED. Gradient synchronization mode POSSIBLE VALUES: 00 - Gradient Sync on Instruction 01 - Gradient Sync on Phase
SYNC_WALKER	25	0x1	This has been deprecated for 7xx and beyond. Treat as RESERVED. Walker synchronization mode POSSIBLE VALUES: 00 - Walker Sync on Instruction 01 - Walker Sync on Phase
SYNC_ALIGNER	26	0x1	This has been deprecated for 7xx and beyond. Treat as RESERVED. Aligner synchronization mode POSSIBLE VALUES: 00 - Aligner Sync on Instruction 01 - Aligner Sync on Phase
TEX_COORD_PRECISION	28	0x0	Texture cooordinate precision setting POSSIBLE VALUES: 00 - 8-bit weight precision 01 - 6-bit weight precision
LOD_LOG2_TRUNC	29	0x0	LOD log2 truncate or round setting POSSIBLE VALUES:

			00 - Round log2 approximation 01 - Truncate log2 approximation
MIPFILTER_POINT_PRECLAMP	30	0x0	Controls the location of the 0.5 addition to LOD for mipfilter_point to before or after the Sampler and Resource clamps POSSIBLE VALUES: 00 - Addition of 0.5 after Sampler and Resource Clamp 01 - Addition of 0.5 before the Sampler and Resource Clamp
LOD_SF_SCALING	31	0x0	Apply the scaling method for sf_int_gt_0 or revert back to the hard clamp POSSIBLE VALUES: 00 - Apply scaling for sf_int_gt_0 in the LOD algorithm 01 - Do not apply scaling

14. Depth Buffer Registers

DB:DB_ALPHA_TO_MASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b70			
Field Name	Bits	Default	Description
ALPHA_TO_MASK_ENABLE	0	none	If enabled, the sample mask is ANDed with a mask produced from the alpha value. This field can be overriden by setting DB_SHADER_CONTROL.ALPHA_TO_MASK_DISABLE.
ALPHA_TO_MASK_OFFSET0	9:8	none	Dither threshold for pixel (0,0) in each quad if alpha to mask is enabled. Set to 2 for non-dithered, or a unique 0-3 value for dithered.
ALPHA_TO_MASK_OFFSET1	11:10	none	Dither threshold for pixel (0,1) in each quad if alpha to mask is enabled. Set to 2 for non-dithered, or a unique 0-3 value for dithered.
ALPHA_TO_MASK_OFFSET2	13:12	none	Dither threshold for pixel (1,0) in each quad if alpha to mask is enabled. Set to 2 for non-dithered, or a unique 0-3 value for dithered.
ALPHA_TO_MASK_OFFSET3	15:14	none	Dither threshold for pixel (1,1) in each quad if alpha to mask is enabled. Set to 2 for non-dithered, or a unique 0-3 value for dithered.
OFFSET_ROUND	16	none	Round dither threshold. Set to 0 for a non-dithered look, or 1 for a dithered look.

DB:DB_COUNT_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28004			
Field Name	Bits	Default	Description
ZPASS_INCREMENT_DISABLE	0	none	Disable incrementing the ZPass count for this context.
PERFECT_ZPASS_COUNTS	1	none	Forces zpass counts to be accurate by turning off no-op culling optimizations where skipping rasterization may lead to incorrect zpass counts (partially covered tiles).
SAMPLE_RATE	6:4	0x0	Sets how many samples per pixel are counted. Area is accurate no matter how many samples per pixel there really are.

DB:DB_DEPTH_CLEAR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2802c			
Field Name	Bits	Default	Description
DEPTH_CLEAR	31:0	none	Depth value when ZMASK==0, which indicates that the tile has been cleared to the background depth. This register holds a 32bit float value. This value must be in the range of 0.0 to 1.0

DB:DB_DEPTH_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28800
DESCRIPTION: *This register controls depth and stencil tests.*

Field Name	Bits	Default	Description
STENCIL_ENABLE	0	none	Enables stencil testing. If disabled, all pixels pass the stencil test. If there is no stencil buffer this is treated as disabled.
Z_ENABLE	1	none	Enables depth testing. If disabled, all pixels pass the depth test. If there is no depth buffer this is treated as disabled.
Z_WRITE_ENABLE	2	none	Enables writing to the depth buffer if the depth test passes.
ZFUNC	6:4	none	Specifies the function that compares the depth at each sample in the fragment to the destination depth at the corresponding sample point. POSSIBLE VALUES: 00 - FRAG_NEVER: never pass 01 - FRAG_LESS: pass if fragment < dest 02 - FRAG_EQUAL: pass if fragment = dest 03 - FRAG_LEQUAL: pass if fragment <= dest 04 - FRAG_GREATER: pass if fragment > dest 05 - FRAG_NOTEQUAL: pass if fragment != dest 06 - FRAG_GEQUAL: pass if fragment >= dest 07 - FRAG_ALWAYS: always pass
BACKFACE_ENABLE	7	none	If false, forces all quads to be stencil tested as frontface quads.
STENCILFUNC	10:8	none	Specifies the function that compares STENCILREF to the destination stencil value for frontface quads. The stencil test passes if ref OP dest is true. POSSIBLE VALUES: 00 - REF_NEVER: never pass 01 - REF_LESS: pass if left < right 02 - REF_EQUAL: pass if left = right 03 - REF_LEQUAL: pass if left <= right 04 - REF_GREATER: pass if left > right 05 - REF_NOTEQUAL: pass if left != right 06 - REF_GEQUAL: pass if left >= right 07 - REF_ALWAYS: always pass
STENCILFAIL	13:11	none	Specifies the stencil operation for frontface quads if the stencil function fails. POSSIBLE VALUES: 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0 02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value

			06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)
STENCILZPASS	16:14	none	Specifies the stencil operation for frontface quads if the stencil and depth functions both pass. <u>POSSIBLE VALUES:</u> 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0 02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value 06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)
STENCILZFAIL	19:17	none	Specifies the stencil operation for frontface quads if the stencil function passes and the depth function fails. <u>POSSIBLE VALUES:</u> 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0 02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value 06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)
STENCILFUNC_BF	22:20	none	Specifies the function that compares STENCILREF_BF to the destination stencil for backface quads. The stencil test passes if ref OP dest is true. <u>POSSIBLE VALUES:</u> 00 - REF_NEVER: never pass 01 - REF_LESS: pass if left < right 02 - REF_EQUAL: pass if left = right 03 - REF_LEQUAL: pass if left <= right 04 - REF_GREATER: pass if left > right 05 - REF_NOTEQUAL: pass if left != right 06 - REF_GEQUAL: pass if left >= right 07 - REF_ALWAYS: always pass
STENCILFAIL_BF	25:23	none	Specifies the stencil operation for backface quads if the stencil function fails. <u>POSSIBLE VALUES:</u> 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0

			02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value 06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)
STENCILZPASS_BF	28:26	none	Specifies the stencil operation for backface quads if the stencil and depth functions both pass. POSSIBLE VALUES: 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0 02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value 06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)
STENCILZFAIL_BF	31:29	none	Specifies the stencil operation for backface quads if the stencil function passes and the depth function fails. POSSIBLE VALUES: 00 - STENCIL_KEEP: New value = Old Value 01 - STENCIL_ZERO: New value = 0 02 - STENCIL_REPLACE: New value = STENCILREF 03 - STENCIL_INCR_CLAMP: New value++ (clamp) 04 - STENCIL_DECR_CLAMP: New value-- (clamp) 05 - STENCIL_INVERT: New value=~Old value 06 - STENCIL_INCR_WRAP: New value++ (wrap) 07 - STENCIL_DECR_WRAP: New value-- (wrap)

DB:DB_DEPTH_INFO · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2803c

Field Name	Bits	Default	Description
ADDR5_SWIZZLE_MASK	3:0	none	For 32B tiles, indicates whether the data should be stored in the upper or lower half of a 64B word. if the XOR reduce of ADDR5_SWIZZLE_MASK & {TILE_Y[1:0], TILE_X[1:0]} is set, use the upper half, otherwise, use the lower half.

DB:DB_DEPTH_SIZE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28058			
Field Name	Bits	Default	Description
PITCH_TILE_MAX	10:0	none	Width in 8x8 pixel tiles. (Pitch/8 - 1)
HEIGHT_TILE_MAX	21:11	none	Height of the depth buffer in 8x8 pixels (height/8 - 1)

DB:DB_DEPTH_SLICE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2805c			
Field Name	Bits	Default	Description
SLICE_TILE_MAX	21:0	none	Number of 8x8 pixel tiles until the next slice plus some small number to be able to rotate the tile pattern. (pitch*height/64 - 1)

DB:DB_DEPTH_VIEW · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28008			
DESCRIPTION: *Selects slice index range for render target 0.*			
Field Name	Bits	Default	Description
SLICE_START	10:0	none	Specifies the starting slice number for this view. This field is added to the RenderTargetArrayIndex to compute the slice to render. SLICE_START must less than or equal to SLICE_MAX
SLICE_MAX	23:13	none	Specifies the maximum allowed Z slice index for this resource, which is one less than the total number of slices.
Z_READ_ONLY	24	none	read only Z buffer. i.e. Force off writes to Z buffer
STENCIL_READ_ONLY	25	none	read only Stencil buffer.i.e. Force off writes to Stencil buffer

DB:DB_EQAA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28804			
DESCRIPTION: *This register controls EQAA in the DB.*			
Field Name	Bits	Default	Description
MAX_ANCHOR_SAMPLES	2:0	none	Sets the most number of anchor samples that the CB is allowe to use. Set this to the mininum allocated sample amount of the DB surfaces to limit the potential of needing the CB to toss non-anchored fragments after they are created.
PS_ITER_SAMPLES	6:4	none	Specifies how many samples to iterate across when PS_ITER_SAMPLE is set thus setting the amount of super-sampling. Typically this is the number of app exposed samples. Values greater than the depth surface samples is not supported.
MASK_EXPORT_NUM_SAMPLES	10:8	none	Specifies how many samples to use for shader mask exports.
ALPHA_TO_MASK_NUM_SAMPLES	14:12	none	How many samples of quality are generated for A2M.

			Set this in between the number of app exposed samples and higher EQAA samples for speed/quality tradeoff. If ALPHA_TO_MASK_EQAA_DISABLE=1, it must be set to the number of app exposed samples.
HIGH_QUALITY_INTERSECTIONS	16	none	If not set, all fully covered tiles run through the detail walker at tile rate, only later slowing down to the DB`s surface rate if it exists and the depth test results are not known or down to pixel rate if the shader executes. If set, will only speed up fully covered tiles that have known Z test results, but still allows tiles that have potential Z intersections to run at the detail rate and therefore get AAed intersections. Should be used with INTERPOLATE_COMP_Z for best quality.
INCOHERENT_EQAA_READS	17	none	Disables the coherency check for abutting triangles that share anchor samples, but not detail samples Important for performance on abutting strips if data forwarding doesn`t exist Introduces latency dependant results, so force to 0 for all testing except perhaps for unit tests.
INTERPOLATE_COMP_Z	18	none	Allows unanchored samples to interpolate a unique Z from compressed Z Planes. Creates nice AAed intersections on first intersection per pixel Introduces latency dependant results Therefore force this to 0 for all testing except perhaps unit directed tests that are visually checked
INTERPOLATE_SRC_Z	19	none	Forces unanchored samples to interpolate a unique source Z even when destination Z is not compressed for a smoother intersection even with uncompressed Z. May cause blending with ZFUNC==EQUALS on uncompressed Z to fail due to comparing against non-interpolated dest Z. Likely will never be set except for experimentation.
STATIC_ANCHOR_ASSOCIATIONS	20	none	Forces replicated destination data to always come from the statically associated anchor sample as opposed to trying to pull destination data from the nearest anchor sample that is inside the primitive. When set, may cause additional coherency stalls and a degradation of quality for abutting triangles
ALPHA_TO_MASK_EQAA_DISABLE	21	none	Makes Alpha to Mask set samples exactly like the previous GPUs. Should only be set if previous generation behavior is desired, otherwise the new behavior is optimized for EQAA which improves the quality when mixing AA modes and even when not.

DB:DB_HTILE_DATA_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28014

Field Name	Bits	Default	Description
BASE_256B	31:0	none	Location of the first byte of the HTileData surface in Device Address Space, which must be 256 byte aligned.

			High 32-bits of 40-bit address. This surface contains the HiZ data.

DB:DB_HTILE_SURFACE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28abc			
Field Name	Bits	Default	Description
HTILE_WIDTH	0	none	How many pixels wide each entry in the htile buffer represents. Must be set to 1. 0 = 4 (deprecated), 1 = 8
HTILE_HEIGHT	1	none	How many pixels high each entry in the htile buffer represents. Must be set to 1. 0 = 4 (deprecated), 1 = 8
LINEAR	2	none	Surface is stored linearly in swaths of 8 htiles high until the surface is complete.
FULL_CACHE	3	none	This htile buffer uses the entire htile cache. if set to 0 and the htile surface will not fit in half the cache, then the SC`s partial vector deadlock timer must also be enabled
HTILE_USES_PRELOAD_WIN	4	none	If set, the htile surface dimensions will be that of the preload window; otherwise, it will be that of the depth buffer
PRELOAD	5	none	Preload all data that fits as soon as room is available once the VGT_DRAW_INITIATOR is seen on a context.
PREFETCH_WIDTH	11:6	none	The Prefetch window width (in 64 pixel increments). Prefetcher tries to keep this window around the last rasterized htile in cache at all times.
PREFETCH_HEIGHT	17:12	none	The Prefetch window height (in 64 pixel increments). Prefetcher tries to keep this window around the last rasterized htile in cache at all times.

DB:DB_PRELOAD_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ac8			
Field Name	Bits	Default	Description
START_X	7:0	none	Starting X position of the preload window, in 64 pixel increments
START_Y	15:8	none	Starting Y position of the preload window, in 64 pixel increments
MAX_X	23:16	none	Ending X position of the preload window, in 64 pixel increments
MAX_Y	31:24	none	Ending Y position of the preload window, in 64 pixel increments

DB:DB_RENDER_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28000			
Field Name	Bits	Default	Description
DEPTH_CLEAR_ENABLE	0	none	Clears Z to the Clear Value.

STENCIL_CLEAR_ENABLE	1	none	Clears Stencil to the Clear Value
DEPTH_COPY	2	none	Enables Z expansion to color render target 0. CB must be programmed to the desired destination format.
STENCIL_COPY	3	none	Enables Stencil expansion to color render target 0. CB must be programmed to the desired destination format.
RESUMMARIZE_ENABLE	4	none	If set, all tiles touched will update the HTILE surface info.
STENCIL_COMPRESS_DISABLE	5	none	Forces stencil to decompress on any rendered tile not hierarchically culled
DEPTH_COMPRESS_DISABLE	6	none	Forces z to decompress on any rendered tile not hierarchically culled
COPY_CENTROID	7	none	If set, copy the 1st lit sample in the pixel starting at the COPY_SAMPLE`th sample (wraps back to lower samples). If COPY_CENTROID==0 and z or stencil writes are on (which doesn`t happen in production drivers), DB_RENDER_OVERRIDE.FORCE_QC_SMASK_CONFLICT must be set. Also, COPY_CENTROID must be set to 1 when doing z or stencil copies and ps_iter is on.
COPY_SAMPLE	11:8	none	If COPY_CENTROID, copy 1st lit starting at this sample number. Else copy this sample whether lit or not.

DB:DB_RENDER_OVERRIDE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2800c			
Field Name	Bits	Default	Description
FORCE_HIZ_ENABLE	1:0	none	Forces hierarchical depth culling to be enabled ignoring what is in DB_SHADER_CONTROL and all other render states. POSSIBLE VALUES: 00 - FORCE_OFF 01 - FORCE_ENABLE 02 - FORCE_DISABLE 03 - FORCE_RESERVED
FORCE_HIS_ENABLE0	3:2	none	Forces hierarchical stencil culling to be enabled for compare state 0, ignoring what is in DB_SHADER_CONTROL and all other render states. POSSIBLE VALUES: 00 - FORCE_OFF 01 - FORCE_ENABLE 02 - FORCE_DISABLE 03 - FORCE_RESERVED
FORCE_HIS_ENABLE1	5:4	none	Forces hierarchical stencil culling to be enabled for compare state 1, ignoring what is in DB_SHADER_CONTROL and all other render states. POSSIBLE VALUES: 00 - FORCE_OFF 01 - FORCE_ENABLE

			02 - FORCE_DISABLE 03 - FORCE_RESERVED
FORCE_SHADER_Z_ORDER	6	none	Forces the setting specified in DB_SHADER_CONTROL.Z_ORDER to be used for early/late/re Z+S test. If not set the shader preference is used unless precluded by other render states.
FAST_Z_DISABLE	7	none	Do not accelerate Z clears or write operations. Prevents killing quads before detail rasterization if depth operations are needed.
FAST_STENCIL_DISABLE	8	none	Do not accelerate stencil clears or write operations. Prevents killing quads before detail rasterization if stencil operations are needed.
NOOP_CULL_DISABLE	9	none	Prevents hierarchically killing quads that will pass Z and Stencil, but do not write Z, Stencil or Color.
FORCE_COLOR_KILL	10	none	DB does any possible depth optimizations assuming the shader results are not needed and kills all samples before the color operation.
FORCE_Z_READ	11	none	Read all Z data for a tile even if it is not needed. Used for resummarization blts.
FORCE_STENCIL_READ	12	none	Read all stencil data for a tile even if it is not needed. Used for resummarization blts.
FORCE_FULL_Z_RANGE	14:13	none	Forces hierarchical depth to treat each primitive as if its range is 0.0 -> 1.0f or not. If disabled, it is implicitly derived from DB_SHADER_CONTROL.Z_EXPORT_ENABLE and other enabling registers. Can be used to reset the Z range to 0-1 as well. May be set to FORCE_DISABLE only if DB_SHADER_CONTROL.Z_EXPORT_ENABLE is set to 0. Production drivers are expected to set this field to FORCE_OFF POSSIBLE VALUES: 00 - FORCE_OFF 01 - FORCE_ENABLE 02 - FORCE_DISABLE 03 - FORCE_RESERVED
FORCE_QC_SMASK_CONFLICT	15	none	Forces Quad Coherency to mark a squad with a matching dtileid, x, and y as a conflict and stall it even if the sample mask doesn`t overrlap.
DISABLE_VIEWPORT_CLAMP	16	none	Disables the viewport clamp, which allows Z data to go through untouched.
IGNORE_SC_ZRANGE	17	none	Ignore the SC`s vertex bounds on the minZ/maxZ for a tile during HiZ.
DISABLE_FULLY_COVERED	18	none	Disable the fully covered tile bit coming into the DB, which turns off all fully covered optimizations.
FORCE_Z_LIMIT_SUMM	20:19	none	Forces summarization of minz or maxz or both.

			POSSIBLE VALUES: 00 - FORCE_SUMM_OFF 01 - FORCE_SUMM_MINZ 02 - FORCE_SUMM_MAXZ 03 - FORCE_SUMM_BOTH
MAX_TILES_IN_DTT	25:21	0x0	Maximum number of tiles allowed in dtt block before causing a stall. If DB_DEBUG.NEVER_FREE_Z_ONLY is set, the MAX_TILES_IN_DTT must be less than or equal to the following, depending on the number of samples in the z buffer: 1xaa: 21 2xaa: 11 4xaa: 5 8xaa: 2 Note: Production drivers are expected to leave this register to the default of 0, which will satisfy the constraint
DISABLE_TILE_RATE_TILES	26	0x0	Disable the optimization which allows some fully covered 8x8s to run at tile rate.
FORCE_Z_DIRTY	27	none	Forces Z data to be written even if it has not changed. Can be used to copy Z data to an alternate surface.
FORCE_STENCIL_DIRTY	28	none	Forces Stencil data to be written even if it has not changed. Can be used to copy Stencil data to an alternate surface.
FORCE_Z_VALID	29	none	Forces the Z data to be read unless it is being overwritten. Can be used to copy Z data to an alternate surface.
FORCE_STENCIL_VALID	30	none	Forces the Stencil data to be read unless it is being overwritten. Can be used to copy Stencil data to an alternate surface.
PRESERVE_COMPRESSION	31	none	Can be used when decompressing to an alternate surface so that the htile`s compression state is not inadvertently marked as expanded. Stops all writes to the zmask and smem fields of the htile buffer.

DB:DB_RENDER_OVERRIDE2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28010			
Field Name	Bits	Default	Description
PARTIAL_SQUAD_LAUNCH_CONTROL	1:0	none	Sets how partial squads are launched. POSSIBLE VALUES: 00 - PSLC_AUTO: Let DB automatically control partial squad launches. 01 - PSLC_ON_HANG_ONLY: Partial squad only launched on hang detect. 02 - PSLC_ASAP: Enable countdown to

			partial launch. PARTIAL_SQUAD_LAUNCH_COUNTDOWN value of 7 means launch immedicately. 03 - PSLC_COUNTDOWN: Enable countdown to partial launch. PARTIAL_SQUAD_LAUNCH_COUNTDOWN value of 7 indicates to never launch partials.
PARTIAL_SQUAD_LAUNCH_COUNTDOWN	4:2	none	Sets countdown after which partial squads are launched as (1 << N). 7 Means disable countdown.
DISABLE_ZMASK_EXPCLEAR_OPTIMIZATION	5	none	Only matters with DB_Z_INFO.ALLOW_EXPCLEAR=1. To be used on first clear on uninitialized surfaces when the zmask can not be trusted.
DISABLE_SMEM_EXPCLEAR_OPTIMIZATION	6	none	Only matters with DB_STENCIL_INFO.ALLOW_EXPCLEAR=1. To be used on first clear on uninitialized surfaces when the stencil memory format can not be trusted.
DISABLE_COLOR_ON_VALIDATION	7	none	Disables DB from looking at CB_COLOR_INFO, CB_SHADER_MASK, and CB_TARGET_MASK to determine if the color is on.
DECOMPRESS_Z_ON_FLUSH	8	none	0: Z Decompresses are performed within the pipeline by allocating cache space and decompressing in the pipe. Has cache pressure in higher AA modes. 1: Z Decmpresses are performed while flushing out to memory without allocating cache space but incurs a startup latency per tile`s flush. Should be set to 0 for 1xAA and 2xAA and 1 for 4xAA and 8xAA.

DB:DB_SHADER_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2880c			
Field Name	Bits	Default	Description
Z_EXPORT_ENABLE	0	none	Use DB Shader Export`s Red channel as Z instead of the intepolated Z value. DB_SOURCE_FORMAT must be FULL or TWO.
STENCIL_REF_EXPORT_ENABLE	1	none	Use DB Shader Export`s Green[7:0] as the Stencil Reference Value. No restriction on DB_SOURCE_FORMAT.
Z_ORDER	5:4	none	Indicates Shader`s preference for which type of Z testing. The _THEN_ for early Z allows the shader to indicate a preference when EARLY_Z can`t be used. If RE_Z can`t be used then LATE_Z is. POSSIBLE VALUES: 00 - LATE_Z

			01 - EARLY_Z_THEN_LATE_Z 02 - RE_Z 03 - EARLY_Z_THEN_RE_Z
KILL_ENABLE	6	none	Shader can kill pixels through texkill.
COVERAGE_TO_MASK_ENABLE	7	none	Use DB Shader Export`s Alpha Channel as an independent Alpha to Mask operation. DB_SOURCE_FORMAT must be FULL or FOUR16, where if FOUR16 the shader converts oDB.alpha to a Float16.
MASK_EXPORT_ENABLE	8	none	Use DB Shader Export`s Blue Channel as sample mask for pixel. The lowest NUM_SAMPLES bits are used. DB_SOURCE_FORMAT must be FULL or FOUR16.
DUAL_EXPORT_ENABLE	9	none	Allows the shader export block to pack two quads into each export to the backend.
EXEC_ON_HIER_FAIL	10	none	Will execute the shader even if Hierarchical Z or Stencil would kill the quad. Enable if the pixel shader has a desired side effect not covered by the above flags for any failing or passing samples (when DEPTH_BEFORE_SHADER=0). Note that EarlyZ and ReZ kills will still stop the shader from running.
EXEC_ON_NOOP	11	none	Will execute the shader even if nothing uses the shader`s color or depth exports. Enable if the pixel shader has a desired side effect not caused by the above flags only for passing pixels.
ALPHA_TO_MASK_DISABLE	12	none	If set, disables alpha to mask, overriding DB_ALPHA_TO_MASK.ALPHA_TO_MASK_ENABLE
DB_SOURCE_FORMAT	14:13	none	This field indicates the format for the DB export from the pixel shader. Red = oDepth, Green = oStencilRef, Blue = oMask, Alpha = oCoverageToMask. See above for the per-channel enables. oDepth can`t use FOUR16 since it is a Float32. oMask and oCoverageToMask can`t use TWO since it doesn`t contain blue or alpha. Therefore if oDepth and oMask and/or oCoverageToMask are on, this must be FULL. If SQ_PGM_EXPORTS_PS.EXPORT_MODE[0] is not 1, this field and the above per-channel enables are ignored and considered disabled. <u>POSSIBLE VALUES:</u> 00 - EXPORT_DB_FULL: Four 32 bit exports over two cycles. 01 - EXPORT_DB_FOUR16: Four 16 bit exports over one cycle extracted from the bottom 16 bits of each channel. 02 - EXPORT_DB_TWO: Two 32 bit exports over one cycle using only Red and Green dropping Blue and Alpha.
DEPTH_BEFORE_SHADER	15	none	The shader is declared to run AFTER depth by definition, which will prevent shader killing of samples and/or pixels (alpha test, alpha to coverage, coverage to mask, mask export, z/stencil exports) from affecting the depth operation and therefore does not allow these to disallow

			EarlyZ. Also ZPass counts are defined to be counted after the Z test, so this mode makes shader and alpha based culling no longer reduce the ZPass counts.
CONSERVATIVE_Z_EXPORT	17:16	none	Forces z exports to be either less than or greater than the source z value. POSSIBLE VALUES: 00 - EXPORT_ANY_Z: Exported Z can be any value 01 - EXPORT_LESS_THAN_Z: Exported Z will be assumed to be less than the source z value 02 - EXPORT_GREATER_THAN_Z: Exported Z will be assumed to be greater than the source z value 03 - EXPORT_RESERVED: Reserved

DB:DB_SRESULTS_COMPARE_STATE0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ac0

Field Name	Bits	Default	Description
COMPAREFUNC0	2:0	none	Used to determine the meaning of the MayPass and MayFail smask bits during hierarchical stencil testing. NEVER or ALWAYS invalidates the SResults in the HTile Buffer POSSIBLE VALUES: 00 - REF_NEVER: never pass 01 - REF_LESS: pass if left < right 02 - REF_EQUAL: pass if left = right 03 - REF_LEQUAL: pass if left <= right 04 - REF_GREATER: pass if left > right 05 - REF_NOTEQUAL: pass if left != right 06 - REF_GEQUAL: pass if left >= right 07 - REF_ALWAYS: always pass
COMPAREVALUE0	11:4	none	Stencil value compared against the stencil reference value during hierarchical stencil testing.
COMPAREMASK0	19:12	none	This value is ANDed with the SResults compare value. A mask of 0 invalidates the SResults in the HTile Buffer
ENABLE0	24	none	If set, use SResults in HiS test. Set when compare state is known and clear when doing a resummarize.

DB:DB_SRESULTS_COMPARE_STATE1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28ac4

Field Name	Bits	Default	Description
COMPAREFUNC1	2:0	none	Used to determine the meaning of the MayPass and MayFail smask bits during hierarchical stencil testing. NEVER or ALWAYS invalidates the SResults in the HTile Buffer POSSIBLE VALUES: 00 - REF_NEVER: never pass 01 - REF_LESS: pass if left < right

			02 - REF_EQUAL: pass if left = right 03 - REF_LEQUAL: pass if left <= right 04 - REF_GREATER: pass if left > right 05 - REF_NOTEQUAL: pass if left != right 06 - REF_GEQUAL: pass if left >= right 07 - REF_ALWAYS: always pass
COMPAREVALUE1	11:4	none	Stencil value compared against the stencil reference value during hierarchical stencil testing.
COMPAREMASK1	19:12	none	This value is ANDed with the SResults compare value. A mask of 0 invalidates the SResults in the HTile Buffer
ENABLE1	24	none	If set, use SResults in HiS test. Set when compare state is known and clear when doing a resummarize.

DB:DB_STENCILREFMASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28430			
Field Name	Bits	Default	Description
STENCILREF	7:0	none	Specifies the reference stencil value for front facing primitives.
STENCILMASK	15:8	none	This value is ANDed with both the reference and the current stencil value prior to the stencil test for front facing primitives.
STENCILWRITEMASK	23:16	none	Specifies the write mask for the stencil planes for front facing primitives.

DB:DB_STENCILREFMASK_BF · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28434			
Field Name	Bits	Default	Description
STENCILREF_BF	7:0	none	Specifies the reference stencil value for back facing primitives.
STENCILMASK_BF	15:8	none	This value is ANDed with both the reference and the current stencil value prior to the stencil test for back facing primitives.
STENCILWRITEMASK_BF	23:16	none	Specifies the write mask for the stencil planes for back facing primitives.

DB:DB_STENCIL_CLEAR · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28028			
Field Name	Bits	Default	Description
CLEAR	7:0	none	Stencil value when SMEM==0, which specifies that the tile is cleared to background stencil values. Cannot be changed without clearing or previously expanding the stencil buffer.

DB:DB_STENCIL_INFO · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28044

Field Name	Bits	Default	Description
FORMAT	0	none	Specifies the size of the Stencil component. POSSIBLE VALUES: 00 - STENCIL_INVALID: Invalid stencil surface. 01 - STENCIL_8: 8-bit INT stencil surface.
TILE_SPLIT	10:8	none	Specifies the number of bytes that will be stored contiguously for each tile. If the tile data requires more storage than this amount, it is split into multiple slices. This field must not be larger than GB_ADDR_CONFIG.DRAM_ROW_SIZE. Only applies to 2D tiling modes POSSIBLE VALUES: 00 - ADDR_SURF_TILE_SPLIT_64B: 01 - ADDR_SURF_TILE_SPLIT_128B: 02 - ADDR_SURF_TILE_SPLIT_256B: 03 - ADDR_SURF_TILE_SPLIT_512B: 04 - ADDR_SURF_TILE_SPLIT_1KB: 05 - ADDR_SURF_TILE_SPLIT_2KB: 06 - ADDR_SURF_TILE_SPLIT_4KB:
ALLOW_EXPCLEAR	27	none	Allow Stencil Memory Format to keep track of expanded and clear.
TILE_STENCIL_DISABLE	29	none	Indicates that htile buffer has no stencil metadata. This improves hiz precision at the cost of having no stencil compression or HiStencil optimizations.

DB:DB_STENCIL_READ_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2804c			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	Location of the first byte of the Stencil surface for READ in Device Address Space, which must be 256 byte aligned. High 32-bits of 40-bit address.

DB:DB_STENCIL_WRITE_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28054			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	Location of the first byte of the Stencil surface for WRITE in Device Address Space, which must be 256 byte aligned. High 32-bits of 40-bit address.

DB:DB_SUBTILE_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9858			
DESCRIPTION: *Controls subtile X and Y size for each MSAA level, for a squad`s size to a full tile*			
Field Name	Bits	Default	Description
MSAA1_X	1:0	0x0	1xMSAA squad is 4x4 : auto, 4, 4, 8 (2 not allowed since < a squad in X)

MSAA1_Y	3:2	0x0	1xMSAA squad is 4x4 : auto, 4, 4, 8 (2 not allowed since < a squad in Y)
MSAA2_X	5:4	0x0	2xMSAA squad is 4x2 : auto, 4, 4, 8 (2 not allowed since < a squad in X)
MSAA2_Y	7:6	0x0	2xMSAA squad is 4x2 : auto, 2, 4, 8
MSAA4_X	9:8	0x0	4xMSAA squad is 2x2 : auto, 2, 4, 8
MSAA4_Y	11:10	0x0	4xMSAA squad is 2x2 : auto, 2, 4, 8
MSAA8_X	13:12	0x0	8xMSAA squad is 2x1 : auto, 2, 4, 8
MSAA8_Y	15:14	0x0	8xMSAA squad is 2x1 : auto, 2, 4, 8 (1 not allowed since want a mininum of a full quad)
MSAA16_X	17:16	0x0	
MSAA16_Y	19:18	0x0	

DB:DB_ZPASS_COUNT_HI · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9874

Field Name	Bits	Default	Description
COUNT_HI	30:0	none	zpass counter [62:32]

DB:DB_ZPASS_COUNT_LOW · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x9870

Field Name	Bits	Default	Description
COUNT_LOW	31:0	none	zpass counter [31:0]

DB:DB_Z_INFO · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28040

Field Name	Bits	Default	Description
FORMAT	1:0	none	Specifies the size of the depth component and whether depth is floating point. POSSIBLE VALUES: 00 - Z_INVALID: Invalid depth surface. 01 - Z_16: 16-bit UNORM depth surface. 02 - Z_24: 24-bit UNORM depth surface. 03 - Z_32_FLOAT: 32-bit FLOAT depth surface.
NUM_SAMPLES	3:2	none	Specifies thye MSAA surface footprint of the Z surface.
ARRAY_MODE	7:4	none	Specifies the tiling format for this array. POSSIBLE VALUES: 02 - ARRAY_1D_TILED_THIN1: Uses 1D 8x8x1 tiles. Not valid for AA modes. 04 - ARRAY_2D_TILED_THIN1: Uses 8x8x1 macro-tiles
TILE_SPLIT	10:8	none	Specifies the number of bytes that will be stored contiguously for each tile. If the tile data requires more storage than this amount, it is split into multiple slices.

			This field must not be larger than GB_ADDR_CONFIG.DRAM_ROW_SIZE. Only applies to 2D tiling modes POSSIBLE VALUES: 00 - ADDR_SURF_TILE_SPLIT_64B: 01 - ADDR_SURF_TILE_SPLIT_128B: 02 - ADDR_SURF_TILE_SPLIT_256B: 03 - ADDR_SURF_TILE_SPLIT_512B: 04 - ADDR_SURF_TILE_SPLIT_1KB: 05 - ADDR_SURF_TILE_SPLIT_2KB: 06 - ADDR_SURF_TILE_SPLIT_4KB:
NUM_BANKS	13:12	none	Specifies the number of memory banks for tiling purposes POSSIBLE VALUES: 00 - ADDR_SURF_2_BANK: 01 - ADDR_SURF_4_BANK: 02 - ADDR_SURF_8_BANK: 03 - ADDR_SURF_16_BANK:
BANK_WIDTH	17:16	none	Specifies the number of tiles in the x direction to be incorporated into the same bank. Only applies to 2D tiling modes POSSIBLE VALUES: 00 - ADDR_SURF_BANK_WIDTH_1: 01 - ADDR_SURF_BANK_WIDTH_2: 02 - ADDR_SURF_BANK_WIDTH_4: 03 - ADDR_SURF_BANK_WIDTH_8:
BANK_HEIGHT	21:20	none	Specifies the number of tiles in the y direction to be incorporated into the same bank. Only applies to 2D tiling modes POSSIBLE VALUES: 00 - ADDR_SURF_BANK_HEIGHT_1: 01 - ADDR_SURF_BANK_HEIGHT_2: 02 - ADDR_SURF_BANK_HEIGHT_4: 03 - ADDR_SURF_BANK_HEIGHT_8:
MACRO_TILE_ASPECT	25:24	none	Specifies the macro tile aspect ratio. Only applies to 2D tiling modes POSSIBLE VALUES: 00 - ADDR_SURF_MACRO_ASPECT_1: 01 - ADDR_SURF_MACRO_ASPECT_2: 02 - ADDR_SURF_MACRO_ASPECT_4: 03 - ADDR_SURF_MACRO_ASPECT_8:
ALLOW_EXPCLEAR	27	none	Allow ZMask to keep track of expanded and clear.
READ_SIZE	28	none	Sets the minimum size for reads to be 512 bits. Set if the surface is in a memory pool that has granularity penalty with < 512 bit accesses.

			POSSIBLE VALUES: 00 - READ_256_BITS 01 - READ_512_BITS
TILE_SURFACE_ENABLE	29	none	Enables reading and writing of the htile data. If off HiZ+S is off.
ZRANGE_PRECISION	31	none	0 = ZMin is the base, generally set when doing a Z > test, 1 = ZMax is the base, set when generally using a Z < test. The value used as base has full 14 bit precision. By setting the base to Max culling has less error in a < test. Can only be changed after a full surface clear. This field is only meaningful if TILE_Z_ONLY == 0

DB:DB_Z_READ_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28048			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	Location of the first byte of the Z surface for READ in Device Address Space, which must be 256 byte aligned. High 32-bits of 40-bit address.

DB:DB_Z_WRITE_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28050			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	Location of the first byte of the Z surface for WRITE in Device Address Space, which must be 256 byte aligned. High 32-bits of 40-bit address.

15. Color Buffer Registers

CB:CB_BLEND[0-7]_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28780-0x2879c			
DESCRIPTION: *Per-MRT blend control for MRTs 0..7.*			
Field Name	Bits	Default	Description
COLOR_SRCBLEND	4:0	none	Source blend function for RGB components. BLEND_X name corresponds to GL_X blend function. POSSIBLE VALUES: 00 - BLEND_ZERO: (d3d_zero) 01 - BLEND_ONE: (d3d_one) 02 - BLEND_SRC_COLOR: (d3d_srccolor) 03 - BLEND_ONE_MINUS_SRC_COLOR: (d3d_invsrccolor) 04 - BLEND_SRC_ALPHA: (d3d_srcalpha) 05 - BLEND_ONE_MINUS_SRC_ALPHA: (d3d_invsrcalpha) 06 - BLEND_DST_ALPHA: (d3d_destalpha) 07 - BLEND_ONE_MINUS_DST_ALPHA: (d3d_invdestalpha) 08 - BLEND_DST_COLOR: (d3d_destcolor) 09 - BLEND_ONE_MINUS_DST_COLOR: (d3d_invdestcolor) 10 - BLEND_SRC_ALPHA_SATURATE: (d3d_srcalphasat) 11 - BLEND_BOTH_SRC_ALPHA: DEPRECATED. Do not use. 12 - BLEND_BOTH_INV_SRC_ALPHA: DEPRECATED. Do not use. 13 - BLEND_CONSTANT_COLOR: (d3d_blendfactor, uses corresponding RB_BLEND component) 14 - BLEND_ONE_MINUS_CONSTANT_COLOR: (d3d_invblendfactor) 15 - BLEND_SRC1_COLOR: DX10 dual-source mode 16 - BLEND_INV_SRC1_COLOR: DX10 dual-source mode 17 - BLEND_SRC1_ALPHA: DX10 dual-source mode 18 - BLEND_INV_SRC1_ALPHA: DX10 dual-source mode 19 - BLEND_CONSTANT_ALPHA: (uses RB_BLEND_ALPHA) 20 - BLEND_ONE_MINUS_CONSTANT_ALPHA:
COLOR_COMB_FCN	7:5	none	Source/dest combination function for RGB components. Result is clamped to the representable range. POSSIBLE VALUES:

			00 - COMB_DST_PLUS_SRC: (ADD): Source*SRCBLEND + Dest*DSTBLEND 01 - COMB_SRC_MINUS_DST: (SUBTRACT): Source*SRCBLEND - Dest*DSTBLEND 02 - COMB_MIN_DST_SRC: (MIN): min(Source, Dest) 03 - COMB_MAX_DST_SRC: (MAX): max(Source, Dest) 04 - COMB_DST_MINUS_SRC: (REVSUBTRACT): Dest*DSTBLEND - Source*SRCBLEND
COLOR_DESTBLEND	12:8	none	Destination blend function for RGB components. BLEND_X name corresponds to GL_X blend function. POSSIBLE VALUES: 00 - BLEND_ZERO: (d3d_zero) 01 - BLEND_ONE: (d3d_one) 02 - BLEND_SRC_COLOR: (d3d_srccolor) 03 - BLEND_ONE_MINUS_SRC_COLOR: (d3d_invsrccolor) 04 - BLEND_SRC_ALPHA: (d3d_srcalpha) 05 - BLEND_ONE_MINUS_SRC_ALPHA: (d3d_invsrcalpha) 06 - BLEND_DST_ALPHA: (d3d_destalpha) 07 - BLEND_ONE_MINUS_DST_ALPHA: (d3d_invdestalpha) 08 - BLEND_DST_COLOR: (d3d_destcolor) 09 - BLEND_ONE_MINUS_DST_COLOR: (d3d_invdestcolor) 10 - BLEND_SRC_ALPHA_SATURATE: (d3d_srcalphasat) 11 - BLEND_BOTH_SRC_ALPHA: DEPRECATED. Do not use. 12 - BLEND_BOTH_INV_SRC_ALPHA: DEPRECATED. Do not use. 13 - BLEND_CONSTANT_COLOR: (d3d_blendfactor, uses corresponding RB_BLEND component) 14 - BLEND_ONE_MINUS_CONSTANT_COLOR: (d3d_invblendfactor) 15 - BLEND_SRC1_COLOR: DX10 dual-source mode 16 - BLEND_INV_SRC1_COLOR: DX10 dual-source mode 17 - BLEND_SRC1_ALPHA: DX10 dual-source mode 18 - BLEND_INV_SRC1_ALPHA: DX10 dual-source mode 19 - BLEND_CONSTANT_ALPHA: (uses RB_BLEND_ALPHA) 20 - BLEND_ONE_MINUS_CONSTANT_ALPHA:
ALPHA_SRCBLEND	20:16	none	Source blend function for alpha component. BLEND_X

			name corresponds to GL_X blend function. POSSIBLE VALUES: 00 - BLEND_ZERO: (d3d_zero) 01 - BLEND_ONE: (d3d_one) 02 - BLEND_SRC_COLOR: (d3d_srccolor) 03 - BLEND_ONE_MINUS_SRC_COLOR: (d3d_invsrccolor) 04 - BLEND_SRC_ALPHA: (d3d_srcalpha) 05 - BLEND_ONE_MINUS_SRC_ALPHA: (d3d_invsrcalpha) 06 - BLEND_DST_ALPHA: (d3d_destalpha) 07 - BLEND_ONE_MINUS_DST_ALPHA: (d3d_invdestalpha) 08 - BLEND_DST_COLOR: (d3d_destcolor) 09 - BLEND_ONE_MINUS_DST_COLOR: (d3d_invdestcolor) 10 - BLEND_SRC_ALPHA_SATURATE: (d3d_srcalphasat) 11 - BLEND_BOTH_SRC_ALPHA: DEPRECATED. Do not use. 12 - BLEND_BOTH_INV_SRC_ALPHA: DEPRECATED. Do not use. 13 - BLEND_CONSTANT_COLOR: (d3d_blendfactor, uses corresponding RB_BLEND component) 14 - BLEND_ONE_MINUS_CONSTANT_COLOR: (d3d_invblendfactor) 15 - BLEND_SRC1_COLOR: DX10 dual-source mode 16 - BLEND_INV_SRC1_COLOR: DX10 dual-source mode 17 - BLEND_SRC1_ALPHA: DX10 dual-source mode 18 - BLEND_INV_SRC1_ALPHA: DX10 dual-source mode 19 - BLEND_CONSTANT_ALPHA: (uses RB_BLEND_ALPHA) 20 - BLEND_ONE_MINUS_CONSTANT_ALPHA:
ALPHA_COMB_FCN	23:21	none	Source/dest combination function for alpha component. Result is clamped to the representable range. Note that Min and Max do not force src and dst blend functions to ONE. POSSIBLE VALUES: 00 - COMB_DST_PLUS_SRC: (ADD): Source*SRCBLEND + Dest*DSTBLEND 01 - COMB_SRC_MINUS_DST: (SUBTRACT): Source*SRCBLEND - Dest*DSTBLEND 02 - COMB_MIN_DST_SRC: (MIN): min(Source, Dest) 03 - COMB_MAX_DST_SRC: (MAX): max(Source,

			Dest) 04 - COMB_DST_MINUS_SRC: (REVSUBTRACT): Dest*DSTBLEND - Source*SRCBLEND
ALPHA_DESTBLEND	28:24	none	Destination blend function for alpha component. BLEND_X name corresponds to GL_X blend function. POSSIBLE VALUES: 00 - BLEND_ZERO: (d3d_zero) 01 - BLEND_ONE: (d3d_one) 02 - BLEND_SRC_COLOR: (d3d_srccolor) 03 - BLEND_ONE_MINUS_SRC_COLOR: (d3d_invsrccolor) 04 - BLEND_SRC_ALPHA: (d3d_srcalpha) 05 - BLEND_ONE_MINUS_SRC_ALPHA: (d3d_invsrcalpha) 06 - BLEND_DST_ALPHA: (d3d_destalpha) 07 - BLEND_ONE_MINUS_DST_ALPHA: (d3d_invdestalpha) 08 - BLEND_DST_COLOR: (d3d_destcolor) 09 - BLEND_ONE_MINUS_DST_COLOR: (d3d_invdestcolor) 10 - BLEND_SRC_ALPHA_SATURATE: (d3d_srcalphasat) 11 - BLEND_BOTH_SRC_ALPHA: DEPRECATED. Do not use. 12 - BLEND_BOTH_INV_SRC_ALPHA: DEPRECATED. Do not use. 13 - BLEND_CONSTANT_COLOR: (d3d_blendfactor, uses corresponding RB_BLEND component) 14 - BLEND_ONE_MINUS_CONSTANT_COLOR: (d3d_invblendfactor) 15 - BLEND_SRC1_COLOR: DX10 dual-source mode 16 - BLEND_INV_SRC1_COLOR: DX10 dual-source mode 17 - BLEND_SRC1_ALPHA: DX10 dual-source mode 18 - BLEND_INV_SRC1_ALPHA: DX10 dual-source mode 19 - BLEND_CONSTANT_ALPHA: (uses RB_BLEND_ALPHA) 20 - BLEND_ONE_MINUS_CONSTANT_ALPHA:
SEPARATE_ALPHA_BLEND	29	none	If false, use color blend modes for blending the alpha channel. If true, use the ALPHA_ fields to control blending to the alpha channel.
ENABLE	30	none	Enables blending for the corresponding MRT if it is 1, else disables blending for that MRT if it is 0. Blending and ROP3 cannot both be enabled.

CB:CB_BLEND_ALPHA · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28420			
DESCRIPTION: *Blend colour constant.*			
Field Name	Bits	Default	Description
BLEND_ALPHA	31:0	none	FP32 alpha component of constant blend color.

CB:CB_BLEND_BLUE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2841c			
DESCRIPTION: *Blend colour constant.*			
Field Name	Bits	Default	Description
BLEND_BLUE	31:0	none	FP32 blue component of constant blend color.

CB:CB_BLEND_GREEN · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28418			
DESCRIPTION: *Blend colour constant.*			
Field Name	Bits	Default	Description
BLEND_GREEN	31:0	none	FP32 green component of constant blend color.

CB:CB_BLEND_RED · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28414			
DESCRIPTION: *Blend colour constant.*			
Field Name	Bits	Default	Description
BLEND_RED	31:0	none	FP32 red component of constant blend color.

CB:CB_COLOR[0-11]_ATTRIB · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c74-0x28ea8			
DESCRIPTION: *Surface address information*			
Field Name	Bits	Default	Description
IGNORE_SHADER_ENGINE_TILING	3	none	When asserted will force NUM_SHADER_ENGINES to 0 into the addrlib.
NON_DISP_TILING_ORDER	4	none	Indicates the micro-tiling used. Depth, stencil, and FMask surfaces implicitly utilize NON_DISPLAY. ARRAY_MODE == ARRAY_LINEAR_* render targets must use NON_DISPLAY. ARRAY_MODE == ARRAY_LINEAR_* textures must use DISPLAY. POSSIBLE VALUES: 00 - ADDR_SURF_MICRO_TILING_DISPLAY: Not supported for 128bpp 01 - ADDR_SURF_MICRO_TILING_NON_DISPLAY: TC only supports 8, 16 , 32, 64, 128 and BC* textures. CB supports for all render target formats.
TILE_SPLIT	8:5	none	Specifies the number of bytes that will be stored contiguously for each tile. If the tile data requires more storage than this amount, it is split into multiple slices.

			This field must not be larger than GB_ADDR_CONFIG.DRAM_ROW_SIZE. Only applies to 2D tiling modes. POSSIBLE VALUES: 00 - ADDR_SURF_TILE_SPLIT_64B: 01 - ADDR_SURF_TILE_SPLIT_128B: 02 - ADDR_SURF_TILE_SPLIT_256B: 03 - ADDR_SURF_TILE_SPLIT_512B: 04 - ADDR_SURF_TILE_SPLIT_1KB: 05 - ADDR_SURF_TILE_SPLIT_2KB: 06 - ADDR_SURF_TILE_SPLIT_4KB:
NUM_BANKS	11:10	none	Specifies the number of memory banks for tiling purposes. POSSIBLE VALUES: 00 - ADDR_SURF_2_BANK: 01 - ADDR_SURF_4_BANK: 02 - ADDR_SURF_8_BANK: 03 - ADDR_SURF_16_BANK:
BANK_WIDTH	14:13	none	Specifies the number of tiles in the x direction to be incorporated into the same bank. Only applies to 2D tiling modes. POSSIBLE VALUES: 00 - ADDR_SURF_BANK_WIDTH_1: 01 - ADDR_SURF_BANK_WIDTH_2: 02 - ADDR_SURF_BANK_WIDTH_4: 03 - ADDR_SURF_BANK_WIDTH_8:
BANK_HEIGHT	17:16	none	Specifies the number of tiles in the y direction to be incorporated into the same bank. Only applies to 2D tiling modes. (Note: Fmask uses a different register field for bank height.) POSSIBLE VALUES: 00 - ADDR_SURF_BANK_HEIGHT_1: 01 - ADDR_SURF_BANK_HEIGHT_2: 02 - ADDR_SURF_BANK_HEIGHT_4: 03 - ADDR_SURF_BANK_HEIGHT_8:
MACRO_TILE_ASPECT	20:19	none	Specifies the macro tile aspect ratio. Only applies to 2D tiling modes. POSSIBLE VALUES: 00 - ADDR_SURF_MACRO_ASPECT_1: 01 - ADDR_SURF_MACRO_ASPECT_2: 02 - ADDR_SURF_MACRO_ASPECT_4: 03 - ADDR_SURF_MACRO_ASPECT_8:
FMASK_BANK_HEIGHT	23:22	none	POSSIBLE VALUES: 00 - ADDR_SURF_BANK_HEIGHT_1: 01 - ADDR_SURF_BANK_HEIGHT_2:

			02 - ADDR_SURF_BANK_HEIGHT_4: 03 - ADDR_SURF_BANK_HEIGHT_8:
NUM_SAMPLES	26:24	none	Specifies log2 of the number of samples. This may not be greater than 4.
NUM_FRAGMENTS	28:27	none	Specifies log2 of the number of fragments. This may not be greater than NUM_SAMPLES.
FORCE_DST_ALPHA_1	31	none	Forces the hw to use 1 for the dst alpha value. For use with formats that do not have an alpha component. Must be 0 if CB_COLORn_INFO.RAT=1.

CB:CB_COLOR[0-11]_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c60-0x28e94

DESCRIPTION: *Base address for colour surface.*

Field Name	Bits	Default	Description
BASE_256B	31:0	none	This specifies bits [39:8] of the byte address of the start of the resource in device address space. For linear-general surfaces, bits [7:0] of the byte address are specified in CB_COLOR*_VIEW.SLICE_START; for all other surfaces, bits [7:0] of the byte address are always zero. Pipe and bank swizzles can be specified here. Bits [p-1:0] of this field, where p = log2(numPipes), specifiy the pipe swizzle. Bits [p+b-1:p], where b = log2(numBanks) specify the bank swizzle. Since RV770 has a 36-bit address space, bits [39:36] of any 40-bit address should be zero.

CB:CB_COLOR[0-7]_CLEAR_WORD0 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c8c-0x28e30

DESCRIPTION: *Bits [31:0] of the per-MRT formatted fast clear color. Pixel size Clear color 8bpp WORD0[7:0] 16bpp WORD0[15:0] 32bpp WORD0[31:0] 64bpp {WORD1[31:0], WORD0[31:0]} 128bpp Unsupported*

Field Name	Bits	Default	Description
CLEAR_WORD0	31:0	none	

CB:CB_COLOR[0-7]_CLEAR_WORD1 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c90-0x28e34

DESCRIPTION: *Bits [63:32] of the per-MRT formatted fast clear color. Pixel size Clear color 8bpp WORD0[7:0] 16bpp WORD0[15:0] 32bpp WORD0[31:0] 64bpp {WORD1[31:0], WORD0[31:0]} 128bpp Unsupported*

Field Name	Bits	Default	Description
CLEAR_WORD1	31:0	none	

CB:CB_COLOR[0-7]_CLEAR_WORD2 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c94-0x28e38

Field Name	Bits	Default	Description
CLEAR_WORD2	31:0	none	

CB:CB_COLOR[0-7]_CLEAR_WORD3 · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c98-0x28e3c			
Field Name	Bits	Default	Description
CLEAR_WORD3	31:0	none	

CB:CB_COLOR[0-7]_CMASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c7c-0x28e20			
DESCRIPTION: *Base address for cmask surface.*			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	This specifies bits [39:8] of the byte address of the start of the CMASK per-tile data, if any, in device address space. Since Wekiva has a 36-bit address space, bits [39:36] of any 40-bit address should be zero.

CB:CB_COLOR[0-7]_CMASK_SLICE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c80-0x28e24			
DESCRIPTION: *Cmask slice settings for MRT 0...7*			
Field Name	Bits	Default	Description
TILE_MAX	13:0	none	This field equals one less than the number of 128x128 blocks (16x16 tiles) of CMASK data per slice.

CB:CB_COLOR[0-11]_DIM · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c78-0x28eac			
DESCRIPTION: *Sets the range of the render target resource.*			
Field Name	Bits	Default	Description
WIDTH_MAX	15:0	none	For BUFFER resource type this is ELEMENTS_MAX[15:0]. For TEXTURE* resource type it is Width-1 of the surface.
HEIGHT_MAX	31:16	none	For BUFFER resource type this is ELEMENTS_MAX[31:16]. For TEXTURE* resource type it is Height-1 of the surface.

CB:CB_COLOR[0-7]_FMASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c84-0x28e28			
DESCRIPTION: *Base address for fmask surface.*			
Field Name	Bits	Default	Description
BASE_256B	31:0	none	This specifies bits [39:8] of the byte address of the start of the resource in device address space. Pipe and bank swizzles can be specified here. Bits [p-1:0] of this field, where p = log2(numPipes), specifiy the pipe swizzle. Bits [p+b-1:p], where b = log2(numBanks) specify the bank swizzle. Since Wekiva has a 36-bit address space, bits [39:36] of any 40-bit address should be zero.

CB:CB_COLOR[0-7]_FMASK_SLICE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c88-0x28e2c			
DESCRIPTION: *Mask settings for MRT 0..7.*			
Field Name	Bits	Default	Description
TILE_MAX	21:0	none	This field equals one less than the number of 8x8 tiles of FMASK data per slice. See the Pele Memory Format Specification for details on the FMASK memory layout.

CB:CB_COLOR[0-11]_INFO · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c70-0x28ea4			
DESCRIPTION: *Surface format information*			
Field Name	Bits	Default	Description
ENDIAN	1:0	none	Specifies what kind of byte swapping to perform, if any, for different endian modes. The byte swap is equivalent to computing dest[A] = src[A XOR N] for byte address A and the XOR values listed below. See the COMP_SWAP field for component swapping options. POSSIBLE VALUES: 00 - ENDIAN_NONE: No endian swapping (XOR by 0) 01 - ENDIAN_8IN16: 8 bit swap within 16 bit word (XOR by 1): 0xAABBCCDD -> 0xBBAADDCC 02 - ENDIAN_8IN32: 8 bit swap within 32 bit word (XOR by 3): 0xAABBCCDD -> 0xDDCCBBAA 03 - ENDIAN_8IN64: 8 bit swap in 64 bits (XOR by 7): 0xaabbccddeeffgghh -> 0xhhggffeeddccbbaa
FORMAT	7:2	none	Specifies the size of the color components and in some cases the number format. See the COMP_SWAP field below for mappings of RGBA (XYZW) shader pipe results to color component positions in the pixel format. When reading from the surface, missing components in the format will be substituted with the default value: 0.0 for RGB or 1.0 for alpha. POSSIBLE VALUES: 00 - COLOR_INVALID: this resource is disabled 01 - COLOR_8: norm, int 02 - RESERVED 03 - RESERVED 04 - RESERVED 05 - COLOR_16: norm, int, float 06 - COLOR_16_FLOAT: float only 07 - COLOR_8_8: norm, int 08 - COLOR_5_6_5: norm only 09 - RESERVED 10 - COLOR_1_5_5_5: norm only, 1-bit component is always unorm 11 - COLOR_4_4_4_4: norm only 12 - COLOR_5_5_5_1: norm only, 1-bit component

			is always unorm 13 - COLOR_32: int, float 14 - COLOR_32_FLOAT: float only 15 - COLOR_16_16: norm, int, float 16 - COLOR_16_16_FLOAT: float only 17 - COLOR_8_24: unorm depth, uint stencil 18 - RESERVED 19 - COLOR_24_8: unorm depth, uint stencil 20 - RESERVED 21 - COLOR_10_11_11: float only 22 - COLOR_10_11_11_FLOAT: float only 23 - RESERVED 24 - RESERVED 25 - COLOR_2_10_10_10: norm, int 26 - COLOR_8_8_8_8: norm, int, srgb 27 - COLOR_10_10_10_2: norm, int 28 - COLOR_X24_8_32_FLOAT: float depth, uint stencil 29 - COLOR_32_32: int, float 30 - COLOR_32_32_FLOAT: float only 31 - COLOR_16_16_16_16: norm, int, float 32 - COLOR_16_16_16_16_FLOAT: norm, int, float 33 - RESERVED 34 - COLOR_32_32_32_32: int, float 35 - COLOR_32_32_32_32_FLOAT: float only
ARRAY_MODE	11:8	none	Specifies the tiling format of this MRT. Each format can be used for 1D, 2D, or 3D arrays. Formats 0 and 1 are not displayable. THICK tiles use an 8x8x4 tile size (volume textures). THIN1 tiles use an 8x8x1 tile size (most other cases). THIN2 tiles double the height alignment. THIN4 tiles quadruple the height alignment. 2D tilings rotate banks for successive render target slices. 3D tilings rotate pipes and banks for render target slices. 2B/3B tilings also rotate banks at row boundaries <u>POSSIBLE VALUES:</u> 00 - ARRAY_LINEAR_GENERAL: Unaligned linear array 01 - ARRAY_LINEAR_ALIGNED: Aligned linear array 02 - ARRAY_1D_TILED_THIN1: Uses 1D 8x8x1 tiles. Not valid for AA modes. 04 - ARRAY_2D_TILED_THIN1: Uses 8x8x1 macro-tiles
NUMBER_TYPE	14:12	none	Specifies the numeric type of the color components. <u>POSSIBLE VALUES:</u> 00 - NUMBER_UNORM: unsigned repeating fraction (urf): range [0..1], scale factor (2^n)-1 01 - NUMBER_SNORM: Microsoft-style signed rf: range [-1..1], scale factor (2^(n-1))-1 02 - RESERVED

			03 - RESERVED 04 - NUMBER_UINT: zero-extended bit field, int in shader: not blendable or filterable 05 - NUMBER_SINT: sign-extended bit field, int in shader: not blendable or filterable 06 - NUMBER_SRGB: gamma corrected, range [0..1] (only supported for COLOR_8_8_8_8 format; always rounds color channels) 07 - NUMBER_FLOAT: floating point: 32-bit: IEEE float, SE8M23, bias 127, range (-2^129..2^129); 16-bit: Short float SE5M10, bias 15, range (-2^17..2^17); 11-bit: Packed float, E5M6 bias 15, range [0..2^17); 10-bit: Packed float, E5M5 bias 15, range [0..2^17)
COMP_SWAP	16:15	none	Specifies how to map the red, green, blue, and alpha components from the shader to the components in the render target pixel format (components 0, 1, 2, 3 with 0 begin least significant, 3 begin most). With one component, this selects which colour channel to map to the single render target component (STD: R=>0; ALT: G=>0; STD_REV: B=>0; ALT_REV: A=>0). With 2-4 components, SWAP_STD always maps shader components starting with R=>0 up to the number of components available (component R=>0, G=>1, B=>2, A=>3). With 2-3 components, SWAP_ALT mimics SWAP_STD except alpha from the shader is always sent to the last render target component (2 components: R=>0, A=>1; 3 components: R=>0, G=>1, A=>2). With 4 components, SWAP_ALT selects an alternate order (B=>0, G=>1, R=>2, A=>3). With 2-4 components, SWAP_STD_REV and SWAP_ALT_REV reverse the component order. <u>POSSIBLE VALUES:</u> 00 - SWAP_STD: standard little-endian comp order 01 - SWAP_ALT: alternate components or order 02 - SWAP_STD_REV: reverses SWAP_STD order 03 - SWAP_ALT_REV: reverses SWAP_ALT order
FAST_CLEAR	17	none	Enables fast clear.
COMPRESSION	18	none	Enables compression.
BLEND_CLAMP	19	none	Specifies whether to clamp source data to the format range prior to blending, in addition to the post-blend clamp. This bit must be cleared if BLEND_BYPASS is set. Otherwise, it must be set iff any component is SNORM, UNORM, SRGB.
BLEND_BYPASS	20	none	If false, the blender for this MRT is enabled/disabled as specified in CB_BLENDn_CONTROL.ENABLE. If true, blending is disabled. This bit should be set iff any component is SINT/UINT (NUMBER_TYPE = SINT, UINT, or FORMAT = COLOR_8_24, COLOR_24_8, COLOR_X24_8_32_FLOAT).
SIMPLE_FLOAT	21	none	If false, floating point processing follows full IEEE rules

			for INF, NaN, and -0. If true, 0*dst produces 0. This controls a hardware optimization for destination reads for FP16/FP32 component formats. When this bit is set for float surfaces, destination reads will be optimized out when the source is zero. This bit is ignored for other component formats.
ROUND_MODE	22	none	This field selects between truncating (standard for floats) and rounding (standard for most other cases) to convert blender results to frame buffer components. This should be set to ROUND_BY_HALF iff any component is UNORM, SNORM or SRGB (this field is ignored for COLOR_8_24 and COLOR_24_8). POSSIBLE VALUES: 00 - ROUND_BY_HALF: add 1/2 lsb and then truncate 01 - ROUND_TRUNCATE: truncate toward zero for float, else toward negative
TILE_COMPACT	23	none	Tile compact attempts to save memory footprint in a multi-GPU system by collapsing the holes in the surface that are stored on other GPUs. This feature may not be implemented in the DX10 driver because the texture cannot read the surface so we cannot do shader resolves. This is potentially useful for DX9, but there are no plans to implement it. Currently this bit is hooked up in the CB but has not been thoroughly tested.
SOURCE_FORMAT	25:24	none	This field indicates the allowed format for color data being exported from the pixel shader into the output merge block. This field must be set to EXPORT_4C_32BPC if any component is a SNORM/UNORM greater than 11 bits, or FLOAT greater than 16 bits, or any SINT/UINT number type. Otherwise this field must be set to EXPORT_4C_16BPC. POSSIBLE VALUES: 00 - EXPORT_4C_32BPC: PS exports are 4 pixels with 4 components with 32-bits-per-component. (two clocks per export) 01 - EXPORT_4C_16BPC: PS exports are 4 pixels with 4 components with 16-bits-per-component. (one clock per export) 02 - EXPORT_2C_32BPC_GR: PS exports are 4 pixels with 2 components with Green and Red channels only. (one clock per export) 03 - EXPORT_2C_32BPC_AR: PS exports are 4 pixels with 2 components with Alpha and Red channels only. (one clock per export)
RAT	26	none	
RESOURCE_TYPE	29:27	none	POSSIBLE VALUES: 00 - BUFFER:

			01 - TEXTURE1D: 02 - TEXTURE1DARRAY: 03 - TEXTURE2D: 04 - TEXTURE2DARRAY: 05 - TEXTURE3D: 06 - STRUCTUREDBUFFER:
SOURCE_NUMBER_TYPE	31:30	none	POSSIBLE VALUES: 00 - EXPORT_FLOAT: 01 - EXPORT_INT: Look at the surface format to distinguish between UINT and SINT 02 - EXPORT_UNORM: 03 - EXPORT_SNORM:

CB:CB_COLOR[0-11]_PITCH · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c64-0x28e98			
DESCRIPTION: *Pitch of color surface.*			
Field Name	Bits	Default	Description
TILE_MAX	10:0	none	Encodes the pitch of a scanline; if Pitch is the number of data elements per scanline, this field is (Pitch / 8) - 1 and is equal to the maximum 8x8 tile number allowed in the X dimension.

CB:CB_COLOR[0-11]_SLICE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c68-0x28e9c			
DESCRIPTION: *Size of color surface.*			
Field Name	Bits	Default	Description
TILE_MAX	21:0	none	Encodes the size of a slice; if SliceTiles is the maximum number of tiles in a slice (equal to Pitch * Height / 64), this field is SliceTiles - 1 and is equal to the maximum tile number tile number allowed in a slice.

CB:CB_COLOR[0-11]_VIEW · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28c6c-0x28ea0			
DESCRIPTION: *Selects slice index range for MRT 0..7.*			
Field Name	Bits	Default	Description
SLICE_START	10:0	none	For ARRAY_LINEAR_GENERAL, bits [7:0] of this field specify bits [7:0] of the byte address of the resource. This together with CB_COLOR*_BASE.BASE_256B specify the 40-bit start address. The address must be element-aligned. In ARRAY_LINEAR_GENERAL, for clamping purposes, this field is forced to zero. For all other surfaces, this specifies the starting slice number for this view: this field is added to rtindex to compute the slice to render.
SLICE_MAX	23:13	none	Specifies the maximum allowed render target slice index (rtindex) for this resource, which is one less than the total number of slices. rtindex is clamped to SLICE_START

			if this value is exceeded.

CB:CB_COLOR_CONTROL · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28808			
DESCRIPTION: *Controls general CB behaviour across all MRTs.*			
Field Name	Bits	Default	Description
DEGAMMA_ENABLE	3	none	If true, then each UNORM format COLOR_8_8_8_8 MRT is treated as an SRGB format instead. This affects both normal draw and resolve. This bit exists for compatibility with older architectures that did not have an SRGB number type.
MODE	6:4	none	This field selects standard color processing or one of several major operation modes. POSSIBLE VALUES: 00 - CB_DISABLE: Disables drawing to color buffer. Causes DB to not send tiles/quads to CB. CB itself ignores this field. 01 - CB_NORMAL: Normal rendering mode. DB should send tiles and quads for pixel exports or just quads for compute exports. 02 - CB_ELIMINATE_FAST_CLEAR: Fill fast cleared color surface locations with clear color. DB should send only tiles. 03 - CB_RESOLVE: Read from MRT0, average all samples, and write to MRT1, which is one-sample. DB should send only tiles. 04 - CB_DECOMPRESS: Decompress MRT0 to a uncompressed color format. This is required before a multisampled surface is accessed by the CPU, or used as a texture. This also decompresses the FMASK buffer. A CB_ELIMINATE_FAST_CLEAR pass before this is unnecessary. DB should send tiles and quads. 05 - CB_FMASK_DECOMPRESS: Decompress the FMASK buffer into a texture readable format. A CB_ELIMINATE_FAST_CLEAR pass before this is unnecessary. DB should send only tiles.
ROP3	23:16	none	This field supports the 28 boolean ops that combine either source and dest or brush and dest, with brush provided by the shader in place of source. The code 0xCC (11001100) copies the source to the destination, which disables the ROP function. ROP must be disabled if any MRT enables blending. POSSIBLE VALUES: 00 - 0x00: BLACKNESS 05 - 0x05 10 - 0x0A 15 - 0x0F 17 - 0x11: NOTSRCERASE

			34 - 0x22 51 - 0x33: NOTSRCCOPY 68 - 0x44: SRCERASE 80 - 0x50 85 - 0x55: DSTINVERT 90 - 0x5A: PATINVERT 95 - 0x5F 102 - 0x66: SRCINVERT 119 - 0x77 136 - 0x88: SRCAND 153 - 0x99 160 - 0xA0 165 - 0xA5 170 - 0xAA 175 - 0xAF 187 - 0xBB: MERGEPAINT 204 - 0xCC: SRCCOPY 221 - 0xDD 238 - 0xEE: SRCPAINT 240 - 0xF0: PATCOPY 245 - 0xF5 250 - 0xFA 255 - 0xFF: WHITENESS

CB:CB_IMMED[0-11]_BASE · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28b9c-0x28bc8

Field Name	Bits	Default	Description
BASE_256B	31:0	none	

CB:CB_SHADER_MASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x2823c

DESCRIPTION: *Contains color component mask fields for the colors output by the shader. The bits in OUTPUT*_ENABLE are in the same order as for TARGET*_ENABLE. Outputs 1-7 are defined equivalently to output 0.*

Field Name	Bits	Default	Description
OUTPUT0_ENABLE	3:0	none	If zero, this field disables MRT 0, else it specifies which components are enabled in the shader. The low order bit corresponds to the red channel. A one bit passes the shader output component value to the color block.
OUTPUT1_ENABLE	7:4	none	Enables output of color 1 components.
OUTPUT2_ENABLE	11:8	none	Enables output of color 2 components.
OUTPUT3_ENABLE	15:12	none	Enables output of color 3 components.
OUTPUT4_ENABLE	19:16	none	Enables output of color 4 components.
OUTPUT5_ENABLE	23:20	none	Enables output of color 5 components.
OUTPUT6_ENABLE	27:24	none	Enables output of color 6 components.
OUTPUT7_ENABLE	31:28	none	Enables output of color 7 components.

CB:CB_TARGET_MASK · [R/W] · 32 bits · Access: 32 · GpuF0MMReg:0x28238			
DESCRIPTION: *Contains color component mask fields for writing the MRTs. Red, green, blue, and alpha are components 0, 1, 2, and 3 in the pixel shader and are enabled by bits 0, 1, 2, and 3 in each field. Note that the components may be in a different order in the frame buffer, depending on the COMP_SWAP field; the bits in TARGET*_ENABLE correspond to the order of components after blending and before COMP_SWAP is applied. MRTs 1-7 are defined equivalently to output 0.*			
Field Name	Bits	Default	Description
TARGET0_ENABLE	3:0	none	Enables writing to MRT 0 components. The low order bit corresponds to the red channel. A zero bit disables writing to that channel and a one bit enables writing to that channel.
TARGET1_ENABLE	7:4	none	Enables write to MRT 1 components.
TARGET2_ENABLE	11:8	none	Enables write to MRT 2 components.
TARGET3_ENABLE	15:12	none	Enables write to MRT 3 components.
TARGET4_ENABLE	19:16	none	Enables write to MRT 4 components.
TARGET5_ENABLE	23:20	none	Enables write to MRT 5 components.
TARGET6_ENABLE	27:24	none	Enables write to MRT 6 components.
TARGET7_ENABLE	31:28	none	Enables write to MRT 7 components.