LogiCORE IP AXI Video Direct Memory Access (axi_vdma) (v2.00.a)

DS799 December 14, 2010 **Product Specification**

Introduction

The AXI Video Direct Memory Access (AXI VDMA) core is a soft Xilinx IP core for use with the Xilinx Embedded Development Kit (EDK). The AXI VDMA engine provides high-bandwidth direct memory access between memory and AXI4-Stream-video type target peripherals. Initialization, status, and management registers are accessed through an AXI4-Lite slave interface, suitable for the Xilinx MicroBlaze™ microprocessor.

Features

- AXI4 Compliant
- Independent Scatter/Gather DMA support
- Primary AXI Memory Map data width support of 32, 64, 128, and 256 bits
- Primary AXI4-Stream data width support of 8, 16, 32, 64, 128, and 256 bits
- Optional Data Re-Alignment Engine
- Optional Gen-Lock Synchronization
- Independent, asynchronous channel operation

LogiCORE IP Facts Table						
Core Specifics						
Supported Device Family[1]	Virtex-6, Spartan-6					
Supported User Interfaces	AXI4, AXI4-Lite, AXI4-Stream					
	Resources				**Frequency**	
	LUTs	**FFs**	**DSP Slices**	**Block RAMs**	**Max. Freq.**	
	See Table 27 and Table 28.					
Provided with Core						
Documentation	Product Specification					
Design Files	VHDL					
Example Design	Not Provided					
Test Bench	Not Provided					
Constraints File	Not Provided					
Simulation Model	Not Provided					
Tested Design Tools						
Design Entry Tools	EDK 12.4, XPS					
Simulation	QuestaSim-64 6.5c					
Synthesis Tools	XST					
Support						
Provided by Xilinx, Inc.						

1. For a complete listing of supported devices, see the release notes for this core.

Applications

The AXI VDMA provides high speed data movement between system memory and AXI4-Stream-based target Video IP.

Functional Description

Figure 1 illustrates the functional composition of the core. The core's design has four AXI Memory Map interfaces: AXI4-Lite Slave, AXI Memory Map Read Master, AXI Memory Map Write Master, and AXI Memory Map Scatter/Gather Read/Write Master. Associated with the memory map interfaces are two AXI4-Streaming interfaces: AXI MM2S Stream Master, AXI S2MM Stream Slave.

Optional Gen-Lock and Video Frame Sync Interfaces are also provided for each channel.

Register access and configuration is provided through the AXI4-Lite slave interface. The register module provides control and status for DMA operations.

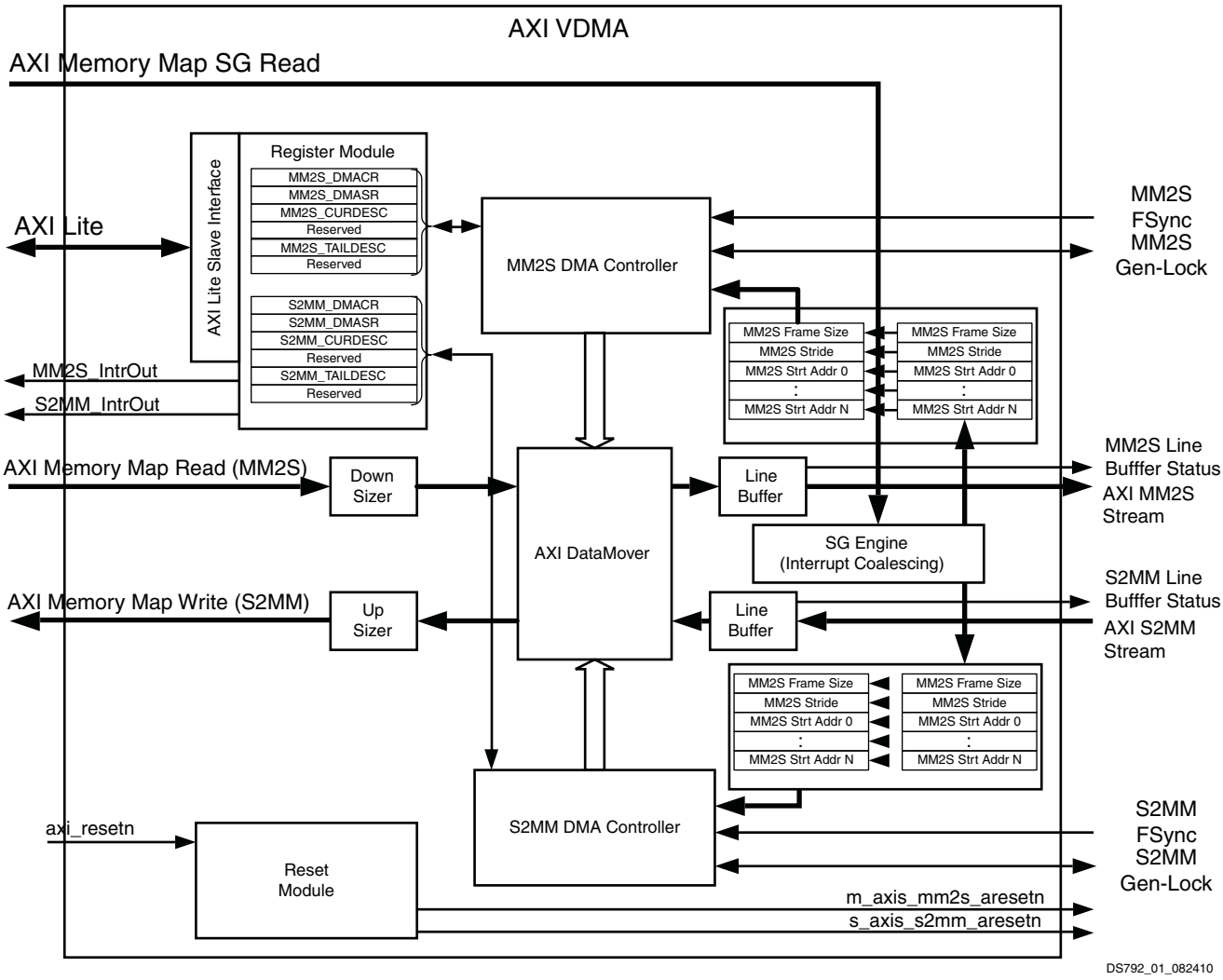

Figure 1: **AXI VDMA Block Diagram**

Primary high-speed DMA data movement between system memory and stream target is through the AXI Memory Map Read Master to AXI MM2S Stream Master, and AXI S2MM Stream Slave to AXI Memory Map Write Master. The AXI DataMover is used for high throughput transfer of data from memory to stream and from stream to memory. The MM2S channel and S2MM channel operate independently and in a full duplex like method. The AXI DataMover provides the AXI VDMA with 4 kbyte address boundary protection, automatic burst partitioning, as well as providing the ability to queue multiple transfer requests using nearly the full bandwidth capabilities of the AXI4-Stream buses. Furthermore, the AXI DataMover provides byte-level data realignment allowing memory reads and writes to any byte offset location.

The AXI VDMA provides a Scatter/Gather Engine for offloading CPU management tasks to hardware. The Scatter/Gather Engine fetches and updates buffer descriptors from system memory through the AXI Memory Map Scatter Gather Read/Write Master interface. Optional descriptor queuing is provided to maximize primary data throughput.

Typical System Interconnect

The AXI VDMA core is designed to be connected via AXI Interconnect in the user's system. A typical MicroBlaze processor configuration is shown in Figure 2. The system's microprocessor has access to the AXI VDMA through the AXI4-Lite interface. An integral Scatter/Gather Engine fetches buffer descriptors from DDRx which then coordinates primary data transfers between Video IP and DDRx. The dual interrupt output of the AXI VDMA core is routed to the System Interrupt Controller.

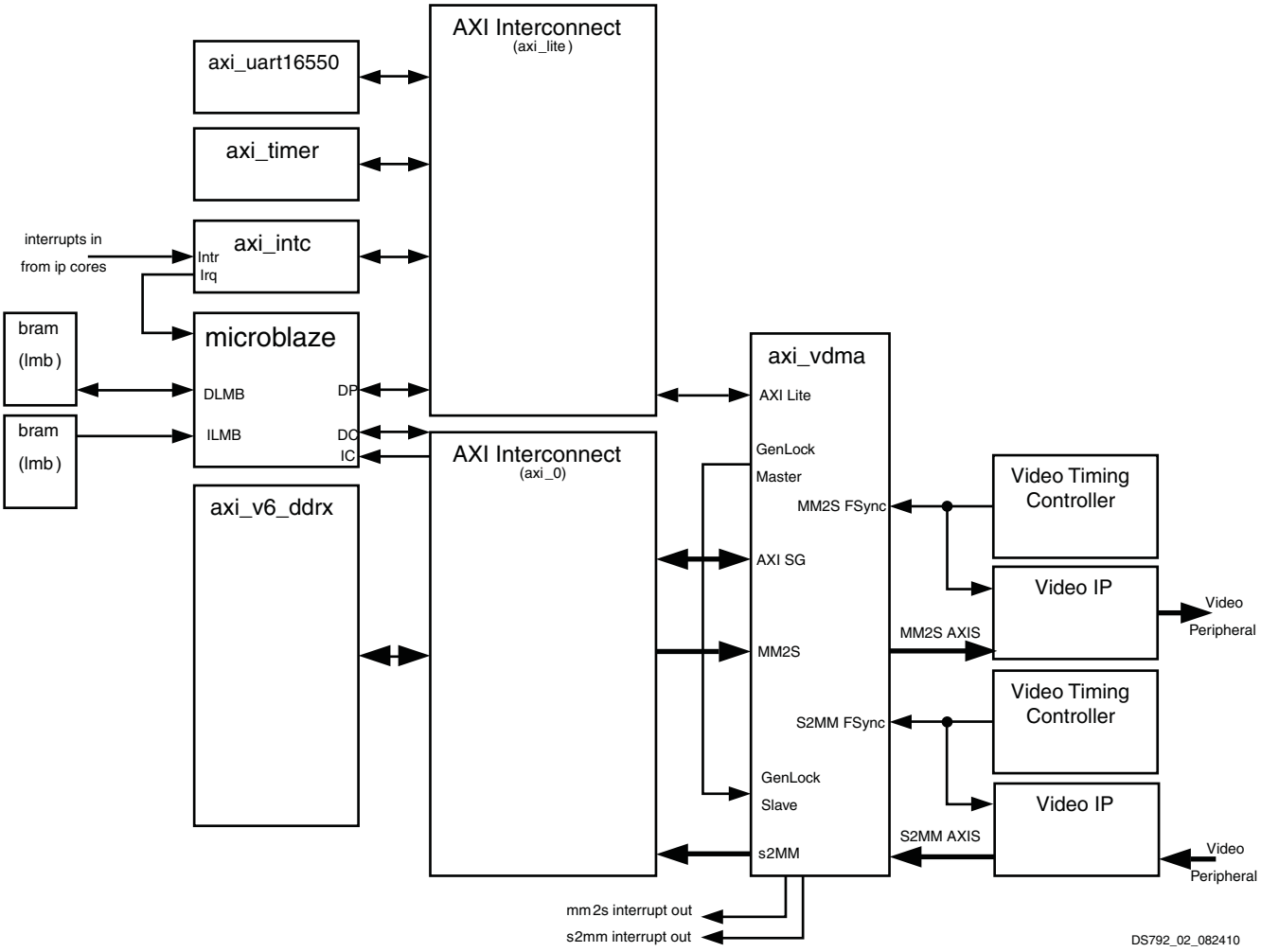

Figure 2: **Typical MicroBlaze Processor System Configuration**

I/O Signals

The AXI VDMA signals are described in Table 1.

Table 1: **AXI VDMA I/O Signal Description**

Signal Name	Interface	Signal Type	Init Status	Description
axi_scndry_aclk	Clock	I		AXI VDMA Secondary Clock. Must be less than or equal to Primary Clock for asynchronous mode. (C_PRMRY_IS_ACLK_ASYNC=1)
axi_mm2s_aclk	Clock	I		AXI VDMA MM2S Clock. MM2S clock must be tied to axi_scndry_aclk source for synchronous mode. (C_PRMRY_IS_ACLK_ASYNC=0)
axi_s2mm_aclk	Clock	I		AXI VDMA S2MM Clock. S2MM clock must be tied to axi_scndry_aclk source for synchronous mode. (C_PRMRY_IS_ACLK_ASYNC=0)
axi_resetn	Reset	I		AXI VDMA Reset. Active low reset. When asserted low, resets entire AXI VDMA core. Must be synchronous to axi_scndry_aclk.
mm2s_introut	Interrupt	O	0	Interrupt Out for Memory Map to Stream Channel.
s2mm_introut	Interrupt	O	0	Interrupt Out for Stream to Memory Map Channel.
Video Synchronization Interface				
mm2s_fsync	Frame Sync	I		MM2S Frame Sync Input. When enabled VDMA Operations begin on each falling edge of fsync. This port is only valid when Use Frame Sync is enabled (C_USE_FSYNC=1)
mm2s_fsync_out	Frame Sync	O	0	MM2S Frame Sync Output. This signal asserts high for 1 axi_mm2s_aclk cycle with each frame boundary. This signals indicates to target video IP when transfer of mms2 new frame data will begin. ***Note:*** In free run mode (C_USE_FSYNC = 0) s2mm_fsync_out will strobe high when all of the data for a frame as been transferred.
mm2s_prmtr_update		O	0	MM2S Parameter Update. This signal indicates new mm2s video parameters will take affect on next frame. This signal is asserted for 1 axi_mm2s_aclk cycle coincident with mm2s_fsync_out.
s2mm_fsync	Frame Sync	I		S2MM Frame Sync Input. When enabled VDMA Operations begin on each falling edge of fsync. This port is only valid when Use Frame Sync is enabled. (C_USE_FSYNC=1)
s2mm_fsync_out	Frame Sync	O	0	S2MM Frame Sync Output. This signal asserts high for 1 axi_s2mm_aclk cycle with each frame boundary. Indicates when S2MM new frame data can be transferred to the S2MM channel by video IP. ***Note:*** In free run mode (C_USE_FSYNC = 0) s2mm_fsync_out will strobe high when all of the data for a frame as been transferred.
s2mm_prmtr_update		O	0	S2MM Parameter Update. This signal indicates new s2mm video parameters will take affect on next frame. This signal is asserted for 1 axi_s2mm_aclk cycle coincident with s2mm_fsync_out.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Gen-Lock Interface				
mm2s_frame_ptr_in((C_MM2S_GENLOCK_NUM_MASTERS*5)-1: 0)	Gen-Lock	I		MM2S Frame Pointer Input. In Gen-Lock Slave mode, specifies the next frame for MM2S to operate on.
mm2s_frame_ptr_out(4:0)	Gen-Lock	O	zeros	MM2S Frame Pointer Output. In Gen-Lock Master mode, specifies the next frame for the slave VDMA to operate on.
s2mm_frame_ptr_in((C_S2MM_GENLOCK_NUM_MASTERS*5)-1: 0)	Gen-Lock	I		S2MM Frame Pointer Input. In Gen-Lock Slave mode, specifies the next frame for S2MM to operate on.
s2mm_frame_ptr_out(4:0)	Gen-Lock	O	zeros	S2MM Frame Pointer Output. In Gen-Lock Master mode, specifies the next frame for the slave VDMA to operate on.
Line Buffer interface				
mm2s_buffer_empty	LineBuffer	O	1	MM2S Line Buffer Empty. Indicates the MM2S line buffer contains no stored data elements.
mm2s_buffer_almost_empty	LineBuffer	O	0	MM2S Line Buffer Almost Empty. Indicates the MM2S line buffer has C_MM2S_LINEBUFFER_THRESH bytes or less stored. When mm2s_buffer_empty asserts mm2s_buffer_almost_empty will deassert.
s2mm_buffer_full	LineBuffer	O	0	S2MM Line Buffer Full. Indicates the S2MM line buffer as no more room to store data elements
s2mm_buffer_almost_full	LineBuffer	O	0	S2MM Line Buffer Almost Full. Indicates the S2MM line buffer has C_S2MM_LINEBUFFER_THRESH bytes or more. When s2mm_buffer_full asserts s2mm_buffer_almost_full will deassert.
AXI4-Lite Interface Signals				
s_axi_lite_awvalid	S_AXI_LITE	I		AXI4-Lite Write Address Channel Write Address Valid. • 1 = Write address is valid. • 0 = Write address is not valid.
s_axi_lite_awready	S_AXI_LITE	O	0	AXI4-Lite Write Address Channel Write Address Ready. Indicates DMA ready to accept the write address. • 1 = Ready to accept address. • 0 = Not ready to accept address.
s_axi_lite_awaddr(31:0)	S_AXI_LITE	I		AXI4-Lite Write Address Bus.
s_axi_lite_wvalid	S_AXI_LITE	I		AXI4-Lite Write Data Channel Write Data Valid. • 1 = Write data is valid. • 0 = Write data is not valid.
s_axi_lite_wready	S_AXI_LITE	O	0	AXI4-Lite Write Data Channel Write Data Ready. Indicates DMA ready to accept the write data. • 1 = Ready to accept data. • 0 = Not ready to accept data.
s_axi_lite_wdata(31:0)	S_AXI_LITE	I		AXI4-Lite Write Data Bus.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
s_axi_lite_bresp(1:0)	S_AXI_LITE	O	zeros	AXI4-Lite Write Response Channel. Indicates results of the write transfer. The AXI VDMA Lite interface always responds with OKAY. • 00b = OKAY - Normal access has been successful. • 01b = EXOKAY - Not supported. • 10b = SLVERR - Not supported. • 11b = DECERR - Not supported.
s_axi_lite_bvalid	S_AXI_LITE	O	0	AXI4-Lite Write Response Channel Response Valid. Indicates response is valid. • 1 = Response is valid. • 0 = Response is not valid.
s_axi_lite_bready	S_AXI_LITE	I		AXI4-Lite Write Response Channel Ready. Indicates target is ready to receive response. • 1 = Ready to receive response. • 0 = Not ready to receive response.
s_axi_lite_arvalid	S_AXI_LITE	I		AXI4-Lite Read Address Channel Read Address Valid. • 1 = Read address is valid. • 0 = Read address is not valid.
s_axi_lite_arready	S_AXI_LITE	O	0	AXI4-Lite Read Address Channel Read Address Ready. Indicates DMA ready to accept the read address. • 1 = Ready to accept address. • 0 = Not ready to accept address.
s_axi_lite_araddr(31:0)	S_AXI_LITE	I		AXI4-Lite Read Address Bus.
s_axi_lite_rvalid	S_AXI_LITE	O	0	AXI4-Lite Read Data Channel Read Data Valid. 1 = Read data is valid. 0 = Read data is not valid.
s_axi_lite_rready	S_AXI_LITE	I		AXI4-Lite Read Data Channel Read Data Ready. Indicates target ready to accept the read data. • 1 = Ready to accept data. • 0 = Not ready to accept data.
s_axi_lite_rdata(31:0)	S_AXI_LITE	O	zeros	AXI4-Lite Read Data Bus.
s_axi_lite_rresp(1:0)	S_AXI_LITE	O	zeros	AXI4-Lite Read Response Channel Response. Indicates results of the read transfer. The AXI VDMA Lite interface always responds with OKAY. • 00b = OKAY - Normal access has been successful. • 01b = EXOKAY - Not supported. • 10b = SLVERR - Not supported. • 11b = DECERR - Not supported.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
MM2S Memory Map Read Interface Signals				
m_axi_mm2s_araddr (C_M_AXI_MM2S_ADDR_ WIDTH-1: 0)	M_AXI_MM2S	O	zeros	Read Address Channel Address Bus.
m_axi_mm2s_arlen(7:0)	M_AXI_MM2S	O	zeros	Read Address Channel Burst Length. In data beats - 1.
m_axi_mm2s_arsize(2:0)	M_AXI_MM2S	O	zeros	Read Address Channel Burst Size. Indicates with of burst transfer. • 000b = 1 byte (8-bit wide burst). • 001b = 2 bytes (16-bit wide burst). • 010b = 4 bytes (32-bit wide burst). • 011b = 8 bytes (64-bit wide burst). • 100b = 16 bytes (128-bit wide burst). • 101b = 32 bytes (256-bit wide burst). • 110b = Not Supported by AXI VDMA. • 111b = Not Supported by AXI VDMA.
m_axi_mm2s_arburst(1:0)	M_AXI_MM2S	O	zeros	Read Address Channel Burst Type. Indicates type burst. • 00b = FIXED - Not supported. • 01b = INCR - Incrementing address. • 10b = WRAP - Not supported. • 11b = Reserved.
m_axi_mm2s_arprot(2:0)	M_AXI_MM2S	O	000b	Read Address Channel Protection. Always driven with a constant output of 000b.
m_axi_mm2s_arcache(3:0)	M_AXI_MM2S	O	0011b	Read Address Channel Cache. This is always driven with a constant output of 0011b.
m_axi_mm2s_arvalid	M_AXI_MM2S	O	0	Read Address Channel Read Address Valid. Indicates m_axi_mm2s_araddr is valid. • 1 = Read address is valid. • 0 = Read address is not valid.
m_axi_mm2s_arready	M_AXI_MM2S	I		Read Address Channel Read Address Ready. Indicates target is ready to accept the read address. • 1 = Target read to accept address. • 0 = Target not ready to accept address.
m_axi_mm2s_rdata (C_M_AXI_MM2S_DATA_ WIDTH-1: 0)	M_AXI_MM2S	I		Read Data Channel Read Data.
m_axi_mm2s_rresp(1:0)	M_AXI_MM2S	I		Read Data Channel Response. Indicates results of the read transfer. • 00b = OKAY - Normal access has been successful. • 01b = EXOKAY - Not supported. • 10b = SLVERR - Slave returned error on transfer. • 11b = DECERR - Decode error, transfer targeted unmapped address.
m_axi_mm2s_rlast	M_AXI_MM2S	I		Read Data Channel Last. Indicates the last data beat of a burst transfer. • 1 = Last data beat. • 0 = Not last data beat.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
m_axi_mm2s_rvalid	M_AXI_MM2S	I		Read Data Channel Data Valid. Indicates m_axi_mm2s_rdata is valid. • 1 = Valid read data. • 0 = Not valid read data.
m_axi_mm2s_rready	M_AXI_MM2S	O	0	Read Data Channel Ready. Indicates the read channel is ready to accept read data. • 1 = Ready. • 0 = Not ready.
MM2S Master Stream Interface Signals				
mm2s_prmry_reset_out_n	M_AXIS_MM2S	O	0	Primary MM2S Reset Out.
m_axis_mm2s_tdata (C_M_AXIS_MM2S_DATA_WIDTH-1: 0)	M_AXIS_MM2S	O	zeros	AXI4-Stream Data Out.
m_axis_mm2s_tstrb (C_M_AXIS_MM2S_DATA_WIDTH/8-1: 0)	M_AXIS_MM2S	O	zeros	AXI4-Stream Write Strobe Out. Indicates valid bytes on stream data. Strobe values are passed from Memory Map side.
m_axis_mm2s_tvalid	M_AXIS_MM2S	O	0	AXI4-Stream Valid Out. Indicates stream data bus, m_axis_mm2s_tdata, is valid • 1 = Write data is valid. • 0 = Write data is not valid.
m_axis_mm2s_tready	M_AXIS_MM2S	I		AXI4-Stream Ready. Indicates to S2MM channel target is ready to receive stream data. • 1 = Ready to receive data. • 0 = Not ready to receive data.
m_axis_mm2s_tlast	M_AXIS_MM2S	O	0	AXI4-Stream Last. Indicates last data beat of stream data. • 1 = Last data beat. • 0 = Not last data beat.
S2MM Memory Map Write Interface Signals				
m_axi_s2mm_awaddr (C_M_AXI_S2MM_ADDR_WIDTH-1: 0)	M_AXI_S2MM	O	zeros	Write Address Channel Address Bus.
m_axi_s2mm_awlen(7: 0)	M_AXI_S2MM	O	zeros	Write Address Channel Burst Length. In data beats - 1.
m_axi_s2mm_awsize(2: 0)	M_AXI_S2MM	O	zeros	Write Address Channel Burst Size. Indicates with of burst transfer. • 000b = 1 byte (8 bit wide burst). • 001b = 2 bytes (16 bit wide burst). • 010b = 4 bytes (32 bit wide burst). • 011b = 8 bytes (64 bit wide burst). • 100b = 16 bytes (128 bit wide burst). • 101b = 32 bytes (256 bit wide burst). • 110b = Not Supported by AXI VDMA. • 111b = Not Supported by AXI VDMA.
m_axi_s2mm_awburst(1:0)	M_AXI_S2MM	O	zeros	Write Address Channel Burst Type. Indicates type burst. • 00b = FIXED - Not supported. • 01b = INCR - Incrementing address. • 10b = WRAP - Not supported. • 11b = Reserved.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
m_axi_s2mm_awprot(2:0)	M_AXI_S2MM	O	000b	Write Address Channel Protection. This is always driven with a constant output of 0000b.
m_axi_s2mm_awcache(3:0)	M_AXI_S2MM	O	0011b	Write Address Channel Cache. This is always driven with a constant output of 0011b.
m_axi_s2mm_awaddr	M_AXI_S2MM	O	0	Write Address Channel Write Address Valid. Indicates if mm2s_axim_awaddr is valid. • 1 = Write Address is valid. • 0 = Write Address is not valid.
m_axi_s2mm_awready	M_AXI_S2MM	I		Write Address Channel Write Address Ready. Indicates target is ready to accept the write address. • 1 = Target read to accept address. • 0 = Target not ready to accept address.
m_axi_s2mm_wdata (C_M_AXI_S2MM_DATA_ WIDTH-1: 0)	M_AXI_S2MM	O	zeros	Write Data Channel Write Data Bus.
m_axi_s2mm_wstrb (C_M_AXI_S2MM_DATA_ WIDTH/8 - 1: 0)	M_AXI_S2MM	O	zeros	Write Data Channel Write Strobe Bus. Indicates which bytes are valid in the write data bus. This value is passed from the stream side strobe bus.
m_axi_s2mm_wlast	M_AXI_S2MM	O	0	Write Data Channel Last. Indicates the last data beat of a burst transfer. • 1 = Last data beat. • 0 = Not last data beat.
m_axi_s2mm_wvalid	M_AXI_S2MM	O	0	Write Data Channel Data Valid. Indicates m_axi_s2mm_wdata is valid. • 1 = Valid write data. • 0 = Not valid write data.
m_axi_s2mm_wready	M_AXI_S2MM	I		Write Data Channel Ready. Indicates the write channel target is ready to accept write data. • 1 = Target is ready • 0 = Target is not ready
m_axi_s2mm_bresp(1:0)	M_AXI_S2MM	I		Write Response Channel Response. Indicates results of the write transfer. • 00b = OKAY - Normal access has been successful. • 01b = EXOKAY - Not supported. • 10b = SLVERR - Slave returned error on transfer. • 11b = DECERR - Decode error, transfer targeted unmapped address.
m_axi_s2mm_bvalid	M_AXI_S2MM	I		Write Response Channel Response Valid. Indicates response, m_axi_s2mm_bresp, is valid. • 1 = Response is valid. • 0 = Response is not valid.
m_axi_s2mm_bready	M_AXI_S2MM	O	0	Write Response Channel Ready. Indicates MM2S write channel is ready to receive response. • 1 = Ready to receive response. • 0 = Not ready to receive response.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
S2MM Slave Stream Interface Signals				
s2mm_prmry_reset_out_n	M_AXIS_S2MM	O	0	Primary S2MM Reset Out.
s_axis_s2mm_tdata (C_S_AXIS_S2MM_DATA_ WIDTH-1: 0)	S_AXIS_S2MM	I		AXI4-Stream Data In.
s_axis_s2mm_tstrb (C_S_AXIS_S2MM_DATA_ WIDTH/8-1: 0)	S_AXIS_S2MM	I		AXI4-Stream Write Strobe In. Indicates valid bytes on stream data. Strobe values passed to Memory Map side.
s_axis_s2mm_tvalid	S_AXIS_S2MM	I		AXI4-Stream Valid In. Indicates stream data bus, s_axis_s2mm_tdata, is valid. • 1 = Write data is valid. • 0 = Write data is not valid.
s_axis_s2mm_tready	S_AXIS_S2MM	O	0	AXI4-Stream Ready. Indicates MM2S channel stream interface ready to receive stream data. • 1 = Ready to receive data. • 0 = Not ready to receive data.
s_axis_s2mm_tlast	S_AXIS_S2MM	I		AXI4-Stream Last. Indicates last data beat of stream data. • 1 = Last data beat. • 0 = Not last data beat.
Scatter Gather Memory Map Read Interface Signals				
m_axi_sg_araddr (C_M_AXI_SG_ADDR_ WIDTH-1: 0)	M_AXI_SG	O	zeros	Scatter Gather Read Address Channel Address Bus.
m_axi_sg_arlen(7: 0)	M_AXI_SG	O	zeros	Scatter Gather Read Address Channel Burst Length. Length in data beats - 1.
m_axi_sg_arsize(2: 0)	M_AXI_SG	O	zeros	Scatter Gather Read Address Channel Burst Size. Indicates with of burst transfer. • 000b = Not Supported by AXI VDMA SG Engine. • 001b = Not Supported by AXI VDMA SG Engine. • 010b = 4 bytes (32 bit wide burst). • 011b = Not Supported by AXI VDMA SG Engine. • 100b = Not Supported by AXI VDMA SG Engine. • 101b = Not Supported by AXI VDMA SG Engine. • 110b = Not Supported by AXI VDMA SG Engine. • 111b = Not Supported by AXI VDMA SG Engine.
m_axi_sg_arburst(1:0)	M_AXI_SG	O	zeros	Scatter Gather Read Address Channel Burst Type. Indicates type burst. • 00b = FIXED - Not supported. • 01b = INCR - Incrementing address. • 10b = WRAP - Not supported. • 11b = Reserved.
m_axi_sg_arprot(2:0)	M_AXI_SG	O	000b	Scatter Gather Read Address Channel Protection. This is always driven with a constant output of 000b.
m_axi_sg_arcache(3:0)	M_AXI_SG	O	0011b	Scatter Gather Read Address Channel Cache. This is always driven with a constant output of 0011b.

Table 1: **AXI VDMA I/O Signal Description *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
m_axi_sg_arvalid	M_AXI_SG	O	0	Scatter Gather Read Address Channel Read Address Valid. Indicates if m_axi_sg_araddr is valid. • 1 = Read Address is valid. • 0 = Read Address is not valid.
m_axi_sg_arready	M_AXI_SG	I		Scatter Gather Read Address Channel Read Address Ready. Indicates target is ready to accept the read address. • 1 = Target read to accept address. • 0 = Target not ready to accept address.
m_axi_sg_rdata (C_M_AXI_SG_DATA_ WIDTH-1: 0)	M_AXI_SG	I		Scatter Gather Read Data Channel Read Data.
m_axi_sg_rresp(1:0)	M_AXI_SG	I		Scatter Gather Read Data Channel Response. Indicates results of the read transfer. • 00b = OKAY - Normal access has been successful. • 01b = EXOKAY - Not supported. • 10b = SLVERR - Slave returned error on transfer. • 11b = DECERR - Decode error, transfer targeted unmapped address.
m_axi_sg_rlast	M_AXI_SG	I		Scatter Gather Read Data Channel Last. Indicates the last data beat of a burst transfer. • 1 = Last data beat. • 0 = Not last data beat.
m_axi_sg_rdata	M_AXI_SG	I		Scatter Gather Read Data Channel Data Valid. Indicates m_sg_aximry_rdata is valid. • 1 = Valid read data. • 0 = Not valid read data.
m_axi_sg_rready	M_AXI_SG	O	0	Scatter Gather Read Data Channel Ready. Indicates the read channel is ready to accept read data. • 1 = Is ready. • 0 = Is not ready.

Design Parameters

The AXI VDMA Design Parameters are listed and described in Table 2.

In addition to the parameters listed in this table, there are also parameters that are inferred for each AXI interface in the EDK tools. Through the design, these EDK-inferred parameters control the behavior of the AXI Interconnect. For a complete list of the interconnect settings related to the AXI interface, see DS768, *AXI Interconnect IP Data Sheet*.

Table 2: **AXI VDMA I/O Signal Description**

Parameter Name	Allowable Values	Default Values	VHDL Type	Feature/Description
AXI VDMA General Parameters				
C_S_AXI_LITE_DATA_WIDTH	32	32	integer	Data width in bits of AXI4-Lite Interface
C_S_AXI_LITE_ADDR_WIDTH	32	32	integer	Address width in bits of AXI4-Lite Interface
C_DLYTMR_RESOLUTION	1 - 1000000	125	integer	Resolution of the interrupt delay timer in axi_scndry_aclk cycles
C_PRMRY_IS_ACLK_ASYNC	0,1	0	integer	Primary clock is asynchronous.
C_NUM_FSTORES	1 - 16	3	Integer	Value used to specify the number of frame buffers to create and use in external memory. This value also specifies the number of Scatter Gather Descriptors to use in a descriptor chain.
C_AXI_MM2S_ACLK_FREQ_HZ		100000000	Integer	Frequency in hertz of the axi_mm2s_aclk clock input.
C_AXI_S2MM_ACLK_FREQ_HZ		100000000	Integer	Frequency in hertz of the axi_s2mm_aclk clock input.
C_USE_FSYNC	0, 1	0	Integer	Specifies DMA operations synchronized to frame sync input. • 0 = Free Running • 1 = Synchronous
C_FAMILY	virtex6, spartan6	virtex6	String	Specifies the target FPGA family
Scatter Gather Engine Parameters				
C_M_AXI_SG_DATA_WIDTH	32	32	integer	Data width of AXI Scatter Gather Engine
C_M_AXI_SG_ADDR_WIDTH	32	32	integer	Address width of AXI Scatter Gather Engine
Memory Map to Stream Parameters				
C_INCLUDE_MM2S	0,1	1	integer	Include or exclude the Memory Map to Stream channel. When excluded, all unused ports are tied off or driven to zero. This will also exclude MM2S AXI Control Stream. • 0 = Exclude MM2S • 1 = Include MM2S

Table 2: **AXI VDMA I/O Signal Description *(Cont'd)***

Parameter Name	Allowable Values	Default Values	VHDL Type	Feature/Description
C_MM2S_GENLOCK_MODE	0,1	0	Integer	Specifies the Gen-Lock mode for the MM2S Channel. Master mode specifies that the Video DMA will operate as the Gen-Lock Master. Slave mode specifies that the Video DMA will operate as a Gen-lock Slave. • 0 = Master Mode • 1 = Slave Mode
C_MM2S_GENLOCK_NUM_ MASTERS	1-16	1	Integer	Specifies the number of Masters to which the Slave can synchronize. A register is used to dynamically specify which master is in control at any given time. ***Note:*** This parameter is only valid in Gen-Lock Slave mode (C_MM2S_GENLOCK_MODE = 1)
C_INCLUDE_MM2S_DRE	0,1	0	integer	Include or exclude the Memory Map to Stream channel Data Realignment Engine. • 0 = Exclude DRE • 1 = Include DRE ***Note:*** DRE support not available for AXI4-Stream data widths of 128 bits and 256 bits.
C_MM2S_LINEBUFFER_DEPTH	0 - 65536	128	Integer	Depth of Line Buffer FIFO. Depth specified in bytes. Depth parameter must be a power of 2 value, that is, 1,2,4,8,16,...1024, 2048, etc. ***Note:*** A value of zero will exclude the line buffer. ***Note:*** Valid minimum depth, excluding 0, equals C_M_AXIS_MM2S_DATA_WIDTH/8.
C_MM2S_LINEBUFFER_THRESH	1 - 65536	1	Integer	Almost Empty Threshold. Threshold point at which MM2S line buffer almost empty flag asserts high. Threshold specified in bytes and must be C_M_AXIS_MM2S_DATA_WIDTH/8 resolution. That is, for a Data Width of 32-bits valid threshold settings would be 4, 8, 12, 16, 20, etc. ***Note:*** Valid minimum threshold value equals C_M_AXIS_MM2S_DATA_WIDTH/8. Maximum threshold value limited by C_MM2S_LINEBUFFER_DEPTH ***Note:*** Value valid when MM2S line buffer is included (C_MM2S_LINEBUFFER_DEPTH > 0)
C_M_AXI_MM2S_ADDR_WIDTH	32	32	integer	Address width of AXI Memory Map on the Memory Map to Stream interface
C_M_AXI_MM2S_DATA_WIDTH	32,64,128, 256	32	integer	Data width of AXI Memory Map on the Memory Map to Stream Interface

Table 2: **AXI VDMA I/O Signal Description *(Cont'd)***

Parameter Name	Allowable Values	Default Values	VHDL Type	Feature/Description
C_M_AXIS_MM2S_DATA_WIDTH	8, 16, 32,64,128, 256	32	integer	Data width of AXI4-Stream on the Stream to Memory Map Interface. Width must be less than or equal to C_M_AXI_MM2S_DATA_WIDTH.
C_MM2S_MAX_BURST_LENGTH	16,32,64,128,256	16	integer	Maximum burst length in data beats per burst request on Memory Map Read interface
Stream to Memory Map Parameters				
C_INCLUDE_S2MM	0,1	1	integer	Include or exclude the Stream to Memory Map. When excluded, all unused ports are tied off or driven to zero. This will also exclude S2MM AXI Status Stream. • 0 = Exclude S2MM • 1 = Include S2MM
C_S2MM_GENLOCK_MODE	0,1	0	Integer	Specifies the Gen-Lock mode for the S2MM Channel. Master mode specifies that the Video DMA will operate as the Gen-Lock Master. Slave mode specifies that the Video DMA will operate as a Gen-lock Slave. • 0 = Master Mode • 1 = Slave Mode
C_S2MM_GENLOCK_NUM_MASTERS	1 - 16	1	Integer	Specifies the number of Masters to which the Slave can synchronize. A register is used to dynamically specify which master is in control at any given time. ***Note:*** This parameter is only valid in Gen-Lock Slave mode (C_S2MM_GENLOCK_MODE = 1)
C_INCLUDE_S2MM_DRE	0,1	0	integer	Include or exclude the Stream to Memory Map channel Data Realignment Engine. • 0 = Exclude DRE • 1 = Include DRE ***Note:*** DRE support not available for AXI4-Stream data widths of 128 bits and 256 bits.
C_S2MM_LINEBUFFER_DEPTH	0 - 65536	128	Integer	Depth of Line Buffer FIFO. Depth specified in bytes. Depth parameter must be a power of 2 value, that is, 2,4,8,16,...1024, 2048, etc. ***Note:*** A value of zero will exclude the line buffer. ***Note:*** Valid minimum depth, excluding 0, equals C_S_AXIS_S2MM_DATA_WIDTH/8.

Table 2: **AXI VDMA I/O Signal Description *(Cont'd)***

Parameter Name	Allowable Values	Default Values	VHDL Type	Feature/Description
C_S2MM_LINEBUFFER_THRESH	1 - 65536	64	Integer	Almost Full Threshold. Threshold point at which S2MM line buffer almost full flag asserts high. Threshold specified in bytes and must be C_S_AXIS_S2MM_DATA_WIDTH/8 resolution. That is, for a Data Width of 32-bits valid threshold settings would be 4, 8, 12, 16, 20, etc. ***Note:*** Valid minimum threshold value equals C_S_AXIS_S2MM_DATA_WIDTH/8. Maximum threshold value limited by C_S2MM_LINEBUFFER_DEPTH ***Note:*** Value valid when S2MM line buffer is included (C_S2MM_LINEBUFFER_DEPTH > 0)
C_M_AXI_S2MM_ADDR_WIDTH	32	32	integer	Address width of AXI Memory Map on the Stream to Memory Map interface
C_M_AXI_S2MM_DATA_WIDTH	32,64,128, 256	32	integer	Data width of AXI Memory Map on the Stream to Memory Map Interface
C_S_AXIS_S2MM_DATA_WIDTH	8, 16, 32,64,128, 256	32	integer	Data width of AXI4-Stream on the Stream to Memory Map Interface. Width must be less than or equal to C_M_AXI_S2MM_DATA_WIDTH.
C_S2MM_MAX_BURST_LENGTH	16,32,64,128,256	16	integer	Maximum burst length in data beats per burst request on Memory Map Write interface

Allowable Parameter Combinations

Table 3: **Allowable Parameter Combination**

Parameter Name	Affects Parameter	Relationship Description
C_INCLUDE_MM2S	C_INCLUDE_MM2S_DRE C_MM2S_BURST_SIZE	Affected Parameters are ignored when C_INCLUDE_MM2S = 0
C_INLCLUDE_S2MM	C_INCLUDE_S2MM_DRE C_S2MM_BURST_SIZE	Affected Parameters are ignored when C_INCLUDE_S2MM = 0
C_MM2S_GENLOCK_MODE	C_MM2S_GENLOCK_NUM_ MASTERS	Affected Parameter is ignored when C_MM2S_GENLOCK_MODE = 0
C_S2MM_GENLOCK_MODE	C_S2MM_GENLOCK_NUM_ MASTERS	Affected Parameter must be less than or equal to C_S2MM_GENLOCK_MODE
C_M_AXI_MM2S_DATA_WIDTH	C_M_AXIS_MM2S_DATA_WIDTH	Affected Parameter must be less than or equal to C_M_AXI_MM2S_DATA_WIDTH
C_M_AXI_S2MM_DATA_WIDTH	C_S_AXIS_S2MM_DATA_WIDTH	Affected Parameter must be less than or equal to C_M_AXI_S2MM_DATA_WIDTH
C_MM2S_LINEBUFFER_DEPTH and C_M_AXIS_MM2S_DATA_WIDTH	C_MM2S_LINEBUFFER_THRESH	Affected Parameter is ignored when C_MM2S_LINEBUFFER_DEPTH = 0 Valid minimum threshold value equals C_M_AXIS_MM2S_DATA_WIDTH/8. Maximum threshold value limited by C_MM2S_LINEBUFFER_DEPTH
C_S2MM_LINEBUFFER_DEPTH and C_S_AXIS_S2MM_DATA_WIDTH	C_S2MM_LINEBUFFER_THRESH	Affected Parameter is ignored when C_S2MM_LINEBUFFER_DEPTH = 0 Valid minimum threshold value equals C_S_AXIS_S2MM_DATA_WIDTH/8 Maximum threshold value limited by C_S2MM_LINEBUFFER_DEPTH

Parameter - I/O Signal Dependencies

Table 4: **Parameter - I/O Signal Dependencies**

Parameter Name	Affects Signal	Depends on Parameter	Relationship Description
C_M_AXI_SG_DATA_WIDTH	m_axi_sg_rdata		The setting of the parameter sets the vector width of the port.
C_M_AXI_SG_ADDR_WIDTH	m_axi_sg_araddr		The setting of the parameter sets the vector width of the port.
C_USE_FSYNC	mm2s_fsync, s2mm_fsync		If the parameter is assigned a value of zero, the input ports are left open.
C_MM2S_GENLOCK_NUM_MASTERS	mm2s_frame_ptr_in		The setting of the parameter sets the vector width of the port.
C_S2MM_GENLOCK_NUM_MASTERS	s2mm_frame_ptr_in		The setting of the parameter sets the vector width of the port.
C_MM2S_GENLOCK_MODE	mm2s_frame_ptr_in, mm2s_frame_ptr_out		If the parameter is assigned a value of zero, the output ports are tied to 0, and the input ports are left open.
C_S2MM_GENLOCK_MODE	s2mm_frame_ptr_in, s2mm_frame_ptr_out		If the parameter is assigned a value of zero, the output ports are tied to 0, and the input ports are left open.
C_INCLUDE_MM2S	m_axis_mm2s_araddr, m_axi_mm2s_arlen, m_axi_mm2s_arsize, m_axi_mm2s_arburst, m_axi_mm2s_arprot, m_axi_mm2s_arcache, m_axi_mm2s_arvalid, m_axi_mm2s_arready, m_axi_mm2s_rdata, m_axi_mm2s_rresp, m_axi_mm2s_rlast, m_axi_mm2s_rvalid, m_axi_mm2s_rready, mm2s_prmry_reset_out_n, m_axis_mm2s_tdata, m_axis_mm2s_tstrb, m_axis_mm2s_tvalid, m_axis_mm2s_tready, m_axis_mm2s_tlast, mm2s_frame_ptr_in, mm2s_frame_ptr_out, mm2s_fsync		If the parameter is assigned a value of zero, the output ports are tied to 0, and the input ports are left open.
C_M_AXI_MM2S_ADDR_WIDTH	m_axi_mm2s_araddr		The setting of the parameter sets the vector width of the port.
C_M_AXI_MM2S_DATA_WIDTH	m_axi_mm2s_rdata		The setting of the parameter sets the vector width of the port.
C_M_AXIS_MM2S_DATA_WIDTH	m_axis_mm2s_tdata, m_axis_mm2s_tstrb		The setting of the parameter sets the vector width of the port.

Table 4: **Parameter - I/O Signal Dependencies *(Cont'd)***

Parameter Name	Affects Signal	Depends on Parameter	Relationship Description
C_INCLUDE_S2MM	m_axis_s2mm_awaddr, m_axi_s2mm_awlen, m_axi_s2mm_awsize, m_axi_s2mm_awburst, m_axi_s2mm_awprot, m_axi_mm2s_awcache, m_axi_s2mm_awvalid, m_axi_s2mm_awready, m_axi_s2mm_wdata, m_axi_s2mm_wstrb, m_axi_s2mm_wlast, m_axi_s2mm_wvalid, m_axi_s2mm_wready, m_axi_s2mm_bresp, m_axi_s2mm_bvalid, m_axi_s2mm_bready, s2mm_prmry_reset_out_n, s_axis_s2mm_tdata, s_axis_s2mm_tstrb, s_axis_s2mm_tvalid, s_axis_s2mm_tready, s_axis_s2mm_tlast, s2mm_frame_ptr_in, s2mm_frame_ptr_out, s2mm_fsync		If the parameter is assigned a value of zero, the output ports are tied to 0 and the input ports are left open.
C_M_AXI_S2MM_ADDR_WIDTH	m_axis_s2mm_awaddr		The setting of the parameter sets the vector width of the port.
C_M_AXI_S2MM_DATA_WIDTH	m_axi_s2mm_wdata, m_axi_s2mm_wstrb		The setting of the parameter sets the vector width of the port.
C_S_AXIS_S2MM_DATA_WIDTH	s_axis_s2mm_tdata, s_axis_s2mm_tstrb		The setting of the parameter sets the vector width of the port.

Parameter Descriptions

C_PRMRY_IS_ACLK_ASYNC

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 0)
- **Definition:** 0 = `axi_prmry_aclk` is synchronous to `axi_scndry_aclk`; 1 = `axi_prmry_aclk` is asynchronous to `axi_scndry_aclk`
- **Description:** Provides ability to operate the primary data path asynchronously to the AXI4-Lite and Scatter/Gather Engine. This is used for applications where there is a requirement to operate the primary data path at high frequencies, but this same high frequency requirement is not required for reading and writing control registers or for fetching and updating descriptors. In some cases, this allows for easier placement and timing closure at system build time.

C_AXI_MM2S_ACLK_FREQ_HZ

- **Type:** Integer
- **Allowed Values:** All integer values
- **Definition:** Frequency in hertz of the `axi_mm2s_aclk` clock input.
- **Description:** This integer parameter is used by AXI VDMA to correctly configure clock domain crossing logic. This parameter is only used when C_PRMRY_IS_ACLK_ASYNC = 1. The EDK tool suite will assign this parameter based on the clock frequency of the `axi_mm2s_aclk` source.

C_AXI_S2MM_ACLK_FREQ_HZ

- **Type:** Integer
- **Allowed Values:** All integer values
- **Definition:** Frequency in hertz of the `axi_s2mm_aclk` clock input.
- **Description:** This integer parameter is used by AXI VDMA to correctly configure clock domain crossing logic. This parameter is only used when C_PRMRY_IS_ACLK_ASYNC = 1. The EDK tool suite will assign this parameter based on the clock frequency of the `axi_s2mm_aclk` source.

C_DLYTMR_RESOLUTION

- **Type:** Integer
- **Allowed Values:** 1 to 100,000 (default = 256)
- **Definition:** Interrupt Delay Timer Resolution in AXI Secondary Clock cycles
- **Description:** This integer parameter is used to set the resolution of the Interrupt Delay Timer. Values specify the number of `axi_scndry_aclk` clock cycles between each tick of the delay timer.

C_NUM_FSTORES

- **Type:** Integer
- **Allowed Values:** 0 to 16 (default = 3)
- **Definition:** Number of frame stores
- **Description:** This integer parameter is used to define the number of frame storage locations to be processed by the AXI VDMA. This parameter also defines the number Scatter Gather descriptors per channel in the descriptor chain required to initialize the AXI VDMA.

C_USE_FSYNC

- **Type:** Integer
- **Allowed Values:** 0, 1 (default = 0)
- **Definition:** 0 = Free run mode, 1 = Frame sync mode
- **Description:** This integer parameter is used to set the synchronization mode of AXI VDMA. When in free mode the AXI VDMA will transfer data as quickly as it is able to. When in frame sync mode, the AXI VDMA will transfer data starting with the falling edge of each `mm2s_fsync` or `s2mm_fsync` for the associated channel.

C_S_AXI_LITE_ADDR_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Address bus width of attached AXI on the AXI4-Lite interface
- **Description:** This integer parameter is used by the AXI4-Lite interface to size the AXI read and write address bus related components within the Lite interface. The EDK tool suite will assign this parameter a fixed value of 32.

C_S_AXI_LITE_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI4-Lite interface
- **Description:** This integer parameter is used by the AXI4-Lite interface to size the AXI read and write data bus related components within the Lite interface. The EDK tool suite will assign this parameter a fixed value of 32.

C_M_AXI_SG_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI Scatter/Gather interface
- **Description:** This integer parameter is used by the AXI Scatter/Gather interface to size the AXI read data bus related components within the Scatter/Gather Engine. The EDK tool suite will assign this parameter a fixed value of 32.

C_M_AXI_SG_ADDR_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Address bus width of attached AXI on the AXI Scatter Gather interface
- **Description:** This integer parameter is used by the AXI Scatter Gather interface to size the AXI read address bus related components within the Scatter Gather Engine. The EDK tool suite will assign this parameter a fixed value of 32.

C_INCLUDE_MM2S

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Exclude MM2S Channel; 1 = Include MM2S Channel
- **Description:** Include or exclude MM2S Channel. Setting this parameter to 0 will cause all output ports for the MM2S channel to be tied to zero, and all of the input ports for the respective channel to be left open.

  Note: Setting both C_INCLUDE_MM2S = 0 and C_INCLUDE_S2MM = 0 will disable all logic within the AXI VDMA and is not a valid configuration.

C_INCLUDE_S2MM

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Exclude S2MM Channel; 1 = Include S2MM Channel
- **Description:** Include or exclude S2MM Channel. Setting this parameter to 0 will cause all output ports for the S2MM channel to be tied to zero, and all of the input ports for the respective channel to be left open.

  Note: Setting both C_INCLUDE_MM2S = 0 and C_INCLUDE_S2MM = 0 will disable all logic within the AXI VDMA and is not a valid configuration.

C_INCLUDE_MM2S_DRE

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Exclude MM2S Data Realignment Engine; 1 = Include MM2S Data Realignment Engine
- **Description:** Include or exclude MM2S Data Realignment Engine. For use cases where all transfers will be C_M_AXIS_MM2S_DATA_WIDTH aligned, this parameter can be set to 0 to exclude DRE-saving FPGA resources. Setting this parameter to 1 allows data realignment to the byte (8 bits) level on the primary memory map data paths. For the MM2S channel, vertical size (vsize) number of video lines each horizontal size (hsize) bytes long and spaced stride bytes apart (stride is number of bytes between first pixel of each line) are read from memory.

  For the case where C_INCLUDE_MM2S_DRE = 1, data reads can start from any Start Address byte offset, be of any horizontal size and stride value and the read data will be aligned such that the first byte read will be the first valid byte out on the AXI4-Stream.

  For the case where C_IN CLUDE_MM2S_DRE = 0, then the Start Address must be aligned to multiples of C_M_AXIS_MM2S_DATA_WIDTH bytes. Also Horizontal Size and Stride must be specified in even multiples of C_M_AXIS_MM2S_DATA_WIDTH bytes. For example, if C_M_AXIS_MM2S_DATA_WIDTH = 32, data is aligned if the Start Address at word offsets (32-bit offset), that is 0x0, 0x4, 0x8, 0xC, etc., Horizontal Size is 0x4, 0x8, 0xC etc. and Stride is 0x4, 0x8, 0xC, etc. If C_M_AXIS_MM2S_DATA_WIDTH = 64, data is aligned if the Start Address is at double-word offsets (64-bit offsets), that is 0x0, 0x8, 0x10, 0x18, etc., and Horizontal Size, and Stride are at 0x4, 0x8, 0xC, etc.

  Note: If DRE is disabled (C_INCLUDE_MM2S_DRE = 0), unaligned start addresses, hsizes, or strides, are not supported. Having an unaligned Start Address, HSize, and/or Stride will result in undefined behavior.

  Note: DRE Support is only available for AXI4-Stream data width setting of 64-bits and less.

C_INCLUDE_S2MM_DRE

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Exclude S2MM Data Realignment Engine, 1 = Include S2MM Data Realignment Engine
- **Description:** Include or exclude S2MM Data Realignment Engine. For use cases where all transfers will be C_S_AXIS_S2MM_DATA_WIDTH aligned, this parameter can be set to 0 to exclude DRE-saving FPGA resources. Setting this parameter to 1 allows data realignment to the byte (8 bits) level on the primary memory map data paths. For the S2MM channel, vertical size (vsize) number of video lines each horizontal size (hsize) bytes long and spaced stride bytes apart (stride is number of bytes between first pixel of each line) are written to memory.

  For the case where C_INCLUDE_S2MM_DRE = 1, data writes can start from any Start Address byte offset, be of any horizontal size and stride value and the write data will be aligned such that first valid byte in on the AXI4-Stream will be the byte written to the memory location specified by the Start Address, Hsize, and Stride.

  For the case where C_IN CLUDE_S2MM_DRE = 0, then the Start Address must be aligned to multiples of C_S_AXIS_S2MM_DATA_WIDTH bytes. Also Horizontal Size and Stride must be specified in even multiples of C_S_AXIS_S2MM_DATA_WIDTH bytes. For example, if C_S_AXIS_S2MM_DATA_WIDTH = 32, data is aligned if the Start Address at word offsets (32-bit offset), that is 0x0, 0x4, 0x8, 0xC, etc., Horizontal Size is 0x4, 0x8, 0xC etc. and Stride is 0x4, 0x8, 0xC, etc. If C_S_AXIS_S2MM_DATA_WIDTH = 64, data is aligned if the Start Address is at double-word offsets (64-bit offsets), that is 0x0, 0x8, 0x10, 0x18, etc., and Horizontal Size, and Stride are at 0x4, 0x8, 0xC, etc.

  Note: If DRE is disabled (C_INCLUDE_S2MM_DRE = 0), unaligned start addresses, hsizes, or strides, are not support. Having an unaligned Start Address, HSize, and/or Stride will result in undefined behavior. DRE Support is only available for AXI4-Stream data width setting of 64-bits and under.

C_S2MM_GENLOCK_MODE

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Gen-Lock Master Mode, 1 = Gen-Lock Slave Mode
- **Description**: This integer values sets the S2MM Channel Gen-Lock synchronization mode. Master mode specifies that the S2MM VDMA channel will operate at the Gen-Lock Master. In Master mode frames are not dropped or repeated. The current master frame being worked on by the S2MM channel is specified on s2mm_frm_ptr_out. Slave mode specifies that the S2MM VDMA channel will operate as a Gen-Lock Slave. Gen-Lock slaves automatically drop and repeat frames based on the master and slave frame rates. The Gen-Lock slave will look at the vector slice of `s2mm_frm_ptr_in` as specified in S2MM DMACR Write Pointer Number field (DMACR.WrPntrNmbr bits 11 downto 8) to determine which frame the master is working on and will operate Frame Delay behind the master.

C_MM2S_GENLOCK_MODE

- **Type:** Integer
- **Allowed Values:** 0,1 (default = 1)
- **Definition:** 0 = Gen-Lock Master Mode, 1 = Gen-Lock Slave Mode
- **Description**: This integer parameter sets the MM2S Channel Gen-Lock synchronization mode. Master mode specifies that the MM2S VDMA channel will operate at the Gen-Lock Master. In Master mode frames are not dropped or repeated. The current master frame being worked on by the S2MM channel is specified on `s2mm_frm_ptr_out`. Slave mode specifies that the MM2S VDMA channel will operate as a Gen-Lock Slave. Gen-Lock slaves automatically drop and repeat frames based on the master and slave frame rates. The Gen-Lock slave will look at the vector slice of `s2mm_frm_ptr_in` as specified in MM2S DMACR Read Pointer Number field (DMACR.RdPntrNmbr bits 11 downto 8) to determine which frame the master is working on and will operate Frame Delay behind the master.

C_MM2S_GENLOCK_NUM_MASTERS

- **Type:** Integer
- **Allowed Values:** 1 to 16(default = 1)
- **Definition:** Number of masters to which the slave will synchronize operations.
- **Description**: This integer parameter specifies to the Gen-Lock slave the total number of masters to synchronize operations to. This parameter also specifies the vector width of the `mm2s_frm_ptr_in` port, where each master requires 5 bits on the `mm2s_frm_ptr_in` vector. Therefore the width of the `mm2s_frm_ptr_in` port is 5 * C_MM2S_GENLOCK_NUM_MASTERS.

  Note: This parameter is only valid for Gen-Lock Slave mode (C_MM2S_GENLOCK_MODE = 1) and is ignored in Gen-Lock Master mode.

C_S2MM_GENLOCK_NUM_MASTERS

- **Type:** Integer
- **Allowed Values:** 1 to 16 (default = 1)
- **Definition:** Number of masters to which the slave will synchronize operations.
- **Description**: This integer parameter specifies to the Gen-Lock slave the total number of masters to synchronize operations to. This parameter also specifies the vector width of the `s2mm_frm_ptr_in` port, where each master requires 5 bits on the mm2s_frm_ptr_in vector. Therefore the width of the `s2mm_frm_ptr_in` port is 5 * C_S2MM_GENLOCK_NUM_MASTERS.

  Note: This parameter is only valid for Gen-Lock Slave mode (C_S2MM_GENLOCK_MODE = 1) and is ignored in Gen-Lock Master mode.

C_S_AXI_MM2S_ADDR_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Address bus width of attached AXI on the AXI MM2S Memory Map Read interface
- **Description:** This integer parameter is used by the MM2S interface to size the AXI read address bus-related components within the MM2S Channel. The EDK tool suite assigns this parameter a fixed value of 32.

C_M_AXI_MM2S_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 32, 64, 128, 256 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI MM2S Memory Map Read interface
- **Description:** This integer parameter is used by the MM2S interface to size the AXI read data bus related components within the MM2S Channel. The EDK tools ensure correct sizing of the AXI data width based on EDK system configuration.

C_M_AXIS_MM2S_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 8, 16, 32, 64, 128, 256 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI MM2S Master Stream interface
- **Description:** This integer parameter is used by the MM2S interface to size the AXI Master Stream data bus-related components within the MM2S Channel.

  Note: This parameter must be set less than or equal to C_M_AXI_MM2S_DATA_WIDTH.

C_M_AXI_S2MM_ADDR_WIDTH

- **Type:** Integer
- **Allowed Values:** 32 (default = 32)
- **Definition:** Address bus width of attached AXI on the AXI S2MM Memory Map Write interface
- **Description:** This integer parameter is used by the S2MM interface to size the AXI write address bus-related components within the S2MM Channel. The EDK tool suite assigns this parameter a fixed value of 32.

C_M_AXI_S2MM_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 32, 64, 128, 256 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI S2MM Memory Map Write interface
- **Description:** This integer parameter is used by the S2MM interface to size the AXI write data bus-related components within the S2MM Channel. The EDK tools ensure correct sizing of the AXI data width based on EDK system configuration.

C_S_AXIS_S2MM_DATA_WIDTH

- **Type:** Integer
- **Allowed Values:** 8, 16, 32, 64, 128, 256 (default = 32)
- **Definition:** Data bus width of attached AXI on the AXI S2MM Slave Stream interface
- **Description:** This integer parameter is used by the S2MM interface to size the AXI Slave Stream data bus related components within the S2MM Channel.

  Note: This parameter must be set less than or equal to C_M_AXI_S2MM_DATA_WIDTH.

C_MM2S_MAX_BURST_LENGTH

- **Type:** Integer
- **Allowed Values:** 16, 32, 64, 128, 256 (default = 16)
- **Definition:** MM2S maximum burst length in data beats
- **Description:** Maximum burst length of the MM2S memory map interface. This parameter sets the granularity of burst partitioning. For example, if the burst length is set to 16, the maximum burst on the memory map interface will be 16 data beats. Smaller values reduce throughput but result in less impact on the AXI infrastructure. Larger values increase throughput but result in a greater impact on the AXI infrastructure.

C_S2MM_MAX_BURST_LENGTH

- **Type:** Integer
- **Allowed Values:** 16, 32, 64, 128, 256 (default = 16)
- **Definition:** S2MM maximum burst length in data beats
- **Description:** Maximum burst length of the S2MM memory map interface. This parameter sets the granularity of burst partitioning. For example, if the burst length is set to 16, the maximum burst on the memory map interface will be 16 data beats. Smaller values reduce throughput but result in less impact on the AXI infrastructure. Larger values increase throughput but result in a greater impact on the AXI infrastructure.

Register Space

The AXI VDMA core register space is shown in Table 5. The AXI VDMA Registers are memory-mapped into non-cacheable memory space. This memory space must be aligned on a AXI word (32-bit) boundary.

AXI VDMA Register Address Mapping

Table 5: **Allowable Parameter Combination**

Name	Description	Address Space Offset(1)
MM2S_DMACR	MM2S DMA Control Register	00h
MM2S_DMASR	MM2S DMA Status Register	04h
MM2S_CURDESC	MM2S Current Descriptor Pointer	08h
Reserved	N/A	0Ch
MM2S_TAILDESC	MM2S Tail Descriptor Pointer	10h
Reserved	N/A	14h to 24h
PARK_PTR_REG	MM2S and S2MM Park Pointer Register	28h
VDMA_VERSION	Video DMA Version Register	2Ch
S2MM_DMACR	S2MM DMA Control Register	30h
S2MM_DMASR	S2MM DMA Status Register	34h
S2MM_CURDESC	S2MM Current Descriptor Pointer	38h
Reserved	N/A	3Ch
S2MM_TAILDESC	S2MM Tail Descriptor Pointer	40h

1. Address Space Offset is relative to C_BASEADDR assignment. C_BASEADDR is defined in AXI VDMA mpd file and set by XPS.

Endianess

All registers are in Little Endian format, as shown in Figure 3.

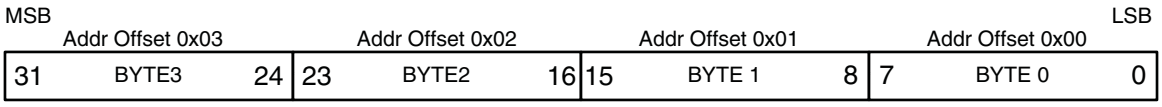

Figure 3: **32-bit Little Endian Example**

Memory Map to Stream Register Detail

MM2S_DMACR (MM2S DMA Control Register - Offset 00h)

This register provides control for the Memory Map to Stream DMA Channel.

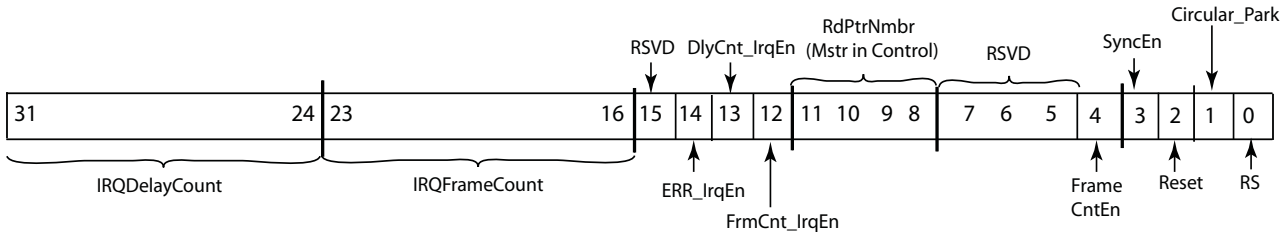

Figure 4: **MM2S DMACR Register**

Table 6: **MM2S_DMACR Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 24	IRQDelayCount	00h	R/W	This value is used for setting the interrupt delay count value. The delay count interrupt is a mechanism for causing the DMA engine to generate an interrupt after the delay period has expired. This is used for cases when the interrupt frame count is not met after a period of time and the CPU desires an interrupt to be generated. Timer begins counting at the start of a frame and will reset with the transfer of a new packet on MM2S stream or when delay count period has expired. When a value different than the current IRQDelayCount is written to this field, the internal delay counter will be reset to zero. ***Note:*** Setting this value to zero will disable the delay counter interrupt.
23 to 16	IRQFrameCount	01h	R/W	This value is used for setting the interrupt threshold. When frame transfer interrupt events occur an internal counter counts down from the Interrupt Frame Count setting. When the count reaches zero an interrupt out is generated by the VDMA engine. When a value different than the current IRQFrameCount is written to this field, the internal frame counter will be reset to the new value. ***Note:*** The minimum setting for the count is 0x01. A write of 0x00 to this register will set the count to 0x01. ***Note:*** When DMACR.FrameCntEn = 1, this value will determines the number of frame buffers to process.
15	Reserved	0	RO	Writing to this bit has no effect and it will always read as zeros.
14	Err_IrqEn	0	R/W	Interrupt on Error Interrupt Enable. When set to 1, will allow DMASR.Err_Irq to generate an interrupt out. • 0 = Error Interrupt disabled • 1 = Error Interrupt enabled
13	DlyCnt_IrqEn	0	R/W	Interrupt on Delay Count Interrupt Enable. When set to 1, will allow DMASR.DlyCnt_Irq to generate an interrupt out. • 0 = Delay Count Interrupt disabled • 1 = Delay Count Interrupt enabled
12	FrmCnt_IrqEn	0	R/W	Frame Count Complete Interrupt Enable. When set to 1, will allow DMASR.FrmCnt_Irq to generate an interrupt out when IRQFrameCount value reaches zero. • 0 = Frame Count Interrupt disabled • 1 = Frame Count Interrupt enabled
11 to 8	RdPntrNum	zeros	RW	Indicates the master in control when MM2S channel is configured for Gen-Lock slave mode (C_MM2S_GENLOCK_MODE = 1). 0000b = Controlling master is Master 1 0001b = Controller master is Master 2 0010b = Controller master is Master 3 etc. ***Note:*** Maximum valid RdPntrNum is C_MM2S_GENLOCK_NUM_MASTER - 1. Setting to a value greater than C_MM2S_GENLOCK_NUM_MASTER - 1 will have undefined results.
7 to 5	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
4	FrameCntEn	0	RW	Configures MM2S channel to only allow IRQFrameCount number of transfers to occur. After IRQFrameCount frames have been transferred the MM2S channel will halt, DMACR.RS bit will be cleared to 0 and DMASR.Halted will assert to 1 once the channel has completely halted.

Table 6: **MM2S_DMACR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
3	SyncEn	0	RW	Enables Gen-Lock synchronization. • 0 = Gen-Lock synchronization disabled. Gen-Lock input ignored by MM2S • 1 = Gen-Lock synchronization enabled. MM2S synchronized to Gen-Lock frame input. ***Note:*** This value is only valid when channel configured as Gen-Lock Slave (C_MM2S_GENLOCK_MODE = 1). If configured for Gen-Lock Master mode (C_MM2S_GENLOCK_MODE = 0) then this bit is reserved and always reads as zero.
2	Reset	0	RW	Soft reset for resetting the AXI VDMA MM2S channel. Setting this bit to a 1 causes the AXI VDMA MM2S channel to be reset. Reset will be accomplished gracefully. Pending commands/transfers will be flushed or completed. AXI4-Stream reset output will be asserted. Setting DMACR.Reset = 1 will only reset the MM2S channel. After completion of a soft reset all MM2S registers and bits will be in the Reset State • 0 = Reset NOT in progress - Normal operation • 1 = Reset in progress
1	Circular_Park	1	RO	Indicates frame buffer Circular mode or frame buffer Park mode. 0 = Park Mode- Engine will park on frame buffer referenced by PARK_PTR_REG.RdFrmPntrRef. 1 = Circular Mode - Engine will continuously circle through C_NUM_FSTORES frame buffers. ***Note:*** Park Mode must only be enabled when AXI VDMA is Idle (DMASR.Idle = 1) and NOT halted (DMASR.Halted = 0). Undefined results will occur if enabled at any other time.
0	RS	0	R/W	Run / Stop controls running and stopping of the VDMA channel. For any DMA operations to commence the AXI VDMA engine must be running (DMACR.RS=1) • 0 = Stop - VDMA stops when current (if any) DMA operations are complete. Queued descriptors for the associated channel will be flushed from the SG Engine. The halted bit in the DMA Status Register will assert to 1 when the DMA engine is halted. This bit will get cleared by AXI VDMA hardware when an error occurs. The CPU may also choose to clear this bit to stop DMA operations. • 1 = Run - Start DMA operations when an error occurs or when the IRQFrameCount is reached when Frame Count Enable is asserted (DMACR.FrameCntEn = 1). The halted bit in the DMA Status Register will deassert to 0 when the DMA engine begins operations. ***Note:*** On Run/Stop clear, in progress stream transfers may terminate early.
RO = Read Only. Writing has no effect. R/W = Read / Write.				

MM2S_DMASR (MM2S DMA Status Register- Offset 04h)

This register provides status for the Memory Map to Stream DMA Channel.

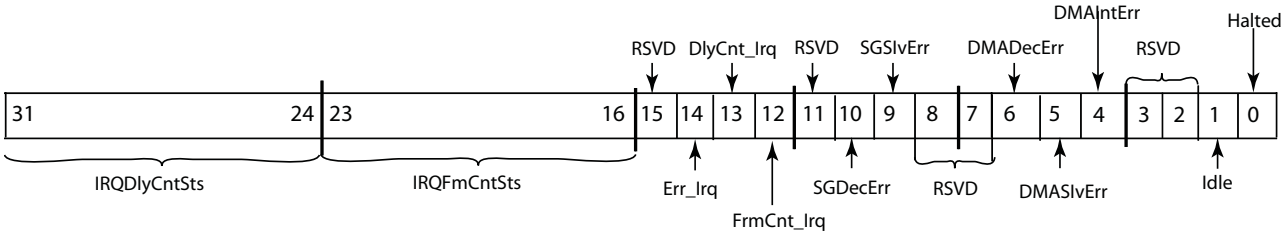

Figure 5: **MM2S DMASR Register**

Table 7: **MM2S_DMASR Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 24	IRQDelayCntSts	00h	RO	Interrupt Delay Count Status. Indicates current interrupt delay time value.
23 to 16	IRQFrameCntSts	01h	RO	Interrupt Frame Count Status. Indicates current interrupt frame count value.
15	Reserved	0	RO	Always read as zero.
14	Err_Irq	0	R/WC	Interrupt on Error. When set to 1, indicates an interrupt event was generated on error. If enabled (DMACR.Err_IrqEn = 1), an interrupt out will be generated from the AXI VDMA. 0 = No error Interrupt. 1 = Error interrupt detected.
13	DlyCnt_Irq	0	R/WC	Interrupt on Delay. Delay counter begins counting at the beginning of each frame and will reset at delay count or transfer of video line. If delay count reached then bit will set to 1 indicating an interrupt event was generated due to delay counter. If enabled (DMACR.DlyCnt_IrqEn = 1) then an interrupt out will be generated from the AXI VDMA. • 0 = No Delay Interrupt. • 1 = Delay Interrupt detected.
12	FrmCnt_Irq	0	R/WC	Frame Count Interrupt. When set to 1, indicates a Frame Count interrupt event was generated. This will occur when DMACR.FrameCount frames have been transferred. If enabled (DMACR.FrmCnt_IrqEn = 1) and if the interrupt threshold has been met, then an interrupt out will be generated from the AXI VDMA. • 0 = No Frame Count Interrupt. • 1 = Frame Count Interrupt detected.
11	Reserved	0	RO	Writing to this bit has no effect, and it will always read as zeros.

Table 7: **MM2S_DMASR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
10	SGDecErr	0	RO	Scatter Gather Decode Error. Decode Error detected by the Scatter Gather AXI DataMover - This error occurs if the address request is to an invalid address (that is, CURDESC and/or NXTDESC points to an invalid address). This error condition will cause both AXI VDMA channels (MM2S and S2MM) to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1 for both channels. • 0 = No SG Decode Errors. • 1 = SG Decode Error detected. DMA Engine will halt. ***Note:*** The CURDESC register will be updated with the errored descriptor pointer when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one address will be updated to the CURDESC. A reset (soft or hard) must be issued to clear the error condition.
9	SGSlvErr	0	RO	Scatter Gather Slave Error. Slave error detected by Scatter Gather AXI DataMover. This error occurs if the slave read from on the Memory Map interface issues a Slave error. This error condition will cause both AXI VDMA channels (MM2S and S2MM) to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1 for both channels. • 0 = No SG Slave Errors. • 1 = SG Slave Error detected. DMA Engine will halt. ***Note:*** The CURDESC_PTR register will be updated with the errored descriptor pointer when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one address will be updated to the CURDESC. A reset (soft or hard) must be issued to clear the error condition.
8 to 7	Reserved	zeros	RO	Writing to these bits has no effect, and they will always read as zeros.
6	DMADecErr	0	RO	DMA Decode Error. Decode error detected by primary AXI DataMover. This error occurs if the address request is to an invalid address (that is, the Descriptor Buffer Address points to an invalid address). This error condition will cause the AXI VDMA MM2S channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Decode Errors. • 1 = DMA Decode Error detected. DMA channel will halt. ***Note:*** The PRK_PTR_REG.RdFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.RdFrmStore. A reset (soft or hard) must be issued to clear the error condition.

Table 7: **MM2S_DMASR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
5	DMASlvErr	0	RO	DMA Slave Error. Slave error detected by primary AXI DataMover. This error occurs if the slave read from the Memory Map interface issues a Slave Error. This error condition will cause the AXI VDMA MM2S channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Slave Errors. • 1 = DMA Slave Error detected. DMA Engine will halt. ***Note:*** The PRK_PTR_REG.RdFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.RdFrmStore. A reset (soft or hard) must be issued to clear the error condition.
4	DMAIntErr	0	RO	DMA Internal Error. Internal error detected by primary AXI DataMover. This error can occur if a 0 length bytes to transfer is fed to the AXI DataMover. This situation only happens if the vertical size (vsize) and/or the horizontal size (hsize) specified in the fetched descriptor is set to 0. This error condition will cause the AXI VDMA MM2S channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the channel has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Internal Errors. • 1 = DMA Internal Error detected. DMA channel will halt. ***Note:*** The PRK_PTR_REG.RdFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.RdFrmStore. A reset (soft or hard) must be issued to clear the error condition.
3 to 2	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
1	Idle	0	RO	DMA Channel Idle. Indicates the state of AXI VDMA operations. The assertion of Idle indicates the SG Engine has reached the tail pointer for the associated channel and all queued descriptors have been processed. If in the Idle state (DMASR.Idle = 1), writing to the TailPointer register will automatically restart DMA operations. • 0 = Not Idle - SG operations in progress. • 1 = Idle - SG operation paused. ***Note:*** DMASR.Idle will only assert after the SG engine has passed through the descriptor chain at least once and has reached the TAILDESC. ***Note:*** Writing to the TAILDESC register when not Idle (DMASR.Idle = 0) will produce undefined results.
0	Halted	1	RO	DMA Channel Halted. DMA Channel Halted. Indicates the run/stop state of the DMA channel. • 0 = DMA channel running • 1 = DMA channel halted. This bit will get set when DMACR.RS = 0 and DMA and SG operations have halted. There may be a lag of time between when DMACR.RS = 0 and when DMASR.Halted = 1. ***Note:*** When halted (RS= 0 and Halted = 1), writing to CURDESC or TAILDESC pointer registers has no effect on DMA operations.
RO = Read Only. Writing has no effect R/WC = Read / Write to Clear. A CPU write of 1 will clear the associated bit to 0.				

MM2S_CURDESC (MM2S DMA Current Descriptor Pointer Register- Offset 08h)

This register provides Current Descriptor Pointer for the Memory Map to Stream DMA Scatter Gather Descriptor Management.

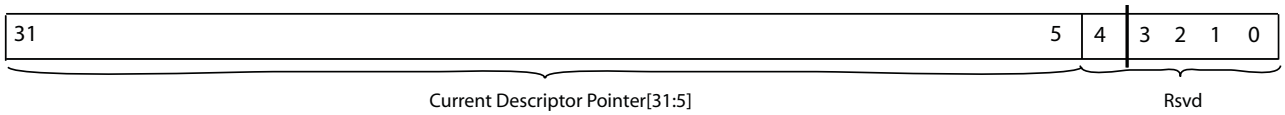

Figure 6: **MM2S CURDESC Register**

Table 8: **MM2S_CURDESC Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 5	Current Descriptor Pointer	zeros	R/W (RO)	Indicates the pointer of the current descriptor being worked on. This register must contain a pointer to a valid descriptor prior to writing the TAILDESC register. Otherwise, undefined results will occur. When DMACR.RS is 1, CURDESC becomes Read Only (RO) and is used to fetch the first descriptor. When the DMA Engine is running (DMACR.RS=1), CURDESC registers will be updated by AXI VDMA to indicate the current descriptor being worked on. On Scatter Gather error detection, CURDESC will be updated to reflect the descriptor associated with the detected error. The register can only be written to by the CPU when the DMA Engine is Halted (DMACR.RS=0 and DMASR.Halted =1). At all other times, this register is Read Only (RO). Descriptors must be 8 word aligned, that is, 0x00, 0x20, 0x40, etc. Any other alignment will have undefined results.
4 to 0	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
RO = Read Only. Writing has no effect. R/W = Read / Write.				

MM2S_TAILDESC (MM2S DMA Tail Descriptor Pointer Register- Offset 10h)

This register provides Tail Descriptor Pointer for the Memory Map to Stream DMA Scatter Gather Descriptor Management.

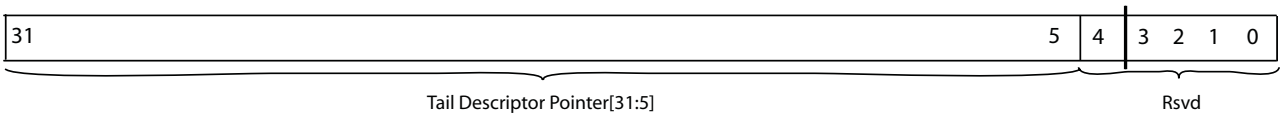

Figure 7: **MM2S_TAILDESC Register**

Table 9: **MM2S_TAILDESC Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 5	Tail Descriptor Pointer	zeros	R/W (RO)	Indicates the pause pointer in a descriptor chain. The AXI VDMA SG Engine will pause descriptor fetching after completing operations on the descriptor whose current descriptor pointer matches the tail descriptor pointer. When AXI VDMA Channel is not Halted (DMASR.Halted = 0) a write by the CPU to the TAILDESC register will cause the AXI VDMA SG Engine to start fetching descriptors or restart if it was idle (DMASR.Idle = 1). Writing to the TAILDESC when not idle (DMASR.Idle = 0) will have undefined results. If the AXI DMA Channel is Halted (DMASR.Halted = 1 and DMACR.RS = 0) a write by the CPU to the TAILDESC register will have no effect except to reposition the pause point. ***Note:*** Descriptors must be 8 word aligned, that is, 0x00, 0x20, 0x40, etc. Any other alignment will have undefined results.
4 to 0	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
RO = Read Only. Writing has no effect. R/W = Read / Write				

PARK_PTR_REG (Park Pointer Register - Offset 28h)

This register provides Park Pointer Registers for the Memory Map to Stream and Stream to Memory Map DMA transfer Management.

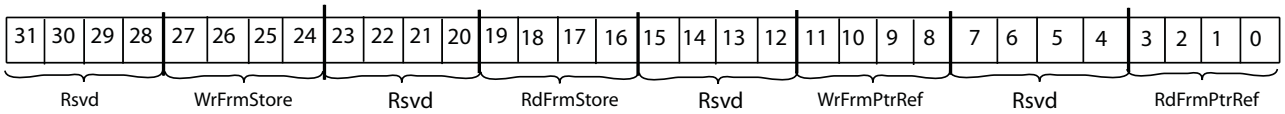

Figure 8: **PARK_PTR_REG Register**

Table 10: **PARK_PTR_REG Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 28	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
27 to 24	WrFrmStore		RO	Write Frame Store. Indicates current frame being operated on by S2MM channel. During DMA operations this value continually updates as each frame is processed. During Error conditions the value will be updated with the current frame being operated on when the error occurred.
23 to 20	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
19 to 16	RdFrmStore		RO	Read Frame Store. Indicates current frame being operated on by MM2S channel. During DMA operations this value continually updates as each frame is processed. During Error conditions the value will be updated with the current frame being operated on when the error occurred.
15 to 12	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
11 to 8	WrFrmPtrRef		RW	Write Frame Pointer Reference. When Parked (DMACR.Circular_Park = 0) references the S2MM Frame to park on.
7 to 4	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
3 to 0	RdFrmPtrRef		RW	Read Frame Pointer Reference. When Parked (DMACR.Circular_Park = 0) references the MM2S Frame to park on.
RO = Read Only. Writing has no effect. RW = Read / Write				

VDMA_VERSION (AXI VDMA Version Register - Offset 2Ch)

This register provides the AXI VDMA Version.

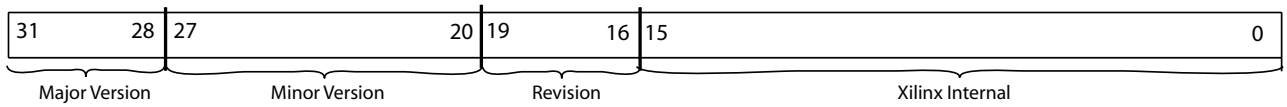

Figure 9: **VDMA_VERSION Register**

Table 11: **VDMA_VERSION Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 28	Major Version	1h	RO	Single 4-bit hexadecimal value. v1 = 1h, v2=2h, v3=3h, etc.
27 to 20	Minor Version	00h	RO	Two separate 4-bit hexadecimal values. 00 = 00h, 01 = 01h, etc.
19 to 16	Revision	Ah	RO	Revision letter as a hexadecimal character from ‘a’ to ‘f’. Mapping is as follows: Ah-> ‘a’, Bh -> ‘b’, Ch-> ‘c’, etc.
15 to 0	Xilinx Internal	various	RO	Reserved for Internal Use Only. Integer value from 0 to 9999
RO = Read Only. Writing has no effect.				

Stream to Memory Map Register Detail

S2MM_DMACR (S2MM DMA Control Register - Offset 30h)

This register provides control for the Stream to Memory Map DMA Channel.

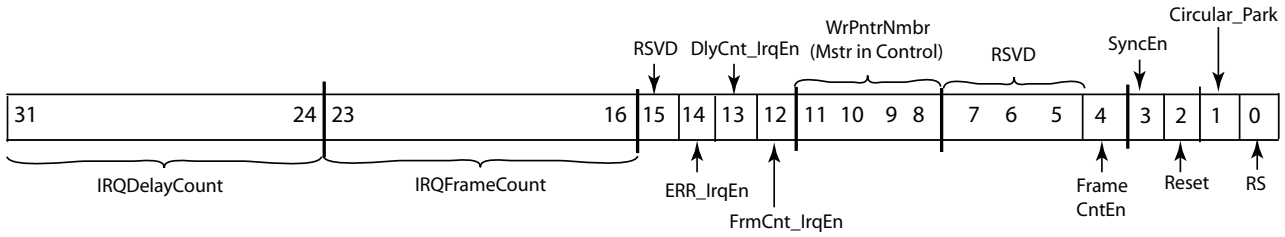

Figure 10: **S2MM DMACR Register**

Table 12: **S2MM_DMACR Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 24	IRQDelayCount	00h	R/W	This value is used for setting the interrupt delay count value. The delay count interrupt is a mechanism for causing the DMA engine to generate an interrupt after the delay period has expired. This is used for cases when the interrupt frame count is not met after a period of time and the CPU desires an interrupt to be generated. Timer begins counting at the start of a frame and will reset with the transfer of a new packet on S2MM stream or when delay count period has expired. When a value different than the current IRQDelayCount is written to this field, the internal delay counter will be reset to zero. ***Note:*** Setting this value to zero will disable the delay counter interrupt.
23 to 16	IRQFrameCount	01h	R/W	This value is used for setting the interrupt threshold. When frame transfer interrupt events occur an internal counter counts down from the Interrupt Frame Count setting. When the count reaches zero an interrupt out is generated by the VDMA engine. When a value different than the current IRQFrameCount is written to this field, the internal frame counter will be reset to the new value. ***Note:*** The minimum setting for the count is 0x01. A write of 0x00 to this register will set the count to 0x01. ***Note:*** When DMACR.FrameCntEn = 1, this value will determines the number of frame buffers to process.
15	Reserved	0	RO	Writing to this bit has no effect and it will always read as zeros.
14	Err_IrqEn	0	R/W	Interrupt on Error Interrupt Enable. When set to 1, will allow DMASR.Err_Irq to generate an interrupt out. • 0 = Error Interrupt disabled • 1 = Error Interrupt enabled
13	DlyCnt_IrqEn	0	R/W	Interrupt on Delay Count Interrupt Enable. When set to 1, will allow DMASR.DlyCnt_Irq to generate an interrupt out. • 0 = Delay Count Interrupt disabled • 1 = Delay Count Interrupt enabled
12	FrmCnt_IrqEn	0	R/W	Frame Count Complete Interrupt Enable. When set to 1, will allow DMASR.FrmCnt_Irq to generate an interrupt out when IRQFrameCount value reaches zero. • 0 = Frame Count Interrupt disabled • 1 = Frame Count Interrupt enabled
11 to 8	WrPntrNum	zeros	RW	Indicates the master in control when S2MM channel is configured for Gen-Lock slave mode (C_S2MM_GENLOCK_MODE = 1). 0000b = Controlling master is Master 1. 0001b = Controller master is Master 2. 0010b = Controller master is Master 3. etc. ***Note:*** Maximum valid RdPntrNum is C_S2MM_GENLOCK_NUM_MASTER - 1. Setting to a value greater than C_S2MM_GENLOCK_NUM_MASTER - 1 will have undefined results.
7 to 5	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
4	FrameCntEn	0	RW	Configures MM2S channel to only allow IRQFrameCount number of transfers to occur. After IRQFrameCount frames have been transferred the S2MM channel will halt, DMACR.RS bit will be cleared to 0 and DMASR.Halted will assert to 1 once the channel has completely halted.

Table 12: **S2MM_DMACR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
3	SyncEn	0	RW	Enables Gen-Lock synchronization. • 0 = Gen-Lock synchronization disabled. Gen-Lock input ignored by S2MM. • 1 = Gen-Lock synchronization enabled. MM2S synchronized to Gen-Lock frame input. ***Note:*** This value is only valid when channel configured as Gen-Lock Slave (C_S2MM_GENLOCK_MODE = 1). If configured for Gen-Lock Master mode (C_S2MM_GENLOCK_MODE = 0) then this bit is reserved and always reads as zero.
2	Reset	0	RW	Soft reset for resetting the AXI VDMA S2MM channel. Setting this bit to a 1 causes the AXI VDMA S2MM channel to be reset. Reset will be accomplished gracefully. Pending commands/transfers will be flushed or completed. AXI4-Stream reset output will be asserted. Setting DMACR.Reset = 1 will only reset the S2MM channel. After completion of a soft reset all S2MM registers and bits will be in the Reset State. • 0 = Reset NOT in progress - Normal operation • 1 = Reset in progress
1	Circular_Park	1	RO	Indicates frame buffer Circular mode or frame buffer Park mode. 0 = Park Mode- Engine will park on frame buffer referenced by PARK_PTR_REG.WrFrmPntrRef. 1 = Circular Mode - Engine will continuously circle through C_NUM_FSTORES frame buffers. ***Note:*** Park Mode must only be enabled when AXI VDMA is Idle (DMASR.Idle = 1) and NOT halted (DMASR.Halted = 0). Undefined results will occur if enabled at any other time.
0	RS	0	R/W	Run / Stop controls running and stopping of the VDMA channel. For any DMA operations to commence the AXI VDMA engine must be running (DMACR.RS=1) • 0 = Stop - VDMA stops when current (if any) DMA operations are complete. Queued descriptors for the associated channel will be flushed from the SG Engine. The halted bit in the DMA Status Register will assert to 1 when the DMA engine is halted. This bit will get cleared by AXI VDMA hardware when an error occurs. The CPU may also choose to clear this bit to stop DMA operations when an error occurs or when the IRQFrameCount is reached when Frame Count Enable is asserted (DMACR.FrameCntEn = 1). • 1 = Run - Start DMA operations. The halted bit in the DMA Status Register will deassert to 0 when the DMA engine begins operations. ***Note:*** On Run/Stop clear in progress stream transfers may terminate early.
RO = Read Only. Writing has no effect. R/W = Read / Write.				

S2MM_DMASR (S2MM DMA Status Register- Offset 34h)

This register provides the status for the Stream to Memory Map DMA Channel.

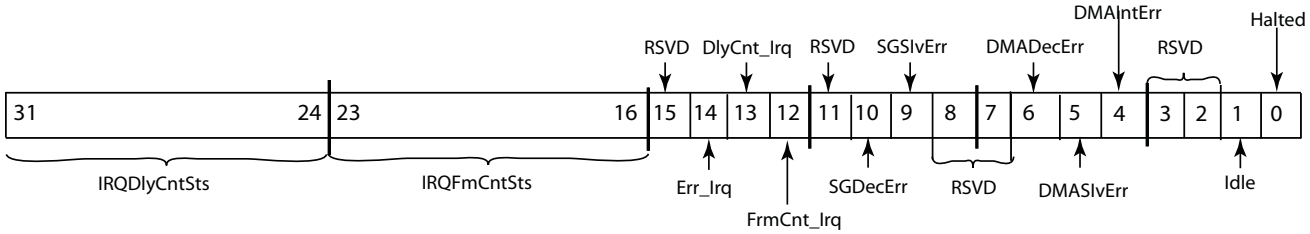

Figure 11: **S2MM DMASR Register**

Table 13: **S2MM_DMASR Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 24	IRQDelayCntSts	00h	RO	Interrupt Delay Count Status. Indicates current interrupt delay time value.
23 to 16	IRQFrameCntSts	01h	RO	Interrupt Frame Count Status. Indicates current interrupt frame count value.
15	Reserved	0	RO	Always read as zero.
14	Err_Irq	0	R/WC	Interrupt on Error. When set to 1, indicates an interrupt event was generated on error. If enabled (DMACR.Err_IrqEn = 1), an interrupt out will be generated from the AXI VDMA. 0 = No error Interrupt. 1 = Error interrupt detected.
13	DlyCnt_Irq	0	R/WC	Interrupt on Delay. Delay counter begins counting at the beginning of each frame and will reset at delay count or transfer of video line. If delay count reached then bit will set to 1 indicating an interrupt event was generated due to delay counter. If enabled (DMACR.DlyCnt_IrqEn = 1) then an interrupt out will be generated from the AXI VDMA. • 0 = No Delay Interrupt. • 1 = Delay Interrupt detected.
12	FrmCnt_Irq	0	R/WC	Frame Count Interrupt. When set to 1, indicates a Frame Count interrupt event was generated. This will occur when DMACR.FrameCount frames have been transferred. If enabled (DMACR.FrmCnt_IrqEn = 1) and if the interrupt threshold has been met, then an interrupt out will be generated from the AXI VDMA. • 0 = No Frame Count Interrupt. • 1 = Frame Count Interrupt detected.
11	Reserved	0	RO	Writing to this bit has no effect, and it will always read as zeros.

Table 13: **S2MM_DMASR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
10	SGDecErr	0	RO	Scatter Gather Decode Error. Decode Error detected by the Scatter Gather AXI DataMover - This error occurs if the address request is to an invalid address (that is, CURDESC and/or NXTDESC points to an invalid address). This error condition will cause both AXI VDMA channels (MM2S and S2MM) to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1 for both channels. • 0 = No SG Decode Errors. • 1 = SG Decode Error detected. DMA Engine will halt. ***Note:*** The CURDESC register will be updated with the errored descriptor pointer when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one address will be updated to the CURDESC. A reset (soft or hard) must be issued to clear the error condition.
9	SGSlvErr	0	RO	Scatter Gather Slave Error. Slave error detected by Scatter Gather AXI DataMover. This error occurs if the slave read from on the Memory Map interface issues a Slave error. This error condition will cause both AXI VDMA channels (MM2S and S2MM) to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1 for both channels. • 0 = No SG Slave Errors. • 1 = SG Slave Error detected. DMA Engine will halt. ***Note:*** The CURDESC register will be updated with the errored descriptor pointer when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one address will be updated to the CURDESC. A reset (soft or hard) must be issued to clear the error condition.
8 to 7	Reserved	zeros	RO	Writing to these bits has no effect, and they will always read as zeros.
6	DMADecErr	0	RO	DMA Decode Error. Decode error detected by primary AXI DataMover. This error occurs if the address request is to an invalid address (that is, the Descriptor Buffer Address points to an invalid address). This error condition will cause the AXI VDMA S2MM channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Decode Errors. • 1 = DMA Decode Error detected. DMA channel will halt. ***Note:*** The PRK_PTR_REG.WrFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.WrFrmStore. A reset (soft or hard) must be issued to clear the error condition.

Table 13: **S2MM_DMASR Register Details *(Cont'd)***

Bits	Field Name	Default Value	Access Type	Description
5	DMASlvErr	0	RO	DMA Slave Error. Slave error detected by primary AXI DataMover. This error occurs if the slave read from the Memory Map interface issues a Slave Error. This error condition will cause the AXI VDMA S2MM channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the engine has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Slave Errors. • 1 = DMA Slave Error detected. DMA Engine will halt. ***Note:*** The PRK_PTR_REG.WrFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.WrFrmStore. A reset (soft or hard) must be issued to clear the error condition.
4	DMAIntErr	0	RO	DMA Internal Error. Internal error detected by primary AXI DataMover. This error can occur if a 0 length bytes to transfer is fed to the AXI DataMover. This situation only happens if the vertical size (vsize) and/or the horizontal size (hsize) specified in the fetched descriptor is set to 0. This error can also occur if a packet greater or less than hsize bytes is received. This error condition will cause the AXI VDMA S2MM channel to gracefully halt. The DMACR.RS bit will be set to 0, and when the channel has completely shut down, the DMASR.Halted bit will be set to 1. • 0 = No DMA Internal Errors. • 1 = DMA Internal Error detected. DMA channel will halt. ***Note:*** The PRK_PTR_REG.WrFrmStore field will be updated with the errored frame reference when this error is detected. If multiple errors are detected, the errors will be logged in the DMASR, but only one reference will be updated to PRK_PTR_REG.WrFrmStore. A reset (soft or hard) must be issued to clear the error condition.
3 to 2	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
1	Idle	0	RO	DMA Channel Idle. Indicates the state of AXI VDMA operations. The assertion of Idle indicates the SG Engine has reached the tail pointer for the associated channel and all queued descriptors have been processed. If in the Idle state (DMASR.Idle = 1), writing to the TailPointer register will automatically restart DMA operations. • 0 = Not Idle - SG operations in progress. • 1 = Idle - SG operation paused. ***Note:*** DMASR.Idle will only assert after the SG engine has passed through the descriptor chain at least once and has reached the TAILDESC. ***Note:*** Writing to the TAILDESC register when not Idle (DMASR.Idel = 0) will produce undefined results.
0	Halted	1	RO	DMA Channel Halted. DMA Channel Halted. Indicates the run/stop state of the DMA channel. • 0 = DMA channel running • 1 = DMA channel halted. This bit will get set when DMACR.RS = 0 and DMA and SG operations have halted. There may be a lag of time between when DMACR.RS = 0 and when DMASR.Halted = 1. ***Note:*** When halted (RS= 0 and Halted = 1), writing to CURDESC or TAILDESC pointer registers has no effect on DMA operations.

RO = Read Only. Writing has no effect
R/WC = Read / Write to Clear. A CPU write of 1 will clear the associated bit to 0.

S2MM_CURDESC (S2MM DMA Current Descriptor Pointer Register- Offset 38h)

This register provides the Current Descriptor Pointer for the Stream to Memory Map DMA Scatter Gather Descriptor Management.

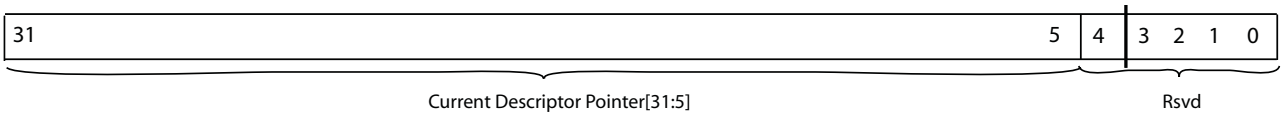

Figure 12: **S2MM CURDESC Register**

Table 14: **S2MM_CURDESC Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 6	Current Descriptor Pointer	zeros	R/W (RO)	Indicates the pointer of the current descriptor being worked on. This register must contain a pointer to a valid descriptor prior to writing the TAILDESC register. Otherwise, undefined results will occur. When DMACR.RS is 1, CURDESC becomes Read Only (RO) and is used to fetch the first descriptor. When the DMA Engine is running (DMACR.RS=1), CURDESC registers will be updated by AXI VDMA to indicate the current descriptor being worked on. On Scatter Gather error detection, CURDESC will be updated to reflect the descriptor associated with the detected error. ***Note:*** The register can only be written to by the CPU when the DMA Engine is Halted (DMACR.RS=0 and DMASR.Halted =1). At all other times, this register is Read Only (RO). Descriptors must be 8 word aligned, that is, 0x00, 0x20, 0x40, etc. Any other alignment will have undefined results.
5 to 0 (Offset 0x38)	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
RO = Read Only. Writing has no effect. R/W = Read / Write.				

S2MM_TAILDESC (S2MM DMA Tail Descriptor Pointer Register- Offset 40h)

This register provides the Tail Descriptor Pointer for the Stream to Memory Map DMA Scatter Gather Descriptor Management.

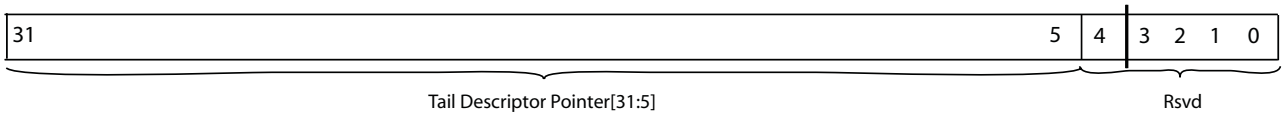

Figure 13: **S2MM TAILDESC Register**

Table 15: **S2MM_TAILDESC Register Details**

Bits	Field Name	Default Value	Access Type	Description
31 to 6	Tail Descriptor Pointer	zeros	R/W (RO)	Indicates the pause pointer in a descriptor chain. The AXI VDMA SG Engine will pause descriptor fetching after completing operations on the descriptor whose current descriptor pointer matches the tail descriptor pointer. When AXI VDMA Channel is not Halted (DMASR.Halted = 0) a write by the CPU to the TAILDESC register will cause the AXI VDMA SG Engine to start fetching descriptors or restart if it was idle (DMASR.Idle = 1). Writing to the TAILDESC when not idle (DMASR.Idle = 0) will have undefined results. If the AXI DMA Channel is Halted (DMASR.Halted = 1 and DMACR.RS = 0) a write by the CPU to the TAILDESC register will have no effect except to reposition the pause point. ***Note:*** Descriptors must be 8 word aligned, that is, 0x00, 0x20, 0x40, etc. Any other alignment will have undefined results.
5 to 0	Reserved	0	RO	Writing to these bits has no effect, and they will always read as zeros.
RO = Read Only. Writing has no effect. R/W = Read / Write.				

Scatter Gather Descriptor

For AXI VDMA the Scatter Gather engine is used to pull in video transfer control information. This is accomplished by defining a linked list of transfer control information referred to as a descriptor chain in system memory. A descriptor chain is required for each channel. The descriptor chain should be made up of C_NUM_FSTORES descriptors where each descriptor is made up of seven 32-bit words. Each descriptor describes a video frame's worth of transfers. The following describes the descriptor fields.

Note: Descriptors must be aligned on 8 32-bit word alignment. Example valid offsets are 0x00, 0x20, 0x40, 0x60, etc.

Table 16: **Descriptor Fields**

Address Space Offset[1]	Name	Description
00h	NXTDESC	Next Descriptor Pointer points to the first word of the next descriptor to fetch
04h	RESERVED	N/A
08h	START_ADDRESS	Start Address points to the starting pixel to transfer for the given frame
0Ch	RESERVED	N/A
10h	VSIZE	Vertical Size specifies the number of video lines to transfer for the given frame
14h	HSIZE	Horizontal Size specifies the size in bytes of the horizontal line to transfer for a given frame
18h	FRMDLY_STRIDE	Stride specifies the number of bytes between the first pixels of each horizontal line. Frame Delay specifies the number of frames a Gen-Lock slave should be behind the Gen-Lock Master

1. Address Space Offset is relative to 8 - 32-bit word alignment in system memory, that is, 0x00, 0x20, 0x40 etc.

NXTDESC (Next Descriptor Pointer)

This value provides the pointer to the next descriptor in the descriptor chain.

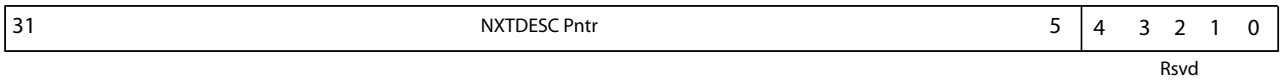

Figure 14: **NXTDESC**

Table 17: **NXTDESC Details**

Bits	Field Name	Description
31 to 5	Next Descriptor Pointer	Indicates the lower order pointer pointing to the first word of the next descriptor. ***Note:*** Descriptors must be 8 word aligned, that is, 0x00, 0x20, 0x40, etc. Any other alignment will have undefined results.
4 to 0	Reserved	These bits are reserved and should be set to zero.

START_ADDRESS (Start Address)

This value provides the pointer to the buffer of data to transfer from system memory to stream.

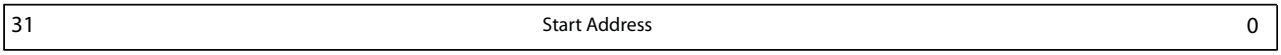

Figure 15: **Start Address**

Table 18: **START_ADDRESS Details**

Bits	Field Name	Description
31 to 0	Start Address	Provides the starting pixel location of the data to transfer. For MM2S channel data is read from system memory starting at this address. For S2MM channel data is written to system memory starting at this address. ***Note:*** If Data Realignment Engine is included (C_INCLUDE_MM2S_DRE = 1 or C_INCLUDE_S2MM_DRE = 1) for the respective channel, then Start Address may be at any byte offset. If the Data Realignment Engine is not included (C_INCLUDE_MM2S_DRE = 0 or C_INCLUDE_S2MM_DRE = 0) for the respective channel, then Start Address must be stream data width aligned.

VSIZE (Vertical Size)

This value provides vertical size (or number of lines) to transfer.

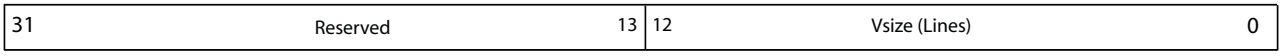

Figure 16: **VSIZE**

Table 19: **VSIZE Details**

Bits	Field Name	Description
31 to 13	Reserved	This bit is reserved and should be written as zero.
12 to 0	VSize	Vertical size in lines of the video data to transfer. On the MM2S stream interface there are vsize number of packets that are hsize bytes long transmitted for each frame. On the S2MM stream interface vsize number of packets that are hsize bytes long are expected to be received for each frame. ***Note:*** A value of zero on Vsize will cause a MM2S_DMAIntErr or S2MM_DMAIntErr for the associated channel to be logged and the channel with the detected error will be shut down.

HSIZE (Horizontal Size)

This value provides horizontal size in bytes of the video line to transfer.

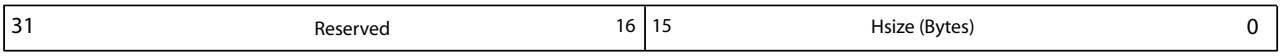

Figure 17: **HSIZE**

Table 20: **HSIZE Details**

Bits	Field Name	Description
31 to 16	Reserved	These bits are reserved and should be set to zero.
15 to 0	HSize	Horizontal size in bytes of the video data to transfer On the MM2S stream interface there are vsize number of packets that are hsize bytes long transmitted for each frame. On the S2MM stream interface vsize number of packets that are hsize bytes long are expected to be received for each frame. ***Note:*** If more or less data is received on S2MM stream interface an underrun or overrun error will occur. This will present itself as a S2MM_DMASR.DMAIntErr and the AXI VDMA S2MM channel will be shut down. ***Note:*** A value of zero for Hsize will cause a MM2S_DMAIntErr or S2MM_DMAIntErr for the associated channel to be logged and the channel with the detected error will be shut down.

FRMDLY_STRIDE (Frame Delay and Stride)

This value provides the Gen-Lock Frame Delay and Stride for the video transfer.

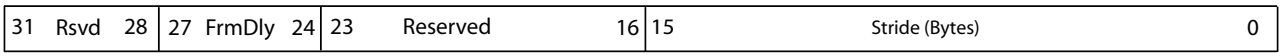

Figure 18: **FRMDLY_STRIDE**

Table 21: **FRMDLY_STRIDE Details**

Bits	Field Name	Description
31 to 28	Reserved	These bits are reserved and should be set to zero.
27 to 24	FrmDly	Frame Delay is the number of frame stores the slave should be behind the locked master. This field is only used if channel is enabled for Gen-Lock Slave Operations (C_MM2S_GENLOCK_MODE = 1 or C_S2MM_GENLOCK_MODE=1) for the respective channel.
23 to 16	Reserved	These bits are reserved and should be set to zero.
15 to 0	Stride	Number of bytes between first pixels of each line. ***Note:*** Stride values less than HSize will result in corrupt data

AXI VDMA Operation

AXI VDMA Operation requires a memory-resident data structure that holds the list of DMA operations to be performed. This list of instructions is organized into what is referred to as a descriptor chain. Each descriptor has a pointer to the next descriptor to be processed. The last descriptor in the chain then points back to the first descriptor in the chain.

VDMA operations began with the set up of the descriptor pointer registers and the DMA control registers. The following lists minimum steps, in order, required for AXI VDMA operations:

1. Write a valid pointer to the channel's CURDESC_PNTR register (Offset 0x08 for MM2S and 0x38 for S2MM).
2. Write control information to channel's DMACR register (Offset 0x00 for MM2S and 0x30 for S2MM) to set interrupt enables if desired, frame count, delay count if desired, and set DMACR.RS=1 to start the AXI VDMA channel running. Note there may be a lag between when the CPU sets DMACR.RS=1 and when AXI VDMA sets DMASR.Halted = 0. The CPU can determine if the AXI VDMA is running when DMACR.RS = 1 and DMASR.Halted = 0.
3. Write a valid point to the channel's TAILDESC_PNTR register (Offset 0x10 for MM2S and 0x40 for S2MM). This will start the channel fetching and processing descriptors.
4. DMA scatter gather operations will continue until the descriptor at TAILDESC_PNTR is processed, then the engine will Idle as indicated by DMASR.Idle = 1.

The descriptor chain is made up of C_NUM_FSTORES descriptors per channel. The SG Engine will fetch descriptors and update an internal register set with the descriptor information (See Example Figure 19 SG Labeled Register Block). The video timing information (vsize, hsize, stride, and frame delay) from the first descriptor fetched after the TAILDESC pointer is written by the CPU will be captured and stored. This video timing information will be used for all transfers in all frames for the specified channel. All other video timing information in the other descriptors will be ignored by the AXI VDMA. Start Addresses from each descriptor will be populated into C_NUM_FSTORES internal address registers.

When the SG Engine reaches the descriptor pointed to by TAILDESC, the SG Engine will process that descriptor, set the internal register set to ping-pong on the next frame sync (Internally generated if C_USE_FSYNC = 0 or externally if C_USE_FSYNC = 1), and enter an Idle state. At the next frame sync the internal register set will switch such that the DMA controller will operate on the newly updated values.

This method for handling data updates internally allows the SG engine to operate on a register set without stepping on the register set that the DMA controller is using for data transfers. It also allows the SG Engine to only fetch descriptors when a change is made to the data set making it unnecessary for the SG engine to fetch a descriptor with each frame. It should be noted that the DMA controller will continue to transfer video data even though the SG engine has reached the tail pointer and paused. This allows un-interrupted video data transfers.

Updating Video Transfer Information

To update the video transfer information for a channel, the CPU writes new start address, vsize, hsize, stride, and frame delay information into the descriptor following the TAILDESC position. Then the CPU simply writes a new TAILDESC pointing to this newly updated descriptor. The SG Engine will automatically fetch the newly updated descriptor and update the start address, vsize, hsize, stride, and frame delay to the internal SG labeled register block. On the next frame sync for the channel these newly updated values will be transferred to the Video labeled register block for use by the channel's DMA controller. The outputs, `mm2s_prmtr_update` and `s2mm_prmtr_update`, for the respective channel will assert coincident with `mm2s_fsync` and `s2mm_fsync` respectively when the new parameters take effect for the channel.

Note: On S2MM channel new video line size and number of video lines need to change following the assertion of `s2mm_prmtr_update` or undefined results will occur.

The entire descriptor chain can be updated in this manner, allowing a completely new set of start addresses and vsize, hsize, stride, and frame delay to be fetched and updated to the DMA controller. Simple update the descriptor chain and then write a TAILDESC pointer pointing to the last descriptor in the chain. All C_NUM_FSTORES descriptors will be fetched.

The CPU should only update the TAILDESC when the engine is Idle otherwise undefined results will occur. Updating the TAILDESC register when the SG Engine is Idle will insure that video timing information will not be ignored in the descriptor.

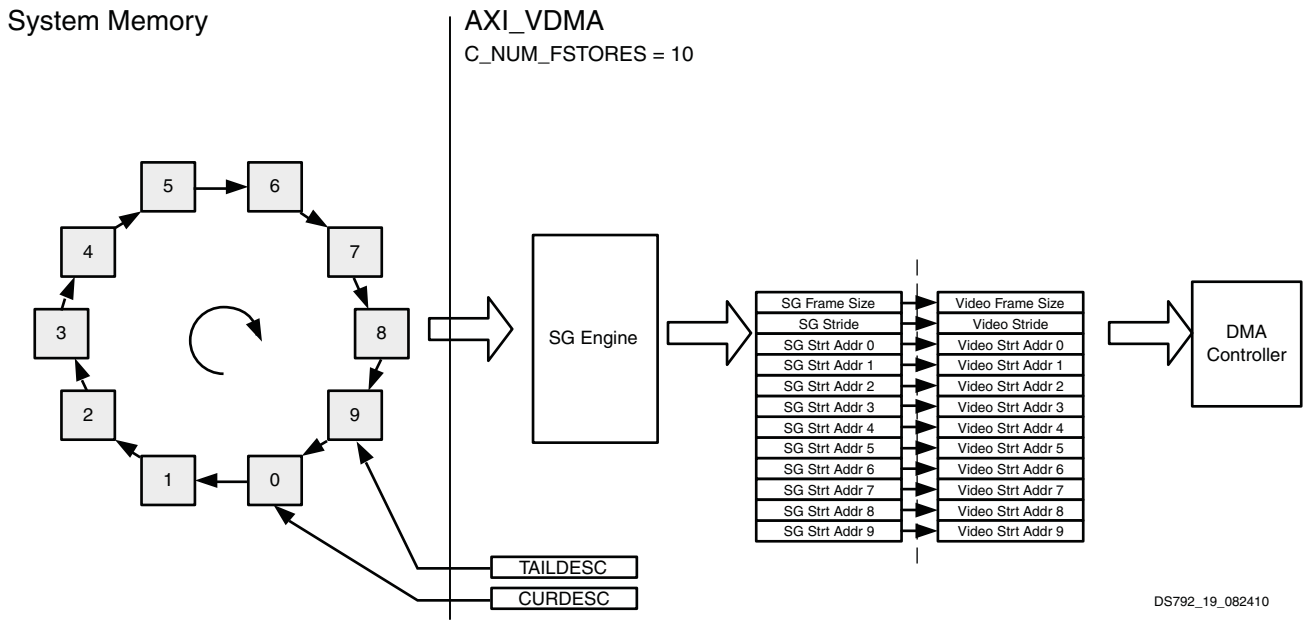

Figure 19: **Example SG Fetch with C_NUM_FSTORES= 10**

The AXI VDMA Scatter Gather engine has independent descriptor queues for fetch descriptors for the MM2S and S2MM channels. Independent register sets and dma controllers for MM2S and S2MM are also provided.

Video Transfer

Data is transferred from System Memory to Stream or Stream to System Memory as defined by the Start Address for the frame currently being operated on and the vertical size, horizontal size, and stride. Figure 20 illustrates a typical video image stored in system memory. The smaller portion of the video image is to be transferred.

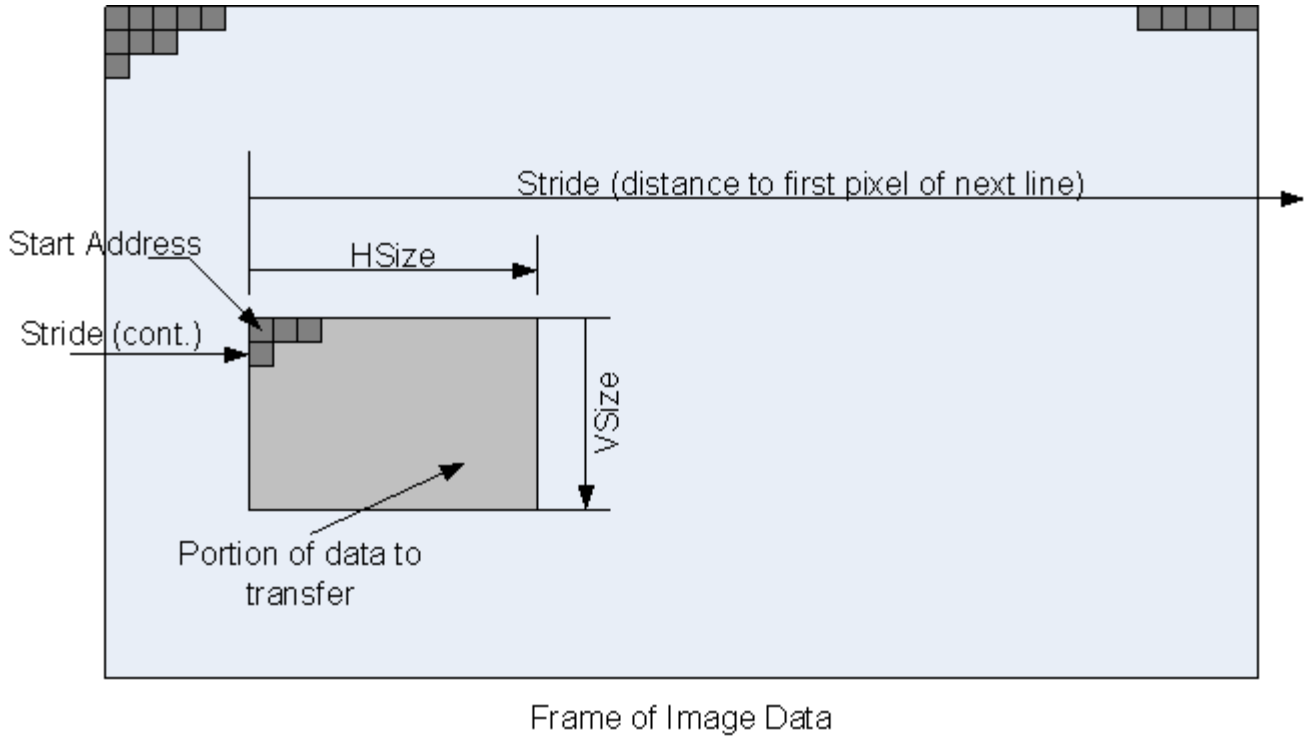

Figure 20: **Example Video Image Transfer**

Vertical Size lines of Horizontal number of bytes are transferred from or two system memory starting at each start address for each video frame. In free run mode (C_USE_FSYNC = 0) video data is transferred as quickly as possible. The output ports `mm2s_fsync_out` and `s2mm_fsync_out` indicate the frame boundaries. On start up, that is, DMACR.RS set to 1 for the respective channel, an initial frame sync (`mm2s_fsync_out` or `s2mm_fsync_out`) will assert after the video parameters have been fetched and updated. This initial assertion of frame sync indicates the beginning of the first frame. For the MM2S channel, video data for a frame will be transferred after the assertion of each `mm2s_fsync_out`. For the S2MM channel, axi_vdma expects new data for the frame to be transferred to the s2mm channel after the assertion of `s2mm_fsync_out`. The AXI4-Stream output port, `s_axis_s2mm_tready`, will assert after `s2mm_fsync_out` indicating the S2MM channel is ready to receive data. When all expected data (vsize lines that are hsize bytes long) has been received, AXI VDMA will deassert `s_axis_s2mm_tready` and will not re-assert until the next `s2mm_fsync_out`. For both channels, MM2S and S2MM, when configured for free run mode, `mm2s_fsync_out` and `s2mm_fsync_out` will assert only after all of the data for a frame has been transferred. In free run mode the progression of video frames is in part, controlled by the target Video IP on the MM2S or S2MM AXI4-Stream. For MM2S, if `m_axis_mm2s_tready` is not asserted then the frame time for mm2s will be extended. Only after all video lines have been transferred will the frame be considered complete and a new `mm2s_fsync_out` will assert. Likewise, on S2MM if `s_axis_s2mm_tvalid` is not asserted then the frame time for s2mm will be extended. Only after all video lines have been transferred will the frame be considered complete and a new `s2mm_fsync_out` will assert.

In frame sync mode (C_USE_FSYNC = 1) video data is synchronized to an external frame sync, mm2s_fsync for the MM2S channel and s2mm_fsync for the S2MM channel. The axi_vdma channel will begin each frame on the falling edge of the associated channel's frame sync input, `mm2s_fsync` or `s2mm_fsync`. For the MM2S channel, data will be driven out on the Master AXI4-Stream port following the assertion of `mm2s_fsync`. It is the responsibility of the target video IP to accept all transferred video lines before the assertion of the next `mm2s_fsync` or undefined results will occur. For the S2MM channel, data will be accepted on the Slave AXI4-Stream port following the assertion of `s2mm_fsync`. Again, the S2MM AXI4-Stream output port `s_axis_s2mm_tready` will assert after `s2mm_fsync_out` indicating the S2MM channel is ready to receive data for a particular video frame, and will deassert `s_axis_s2mm_tready` when all expected data for the frame has been received.

It is the responsibility of the target video IP to only start driving data to the S2MM channel after the assertion of `s2mm_fsync` and to transfer all video lines before the assertion of the next `s2mm_fsync` or undefined results will occur.

Minimum time between assertions of the associated channel's fsync input, `mm2s_fsync` or `s2mm_fsync`, has been verified via simulation down to 1 usec. Fsync rates faster than 1 usec between assertions will produce undefined results.

Figure 21 illustrates three frames worth of image transfer. MM2S channel reads from Frame 0 then on MM2S Frame Sync transitions to reading from Frame 1, then Frame 2, and back to Frame 0 continuing this cycle with each Frame sync. S2MM writes to Frame 1 then transitions to Frame 2, followed by Frame 0. S2MM continues this progressing with each S2MM Frame Sync. To keep S2MM and MM2S from stepping on each other Gen-Lock synchronization is used.

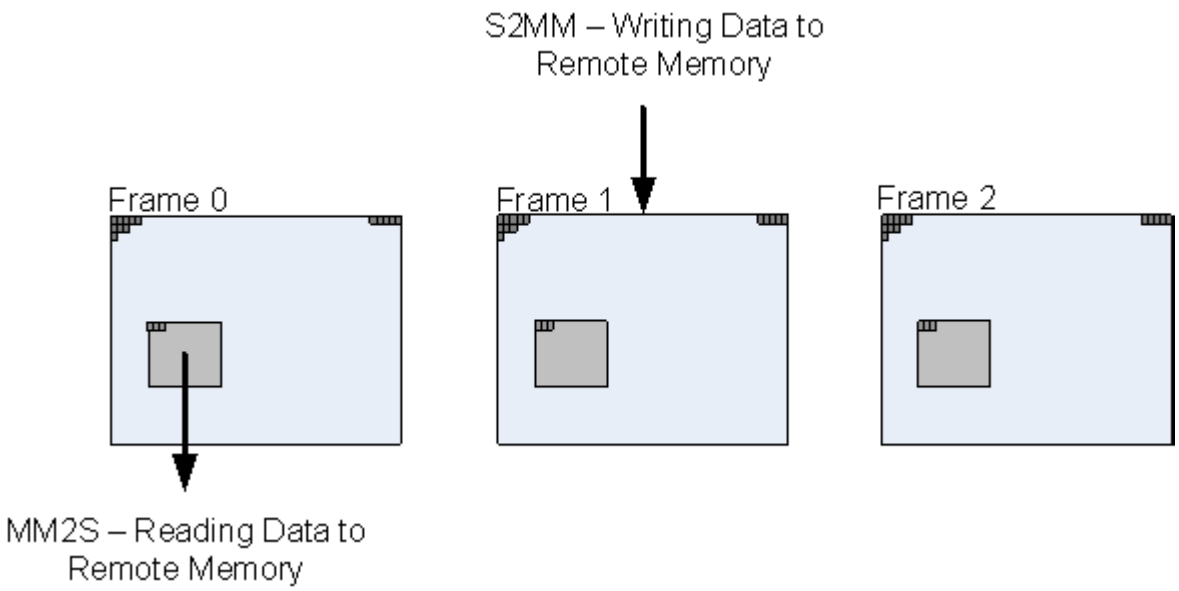

Figure 21: **Three Frame Example with S2MM and MM2S**

Gen-Lock Synchronization

In many video applications, a producer of data will run at a different rate than the consumer of that data. To avoid the potential ill effects that such a rate mismatch can cause, frame buffering is often used. Frame buffering allocates multiple frames worth of memory to be used to hold the data. The data producer writes to one buffer while the consumer reads from another (See Figure 21). The two are kept in sync by not allowing both to use the same buffer at the same time. Typically one of the two will be forced to either skip or repeat frames as necessary. This type of synchronization is called Gen-Lock and is provided when the axi_vdma is configured for frame sync mode (C_USE_FSYNC = 1). Each channel of AXI VDMA has been designed to operate as either a Gen-Lock Master or Slave. In general, a Gen-Lock master specifies to a Gen-Lock slave which frame to operate on at any given time. Figure 22 illustrates simple timing of Gen-Lock operation. In this example C_NUM_FSTORES = 3, C_USE_FSYNC = 1, the frame delay for S2MM has been set to 1. Also, MM2S has been configured as the Gen-Lock Master and S2MM channel has been configured as the Gen-Lock Slave. The MM2S channel's frame rate is faster than that of S2MM so the S2MM Slave will skip frames automatically. As you can see in Figure 22, in the time the MM2S channel cycles through frame 0, 1, 2 and back to 0, the S2MM channel has only cycled through two frame. Due to the slow frame rate of S2MM compared to MM2S, the S2MM channel will process frame 2 then frame 0 then frame 2 again, skipping frame 1.

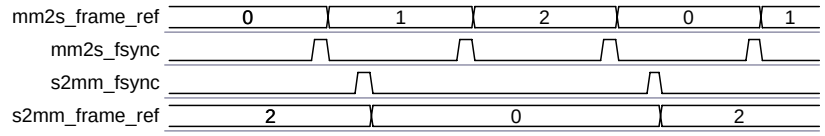

Figure 22: **Example Gen-Lock Timing**

The Gen-Lock mechanism that the AXI VDMA implements is based around the internal Start Address registers. The number of Start Address registers can be configured to be between 1 and 16. The C_NUM_FSTORES parameter specifies this value. The Gen-Lock Master uses the index of the Start Address register to specify which Start Address register the Gen-Lock Slave should use. This Start Address Register index is encoded as a Grey code value. The Grey Code that is used depends upon the number of Frame Stores that was specified. Table 22 lists the Grey Codes that are used for each of the 16 possible Frame Store sizes. The Grey Code cycles through all of the codes on the first line first and then cycles through all of the codes on the second line before repeating the first line. Number of grey codes is double the number of frame stores to allow for non-power-of-two frame store values to be cycled through and still maintain grey code coherency with minimal FPGA resources required.

Table 22: **Gen-Lock Grey Code**

C_NUM_FSTORES	Grey Code															
1	0															
	1															
2	0	1														
	3	2														
3	1	3	2													
	6	7	5													
4	0	1	3	2												
	6	7	5	4												
5	2	6	7	5	4											
	12	13	15	14	10											
6	3	2	6	7	5	4										
	12	13	15	14	10	11										
7	1	3	2	6	7	5	4									
	12	13	15	14	10	11	9									
8	0	1	3	2	6	7	5	4								
	12	13	15	14	10	11	9	8								
9	4	12	13	15	14	10	11	9	8							
	24	25	27	26	30	31	29	28	20							
10	5	4	12	13	15	14	10	11	9	8						
	24	25	27	26	30	31	29	28	20	21						
11	7	5	4	12	13	15	14	10	11	9	8					
	24	25	27	26	30	31	29	28	20	21	23					
12	6	7	5	4	12	13	15	14	10	11	9	8				
	24	25	27	26	30	31	29	28	20	21	23	22				
13	2	6	7	5	4	12	13	15	14	10	11	9	8			
	24	25	27	26	30	31	29	28	20	21	23	22	18			
14	3	2	6	7	5	4	12	13	15	14	10	11	9	8		
	24	25	27	26	30	31	29	28	20	21	23	22	18	19		
15	1	3	2	6	7	5	4	12	13	15	14	10	11	9	8	
	24	25	27	26	30	31	29	28	20	21	23	22	18	19	17	
16	0	1	3	2	6	7	5	4	12	13	15	14	10	11	9	8
	24	25	27	26	30	31	29	28	20	21	23	22	18	19	17	16

The grey codes received by the Gen-Lock slave are then converted to a frame reference to tell the Gen-Lock slave which frame to work on. The slave modifies the Gen-Lock frame reference received by the frame delay such that the Gen-Lock slaves remains Frame Delay behind the Gen-Lock Master. Table 23 illustrates an example conversion from Gen-Lock Grey Code to Frame Reference used by the Gen-Lock Slave.

Table 23: **Example Grey Code Conversion for C_NUM_FSTORES = 5**

Grey To Frame Conversion	Grey Code Progressions for C_NUM_FSTORES = 5									
Grey Code	2	6	7	5	4	12	13	15	14	10
Decoded Frame Reference	0	1	2	3	4	0	1	2	3	4

Line Buffers

Optional line buffer can be utilized to prevent memory controller throttling from causing inner packet throttling on the stream interface. Line buffer depth is configurable by the user (via C_MM2S_BUFFER_DEPTH and C_S2MM_BUFFER_DEPTH) to provide the necessary buffering for a specific video application. For transmit (MM2S), this line buffering will present several clocks of latency at the onset of a video frame transfer between system memory read and initial tvalid on AXI4-Stream as the line buffer queues up a portion of the data to transfer.

Line buffer status is provided for video IP use. `mm2s_linebuffer_empty` and `mm2s_linebuffer_almost` empty are provided for the MM2S channel and `s2mm_linebuffer_full` and `s2mm_linebuffer_almost_full` are provided for the S2MM channel. Threshold parameters are provided to specify the assertion points for `mm2s_linebuffer_almost_empty` and `s2mm_linebuffer_almost_full`.

For MM2S, C_MM2S_LINE_BUFFER_THRESH specifies the assertion point for `mm2s_buffer_almost_empty`. The parameter is specified in bytes and must be a specified in even multiples of the MM2S AXIS data width in bytes (that is, C_M_AXIS_MM2S_DATA_WIDTH / 8 multiples). `mm2s_buffer_almost_empty` will assert when there are C_MM2S_LINE_BUFFER_THRESH bytes or less, excluding 0 bytes, left in the Line Buffer. For example if C_MM2S_LINE_BUFFER_THRESH = 8 then mm2s_buffer_almost empty will assert when the number of stored bytes is 8,7,6,5,4,3,2, or 1. When byte count = 0 then `mm2s_buffer_almost_empty` will deassert and `mm2s_buffer_empty` with assert.

To prevent throttling within a MM2S AXI4-Stream packet, the target Video IP is responsible for not asserting `m_axis_mm2s_tready = 1` until `mm2s_buffer_almost_empty` deasserts and `mm2s_buffer_empty = 0` indicating more than C_MM2S_LINE_BUFFER_THRESH data is stored in the line buffer. Adjustments may need to be made to the C_MM2S_LINE_BUFFER_THRESH value for a user's application to prevent throttling within packets.

For S2MM, C_S2MM_LINEBUFFER_THRESH specifies the assertion point of `s2mm_buffer_almost_full`. The parameter is specified in bytes and must be specified in even multiples of the S2MM AXIS data width in bytes (that is, C_S_AXIS_S2MM_DATA_WIDTH / 8 multiples). `s2mm_buffer_almost_full` will assert when there are C_S2MM_LINE_BUFFER_THRESH bytes or more in the Line Buffer. For example if C_MM2S_LINE_BUFFER_THRESH = 8 then `s2mm_buffer_almost_full` will assert when the number of stored bytes is 8 or greater. When line buffer is full then `s2mm_buffer_almost_full` will deassert and `s2mm_buffer_full` will assert. `s2mm_buffer_full` will assert when the last data beat of space is occupied. A data beat of space is C_S_AXIS_S2MM_DATA_WIDTH bits wide.

Example MM2S Timing

Figure 23 illustrates example timing on MM2S channel for C_USE_FSYNC = 1, Vertical Size = 5 lines, Horizontal Size = 128, bytes, and Stride = 128 bytes. The figure shows the `m_axi_mm2s` and `m_axis_mm2s` interfaces.

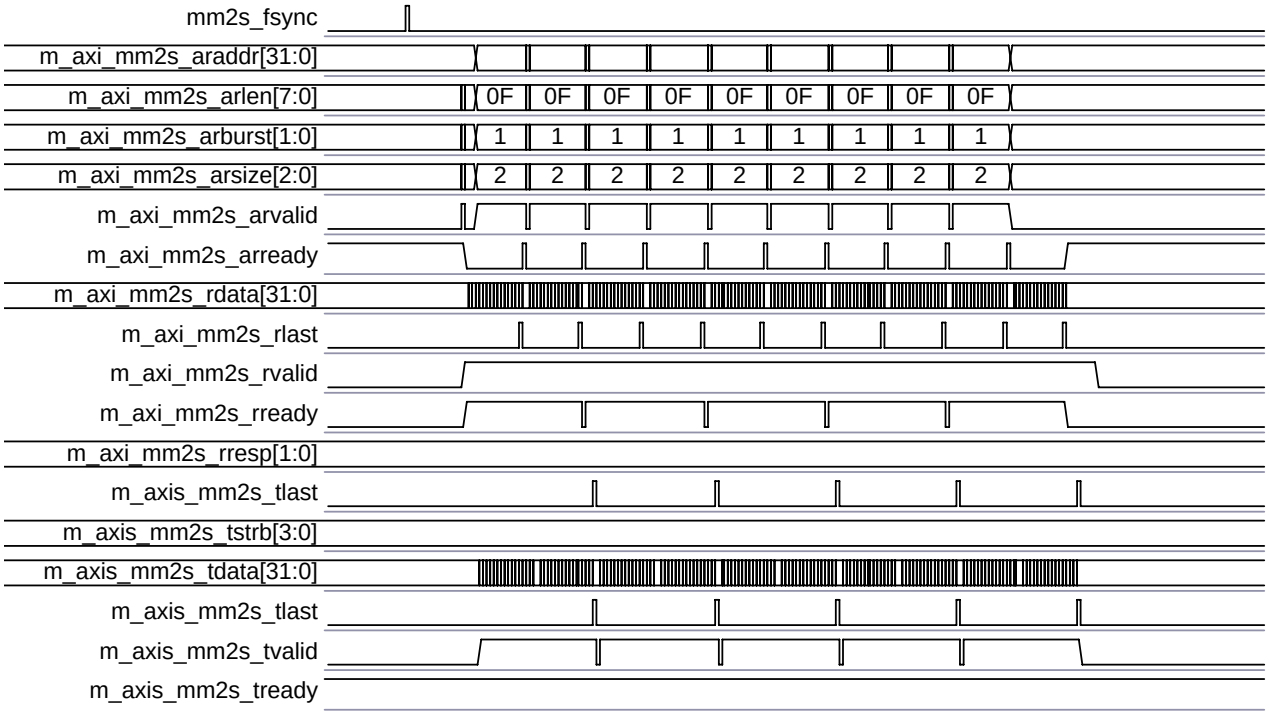

Figure 23: **Example MM2S Interface Timing**

Example S2MM Timing

Figure 24 illustrates example timing on S2MM channel for C_USE_FSYNC = 1, Vertical Size = 5 lines, Horizontal Size = 128, bytes, and Stride = 128 bytes. The figure shows the `m_axi_s2mm` and `s_axis_s2mm` interfaces.

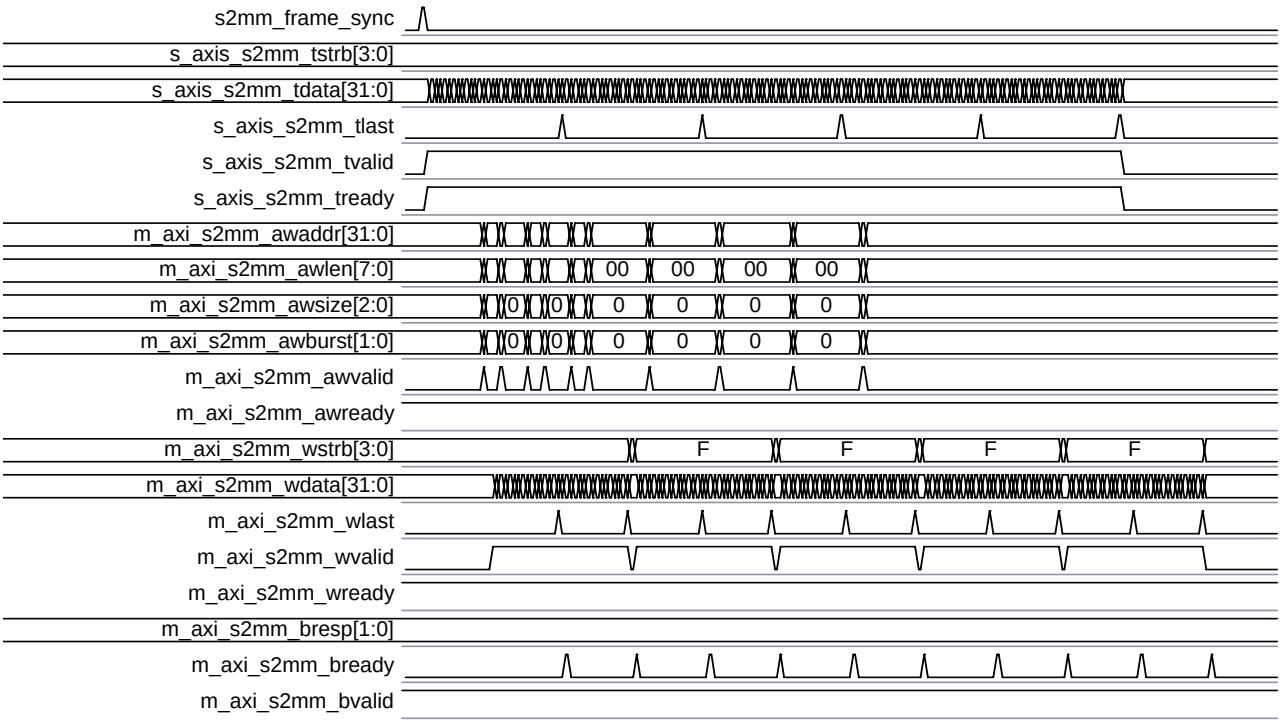

Figure 24: **Example S2MM Interface Timing**

Example Scatter Gather Timing

Figure 25 illustrates example timing for the Scatter Gather interface. The figure shows m_axi_sg interface timing with respect to m_axi_mm2s interface.

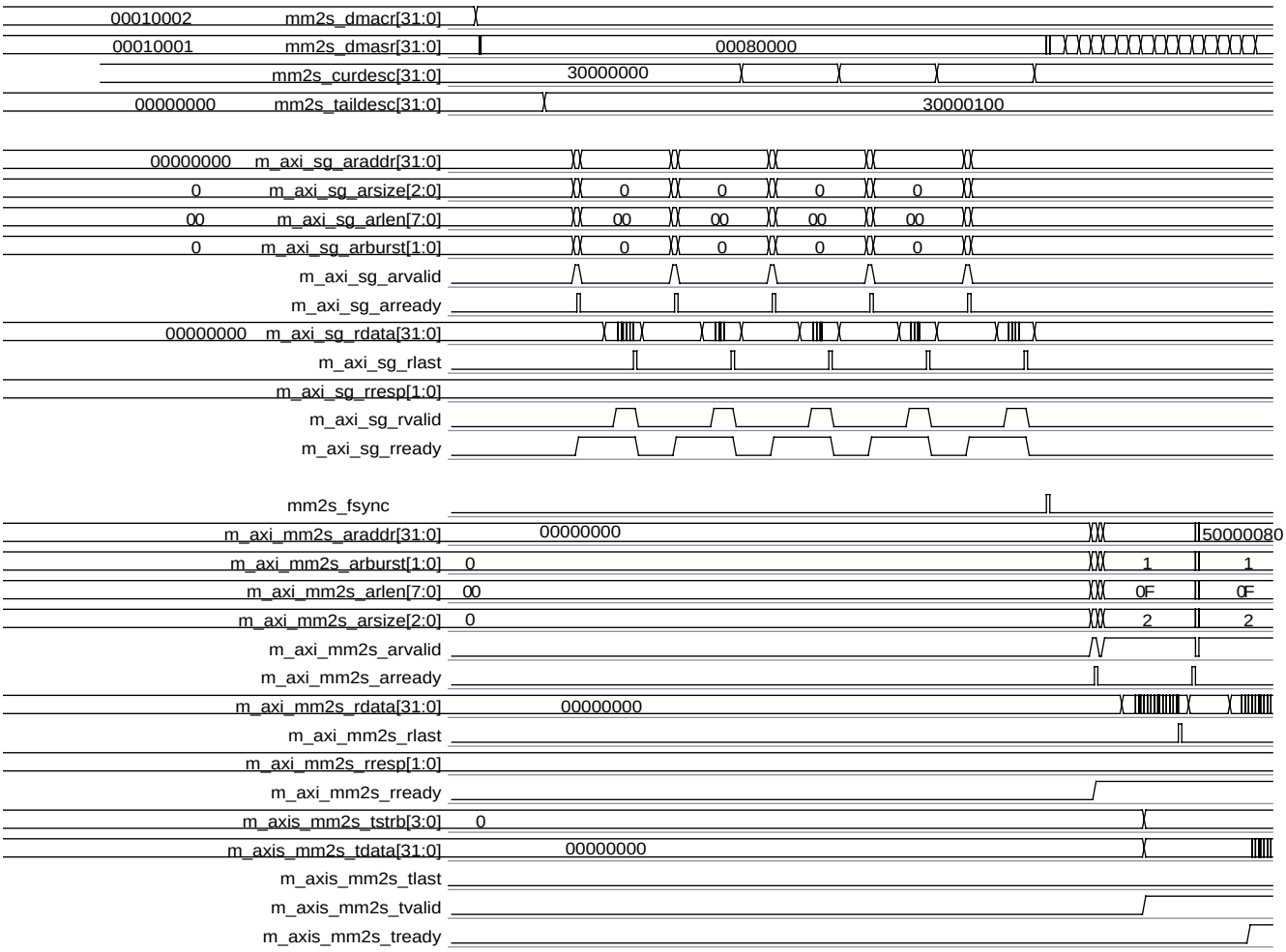

Figure 25: **Example Scatter Gather Timing**

Interrupt Controller

An interrupt output is provided for each channel (MM2S and S2MM). This output will drive high when the interrupt frame count is met, if there is a delay interrupt, or an error if the associated interrupt is enabled, as shown in Figure 26.

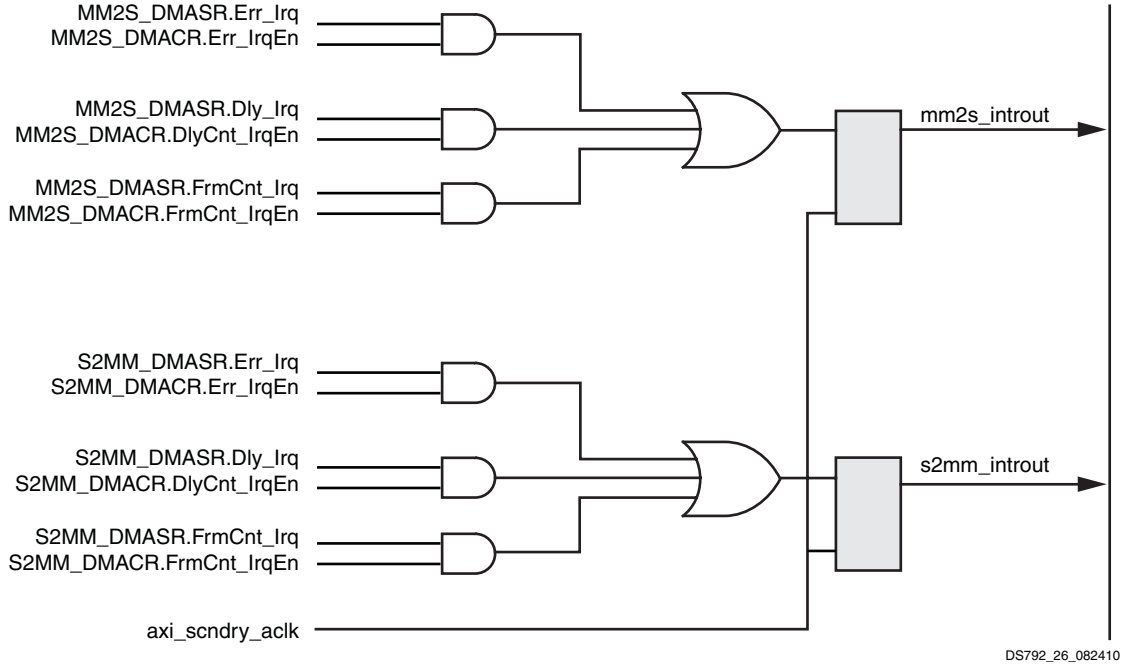

Figure 26: **Interrupt Out Concept**

Threshold Interrupt

Interrupt coalescing can be accomplished by setting the DMACR.IRQFrameCount field. With each frame completion event (that is, transmitted last line of frame or Received last line of frame when C_USE_FSYNC = 0 or on `mm2s_fsync` or `s2mm_fsync` for when C_USE_FSYNC = 1) the frame count will be decremented. When the count reaches zero, FrmCnt_Irq will assert and if FrmCnt_IrqEn = 1 then an interrupt will be generated on the associated channels introut signal (that is, `mm2s_introut` or `s2mm_introut`). When a frame count interrupt is generated the frame count is reloaded in the counter in preparation for the next frame count event. The internal frame count value is presented to software in DMASR.IRQFrameCntSts. A DMACR.IRQFrameCount value of 0x01 (default) cause a single frame count interrupt event to immediately generate an interrupt out.

If the delay interrupt is enabled (DMACR.IRQDelayCount not equal to 0 and DMACR.DlyIrq_En = 1) then a delay interrupt event will also reload the internal frame count counter. Finally with each software write of the frame count value (DMACR.IRQFrameCount) the internal counter is reloaded.

Figure 27 illustrates the functional composition of the interrupt threshold logic.

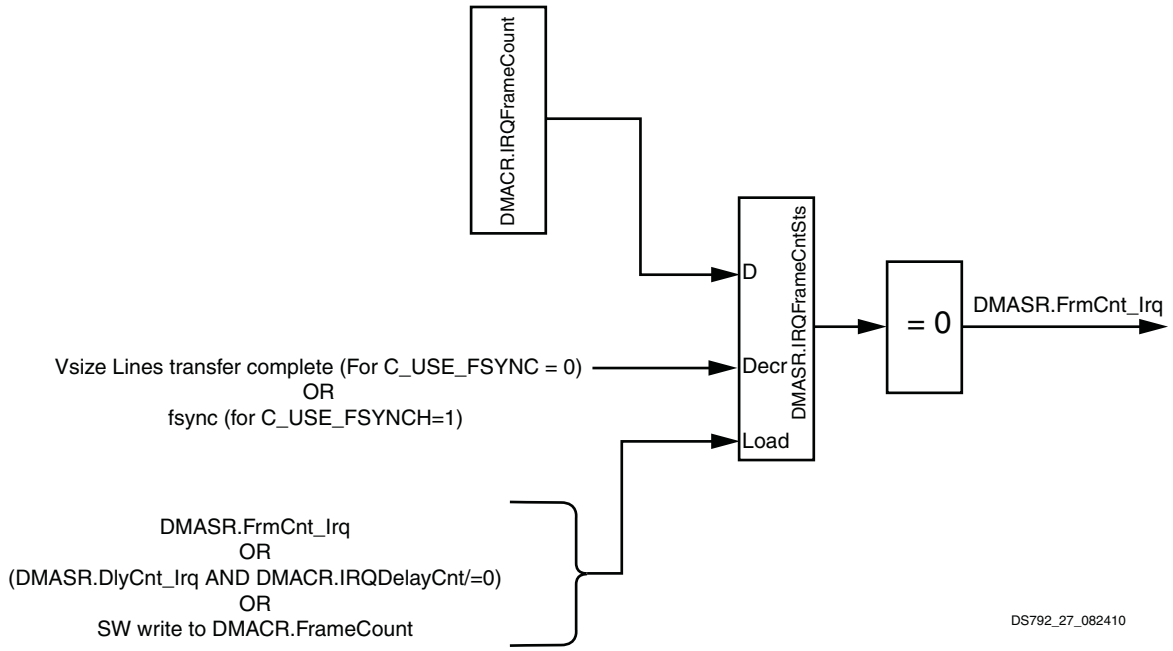

Figure 27: **Frame Count Interrupt Logic Concept**

Delay Interrupt

The delay interrupt is a mechanism by which software can receive an interrupt even when the frame count is not met. This is useful primarily for the S2MM (receive) channel for when receive data is sporadic. The software can still get an interrupt and service what data is received even if the frame count has not been decremented to zero. Figure 28 shows a high-level block diagram of the delay interrupt architecture.

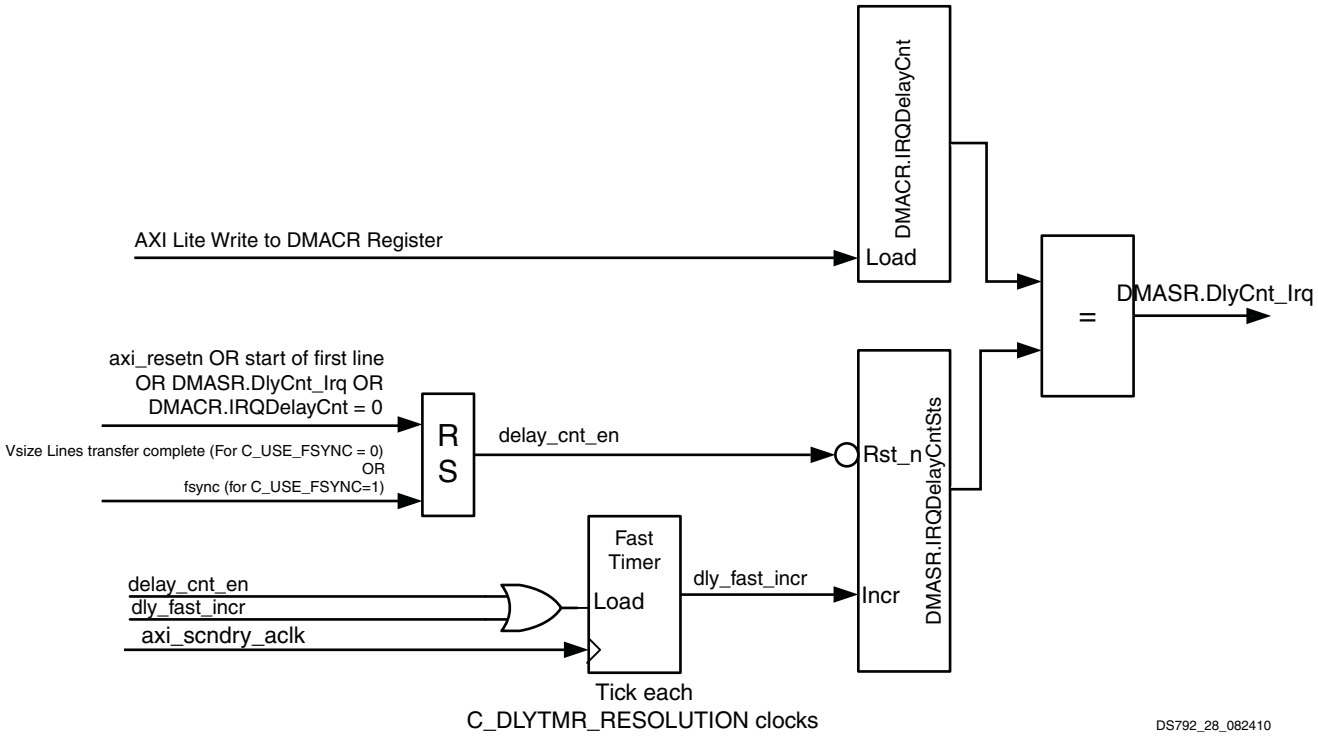

Figure 28: **Delay Interrupt Logic Concept**

The delay count interrupt is enabled by setting DMACR.IRQDelay value to a non-zero value. The delay counter will begin counting up either upon receipt of frame sync (`mm2s_fsync` or `s2mm_fsync`) for C_USE_FSYNC = 1 or the completion of the transfer of vsize lines for respective channel. The delay counter resets with each subsequent fsync or frame transfer completion. When a delay interrupt event occurs, the delay timer is reset to zero generation a IRQDelayCount event. If the DMACR.DlyCnt_IrqEn = 1 for the respective channel then an interrupt out will be generated from AXI VDMA. The delay timer will not count until the CPU services the interrupt by clearing the DMASR.DlyCnt_Irq bit to 0. Figure 29 shows example timing for this situation.

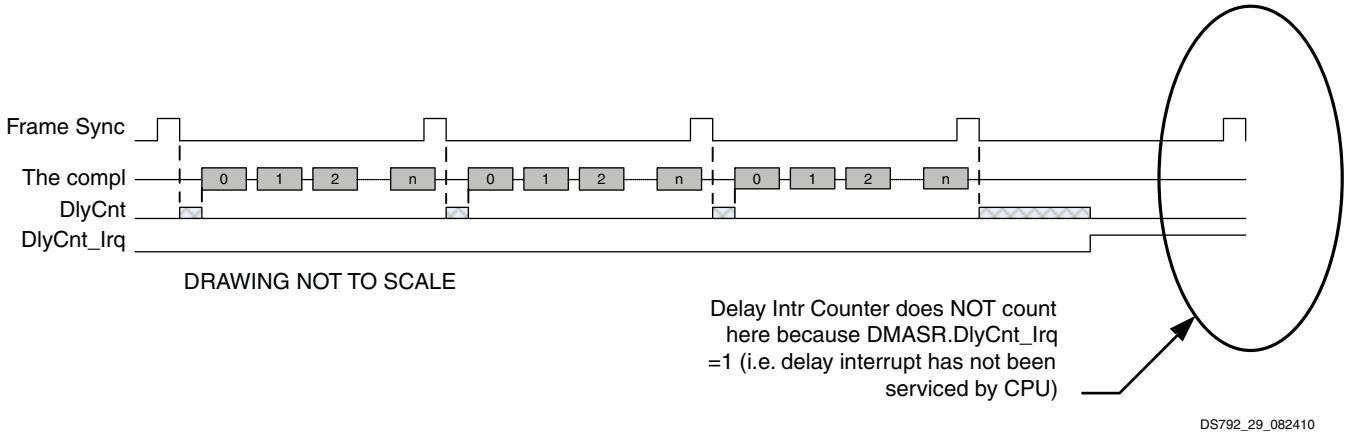

Figure 29: **Example Delay Timer Timing**

Errors

Any detected error on the primary data path (that is, DMAIntErr, DMASlvErr, and DMADecErr) will result in the associated channel (MM2S or S2MM) to gracefully halt. Any Scatter Gather Engine detected error (that is, SGSlvErr or SGDecErr) will cause the entire AXI VDMA engine to halt gracefully.

When an error is detected, the errored channel's DMACR.RS bit will be set to 0. Per AXI protocol all AXI transfers on memory map interfaces must complete, therefore the AXI VDMA will complete all pending transactions (transactions already posted and accepted on memory map interfaces) before setting the errored channel's DMASR.Halted bit. When the DMASR.Halted bit is set to 1 then the AXI VDMA channel is truly halted.

Furthermore for DMA detected errors the associated channel's FrmStore pointer (RdFrmStore or WrFrmStore) will be updated with the frame reference of the errored frame. For SG detected errors the descriptor associated with the errored transfer will be updated to the channel's CRNTDESC pointer register. If multiple, simultaneous errors are detected, then only one of the detected error's CRNTDESC will be updated.

In order to resume operations, a reset must be issued, either soft or hard. If using soft reset, then for Scatter gather errors a reset must be issue to both channels. For primary DMA errors the reset must be issued only to the channel that logged the error. Hard resets, reset the entire AXI VDMA engine.

The following is a list of possible errors:

DMAIntErr

DMA Internal Error flags an internal error in the AXI DataMover was detected. This can occur under two conditions. First, for MM2S and S2MM channels, it can occur when a BTT = 0 is written to the primary AXI DataMover. This would happen if a descriptor is fetched with the VSIZE or HSIZE = 0.

Secondly, for S2MM channels, the internal error can occur if an underflow or overflow condition occurs on the S2MM primary stream interface.

An underflow is defined as a condition where a received packet contains less bytes than what is specified by the hsize field in the descriptor.

An overflow is defined as a condition where a received packet contains more bytes than what is specified by the hsize field in the descriptor.

DMASlvErr

DMA Slave Error occurs when the slave to/from which data is transferred responds with a SLVERR on the memory map interface.

DMADecErr

DMA Decode Error occurs when the address request is targeted to an address that does not exist.

SGSlvErr

Scatter Gather Slave Error occurs when the slave from which descriptors are fetched responds with a SLVERR.

SGDecErr

Scatter Gather Decode Error occurs when the address request is targeted to an address that does not exist.

Error Priority

Table 24 shows the error priorities and when the errors might occur.

Table 24: **Error Priority**

Priority (1= highest priority)	Error	Occurrence
1	Fetch SGSlvErr	Cannot occur simultaneous with Fetch SGDecErr.
1	Fetch SGDecErr	Cannot occur simultaneous with Fetch SGSlvErr.
2	DMAIntErr (that is, zero vsize or hsize error)	Cannot occur simultaneous with DMASlvErr or DMADecErr for the descriptor with the zero vsize or hsize error. Cannot occur simultaneously with Fetch SGDecErr, Fetch SGSlvErr, or Fetch SGIntErr for the descriptor with the zero vsize or hsize error. Can occur simultaneously with errors detected for other descriptors not having zero vsize or hsize error.
3	DMAIntErr (that is, Overflow/Underflow)	Cannot occur simultaneous with zero vsize or hsize error. Cannot occur simultaneously with Fetch SGDecErr, Fetch SGSlvErr, for the descriptor associated with overflow or underflow. Can occur simultaneous with DMASlvErr or DMADecErr
3	DMADecErr	Cannot occur simultaneous with DMASlvErr.
3	DMASlvErr	Cannot occur simultaneous with DMADecErr

Clock Domains

AXI VDMA provides two clocking modes of operation: asynchronous and synchronous. Asynchronous mode DMA control (contained in secondary clock domain) can run asynchronously from the primary data path (contained in primary clock domain) and furthermore the two primary data paths can run asynchronously from each other. Setting C_PRMRY_IS_ACLK_ASYNC = 1 enables this mode and creates 3 clock domains, secondary clock domain clocked by axi_scndry_aclk, mm2s clock domain clocked by `axi_mm2s_aclk`, and s2mm clock domain clocked by `axi_s2mm_aclk` as shown in Figure 30. This allows high performance users to run the primary data path at a higher clock rate than the DMA control (AXI4-Lite interface, SG Engine, DMA Controller, etc.) helping in FPGA placement and timing.

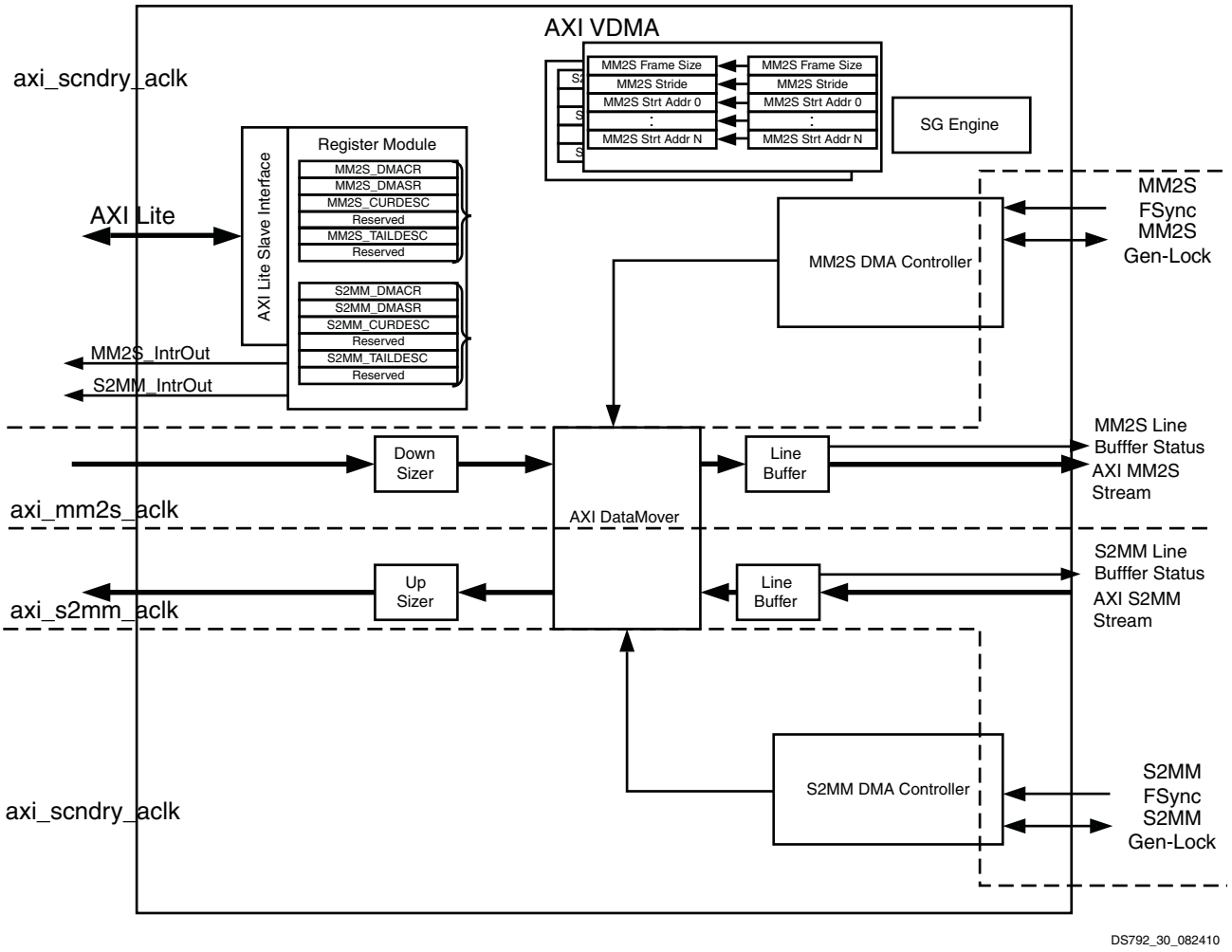

Figure 30: **Asynchronous Mode Clock Domains**

In synchronous mode, C_PRMRY_IS_ACLK_ASYNC = 0, all logic runs in a single clock domain. `axi_scndry_aclk`, `axi_mm2s_aclk`, and `axi_s2mm_aclk` must be tied to the same source otherwise undefined results will occur.

Performance

The following are typical clock frequencies for the target families. The maximum achievable clock frequency could vary. The maximum achievable clock frequency and all resource counts may be affected by other tool options, additional logic in the FPGA device, using a different version of Xilinx tools, and other factors.

Table 25: **System Performance**

Target FPGA	Target F_{MAX} (MHz)
Spartan-6	110
Virtex-6	180

Latency and Throughput

Table 26 and Table describes the latency and throughput for the AXI VDMA. The tables provides performance information for a typical configuration. Throughput test consisted of 8 video frames for each channel with each descriptor describing a 1000 lines at 1000 bytes per line per frame (~1MByte) and each channel operating simultaneously (full duplex). Systems included a loop back module for looping MM2S channel data back to S2MM channel data. Systems were configured for synchronous operation meaning `axi_scndry_aclk` = `axi_mm2s_aclk` = `axi_s2mm_aclk`. Throughput measured from completion of descriptor fetches (DMACR.Idle = 1) to frame count interrupt assertion. Latency measured in simulation and indicates AXI VDMA core latency cycles only and does not include system dependent latency or throttling.

System Configuration

- AXI VDMA Configuration
  - C_USE_FSYNC = 0
  - C_NUM_FSTORES = 8
  - C_M_AXI_MM2S_DATA_WIDTH = 32 and C_M_AXI_S2MM_DATA_WIDTH = 32
  - C_M_AXIS_MM2S_DATA_WIDTH = 32 and C_S_AXIS_S2MM_DATA_WIDTH = 32
  - C_MM2S_MAX_BURST_LENGTH = 16 and C_S2MM_MAX_BURST_LENGTH = 16
  - C_MM2S_GENLOCK_MODE = 0 and C_S2MM_GENLOCK_MODE = 0
  - C_MM2S_LINEBUFFER_DEPTH = 0 and C_S2MM_LINEBUFFER_DEPTH = 0
- ML605 System
  - Single Ported DDRx memory controller (Shared by CPU, AXI VDMA MM2S and S2MM, and AXI VDMA SG
- SP605 System
  - Multiported DDRx memory controller (One port shared by AXI VDMA MM2S and S2MM, one port dedicated AXI VDMA SG, and one port dedicated to CPU)

Table 26: **AXI VDMA Throughput**

AXI VDMA Channel	Primary Clock Frequency (axi_scndry_aclk)	Frame Size (Bytes)	Maximum Total Data Throughput (MBytes/sec)	Percent of Theoretical
Spartan-6 FPGAs				
MM2S Channel	80.00MHz	1MB	319.91	99.97%
S2MM Channel	80.00MHz	1MB	319.91	99.97%
Virtex-6 FPGAs				
MM2S Channel	151.51MHz	1MB	553.00	91.00%
S2MM Channel	151.51MHz	1MB	553.00	91.00%

AXI VDMA Latency (Free Run Mode)

Description	Clocks(1)
MM2S Channel	
Initial m_axi_mm2s_arvalid to m_axis_mm2s_tvalid	2
mm2s_fsync_out to m_axis_mm2s_tvalid	16
m_axis_mm2s_tlast to mm2s_fysnc_out	3
AXI4-Stream packet to packet latency (C_INCLUDE_MM2S_DRE = 0)	1
AXI4-Stream packet to packet latency (C_INCLUDE_MM2S_DRE = 1)	2
S2MM Channel	
Initial s_axis_s2mm_tvalid to m_axi_s2mm_awvalid	14
s2mm_fsync_out to m_axi_s2mm_awvalid	14
s_axis_s2mm_tlast to s2mm_fsync_out	10
AXI4-Stream packet to packet latency (C_INCLUDE_S2MM_DRE = 0)	1
AXI4-Stream packet to packet latency (C_INCLUDE_S2MM_DRE = 1)	2

1. Note: MM2S latency based on axi_mm2s_aclk and S2MM latency based on axi_s2mm_aclk.

Resource Utilization

Resources required for the AXI VDMA core have been estimated for the Spartan-6 FPGA (Table 27) and the Virtex-6 FPGA (Table 28) These values were generated using the Xilinx EDK tools v12.4. They are derived from post-synthesis reports and may change during MAP and PAR.

Table 27: **Spartan-6 FPGA Resource Estimates**

C_M_AXI_MM2S_DATA_WIDTH	C_M_AXI_S2MM_DATA_WIDTH	C_M_AXIS_MM2S_DATA_WIDTH	C_M_AXIS_S2MM_DATA_WIDTH	C_MM2S_MAX_BURST_LENGTH	C_S2MM_MAX_BURST_LENGTH	C_INCLUDE_MM2S_DRE	C_INCLUDE_S2MM_DRE	C_MM2S_LINEBUFFER_DEPTH	C_S2MM_LINEBUFFER_DEPTH	C_USE_FSYNC	C_NUM_FSTORES	C_MM2S_GENLOCK_MODE	C_S2MM_GENLOCK_MODE	Slices	Slice Reg	Slice LUTs	Block RAM
32	32	8	8	16	16	0	0	0	0	0	8	0	0	1705	3674	2351	0
32	32	32	32	16	16	0	0	0	0	0	8	0	0	1548	3773	2158	0
128	128	8	8	16	16	0	0	0	0	0	8	0	0	1949	4046	2941	0
128	128	128	128	16	16	0	0	0	0	0	8	0	0	1774	4937	2545	0
256	256	8	8	16	16	0	0	0	0	0	8	0	0	2180	4511	3525	0
256	256	256	256	16	16	0	0	0	0	0	8	0	0	2153	6460	3201	0
64	64	16	16	64	64	0	0	8192	8192	0	8	0	0	1949	4196	2849	10
64	64	32	32	128	128	0	0	32768	32768	0	8	0	0	2063	4521	3190	34
64	64	64	64	256	256	0	0	65536	65536	0	8	0	0	1896	4794	2803	66
64	64	64	64	32	32	1	1	1024	1024	1	1	0	0	2060	4387	3182	6
64	64	64	64	32	32	1	1	1024	1024	1	2	0	0	2123	4497	3240	6
64	64	64	64	32	32	1	1	1024	1024	1	3	0	0	2141	4606	3177	6
64	64	64	64	32	32	1	1	1024	1024	1	9	0	0	2374	5258	3293	6
64	64	64	64	32	32	1	1	1024	1024	1	15	0	0	2528	5906	3351	6
64	64	64	64	32	32	1	1	1024	1024	1	16	0	1	2549	6063	3444	6
64	64	64	64	32	32	1	1	1024	1024	1	16	1	0	2587	6063	3444	6

Table 28: **Virtex-6 FPGA Resource Estimates**

C_M_AXI_MM2S_DATA_WIDTH	C_M_AXI_S2MM_DATA_WIDTH	C_M_AXIS_MM2S_DATA_WIDTH	C_M_AXIS_S2MM_DATA_WIDTH	C_MM2S_MAX_BURST_LENGTH	C_S2MM_MAX_BURST_LENGTH	C_INCLUDE_MM2S_DRE	C_INCLUDE_S2MM_DRE	C_MM2S_LINEBUFFER_DEPTH	C_S2MM_LINEBUFFER_DEPTH	C_USE_FSYNC	C_NUM_FSTORES	C_MM2S_GENLOCK_MODE	C_S2MM_GENLOCK_MODE	Slices	Slice Reg	Slice LUTs	Block RAM
32	32	8	8	16	16	0	0	0	0	0	8	0	0	1772	3784	2472	0
32	32	32	32	16	16	0	0	0	0	0	8	0	0	1596	3887	2229	0
128	128	8	8	16	16	0	0	0	0	0	8	0	0	2011	4155	3070	0
128	128	128	128	16	16	0	0	0	0	0	8	0	0	1882	5043	2756	0
256	256	8	8	16	16	0	0	0	0	0	8	0	0	2302	4621	3632	0
256	256	256	256	16	16	0	0	0	0	0	8	0	0	2161	6571	3309	0
64	64	16	16	64	64	0	0	8192	8192	0	8	0	0	2056	4306	3001	6
64	64	32	32	128	128	0	0	32768	32768	0	8	0	0	2122	4627	3222	18
64	64	64	64	256	256	0	0	65536	65536	0	8	0	0	1955	4903	2897	34
64	64	64	64	32	32	1	1	1024	1024	1	1	0	0	2169	4476	3255	6
64	64	64	64	32	32	1	1	1024	1024	1	2	0	0	2171	4594	3317	6
64	64	64	64	32	32	1	1	1024	1024	1	3	0	0	2231	4712	3323	6
64	64	64	64	32	32	1	1	1024	1024	1	9	0	0	2501	5379	3401	4
64	64	64	64	32	32	1	1	1024	1024	1	15	0	0	2632	6075	3463	4
64	64	64	64	32	32	1	1	1024	1024	1	16	0	1	2683	6234	3490	4
64	64	64	64	32	32	1	1	1024	1024	1	16	1	0	2683	6234	3490	4

Support

Xilinx provides technical support for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx ISE® Design Suite Embedded Edition software under the terms of the Xilinx End User License. The core is generated using the Xilinx ISE Embedded Edition software (EDK).

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and software, please contact your local Xilinx sales representative.

Reference Documentation

DS768, *AXI Interconnect IP Data Sheet*

List of Acronyms

Acronym	Spelled Out
AXI	Advanced eXtensible Interface
CPU	Central Processing Unit
DDR	Double Data Rate
DMA	Direct Memory Access
DRE	Data Realignment Engine
DSP	Digital Signal Processing
EDK	Embedded Development Kit
FF	Flip-Flop
FIFO	First In First Out
FPGA	Field Programmable Gate Array
IP	Intellectual Property
ISE	Integrated Software Environment
LUT	Lookup Table
MM2S	Memory Map to Stream
R/WC	Read/Write to Clear
RAM	Random Access Memory
RO	Read Only
RW	Read/Write
S2MM	Stream to Memory Map
SG	Scatter Gather
VDMA	Video Direct Memory Access
XPS	Xilinx Platform Studio (part of the EDK software)
XST	Xilinx Synthesis Technology

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
09/21/10	1.0	First release of the core with AXI interface support.
12/14/10	2.0	Updated for v2.00. a core version; updated for 12.4 tools.

Notice of Disclaimer