Audio Formatter v1.0

LogiCORE IP Product Guide

Vivado Design Suite

PG330 (v1.0) July 8, 2020

XILINX®

Table of Contents

Chapter 1: Introduction 4

Features 4

IP Facts 5

Chapter 2: Overview 6

Feature Summary 6

Limitations 7

Licensing and Ordering 8

Chapter 3: Product Specification 9

Performance 10

Resource Use 10

Port Descriptions 10

Register Space 11

Chapter 4: Designing with the Core 20

Clocking 24

Resets 25

Chapter 5: Design Flow Steps 26

Customizing and Generating the Core 26

Constraining the Core 30

Synthesis and Implementation 31

Chapter 6: Example Design 32

Implementing the Example Design 33

Example Design Directory Structure 33

Simulating the Example Design 34

Appendix A: Upgrading 35

Appendix B: Debugging 36

Finding Help on Xilinx.com 36

Debug Tools..37

Appendix C: Additional Resources and Legal Notices............................ 38

Xilinx Resources..38

Documentation Navigator and Design Hubs..38

References..38

Revision History..39

Please Read: Important Legal Notices..39

Introduction

The Xilinx® LogiCORE™ IP Audio Formatter core is a soft Xilinx IP core for use with the Vivado® Design Suite. The Audio Formatter provides high-bandwidth direct memory access between memory and AXI4-Stream target peripherals supporting audio data.

Features

- Supports 2, 4, 6, or 8 audio channels
- Supports 8, 16, 20, 24, and 32 bits PCM data width
- Independent S2MM (write to Memory) and MM2S (read from memory) operations
- Supports Interleaved and Non-interleaved modes of data packaging in memory
- Supported data formats while write from Audio stream to Memory buffer (S2MM):
  - AES to AES
  - AES to PCM (includes AES Decoder)
  - PCM to PCM
- Supported data formats while reading from memory Buffer as audio stream (MM2S):
  - AES to AES
  - AES to PCM
  - PCM to PCM
  - PCM to AES (Includes AES Encoder)
- Supports timeout feature in case of S2MM
- Interrupt generation on completion of every Period size of data or when an error occurs
- Support to pad 0s in case missing samples from some channels in S2MM
- Can handle random order of audio channels in incoming stream in S2MM
- Supports graceful halt and soft reset by completion of pending AXI4 transactions.
- Supports selection of synchronous or asynchronous clocks through GUI interface

IP Facts

LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ UltraScale™ Zynq® UltraScale+™ MPSoC 7 series
Supported User Interfaces	AXI4, AXI4-Lite, AXI4-Stream
Resources	Performance and Resource Use web page
Provided with Core	
Design Files	Register Transfer Level (RTL)
Example Design	Verilog
Test Bench	Verilog
Constraints File	Xilinx Constraints File
Simulation Model	Not Provided
Supported S/W Driver[2]	Standalone and Linux
Tested Design Flows[3]	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado® synthesis
Support	
Release Notes and Known Issues	Master Answer Record: 54489
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775
Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado® IP catalog.
2. Standalone driver information can be found in the Software Development Kit (SDK) (SDK) installation directory. See `xilinx_drivers.htm` in `<install_directory>/SDK/<release>/data/embeddedsw/doc/xilinx_drivers.htm`. Linux OS and driver support information is available on the Xilinx Wiki. For reference, see AXI DMA.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Chapter 2

Overview

The LogiCORE™ IP Audio Formatter (Audio DMA) core is a soft IP core for use with Vivado® Design Suite. The Audio Formatter provides high-bandwidth direct memory access between memory and AXI4-Stream target peripherals.

Feature Summary

The Audio Data can be saved in the memory as PCM data or in AES format. PCM is the actual Audio sample data. AES format has sideband signals added to PCM data as defined in AES-3 Specification.

Note: Refer to *HDMI 1.4/2.0 Transmitter Subsystem Product Guide* (PG235) for details on AES-3 Specification.

AES[3:0] are the LSB sideband 4-bits and AES[7:4] are the MSB sideband 4-bits as per the AES specification.

Table 1: **Supported Data Formats In Memory**

PCM Data width	Format Name	Memory Format
8	SND_PCM_FORMAT_IEC958_SUBFRAME_LE_S8	AES[7:4] 8’bPCM 16’b0 AES[3:0] AES[7:4] 8’bPCM 16’b0 AES[3:0]
8	SND_PCM_S8_LE	B3 B2 B1 B0 B7 B6 B5 B4[1]
16	SND_PCM_FORMAT_IEC958_SUBFRAME_LE_S16	AES[7:4] 16’bPCM 8’b0 AES[3:0] AES[7:4] 16’bPCM 8’b0 AES[3:0]

Table 1: **Supported Data Formats In Memory** *(cont'd)*

PCM Data width	Format Name	Memory Format
16	SND_PCM_S16_LE	B1 | B0 B3 | B2[1]
20	SND_PCM_FORMAT_IEC958_SUBFRAME_LE_S20	AES[7:4] 20’bPCM 4’b0 AES[3:0] AES[7:4] 20’bPCM 4’b0 AES[3:0]
20	SND_PCM_S20_LE	12’b0 20’bPCM 12’b0 20’bPCM
24	SND_PCM_FORMAT_IEC958_SUBFRAME_LE_S24	AES[7:4] 24’bPCM AES[3:0] AES[7:4] 24’bPCM AES[3:0]
24	SND_PCM_S24_LE	8’b0 | B2 B1 B0 8’b0 | B5 B4 B3[1]
32	SND_PCM_S32_LE	B3 B2 B1 B0 B7 B6 B5 B4[1]

Notes:

1. B0, B1, B2 indicates each byte in the PCM data. For example if PCM data width is 24 bits, it is depicted as 3-byte data, B2B1B0 in the table.

- Optional interrupts on transfer of a period of data or on error detection.
- Selection of non-interleaved mode or interleaved mode through the GUI.

Limitations

- No support for non-aligned Buffer address. The Buffer address must be aligned to Memory Map width.
- Period size must be integer multiples of (32 bytes * # of valid channels). For example:
    - Number of valid channels = 4

  - Period size must at least be 4*8*(32 bits) = 4*32 bytes
  - Period size must integer multiples of 4*32 bytes = 128 bytes, 256bytes, 384 bytes, etc.
- When calculating, use 32 as the the period size for PCM data widths of 20, 24 and 32, because they occupy 32 bits space in memory.
- No support for PCM sample data width conversion.

Licensing and Ordering

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License.

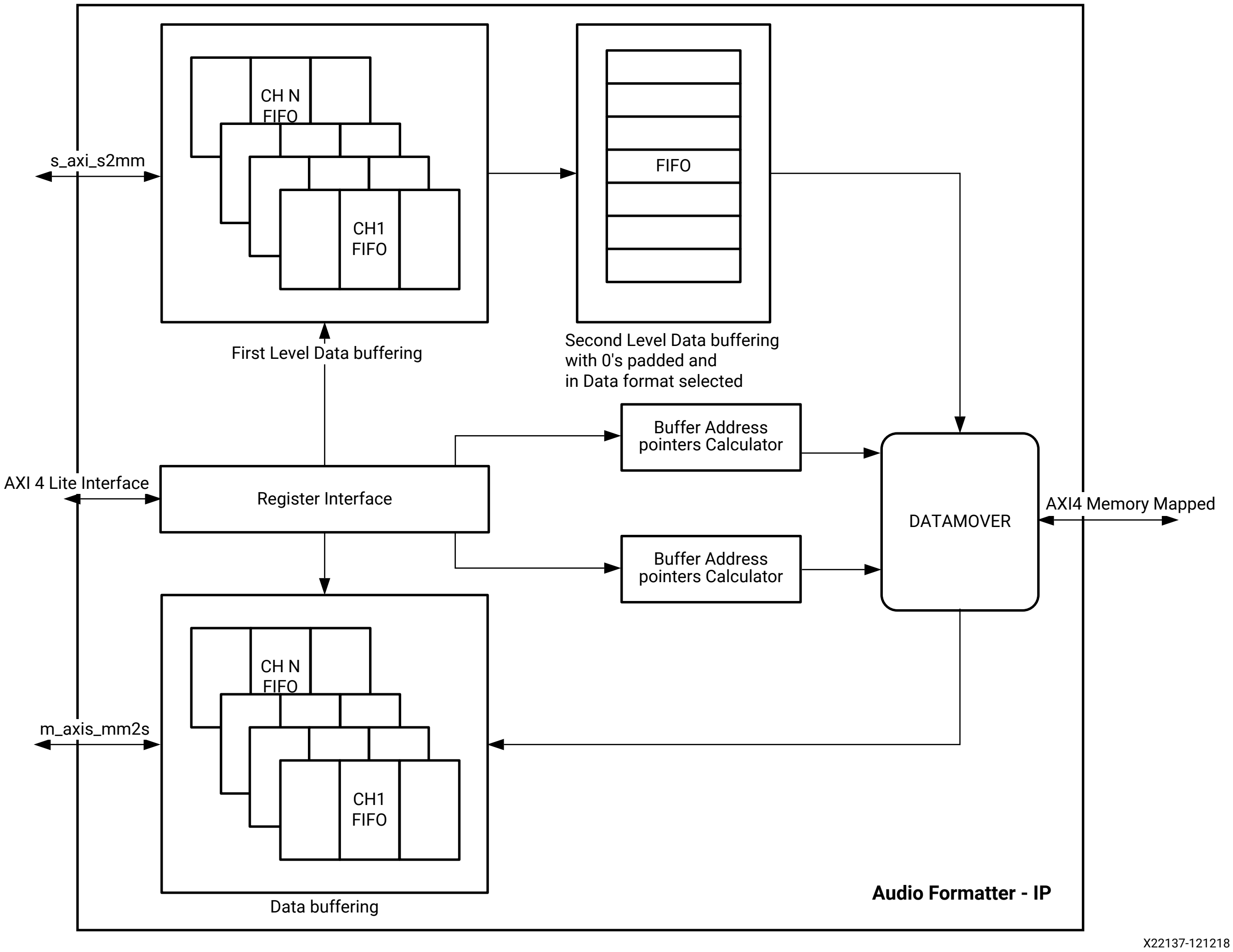
Information about other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

Initialization, status, and management registers are accessed through an AXI4-Lite slave interface.

The functional block diagram of the core is shown in the following figure.

Figure 1: **Core Block Diagram**



Performance

For full details about performance and resource use, visit the Performance and Resource Use web page.

Resource Use

For full details about performance and resource use, visit the Performance and Resource Use web page.

Port Descriptions

Signal Name	Interface	Signal Type	Description
AXI4-Lite Interface Signals			
s_axi_lite_aclk	Clock	I	AXI4-Lite clock for register interface
s_axi_lite_aresetn	Reset	I	AXI4-Lite reset synchronous to s_axi_lite_aclk
s_axi_lite_*	AXI4-Lite	-	AXI4-Lite interface ports for register programming
S2MM Interface			
s_axis_s2mm_aclk	Clock	I	Clock for AXI4 and stream interfaces of S2MM
s_axis_s2mm_aresetn	Reset	I	Synchronous reset for s2mm_aclk
s_axis_s2mm_*	AXI4-Stream[1]	-	Audio AXI4 Streaming input for S2MM data[1]
m_axi_s2mm_*	AXI4 MM	-	AXI4 MM interface ports to write data to memory in S2MM
Irq_s2mm	Signal	O	S2MM Interrupt signal to indicate period transmission and error conditions
MM2S Interface			
m_axis_mm2s_aclk	Clock	I	Clock for AXI4 memory mapped and stream interfaces of MM2S
m_axis_mm2s_aresetn	Reset	I	Synchronous reset for mm2s_aclk
aud_mclk	Clock	I	Audio Master clock for streaming MM2S data at sampling rate
aud_mreset	Reset	I	Active-High Reset corresponding to aud_mclk
m_axi_mm2s_*	AXI4 MM	-	AXI4 MM interface ports to read data from memory in MM2S
m_axis_mm2s_*	AXI4-Stream[1]	-	AXI4-Stream output in MM2S

Signal Name	Interface	Signal Type	Description
Irq_mm2s	Signal	O	MM2S Interrupt signal to indicate period transmission and error conditions

Notes:

1. The streaming interface corresponds to the AXI4-Stream Audio Interface. It consists of TDATA, TID, TREADY, and TVALID signals. More details on the interface can be found in *HDMI 1.4/2.0 Transmitter Subsystem Product Guide* (PG235).

Register Space

The Audio Formatter IP registers are memory-mapped into non-cacheable memory space. This memory space must be aligned on an AXI word (32-bit) boundary.

Note: The AXI4-Lite write access register is updated by the 32-bit AXI Write Data (`*_wdata`) signal, and is not impacted by the AXI Write Data Strobe (`*_wstrb`) signal. For a Write, both the AXI Write Address Valid (`*_awvalid`) and the AXI Write Data Valid (`*_wvalid`) signals should be asserted together.

Table 2: **Audio Formatter Register Address Space**

Address (hex)	Register
0x00	**Core Version**: Returns the core Major and Minor version
0x04	**Core Configuration**: Returns the core configuration details
0x10	**S2MM Control**: S2MM Control options
0x14	**S2MM Status**: Status of S2MM DMA transfer
0x18	**S2MM Timeout**: Timeout count value
0x1C	**S2MM Period Config**: Register to program period size and no. of periods
0x20	**S2MM Buffer Address LSB**: Buffer Start Address LSB
0x24	**S2MM Buffer Address MSB**: Buffer Start Address MSB
0x28	**S2MM Transfer Count**: Count of data transferred to memory
0x2C – 0x40	**S2MM Channel status bits**: Read only Channel status bits
0x44	**S2MM Channel Offset**: Bytes per channel in a period
0x110	**MM2S Control**: MM2S Control Options
0x114	**MM2S Status**: Status of the MM2S DMA transfer
0x118	**MM2S Fs Multiplier**: Register to specify Fs multiplier
0x11C	**MM2S Period Config**: Register to program period size and number of periods
0x120	**MM2S Buffer Address LSB**: Buffer Start Address LSB
0x124	**MM2S Buffer Address MSB**: Buffer Start Address MSB
0x128	**MM2S Transfer Count**: Count of data read from memory
0x12C – 0x140	**AES Encode Channel Status**: Channel Status Bits to Embed
0x144	**MM2S Channel Offset**: Bytes per channel in a period

Core Version Register (0x00)

Bit	Default Value	Access Type	Description
31:24	0x1	RO	**Major Version**: This is the IP major version value. For example, if the IP version is 1.2, then this will return a value of 1.
23:16	0x0	RO	**Minor Version**: This is the IP minor version value. For example, if the IP version is 1.2, then this will return a value of 2.
15:8	0x0	RO	**Revision**: Revision of minor version
7:0	0x0	RO	Reserved

Core Configuration Register (0x04)

Bit	Default Value	Access Type	Description
31	0x1	RO	**S2MM_Included** • 0: S2MM not part of IP • 1: S2MM is included
30:29	0x1	RO	**S2MM_Dataformat** • 0: AES -> AES • 1: AES -> PCM • 2: PCM -> PCM
28	0x0	RO	**S2MM_packaging_mode** • 0: Interleaved • 1: Non-interleaved
27:24	0x2	RO	**Max_channels_S2MM**: Indicates the maximum number of channels supported for S2MM.
23:16	0x0	RO	Reserved
15	0x1	RO	**MM2S_Included** • 0: MM2S not part of IP • 1: MM2S is included
14	0x3	RO	**MM2S_Dataformat** • 0: AES -> AES • 1: AES -> PCM • 2: PCM -> PCM • 3: PCM -> AES

Bit	Default Value	Access Type	Description
12	0x0	RO	**MM2S_packaging_mode** • 0: Interleaved • 1: Non-interleaved
11:8	0x2	RO	**Max_channels_MM2S**: Indicates the maximum number of channels supported for MM2S.
7:0	0x0	RO	Reserved

S2MM Control Register (0x10)

Bit	Default Value	Access Type	Description
31:23	0x0	-	Reserved
22:19	0x2	R/W	**Number of valid channels**: Valid values are 2,4,6,8. This value must be less than or equal to maximum no. of channels programmed through GUI.
18:16	0x2	R/W	**PCM data width**: PCM data width to write to memory. • 0x0: 8 bits • 0x1: 16 bits • 0x2: 20 bits • 0x3: 24 bits • 0x4: 32 bits (Complete AES)
15	-	-	Reserved
14	0x1	R/W	**Timeout_IrqEn**: Enables interrupt on timeouts. • 1: Enables interrupt when timeout occurs • 0: Disables interrupt
13	0x1	R/W	**IOC_IrqEn**: Enables interrupt on completion. • 1: Enables interrupt after each period size of data is transferred • 0: Disables interrupt
12	0x0	R/W	**Err_IrqEn**: Enables interrupt on error.
11:2	-	-	Reserved
1	0x0	R/W	**Reset**: Soft reset for resetting the S2MM core. Setting this bit to a 1 causes the S2MM part of the IP to be reset. Reset is accomplished gracefully. Pending commands/ transfers are flushed or completed. AXI4-Stream outs are terminated early. After completion of a soft reset, all registers and bits are in the Reset State. • 0 = Reset not in progress. Normal operation. Out of reset. • 1 = Reset in progress.

Bit	Default Value	Access Type	Description
0	0x0	R/W	Run/Stop control for controlling running and stopping of the S2MM channel. • 0 = Stop. S2MM part of the IP stops when current (if any) DMA operations are complete. Pending commands/transfers are flushed or completed. AXI4-Streams are potentially terminated. Data integrity on S2MM AXI4 cannot be guaranteed. The halt_in_process bit (Bit 0) in S2MM Status register goes from 1 to 0 when the halt process is complete. • 1 = Run. Start S2MM operations.

S2MM Status Register Control Register (0x14)

Bit	Default Value	Access Type	Description
31	0x0	W1C	**IOC_Irq**: Interrupt on complete (IOC). Set with each period size of data transferred. • 0 = No IOC Interrupt • 1 = IOC Interrupt detected. Writing a 1 to this bit clears it.
30	0x0	W1C	**Err_Irq**: Interrupt on Error. When set to 1, indicates an interrupt event was generated on error. If the corresponding bit in S2MM_DMACR is enabled (Err_IrqEn = 1), an interrupt out is generated from the IP. Writing a 1 to this bit clears it.
29	0x0	R/C	**Channel status detected**: Indication whenever the channel status bits are updated in the registers 0x2C to 0x40. This bit is cleared on reading.
28:20	-	-	Reserved
19	0x0	RO	**Timeout_Error**: Error when there is no input audio stream till counter hits timeout value. The core must be reset to ensure proper functionality. • 0 = No Timeout • 1 = Input timeout
18	0x0	RO	**S2MM Decode Error**: This error occurs if the address request points to an invalid address. This bit is cleared only with a hard or soft reset to S2MM core. • 0 = No DMA Decode Errors • 1 = DMA Decode Error detected
17	0x0	RO	**S2MM Slave Error**: DMA Slave Error. This error occurs if the slave read from the Memory Map interface issues a Slave Error. This bit is cleared only with a hard or soft reset to S2MM core. • 0 = No DMA Slave Errors. • 1 = DMA Slave Error detected.
16:1	-	-	Reserved

Bit	Default Value	Access Type	Description
0	0x0	RO	**Halt in process**: S2MM Channel Halt in process indication. Asserted when DMA Run Stop bit goes from 1 to 0 till all the transactions are gracefully completed. • 1 = Halting in process • 0 = Either running state or DMA operations have stopped

S2MM Timeout Register (0x18)

Bit	Default Value	Access Type	Description
31:0	0x80000000	R/W	**Timeout Counter**: Timeout value after which all the data is flushed to memory if there is no incoming data from any channel. This counter runs on s_axis_s2mm_aclk

S2MM Period Config Register (0x1C)

Bit	Default Value	Access Type	Description
31:24	0x0	-	Reserved
23:16	0x1	R/W	**Number of periods**: Number of periods of data to transfer.
15:0	0x0	R/W	**Period Size**: The number of bytes of data in each period.

S2MM Buffer Address LSB Register (0x20)

Bit	Default Value	Access Type	Description
31:0	0x0	R/W	**Buffer Start Address LSB**: LSB 32 bits of buffer start Address.

S2MM Buffer Address MSB Register (0x24)

Bit	Default Value	Access Type	Description
31:0	0x0	R/W	**Buffer Start Address MSB**: If buffer start address is greater than 32 bits. MSB 32 bits are programmed here. ***Note***: The buffer address should be MMap data width aligned.

S2MM Transfer Count Register (0x28)

Bit	Default Value	Access Type	Description
31:25	-	-	Reserved
24:0	0x0	RO	**DMA transfer Count**: Number of bytes of data written to the memory. Count refreshes when a buffer size of data is transferred.

S2MM Channel Status Bits Register (0x2C – 0x40)

These 6 registers together give the 192-bit Channel Status Information that is received over the Audio stream. A write to any of the six registers would clear all the six registers.

Bit	Default Value	Access Type	Description
31:0	0	RO	**32bit AES value**: 32-bit AES Channel Status value.

The 6 registers give the value in order of LSB to MSB. The register 0x2C returns the bits [31:0] of 192-bit channel status, while register 0x40 returns the bits [191:160]. This value is always taken from Channel0.

S2MM Channel Offset Register (0x44)

Bit	Default Value	Access Type	Description
31:16	-	-	Reserved
15:0	0x0	R/W	**Channel Offset**: Number of bytes of data of each channel in a period. Channel offset = period size/number of valid channels

MM2S Control Register (0x110)

Bit	Default Value	Access Type	Description
31:23	0x0	-	Reserved
22:19	0x2	R/W	**Number of valid channels**: Valid values are 2,4,6,8. This value must be less than or equal to maximum no. of channels programmed through GUI.

Bit	Default Value	Access Type	Description
18:16	0x2	R/W	**PCM data width**: PCM data width of data present in the memory. • 0x0 : 8 bits • 0x1 : 16 bits • 0x2 : 20 bits • 0x3 : 24 bits • 0x4 : 32 bits (Complete AES)
15:14	-	-	Reserved
13	0x1	R/W	**IOC_IrqEn**: Enable interrupt on complete (IOC). • 1: Enables Interrupt after each period of data is transferred • 0: Disables interrupt
12	0x0	R/W	**Err_IrqEn**: Enables interrupt on error.
11:2	-	-	Reserved
1	0x0	R/W	**Reset**: Soft reset for resetting the MM2S core. Setting this bit to a 1 causes the MM2S core to be reset. Reset is accomplished gracefully. Pending commands/transfers are flushed or completed. AXI4-Stream outs are terminated early. After completion of a soft reset, all registers and bits are in the Reset State. • 0 = Reset not in progress. Normal operation/Out of reset. • 1 = Reset in progress.
0	0x0	R/W	**Run/Stop**: Run/Stop control for controlling running and stopping of the DMA channel. • 0 = Stop – MM2S stops when current (if any) DMA operations are complete. Pending commands/transfers are flushed or completed. AXI4-Streams are potentially terminated. The halt in process bit in MM2S Status register shows the status of halt. • 1 = Run – Start MM2S operations.

MM2S Status Register (0x114)

Bit	Default Value	Access Type	Description
31	0x0	R/W1C	**IOC_Irq**: Interrupt on Complete. When set to 1, an interrupt event was generated on the completion of transferring one period size of data.
30	0x0	R/W1C	**Err_Irq**: Interrupt on Error. When set to 1, indicates an interrupt event was generated on error. If the corresponding bit in MM2S_DMACR is enabled (Err_IrqEn = 1), an interrupt out is generated from the IP. Writing a 1 to this bit clears it.
29:19	-	-	Reserved

Bit	Default Value	Access Type	Description
18	0x0	RO	**MM2S Decode Error**: DMA Decode Error. This error occurs if the address request points to an invalid address. After it has been set, it is only cleared with a reset. • 0 = No DMA Decode Errors • 1 = DMA Decode Error detected
17	0x0	RO	**MM2S Slave Error**: DMA Slave Error. This error occurs if the slave read from the Memory Map interface issues a Slave Error. After it has been set, it is only cleared with a reset. • 0 = No DMA Slave Errors • 1 = DMA Slave Error detected
16:1	-	-	Reserved
0	0x1	RO	**Halt in process**: DMA Channel Halt in process indication. Asserted when DMA Run Stop bit goes from 1 to 0 till all the transactions are gracefully completed. • 1 = Halting in process • 0 = Either running state or DMA operations have stopped

MM2S Fs Multiplier Register (0x118)

Bit	Default Value	Access Type	Description
31:16	-	-	Reserved
15:0	0x180	R/W	**Sampling Frequency (Fs) multiplier value**: Specify the multiplier value of the aud_mclk. For example, for Fs = 48Khz, the supplied aud_clk is 18.432Mhz, then this register should be programmed with a value of 384. For example, 48Khz * 384 = 18.432Mhz.

MM2S Period Config Register (0x11C)

Bit	Default Value	Access Type	Description
31:24	0x0	-	Reserved
23:16	0x1	R/W	**Number of periods**: Number of periods of data to transfer.
15:0	0x0	R/W	**Period Size**: The number of bytes of data in each period.

MM2S Buffer Address LSB (0x120)

Bit	Default Value	Access Type	Description
31:0	0x0	R/W	**Buffer Start Address LSB**: LSB 32 bits of buffer start Address.

MM2S Buffer Address MSB (0x124)

Bit	Default Value	Access Type	Description
31:0	0x0	R/W	**Buffer Start Address MSB**: If buffer start Address is greater than 32 bits. MSB 32 bits are programmed here. ***Note***: The buffer address should be MMap data width aligned.

MM2S Transfer Count Register (0x128)

Bit	Default Value	Access Type	Description
31:25	-	-	Reserved
24:0	0x0	RO	**DMA transfer Count**: Number of bytes of data read from memory. Refreshes with every buffer size

AES Encode Channel Status Register (0x12C – 0x140)

These 6 registers together give the 192-bit Channel Status Information to embed over the Audio PCM data available in the memory to send them as AES data out when AES encoder is included.

Bit	Default Value	Access Type	Description
31:0	0	R/W	**32bit AES value**: 32-bit AES Channel Status value.

The 6 registers give the value in order of LSB to MSB. The register 0x12C corresponds to the bits [31:0] of 192-bit channel status, while register 0x140 corresponds to the bits [191:160]. This value is programmed onto all the channels.

MM2S Channel Offset Register (0x144)

Bit	Default Value	Access Type	Description
31:16	-	-	Reserved
15:0	0x0	R/W	**Channel Offset**: Number of bytes of data of each channel in a period. Channel offset = period size/no. of valid channels

Designing with the Core

This section includes guidelines and additional information to facilitate designing with the core.

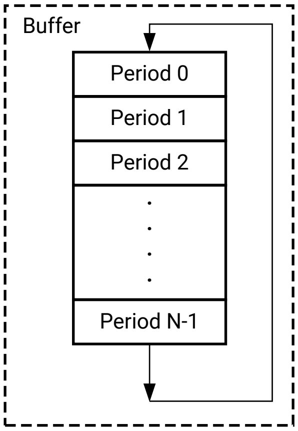
Memory Structure

The Audio Formatter writes or reads from memory, assuming a cyclic buffer with programmed number of periods. S2MM and MM2S are completely independent and the number of periods, period size, and buffer addresses can vary for both.

The core starts writing to the buffer start address as programmed, and comes back to the buffer start address after finishing all the periods. The buffer start address can be modified while DMA is still in operation, but the new address is sampled only after finishing all the transfers, which is the period size of data multiplied by the number of periods programmed (i.e., size of buffer).

In Interleaved mode, the data from all the channels is ordered from 0 to Max number of channels, and placed in memory at consecutive address locations. Whereas in Non-Interleaved mode, all the data from each channel is placed together in order in a period.

Figure 2: **Buffer**



X22138-121218

The following figure shows the alignment of data at consecutive addresses in a period where L1, L2, L3, and L4 are channel 0 data, and R1, R2, R3, R4 are channel 1 data.

Table 3: **Alignment of Data at Consecutive Addresses**

	Address	Non-Interleaved 32-bit	Interleaved Mode 32-bit	Non-Interleaved 16-bit	Interleaved 16-bit
Period N	PSA[1] = BSA[2] + N(PS[3])	L1	L1	L2 L1	R1 L1
	PSA + ‘h4	L2	R1	L4 L3	R2 L2
	PSA + ‘h8	L3	L2	L6 L5	R3 L3
	PSA + ‘hC	L4	R2	L8 L7	R4 L4
		.	.	.	.
		.	.	.	.
		.	.	.	.
		.	.	.	.
	PSA + CO[4]	R1	Lx	R2 R1	Ra La
	PSA + CO + ‘h4	R2	RX	R4 R3	Rb Lb
	PSA + CO + ‘h8	R3	Ly	R6 R5	Rc Lc
	PSA + CO + ‘hC	R4	Ry	R8 R7	Rd Ld
		.	.		.
		.	.		.
		.	.		.
		.	.		.

Notes:

1. Period Start Address (PSA)
2. Buffer Start Address (BSA)
3. Period Size (PS)
4. Channel Offset (CO)

***Note*:**

The period size constraints mentioned in the register description must be strictly followed. The functionality is not guaranteed otherwise.

S2MM Operations

The Audio Formatter core collects audio data from 2,4,6, or 8 audio channels as programmed through register and writes the data to memory in a packaged format. When the input data is AES, the core has an AES decoder block to get the channel status information of the audio stream. S2MM supports any order of data input channels and append zeroes for missing samples. S2MM also has a timeout logic to know if the source stops sending data.

When the core is enabled for S2MM, it readily accepts the audio input stream even before the DMA is enabled through registers. All this data is used only for channel status calculation and is not saved to write to memory when DMA is enabled.

- **AES Decoder:**

AES Decoder block is included in the core when the write is enabled and the input audio stream is selected as AES. The AES Decode logic is reset only with the hard reset to the system and the soft reset to S2MM programmed through register.

This module keeps accumulating the Channel status bit on the AES input and generates a flag when all the 192 bits are updated. The channel status thus inferred and the update indication flag are register readable. The flag is cleared once the register is read and is set when the channel status is updated or differs from the previous value.

- **Timeout Logic:**

  The S2MM core gives an error indication when there is no valid audio input for a long time on the audio channels. A counter runs on the S2MM AXI4 clock when the S2MM DMA is enabled. Counter increments when input TVALID is low and refreshes when TVALID is high. When the counter reaches the programmed value in register, it generated timeout error which can be read through register. It can also generate an interrupt when configured.

  Timeout error is cleared only through hard or soft reset.

  ***Note*:** The timeout count must be updated before starting the DMA. The change in the count value after starting the DMA will not be considered.

- **Missing Samples:**

  In this core, we assume that the input audio samples of different channels can come in any order but before the second set of samples from any of the channels arrives.

  For example, in a 2-channel system, if the incoming samples are in the following order:

  1,2,2,1,1,1,2

  {1,2}, {2,1}, {1 ?}

  Here, the expectation is a 2 after 1 to complete the third set. However, another sample from channel 1 arrived that indicates a sample from channel 2 is missing. Hence, 0 is added as data for second channel for third set. The final data would be:

  {1,2}, {2,1} (will be re-ordered), {1,0}, {1,2}

  {1,2}, {1,2}, {1,0}, {1,2}

***Note*:** The core has a buffer logic inside which saves the channel data up to 8 samples. So the core flushes 8 samples of data per channel onto AXI4 Memory Map interface at once when all the 8 samples per channel are received or when the ninth sample of any of the channel is received in case of missing samples.

MM2S Operations

The core reads the data from the memory, and based on the memory organization programmed, the core rearranges the read data into multiple channels and sends it as an output AXI4-Stream at a sampling rate.

The core has a master clock “aud_mclk” whose frequency is known. Based on the required sampling rate, the Fs multiplier value is programmed through the register.

Sampling frequency = Frequency of aud_mclk/Fs multiplier Value

When the data in memory is selected as PCM and output data format is AES, the core adds AES information to the available PCM data. The channel status information is added as programmed through registers before starting DMA. The first data out is the start of block. The data on MM2S is sent out in order of channels from 0 to N-1.

The data on MM2S is sent out in order of channels from 0 to N-1.

Programming Sequence for S2MM

1. Program bit run/stop in register 0x10 after programming all other S2MM related registers.
2. Programming 0x1C, 0x20, 0x24, and 0x44 is required and to be done only when bit 0 of 0x10 is 0.
    a. Program 0x1C for a valid period size and no. of periods in a buffer. If constraints on period size are not met, functionality of the core is not guaranteed.

    b. Programming 0x24 register can be skipped if Address width is 32 bits.

    c. Program channel offset in 0x44 register with a value = period size / no. of valid channels. This register can be skipped when C_PACKAGING_MODE_S2MM = 0 (Interleaved mode).
3. Program register 0x18, if timeout error interrupt is required.
4. Program 0x10 in the end with the no. of valid channels and PCM data width. Enable the respective interrupts required.
5. Program 0x10 to start DMA making bit 0 to 1.
    - This can be clubbed with step IV.

Programming Sequence for MM2S

1. Program bit run/stop in register 0x110 after programming all other MM2S related registers.
2. Programming 0x118, 0x11C, 0x120, 0x124, and 0x144 is required and to be done only when bit 0 of 0x110 is 0.
    a. Program 0x118 with “fs_multiplier_value” to generate the output stream samples at this rate.

    b. Program 0x11C for a valid period size and no. of periods in a buffer. If constraints on period size are not met, functionality of the core is not guaranteed.

   c. Programming 0x124 register can be skipped if Address width is 32 bits.

   d. Program channel offset in 0x144 register with a value = period size / no. of valid channels. This register can be skipped when C_PACKAGING_MODE_MM2S = 0 (Interleaved mode).

3. Program registers 0x12C to 0x140 with AES channel status when C_MM2S_DATAFORMAT = 3. In other data-formats these registers can be ignored.
4. Program 0x110 in the end with the no. of valid channels and PCM data width. Enable the respective interrupts required.
5. Program 0x110 to start DMA making bit 0 to 1.
   - This can be clubbed with step 4.

***Note*:** Do not modify any of the above-mentioned registers when Run/Stop bit is 1. The core behavior is not guaranteed otherwise.

Clocking

There are 4 clocking inputs possible, and each input has an associated synchronous reset. Ensure that the proper clock is connected to aud_mclk so that Audio samples are generated at the required sampling rate in case of MM2S. Audio clock is typically an integer multiple of required frequency sampling rate (Fs). It is best practice to use a very stable clock source to generate the Audio Clock to minimize jitter.

Table 4: **Clocks**

Clock	Description
s_axi_lite_aclk	Used on AXI4-Lite interface for register reads and writes for both S2MM and MM2S.
m_axis_mm2s_aclk	Used for reading from memory as audio stream output. AXI4 MM2S Stream and AXI4 MM2S Memory mapped interface ports work on this clock.
s_axis_s2mm_aclk	Used to write audio stream input to memory. AXI4 S2MM Stream and AXI4 S2MM Memory mapped interface ports work on this clock.
aud_mclk	Used as a master reference clock to generate audio stream in case of MM2S at a required sampling rate. Typically a multiple of sampling rate frequency.

All the signals are well synchronized to avoid CDC issues. In case of synchronous clocks for s_axi_lite_aclk and S2MM clock or MM2S clock, the corresponding ASYNC_CLOCKS parameter can be disabled to reduce to resource count.

***Note*:**

When ASYNC_CLOCKS parameter is 0 and clocks are not synchronous, IP behavior is not guaranteed.

Resets

Each clock in the system has an associated reset synchronous to that clock.

Table 5: **Clocks**

Clock	Synchronous Reset To	Description
s_axi_lite_aresetn	active-Low to s_axi_lite_aclk	This reset resets the entire core when active.
s_axis_s2mm_aresetn	active-Low to s_axis_s2mm_aclk	Resets the S2MM logic apart from AXI4-Lite register interface.
m_axis_mm2s_aresetn	active-Low to m_axis_mm2s_aclk	Resets the MM2S logic apart from AXI4-Lite register interface.
aud_mreset	active-Low to Audio Master clock	Resets the sampling frequency counter. This when in reset state, does not generate AXI4-Stream out in MM2S.

These are all hard resets to the core. Apart from these, there is also a soft reset feature for the core independently for S2MM and MM2S. It gracefully completes the AXI4 Memory Map transactions and ends AXI4-Stream transactions abruptly.

Chapter 5

Design Flow Steps

This section describes customizing and generating the core, constraining the core, and the simulation, synthesis, and implementation steps that are specific to this IP core. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Getting Started* (UG910)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)

Customizing and Generating the Core

This section includes information about using Xilinx® tools to customize and generate the core in the Vivado® Design Suite.

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

1. Select the IP from the IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or right-click menu.

For details, see the following guides:

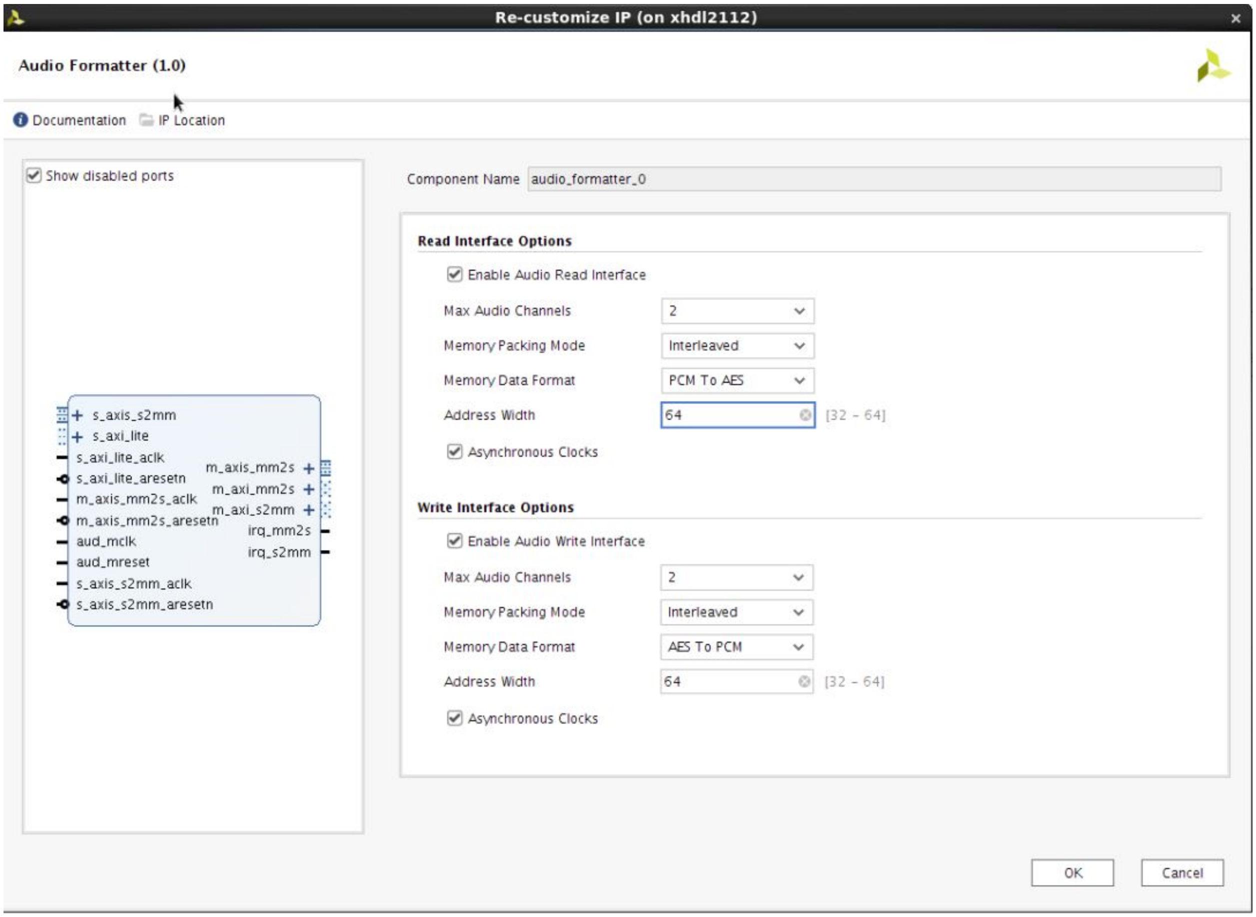
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Getting Started* (UG910)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)

Note:

Figures in this chapter are illustrations of the Vivado IDE. The layout depicted here might vary from the current version.

IP Customization GUI

Figure 3: **Audio Formatter Tab**



Re-customize IP (on xhdl2112)

Audio Formatter (1.0)

Documentation IP Location

☑ Show disabled ports

Component Name: audio_formatter_0

Read Interface Options

☑ Enable Audio Read Interface

Max Audio Channels: 2

Memory Packing Mode: Interleaved

Memory Data Format: PCM To AES

Address Width: 64 [32 - 64]

☑ Asynchronous Clocks

Write Interface Options

☑ Enable Audio Write Interface

Max Audio Channels: 2

Memory Packing Mode: Interleaved

Memory Data Format: AES To PCM

Address Width: 64 [32 - 64]

☑ Asynchronous Clocks

- **Component Name:** The component name is used as the base name of output files generated for the module. Names must begin with a letter and must be composed from characters: a to z, 0 to 9 and "_".
- **Read Interface Options:**
  - **Enable Audio Read Interface:** Enables the MM2S part of the core when selected. All the other Read Interface Options are valid only when this is enabled.
  - **Max Audio Channels:** Specifies the maximum number of Audio channels the core can support in MM2S mode. Can be configured as 2, 4, 6 or 8.

  - **Memory Packaging Mode:** Specifies the Interleaved or Non-Interleaved mode packaging of data in memory.
  - **Memory Data Format:** Configured from a choice of AES and PCM data in memory and audio channels:
    - AES To AES: AES Data in Memory and read as AES data on Stream
    - AES To PCM: AES Data available in Memory and reads only PCM data on stream
    - PCM To PCM: PCM Data available in memory is read on Stream as well
    - PCM To AES (Default): PCM Data in memory is AES encoded and read on Stream
  - **Address Width:** Address width of AXI4 Memory mapped MM2S interface.
  - **Asynchronous Clocks:** When checked, means the `s_axi_lite_aclk` and `m_axis_mm2s_aclk` are asynchronous and all the Clock Domain Crossing of signals within the IP is taken care. When unchecked, the core assumes both clocks to be synchronous, and does not add synchronizers.

- **Write Interface Options:**
  - **Enable Audio Write Interface:** Enables the S2MM part of the core when selected. All the other Write Interface Options are valid only when this is enabled.
  - **Max Audio Channels:** Specifies the maximum number of Audio channels the core can support in S2MM mode. Can be configured as 2, 4, 6 or 8.
  - **Memory Packaging Mode:** Specifies the Interleaved or Non-Interleaved mode packaging of data in memory.
  - **Memory Data Format:** Configured from a choice of AES and PCM data in memory and audio channels:
    - AES To AES: AES Data on stream and the same is written to memory
    - AES To PCM(Default): AES Data on stream is decoded and PCM data is written to memory
    - PCM To PCM: PCM Data on stream is written to memory directly.
  - **Address Width:** Address width of AXI4 Memory mapped S2MM interface.
  - **Asynchronous Clocks:** When checked, this means `s_axi_lite_aclk` and `s_axis_s2mm_aclk` are asynchronous, and all the Clock Domain Crossing of signals within the IP is taken care of. When unchecked, the core assumes both clocks to be synchronous, and does not add synchronizers.

    ***Note*:** The IP requires more resources when using Asynchronous clock mode compared to synchronous clock mode.

User Parameters

The following table shows the relationship between the fields in the Vivado® IDE and the user parameters (which can be viewed in the Tcl Console).

Table 6: **User Parameters**

Vivado IDE Parameter/ Value[1]	User Parameter/Value	Default Value	Description
C_MAX_NUM_CHANNELS_MM2S	2,4,6,8	2	Maximum number of channels supported by the system (MM2S)
C_MAX_NUM_CHANNELS_S2MM	2,4,6,8	2	Maximum number of channels supported by the system (S2MM)
C_INCLUDE_S2MM	0,1	1	1: Include S2MM 0: Do not include
C_INCLUDE_MM2S	0,1	1	1: Include MM2S 0: Do not include
C_S2MM_DATAFORMAT	0,1,2	1	0: AES to AES 1: AES to PCM 2: PCM to PCM
C_MM2S_DATAFORMAT	0,1,2,3	3	0: AES to AES 1: AES to PCM 2: PCM to PCM 3: PCM to AES
C_PACKING_MODE_MM2S	0,1	0	0: Interleaved mode 1: Non-Interleaved mode
C_PACKING_MODE_S2MM	0,1	0	0: Interleaved mode 1: Non-Interleaved mode
C_S2MM_ASYNC_CLOCK	0,1	1	ASYNC_CLK parameter of S2MM logic 0: s_axi_lite_aclk and s_axis_s2mm_aclk must be synchronous 1: Clocks can be asynchronous
C_MM2S_ASYNC_CLOCK	0,1	1	ASYNC_CLK parameter of S2MM logic 0: s_axi_lite_aclk and m_axis_mm2s_aclk must be synchronous 1: Clocks can be asynchronous

Notes:

1. Parameter values are listed in the table where the Vivado IDE parameter value differs from the user parameter value. Such values are shown in this table as indented below the associated parameter.

***Note*:**

- In case of PCM input on AXIS, the valid data should be driven on the applicable LSB bits.

- In case of PCM output on AXIS, the valid data would be driven on applicable LSB bits.
- Width of AXIS data is always 32.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896).

Constraining the Core

Required Constraints

This section is not applicable for this IP core.

Device, Package, and Speed Grade Selections

This section is not applicable for this IP core.

Clock Frequencies

See Clocking.

Clock Management

This section is not applicable for this IP core.

Clock Placement

This section is not applicable for this IP core.

Banking

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

This section is not applicable for this IP core.

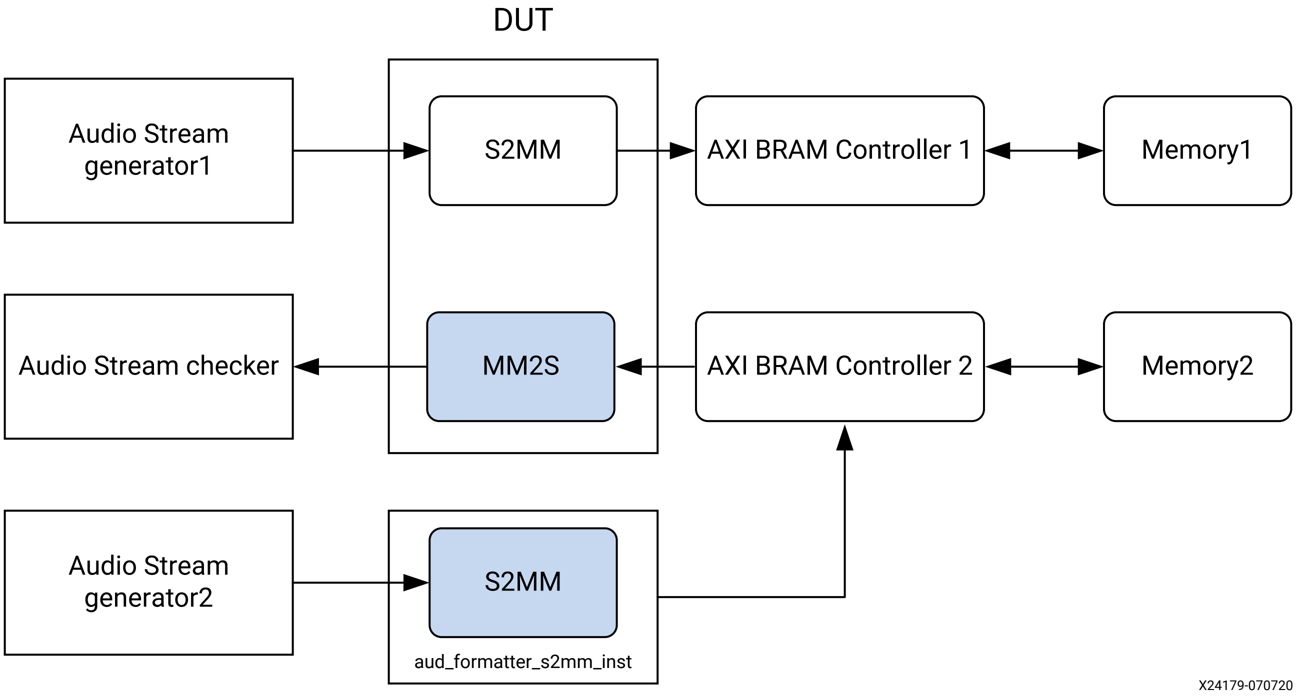
Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896).

Example Design

This chapter contains information about the example design provided in Vivado® Design Suite.

Figure 4: **Core Example Design**



The example design demonstrates the functioning of Audio Formatter IP core for 2 channels. An Audio stream generator module is used to generate either PCM or AES audio input to the IP based on the configuration. Audio stream checker is also implemented to verify the functioning of core by reading the known data from memory and checking the incremental pattern.

Based on the configuration of DUT (Device under Test), the data is written and read back from the memory. After verification of the status of the commands from the core and the data check at the checker module, the test evaluates to PASS or FAIL.

The example design consists of following blocks:

1. Clocking wizard: To generate necessary clocks AXI4-Lite clock and other audio clock
2. Reset Generator: To generate corresponding synchronous resets to the system

3. Audio Formatter IP DUT
4. A generated Audio Formatter S2MM core based on the configuration of DUT MM2S core
5. 2 AXI Traffic Generator IPs to program all the IPs
6. 2 instances of AXI4 Block RAM Controller as a AXI4 Memory Map slave to the DUT and S2MM generated core to interface with Memory
7. 2 instances of XPM Memory Single port RAM
8. Audio Stream generator module
9. Audio stream checker

Note:

The Audio stream generator, checker, and the example design are not generic configurations. The modules and example design are generated based on IP configuration of DUT. Any changes in the models are not recommended.

Irrespective of the number of Channels configuration of DUT, `no_of_valid_channels` is programmed as 2 for the core through registers. So, the Example design always designed to work with 2 channels configuration only.

Implementing the Example Design

After you follow the steps described in Customizing and Generating the Core, implement the example design.

1. Right-click the core in the Hierarchy window, and select **Open IP Example Design**.
2. A new window pops up, asking you to specify a directory for the example design. Select a new directory or keep the default directory.
3. A new project is automatically created in the selected directory and it is opened in a new Vivado® integrated design environment (IDE) window.
4. In the Flow Navigator (left pane), click **Run Implementation**, and follow the directions.

Example Design Directory Structure

In the current project directory, a new project named <component_name>_ex is created, and the files are generated in the `<component_name>_example.src/sources_1/ip/<component_name>/` directory. This directory and its sub-directories contain all the source files that are required to create the AMM Master Bridge example design.

The example design directory is created in imports/. It contains the following generated example design top files:

- `<component_name>_exdes.v`: Top-level HDL file for the example design
- `Exdes_axis_chk.sv`: Checker module
- `Exdes_axis_gen.sv`: AXI4-Stream generator module

Simulating the Example Design

Using the example design delivered as part of the Audio Formatter IP core, you can quickly simulate and observe the behavior of the core.

Setting Up the Simulation

The Xilinx® simulation libraries must be mapped to the simulator. The example design supports functional (behavioral) and post-synthesis simulations.

For comprehensive information about Vivado® simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900).

***Note*:** To switch simulators, click Simulation Settings in the Flow Navigator (left pane). In the Simulation options list, change Target Simulator.

Simulation Results

The simulation script compiles the example design and supporting simulation files. It then runs the simulation and checks that it completed successfully after simulating for 3ms, and displays either of the following messages:

- `Test Completed Successfully`
- `Test FAILED`

Appendix A

Upgrading

This appendix is not applicable for the first release of the core.

Appendix B

Debugging

This appendix includes details about resources available on the Xilinx® Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the core, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support. The Xilinx Community Forums are also available where members can learn, participate, share, and ask questions about Xilinx solutions.

Documentation

This product guide is the main document associated with the core. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx® Documentation Navigator. Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use keywords such as:

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the Core

AR 54489.

Technical Support

Xilinx provides technical support on the Xilinx Community Forums for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To ask questions, navigate to the Xilinx Community Forums.

Debug Tools

There are many tools available to address Audio Formatter design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

The Vivado® Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx® devices.

The Vivado logic analyzer is used to interact with the logic debug LogiCORE IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908).

Appendix C

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

***Note*:** For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

References

These documents provide supplemental material useful with this guide:

1. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
2. *Vivado Design Suite User Guide: Designing with IP* (UG896)
3. *Vivado Design Suite User Guide: Getting Started* (UG910)
4. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
5. *SPDIF/AES3 LogiCORE IP Product Guide* (PG045)
6. *I2S Transmitter and Receiver LogiCORE IP Product Guide* (PG308)
7. *HDMI 1.4/2.0 Transmitter Subsystem Product Guide* (PG235)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
07/08/2020 Version 2020.1	
General Updates	• Provided additional information on how data is mapped in memory • Clarified Example Design figure • Updated Programming Sequence for MM2S
12/05/2018 Version 2018.3	
Initial release.	N/A

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos.

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

Copyright