XAPP1272 (v2.0) May 5, 2016

Modular SMPTE ST 2022-567 on Kintex-7 Evaluation Board

Author: Ilias Ibrahim, Josh Poh, Cunhua Xue

Summary

This application note covers the design considerations of a Video Over IP networks system using a modular form of the Video Over IP Transmitter and Receiver Subsystem designs. The design focuses on high-bit rate, native media transport over a 10 Gb/s Ethernet with built-in Forward Error Correction (FEC) Engine and Seamless Switching Protection. It is able to support up to three standard definition/high definition/serial digital interface (SD/HD/3G-SDI) streams. Overall, the reference design has two platforms: the transmitter platform and the receiver platform. The transmitter platform uses three Society of Motion Picture and Television Engineers (SMPTE) SDI cores to receive incoming serial digital interface (SDI) video streams. The received SDI streams are encapsulated into fixed-size datagrams and multiplexed by the modular Video Over IP transmitter subsystem and sent out through two 10-Gigabit Ethernet media access control (MAC) cores. Each 10-Gigabit link is supported by a 10-Gigabit Ethernet physical coding sublayer/physical medium attachment (PCS/PMA) core which requires an optical cable to link to the receiver end. On the receiver platform, the Ethernet datagrams are received with two 10-Gigabit Ethernet MAC cores. The modular Video Over IP receiver subsystem filters the datagrams, de-encapsulates and de-multiplexes the datagrams into individual streams which are then fed out through the SMPTE SDI cores. The Ethernet datagrams within both the transmitter and receiver subsystem are being buffered using DDR3 SDRAM. The double data rate (DDR) section uses a 7 series device AXI4 memory DDR controller and connects to the subsystem via an AXI4 interconnect. A MicroBlaze™ processor is included in the design to initialize and control the subsystem.

The reference design targets the Xilinx® Kintex®-7 FPGA KC705 Evaluation Kit, which uses the Kintex-7 XC7K325T-2FFG900 FPGA, Inrevium TB-FMCH-3GSDI2A [Ref 1] and Faster Technology FM-S14 Quad small form-factor pluggable (SFP/SFP+) Transceiver FPGA Mezzanine Card (FMC) boards. See the Kintex-7 FPGA KC705 Evaluation Kit [Ref 2] and Faster Technology FM-S14 Quad SFP/SFP+ transceiver FMC [Ref 3] for details.

Reference Design

Included Systems

The reference design was created and built using the Vivado® Design Suite, System Edition. The design also includes software built using the Xilinx Software Development Kit (SDK). The software runs on the MicroBlaze processor subsystem and implements control write and status read functions. Complete project files for the Vivado Design Suite and the SDK are provided with this application note to allow examination and rebuilding of the design or to use it as a template for starting a new design.

Hardware

Introduction

The reference design is built around the Modular IP cores and leverages existing Xilinx IP cores to form the complete video over IP system. The input and output of the complete system are SDI streams. The system consists of two platforms. The transmitter and receiver system each resides in separate platforms. Two optical cables connect the two platforms simulating an IP network. See Figure 1.

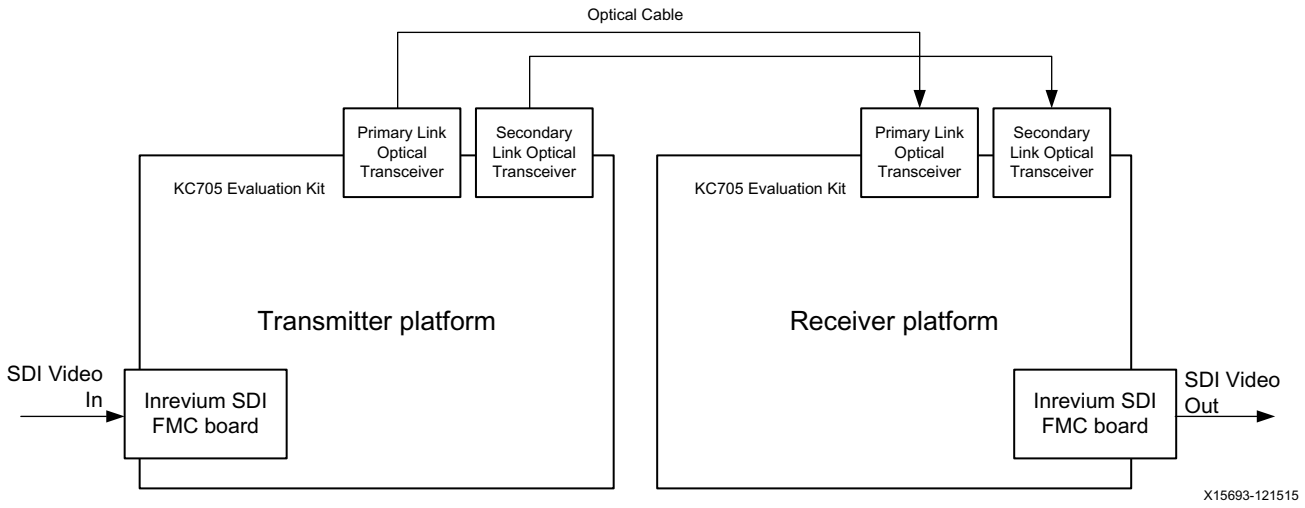

Figure 1: **Block Diagram of the Video over IP Platform Setup**

The SMPTE SDI core allows the SDI Front End Interface to receive and transmit SDI AXI4-Streams while the 10-Gigabit Ethernet Subsystem enables the Modular ST 2022-56 Media over IP subsystem to transfer SDI data in the Ethernet medium. See Figure 2 and Figure 3.

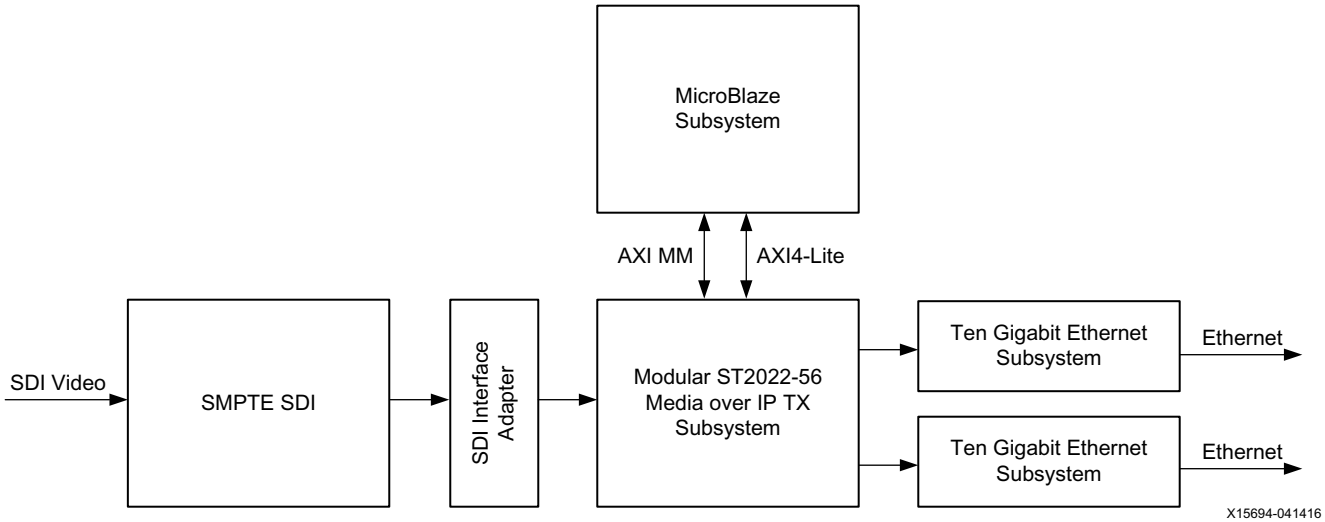

Figure 2: **Block Diagram of the Video over IP Transmitter System**

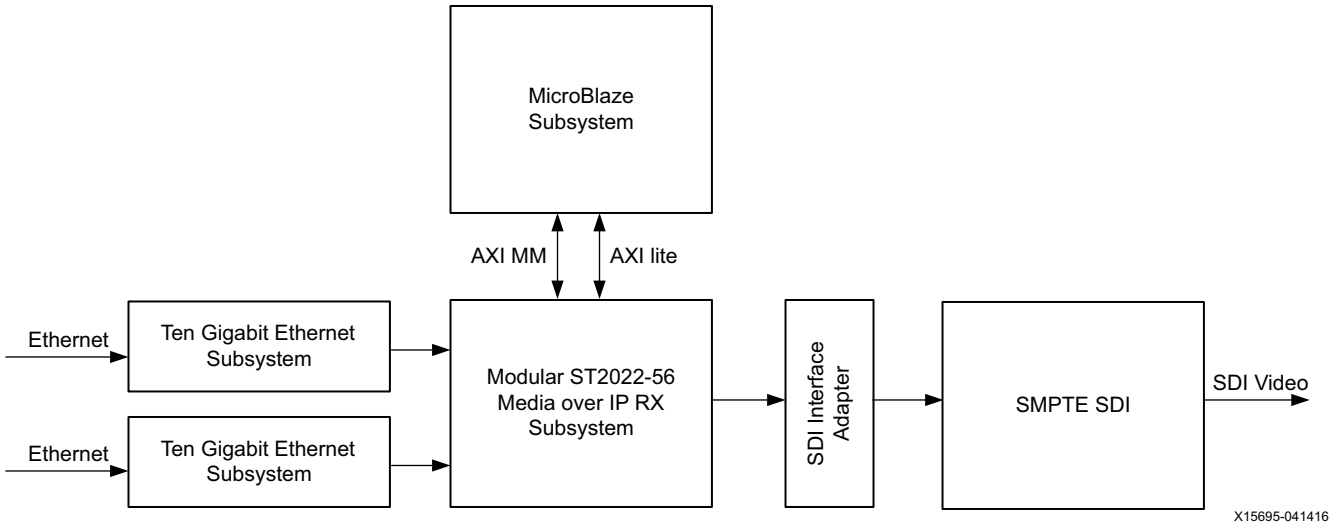

Figure 3: **Block Diagram of the Video over IP Receiver System**

Other than managing the SDI streams, encapsulation and de-encapsulation, the transmitter and receiver subsystem includes FEC and seamless protection switching features. FEC protects the video stream during the transport of high-quality video over IP networks. With FEC, the transmitter adds systematically generated redundant packets to the packet stream. This redundancy allows the receiver to detect and correct a limited number of packet errors occurring anywhere in the data stream without the need to ask the transmitter for additional video data. These errors, in the form of lost video packets, result from a variety of causes ranging from thermal noise to storage system defects and transmission noise introduced by the environment. FEC gives the receiver the ability to correct these errors without needing a reverse channel to request retransmission of data. Seamless protection switching allows transmission and reception of two identical streams over potentially diverse paths to add more reliability to the system. The receiver handles the seamless switching datagram-by-datagram without impacting the content or the stream. These features can be enabled using the core registers.

High-level control of the system is provided by a simplified MicroBlaze embedded processor subsystem containing I/O peripherals and processor support IP. A clock generator block and a processor system reset block supply clock and reset signals for the system, respectively. An AXI4 interconnect and an AXI4 memory interface generator (MIG) is instantiated in the subsystem allowing the video over IP cores access to the on-board DDR3 SDRAM. See Figure 4 and Table 1 for a block diagram of the MicroBlaze processor subsystem and its address map.

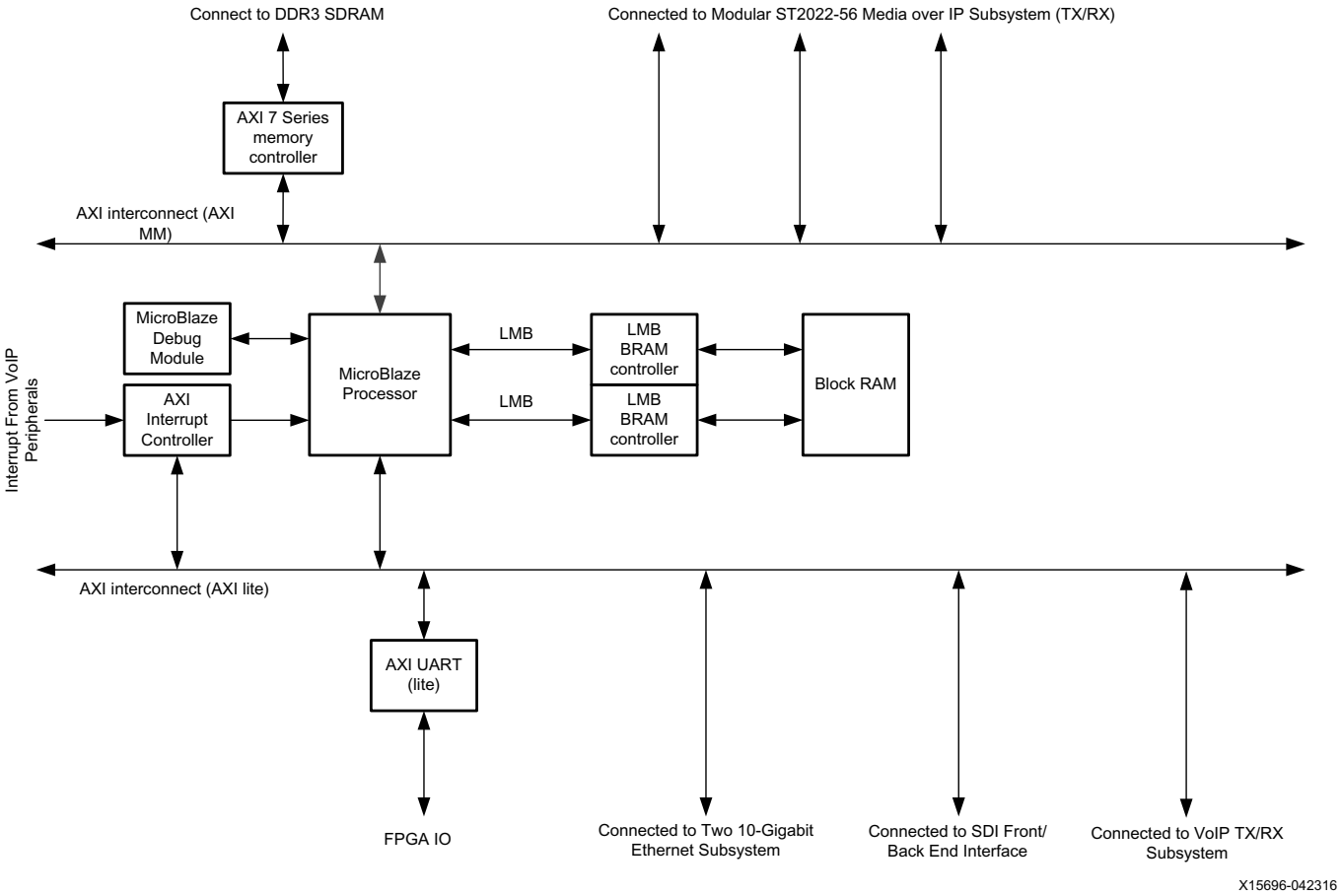

Figure 4: **MicroBlaze Processor Subsystem**

Table 1: **Reference Design Subsystem Address Map**

Peripheral	Instance	Base Address	High Address
Module: MicroBlaze Subsystem (Common)			
lmb_bram_if_ctrl	ilmb_bram_if_ctrl	0x00000000	0x0001FFFF
lmb_bram_if_ctrl	dlmb_bram_if_ctrl	0x00000000	0x0001FFFF
mig_7series	mig_1	0xC0000000	0xFFFFFFFF
axi_uartlite	axi_uartlite_0	0x40600000	0x4060FFFF
axi_intc	axi_intc_0	0x41200000	0x4120FFFF
Module: 10-Gigabit Ethernet Subsystem (Common)			
axi_10g_ethernet	axi_10g_ethernet_0	0x55000000	0x5500FFFF
axi_10g_ethernet	axi_10g_ethernet_1	0x56000000	0x5600FFFF
Module: SDI Front End Interface (VoIP TX)			
v_voip_sdi2axis	sdi_front_intf_0/v_voip_sdi2axis_0	0x60000000	0x6000FFFFF
v_voip_sdi2axis	sdi_front_intf_1/v_voip_sdi2axis_0	0x61000000	0x6100FFFFF
v_voip_sdi2axis	sdi_front_intf_2/v_voip_sdi2axis_0	0x62000000	0x6200FFFFF
Module: SDI Back End Interface (VoIP RX)			
v_voip_axis2sdi	sdi_back_intf_0/v_voip_axis2sdi_0	0x60000000	0x6000FFFFF
v_voip_axis2sdi	sdi_back_intf_1/v_voip_axis2sdi_0	0x61000000	0x6100FFFFF
v_voip_axis2sdi	sdi_back_intf_2/v_voip_axis2sdi_0	0x62000000	0x6200FFFFF
Module: VoIP TX Subsystem			
v_voip_fec_tx	v_voip_fec_tx_0	0x70000000	0x7000FFFF
v_voip_packetizer56	v_voip_packetizer56_0	0x65000000	0x6500FFFF
v_voip_packetizer56	v_voip_packetizer56_1	0x66000000	0x6600FFFF
v_voip_packetizer56	v_voip_packetizer56_2	0x67000000	0x6700FFFF
v_voip_framer	v_voip_framer_0	0x50000000	0x5000FFFF
v_voip_framer	v_voip_framer_1	0x51000000	0x5100FFFF
Module: VoIP RX Subsystem			
v_voip_fec_rx	v_voip_fec_rx_0	0x70000000	0x7000FFFF
v_voip_depacketizer56	v_voip_depacketizer56_0	0x65000000	0x6500FFFF
v_voip_depacketizer56	v_voip_depacketizer56_1	0x66000000	0x6600FFFF
v_voip_depacketizer56	v_voip_depacketizer56_2	0x67000000	0x6700FFFF
v_voip_decap	v_voip_decap_0	0x50000000	0x5000FFFF
v_voip_decap	v_voip_decap_1	0x51000000	0x5100FFFF
v_voip_data_pullout	v_voip_data_pullout_0	0x75000000	0x7500FFFFF

Reference Design Specifics

The reference design includes the following cores:

- AXI Interconnect
- AXI Interrupt Controller
- MicroBlaze
- MicroBlaze Debug Module
- Local Memory Bus (LMB)
- LMB BRAM Controller
- Block Memory Generator
- Clocking Wizard
- Processor System Reset
- AXI UARTLite
- Memory Interface Generator
- SMPTE SD/HD/3G- SDI
- Ten Gigabit Ethernet Subsystem
- AXI4-Stream Switch
- AXI4-Stream Broadcaster
- AXI4-Stream Subset Converter
- Modular VoIP: SDI2AXIS Interface Adapter Module
- Modular VoIP: ST 2022-6 Packetizer Module
- Modular VoIP: Video over IP FEC Transmitter Module
- Modular VoIP: Framer Module
- Modular VoIP: Decapsulator Module
- Modular VoIP: Video over IP FEC Receiver Module
- Modular VoIP: ST 2022-6 Depacketizer Module
- Modular VoIP: AXIS2SDI Interface Adapter Module

Hardware System Specifics

This section describes the high-level features of the reference design, including how the main IP blocks are configured.

Video over IP System Reference Design

The reference design implements SMPTE2022-5/6/7 using modular VoIP cores which support broadcast applications that require bridging between broadcast connectivity standards (SD/HD/3G) and two 10-Gigabit Ethernet networks. The cores are intended for developing Internet protocol-based systems to reduce the overall cost in broadcast facilities for distribution and routing of video data.

Video over IP Transmitter (VoIP TX) Subsystem

In the reference design, the VoIP TX subsystem receives three SDI video streams from three SDI over AXI4-Stream interfaces and transmits combined SMPTE2022-5/6 datagrams into the dual-link 10G Ethernet.

The VoIP TX subsystem consists of multiple modules (Figure 6) which are:

- **ST 2022-6 Packetizer Module**: Converts SDI video stream (transmitted over SDI over AXI4-Stream) from the SDI Front End interface into a media datagram stream in accordance with the SMPTE2022-6 protocol.
- **AXI4-Stream Switch**: Combines multiple datagram streams into one stream and feeds into the VoIP FEC TX module.
- **VoIP FEC TX**: Generates redundant FEC datagrams according to the SMTPE2022-5 protocol.
- **AXI4-Stream Broadcaster**: Broadcasts the incoming stream from the VoIP FEC TX to two Framer modules for seamless protection as in SMPTE2022-7.
- **Framer Module**: Adds the Ethernet, IP and User Datagram Protocol (UDP) headers to the RTP packets.

The VoIP TX system accepts SDI stream data and encapsulates it into media datagram payloads in accordance with SMPTE 2022-6. The systematically-generated redundant forward error correction datagrams are formatted according to SMPTE 2022-5. The system uses IP/UDP/RTP protocols to transport the media and FEC datagrams over the IP network. The SMPTE 2022-5/6 datagrams are replicated using a broadcaster and transmitted to the two 10-Gigabit Ethernet media access control (MAC) cores for a seamless protection according to SMPTE 2022-7.

To run the TX subsystem correctly, the video bandwidth must meet or do not exceed what is required to support the packet header overhead generated by the system. The header overhead required for media and FEC datagram (SMPTE 2022-5/6 packet) generation is approximately 6.6% due to the MAC/IP/UDP/RTP and SMPTE 2022-5/6 headers.

Note: 6.6% is obtained from the ratio of the packet header over the total packet size. Here only packet header overhead is discussed. The bandwidth overhead introduced by possible redundant SMPTE ST 5 FEC packets depends on the configuration of the FEC TX core, for example, fec mode, matrix size, etc.

The VoIP TX subsystem consist of two main components which are the SDI Front End Interface System and the Video over IP TX System shown in Figure 5 and Figure 6.

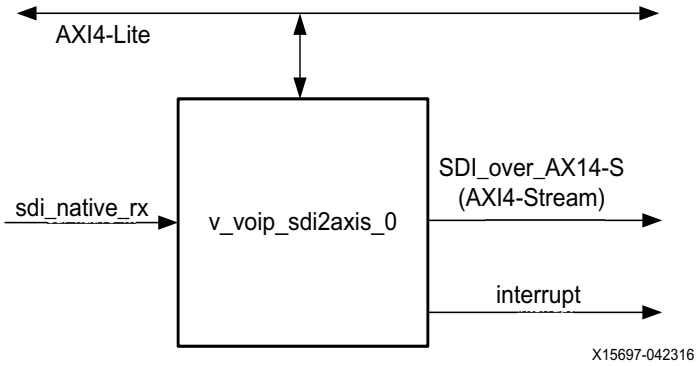

Figure 5: **SDI Front End Interface (Channel 0)**

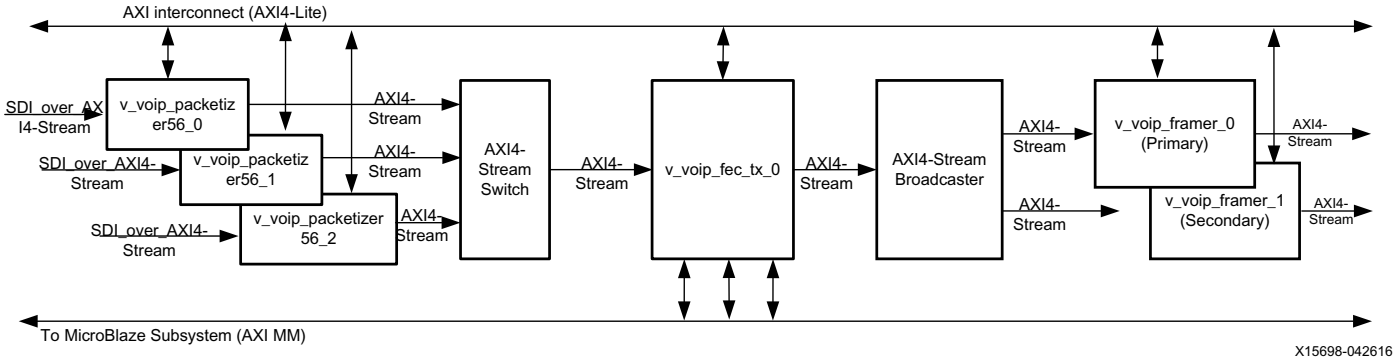

Figure 6: **Video over IP Transmitter Subsystem (3 Channels Configuration)**

SDI Front End Interface (SDI2AXIS Adapter)

The SDI Front End Interface consists of an SDI2AXIS Adapter which converts native SDI-Stream to SDI over AXI4-Stream.

The SDI Front End Interface is instantiated three times in the reference design to support three channels of three incoming SDI-RX streams.

The SDI2AXIS Adapter in the reference design accepts an SDI Stream via a native SDI-RX Interface and transmits the SDI Stream via SDI over the AXI4-Stream interface. The SDI over AXI4-Stream stream interface is described in Table 2.

Table 2: **SDI over AXI4-Stream Protocol**

Signals	Direction	Description		
sdi_axis_tready	In	TREADY indicates that the slave can accept a transfer in the current cycle.		
sdi_axis_tvalid	Out	Valid indicator for sdi_axis_tdata, sdi_axis_tlast, and sdi_axis_tuser signals.		
sdi_axis_tlast	Out	Together with sdi_axis_tdata to indicate the last word of the frame.		
sdi_axis_tdata	Out	**Bit**	**Native SDI Mapping**	**Description**
		9:0	ds1a	Video data stream 1 which is dependent on the SDI mode: • SD-SDI: Multiplexed Y/C data stream • HD-SDI: Y data stream • 3G-SDI level A: Data stream 1 • 3G-SDI level B-DL: Data stream 1 of link A • 3G-SDI level B-DS: Y data stream of HD-SDI signal 1
		19:10	ds2a	Video data stream 2 which is dependent on the SDI mode: • SD-SDI: Not used • HD-SDI: C data stream • 3G-SDI level A: Data stream 2 • 3G-SDI level B-DL: Data stream 2 of link A • 3G-SDI level B-DS: C data stream of HD-SDI signal 1
		29:20	ds1b	This is only used in 3G-SDI level B mode. The data stream on this port is: • 3G-SDI level B-DL: Data stream 1 of link B • 3G-SDI level B-DS: Y data stream of HD-SDI signal 2
		39:30	ds2b	This is only used in 3G-SDI level B mode. The data stream on this port is: • 3G-SDI level B-DL: Data stream 2 of link B • 3G-SDI level B-DS: C data stream of HD-SDI signal 2
		50:40	line_a	Indicates current line number of SDI link A.
		61:51	line_b	Indicates current line number of SDI link B.
		62	rx_sav	Asserted High for 1 sample time when XYZ of SAV is present on data stream output.
		63	rx_eav	Asserted High for 1 sample time when XYZ of EAV is present on data stream output.

Table 2: **SDI over AXI4-Stream Protocol *(Cont'd)***

Signals	Direction	Description			
sdi_axis_tuser	Out	**Bit**	**Abbreviation**	**Description**	
		0	sof	Indicates Start of Frame	
		1	Reserved		
		3:2	sdi_mode	00	HD-SDI
				01	SD-SDI
				10	3G-SDI
				11	Invalid Video Format
		4	level_b_3g	In 3G-SDI mode, this output is asserted High when the input signal is level B and Low when it is level A. This output is only valid when rx_mode_3g is High.	
		5	rx_bit_rate	This input port indicates which bit rate is being received in HD-SDI and 3G-SDI modes. This input only is used to generate the value on the rx_t_rate output port. So if the rx_t_rate output port is not used, then it is not required that the rx_bit_rate input be driven correctly. When using the transceivers in Xilinx FPGAs, the device-specific transceiver control module contains a bit rate detector that generates the signal to be connected to the rx_bit_rate input port. HD-SDI mode: • rx_bit_rate = 0: Bit rate = 1.485 Gb/s • rx_bit_rate = 1: Bit rate = 1.485/1.001 Gb/s 3G-SDI mode: • rx_bit_rate = 0: Bit rate = 2.97 Gb/s • rx_bit_rate = 1: Bit rate = 2.97/1.001 Gb/s	
		29:6	vid_src_fmt	**Bit**	**Description**
				29:26	MAP
				25:18	FRAME
				17:10	FRATE
				9:6	SAMPLE
		31:30	Reserved		

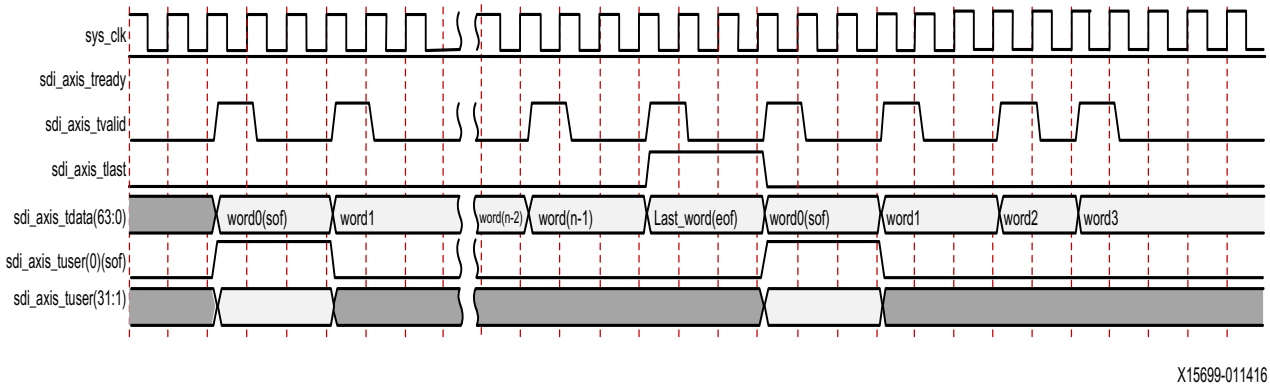

Figure 7: **SDI over AXI4-Stream Timing Diagram**

The module has an AXI4-Lite interface that allows dynamic control of the parameters within the core from a processor. For information about the registers, see SDI2AXIS Adapter Register Map.

ST 2022-6 Packetizer Module

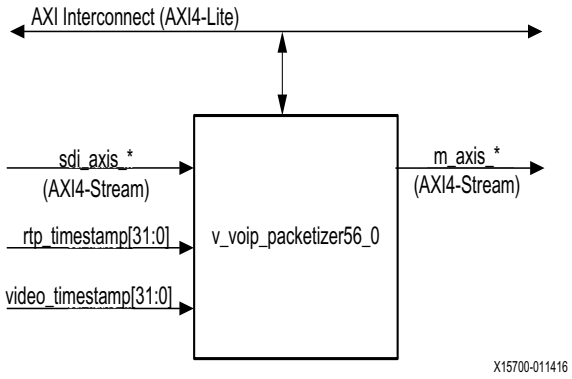

Figure 8: **ST 2022-6 Packetizer Module**

The ST 2022-6 Packetizer module accepts SDI over AXI4-Streams and transmits SMPTE2022-6 media datagrams to the VoIP FEC TX module. The packet information is defined in the TUSER signals. See Table 2-4 in the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

The ST 2022-6 Packetizer has an AXI4 Slave Interface (AXI4-Lite) that allows dynamic control of register settings. For more information on the register space, refer to the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

RTP over AXI4-Stream Interface Protocol (Master Interface)

Table 3: **RTP over AXI4-Stream Interface Protocol (Master Interface)**

Signals	Direction	Description			
m_axis_tvalid	Out	High only from the start to the end of the packet transfer.			
m_axis_tdata[63:0]	Out	Packet Data			
m_axis_tlast	Out	High only at the last word of the output packet			
m_axis_tuser[31:0]	Out	**Bit**	**Abbreviation**	**Description**	
		0	Packet Start	High only at the first valid word of the output packet.	
		2:1	Protocol Version	Protocol version (set to 00)	
		14:3	Channel Number	Shall be valid at packet start	
		15	Reserved		
		26:16	Packet Length	Shall be valid at payload start. It is the sum of packet length in bytes.	
		27	Reserved		
		31:28	Packet Type	0000	UDP Encapsulated Packet
				0001	RTP Encapsulated ST 2022-2 compliant Media Packet
				0010	RTP Encapsulated ST 2022-1 compliant Column FEC Packet
				0011	RTP Encapsulated ST 2022-1 compliant Row FEC packet
				0101	RTP Encapsulated ST 2022-6 compliant Media Packet
				0110	RTP Encapsulated ST 2022-5 compliant Column FEC Packet
				0111	RTP Encapsulated ST 2022-5 compliant Row FEC Packet
m_axis_tready	In	Asserted Low in between packet transfers.			

RTP over AXI4-Stream Interface Protocol (Slave Interface)

Table 4: **RTP over AXI4-Stream Interface Protocol (Slave Interface)**

Signals	Direction	Description			
s_axis_tvalid	In	High from the start to the end of the packet transfer.			
s_axis_tdata[63:0]	In	Packet Data			
s_axis_tlast	In	High at the last word of the output packet			
s_axis_tuser[31:0]	In	**Bit**	**Abbreviation**	**Description**	
		0	Packet Start	High only at the first valid word of the output packet.	
		2:1	Protocol Version	Protocol version (set to 00)	
		14:3	Channel Number	Shall be valid at packet start	
		15	Reserved		
		26:16	Packet Length	Shall be valid at payload start. It is the sum of the packet length in bytes.	
		27	Reserved		
		31:28	Packet Type	0000	UDP encapsulated
				0001	RTP Encapsulated ST 2022-2 compliant Media Packet
				0010	RTP Encapsulated ST 2022-1 compliant Column FEC Packet
				0011	RTP Encapsulated ST 2022-1 compliant Row FEC Packet
				0101	RTP Encapsulated ST 2022-6 compliant Media Packet
				0110	RTP Encapsulated ST 2022-5 compliant Column Packet
				0111	RTP Encapsulated ST 2022-5 compliant Row FEC Packet
s_axis_tready	Out	Asserted Low in between packet transfers.			

Notes:

1. The Video over IP (RTP) Master AXI4-Stream Protocol (Table 3) is compliant with Payload Out Interface for Generic VoIP FEC Transmitter, VoIP FEC Receiver, and Video over IP (RTP) Slave AXI4-Stream Protocol (Table 4) is compliant with the Payload In Interface for Generic VoIP FEC.

AXI4-Stream Switch

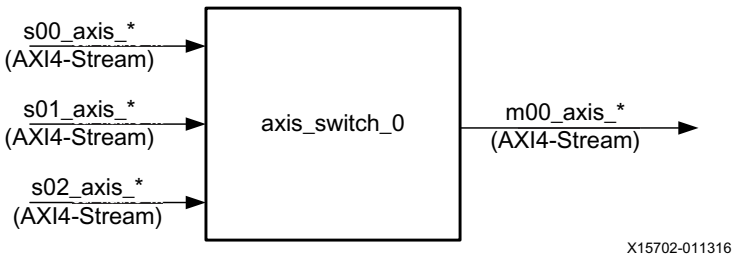

Figure 9: **AXI4-Stream Switch**

The AXI4-Stream switch in the VoIP TX subsystem is used to route three incoming RTP streams from three ST 2022-6 Packetizer module into a single master AXI4-Stream which is routed to VoIP FEC TX. See the *AXI4-Stream Interconnect v1.1 LogiCORE IP Product Guide* (PG035) [Ref 16].

Video over IP (VoIP) FEC TX

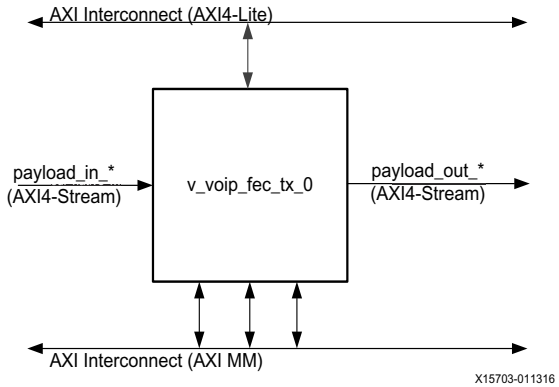

Figure 10: **Video over IP (VoIP) FEC TX**

The Video over IP FEC Transmitter (VoIP FEC TX) in the reference design (Figure 10) is used to generate FEC packets (SMPTE2022-5 Forward Error Correction). The VoIP FEC TX core has three AXI-Memory Map interfaces to access the DDR3 SDRAM through the AXI4 Interconnect and the Memory Interface Group (MIG). The memory map address range is fixed from 0xC0000000 to 0xFFFFFFFF.

The VoIP FEC TX has an AXI4 Slave Interface (AXI4-Lite) that allows dynamic control of register settings. For more information on the register space, refer to the *Video over IP Transmitter LogiCORE IP Product Guide* (PG206) [Ref 4].

In the reference design, the core is set to a specific configuration as shown in Table 5, but you have the flexibility to change the configuration by editing the provided software or configuring via the universal asynchronous receiver-transmitter (UART) terminal.

Table 5: **VoIP FEC TX Reference Design Configuration**

Channel	FEC Mode	Block Align	FEC L	FEC D
1	2D	Block Align	77	77
2	2D	Block Align	77	77
3	2D	Block Align	77	77

AXI4-Stream Broadcaster

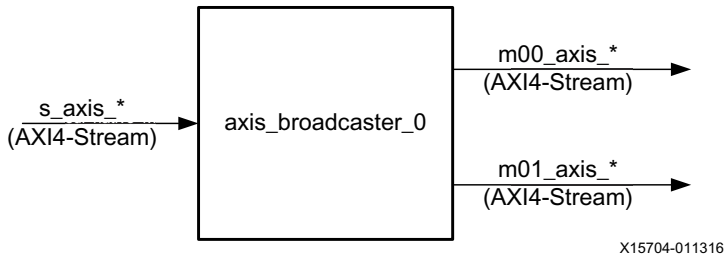

Figure 11: **AXI4-Stream Broadcaster**

The AXI4-Stream Broadcaster in the VoIP TX reference design is used to broadcast an incoming single RTP stream (from VoIP FEC TX) into two identical RTP streams. These identical RTP streams are fed into two Framer modules for seamless switching support. For more information on the AXI4-Stream Broadcaster, refer to the *AXI4-Stream Infrastructure IP Suite LogiCORE IP Product Guide* (PG085) [Ref 15].

Framer Module

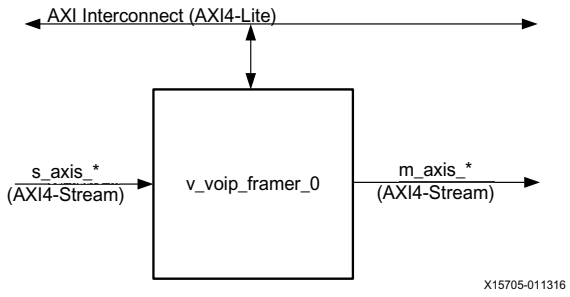

Figure 12: **Framer Module**

The Framer module adds the Ethernet/IP/UDP headers that you configure into the incoming RTP datagram in order to construct the Ethernet Datagram. The Ethernet Datagram is sent out via a 10 Gb/s Ethernet link.

The Framer has an AXI4 Slave Interface (AXI4-Lite) that allows dynamic control of register settings. For more information on the register space, refer to the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

Note: The Framer `s_axis_*` interface is compliant to the RTP over IP RTP AXI4-Stream Protocol and the Framer `m_axis_*` interface is compliant to the 10G Ethernet Subsystem (PG157) [Ref 7]`s_axis_tx` interface.

Video over IP RX Subsystem

The VoIP RX subsystem receives SMPTE2022-5 Ethernet packets (if present) and SMPTE2022-6 Ethernet packets from two 10-Gigabit Ethernet MAC cores. The Ethernet packet headers (MAC, IP, and UDP) are stripped off to form an RTP packet for downstream module operations.

The VoIP RX subsystem is capable of recovering lost SMPTE2022-6 packets (a.k.a RTP packets) from SMPTE2022-5 packets (a.k.a FEC packets) by using the SMPTE2022 Forward Error Correction method. The subsystem supports seamless switching giving additional protection by receiving two identical Ethernet streams (SMPTE2022-7).

The VoIP RX reference design consists of two main component which are the Video over IP RX subsystem and the SDI Back End interface system shown in Figure 13 and Figure 14.

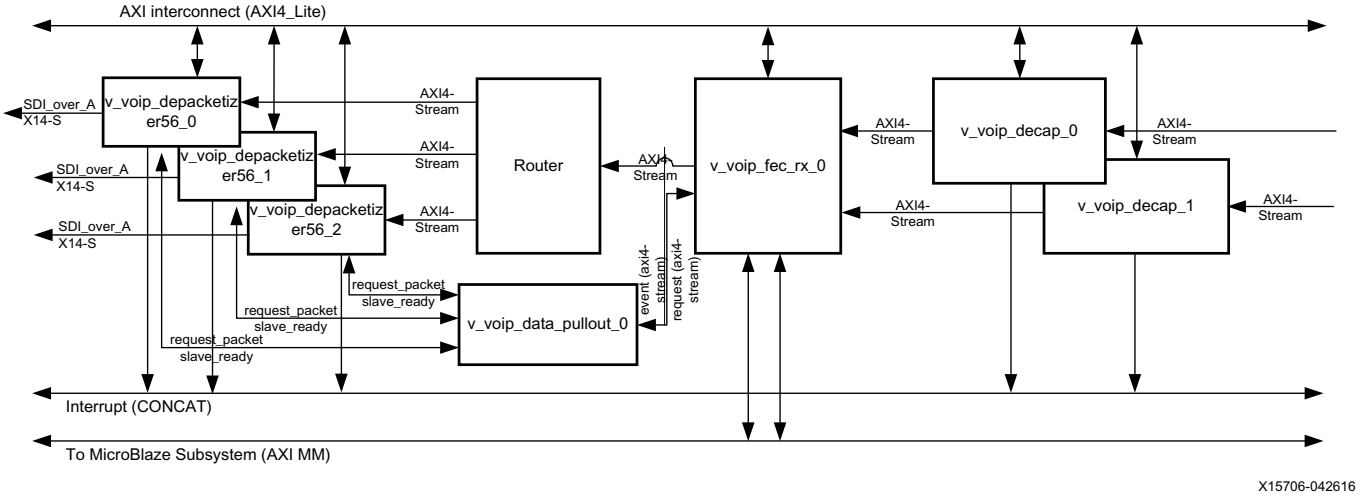

Figure 13: **Video over IP Receiver Subsystem (3 Channel Configuration)**

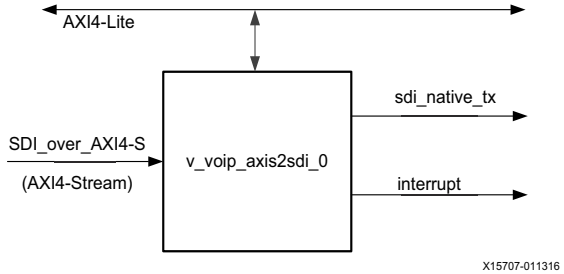

Figure 14: **SDI Back End Interface (Channel 0)**

Note: The AXIS2SDI Adapter `SDI_over_AXI4-S` interface is compliant to the SDI over AXI4-Stream Protocol (Table 2).

In the reference design, the VoIP RX subsystem receives two identical SMPTE2022-5/6 Ethernet datagrams from the Dual-Link 10-Gigabit Ethernet and transmits three streams via SDI over the AXI4-Stream interface.

The VoIP RX subsystem consists of the following modules.

- **Decapsulator Module**: Performs header stripping (output RTP packets), channel matching and filtering, and video stream detection.
- **VoIP FEC RX**: Recovers SMPTE2022-6 lost packets and provides seamless switching support.
- **Data Pullout**: Generates packet requests to the VoIP FEC RX based on the packet requests from the ST 2022-6 Depacketizer module.
- **Router**: Routes accepted RTP packets from VoIP FEC RX to multiple ST 2022-6 Depacketizer modules based on the channel number.
- **ST 2022-6 Depacketizer Module**: Converts received SMPTE2022-6 packets into SDI over AXI4-Stream.

Decapsulator Module

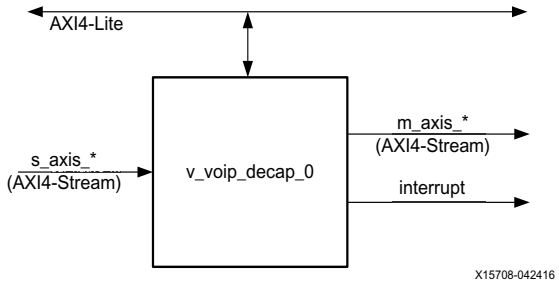

Figure 15: **Decap Module**

The Decapsulator module receives SMPTE2022 5/6 Ethernet datagrams and transmits SMPTE2022 5/6 RTP datagrams to the VoIP FEC RX core via the AXI4-Stream interface. The Decapsulator module performs operations such as channel mapping per channel header filtering settings, stripping the Ethernet, IP and UDP header forming a RTP packet, SDI-Video detection based on the media payload header information in the received number of SMPTE2022-6 packets, and generates a few packet stream interrupts to notify the system about the packet stream change.

The Decapsulator has an AXI4 Slave Interface (AXI4-Lite) that allows dynamic control of register settings. For more information on the register space, refer to the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

Register configuration is required for normal Decapsulator module operation.

Note: The Decapsulator `s_axis_*` interface is compliant to the 10G Ethernet Subsystem (PG157) [Ref 7] `m_axis_rx` interface and the Decapsulator `m_axis_*` interface is compliant to the RTP over IP RTP AXI4-Stream Protocol.

Table 6: **Decap Module Configured Register Before Operation**

Address (HEX)	Register Name	Value				
General Space		**Bit**	**Parameter**	**Value**		
0x0014	Packet Lock/Unlock Window	31:16	Packet Lock Window	6,000		
		15:0	Packet Unlock Window	1,024		
0x002C	module_ctrl	0	Module Enable	1		
Channel Space		**Bit**	**Parameter**	**Ch 1**	**Ch 2**	**Ch 3**
0x0080	Channel Control	1	Lossless Mode	0	0	0
		0	Channel Enable	0	0	0
0x0084	Channel Timeout	31:0	Channel Timeout	156,250,000	156,250,000	156,250,000
0x0090	Match VLAN Header	31	VLAN Filter Enable	0	0	0
		11:0	VLAN Identifier	0xB00	0xB10	0xB20
0x0094	Match Destination IP Address	31:0	IP Destination Address	0xC0A80064	0xC0A80164	0xC0A80264
0x00A4	Match Source IP Address	31:0	IP Source Address	0xC0A80032	0xC0A80132	0xC0A80232
0x00B4	Match UDP Source Port	15:0	UDP Source Port	0x0010	0x0020	0x0030
0x00B8	Match UDP Destination Port	15:0	UDP Destination Port	0x0010	0x0020	0x0030
0x00BC	Match SSRC	31:0	SSRC	0x12345600	0x12345610	0x12345620
0x00C0	Match Select Setting	5	Match SSRC	0	0	0
		4	Match UDP Destination Port	1	1	1
		3	Match UDP Source Port	0	0	0
		2	Match IP Destination Addr.	0	0	0
		1	Match IP Source Addr.	0	0	0
		0	Match VLAN Identifier	0	0	0

In the reference design, the Decapsulator module controls the incoming packet flow. The Decapsulator module generates three types of interrupts (per channel) based on changes in the incoming stream behavior which are:

- Packet Lock Interrupt:
  Packet Lock Interrupt occurs when the Decapsulator module receives N-number of consecutive ST 2022-6 Ethernet Packets with the same video format (by looking at the media payload header).

  Note: You can configure the N-Number which is the Packet Lock window.

- Packet Unlock Interrupt:
  Packet Unlock Interrupt occurs when the Decapsulator module receives N-number of consecutive ST 2022-6 Ethernet Packets with different video formats compared with the locked video format.

  Note: You can configure the N-Number which is the Packet Unlock window.

- Packet Stop Interrupt (Timeout):
  No ST 2022-6 Ethernet Packet is received within the configured channel timeout time after a stable and locked video stream (packet stream).

Video over IP (VoIP) FEC RX

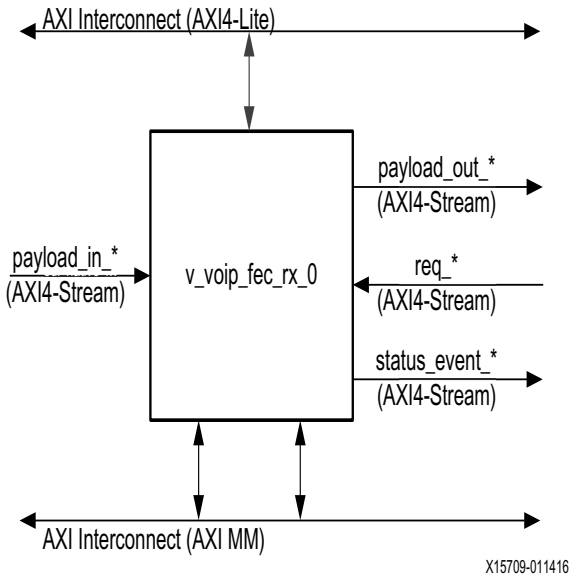

Figure 16: **VoIP FEC RX**

The Video over IP FEC Receiver (VoIP FEC RX) in the reference design (Figure 16) is used to recover lost SMPTE2022-6 packets (a.k.a RTP packets) by using the SMPTE2022-5 Forward Error Correction method. VoIP FEC RX supports seamless switching defined in SMPTE2022-7 by receiving two identical streams.

The VoIP FEC RX core has two AXI-Memory Map interfaces to access the DDR3 SDRAM through the AXI4 interconnect and Memory Interface Group (MIG). The memory map address range is fixed from 0xC0000000 to 0xFFFFFFFF.

The VoIP FEC RX has an AXI4-Lite interface that allows dynamic control of register settings. For more information on the register space, refer to the *Video Over IP FEC Receiver LogiCORE IP Product Guide* (PG207) [Ref 5].

In the reference design, this core is configured as in Table 7. You have the flexibility to change the configuration to the provided software or configure via a UART Terminal.

Table 7: **VoIP FEC RX Configuration**

Channel	FEC Recovery Enable	Media Packet Bypass	Channel Enable
1	Enable	Disable	Enable
2	Enable	Disable	Enable
3	Enable	Disable	Enable

Router

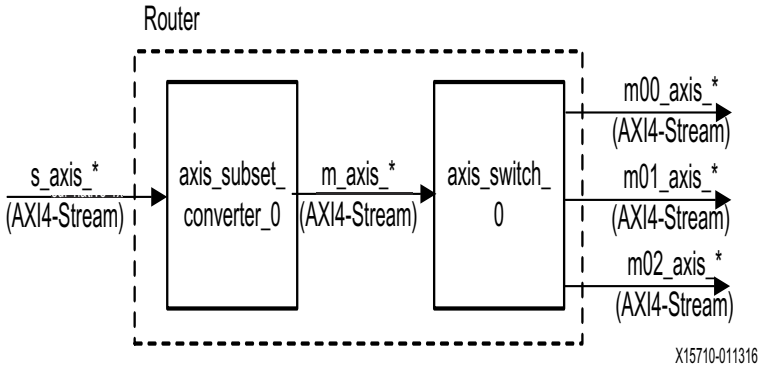

Figure 17: **AXI4-Stream Switch**

The Router is used to map the channel number and route the incoming stream from VoIP FEC RX into three different streams that are targeted to dedicated ST 2022-6 Depacketizer modules based on the channel number. The AXI4-Stream Subset Converter is used to map the Channel Number from PAYLOAD_OUT_TUSER into M_AXIS_TDEST of the AXI4-Stream Subset Converter.

Data Pullout

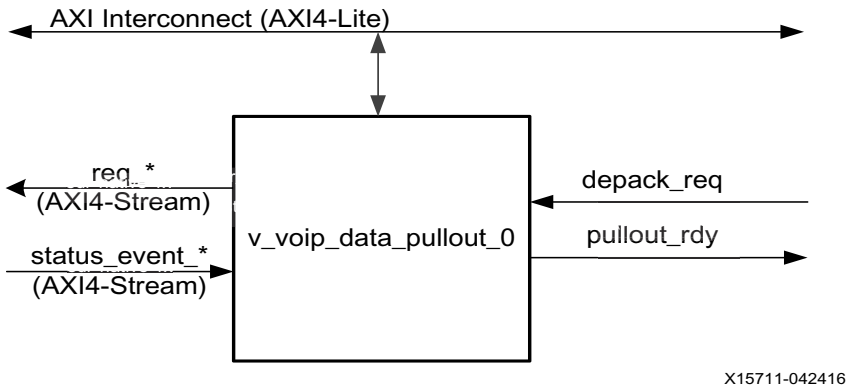

Figure 18: **Data Pullout**

The Data Pullout module is a channel playout controller that receives events from and sends packet requests to the FEC RX. It monitors the current packet buffer for each channel and sends a playout ready signal to the ST 2022-6 Depacketizer module when a user-targeted threshold is reached. The ST 2022-6 Depacketizer module requests new packets through the Data Pullout module by sending a pulse for each request.

The Data Pullout module has an AXI4-Lite interface that allows dynamic control of register settings. See Table 9.

Note: The Data Pullout `status_event_*` and the Data Pullout `req_*` interfaces are compliant to the Stream Event Signals of VoIP FEC RX (PG207) [Ref 5].

ST 2022-6 Depacketizer Module

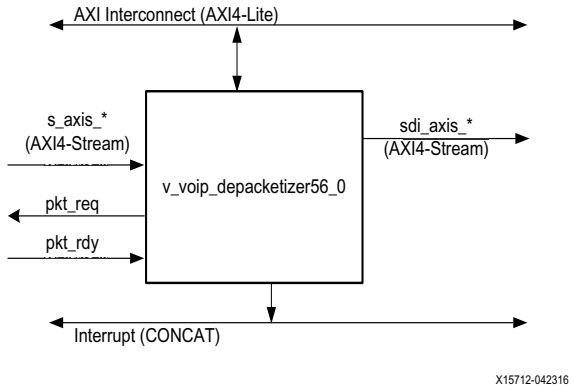

Figure 19: **ST 2022-6 Depacketizer Module**

The ST 2022-6 Depacketizer module converts RTP packets back to SDI over an AXI4-Stream video stream. The incoming stream packet type must be an SMPTE2022-6 RTP packet. The RTP header and media payload header are stripped off. The input to ST 2022-6 Depacketizer module must comply with the v_voip_fex_rx payload out of AXI4-Stream protocol (RTP over AXI4-Stream protocol).

The `pkt_req` output port of the ST 2022-6 Depacketizer module is connected to the data_pullout module for the packet request process. Whenever a packet is processed and sent out, a new `pkt_req` pulse is sent from the ST 2022-6 Depacketizer module to the v_voip_data_pullout module. Note that `pkt_req` can only be sent when `pkt_rdy` is High. (Once `pkt_rdy` is High, it will always remain High until the ST 2022-6 Depacketizer module reset). For more information on this `pkt_req` and `pkt_rdy` timing protocol, refer to the ST 2022-6 Depacketizer section in the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

Note: The ST 2022-6 Depacketizer `s_axis_*` interface is compliant to the RTP over AXI4-Stream Protocol and the `m_axis_*` is compliant to the SDI over AXI4-Stream Protocol (Table 2).

The ST 2022-6 Depacketizer has an AXI4 Slave Interface (AXI4-Lite) that allows dynamic control of register settings. For more information on the register space, refer to the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

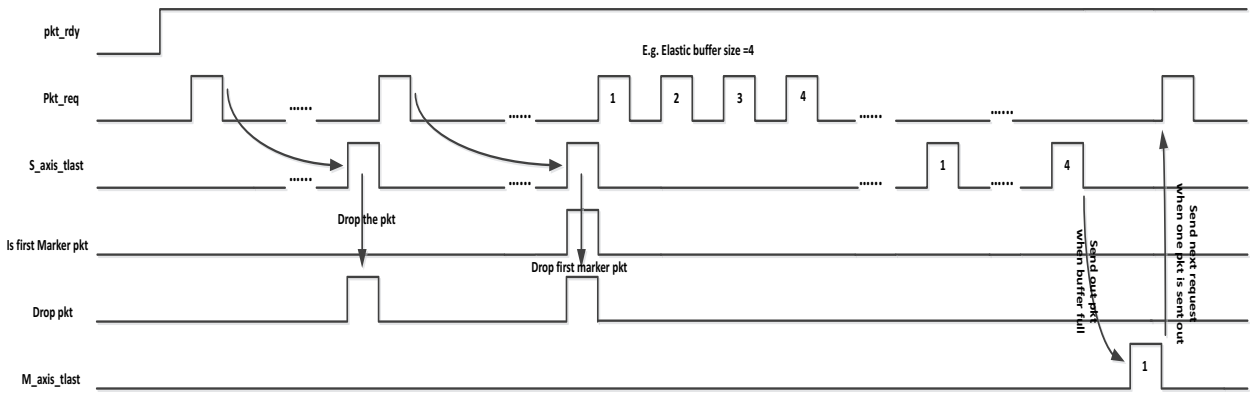

Figure 20: **Request Packet Timing Diagram**

SDI Back End Interface (AXIS2SDI Adapter)

The AXIS2SDI Adapter (Figure 14) converts the SDI over AXI4-Stream video stream back to a native SDI video stream. On the sdi_over_axis interface, it is in the core clock domain. On the native SDI interface, it is in the SDI clock domain.

The AXIS2SDI Adapter module has an AXI4-Lite interface that allows dynamic control of register settings. See SDI2AXIS Adapter Register Map.

SMPTE SD/HD/3G-SDI

The SMPTE SDI core provides transmitter and receiver interfaces for the SMPTE SD-SDI, HD-SDI and 3G-SDI standards. The core is connected to 7 series FPGA GTX transceivers for serialization and deserialization of the SDI video streams. The SMPTE SDI receiver uses a 148.5 MHz GTX transceiver reference clock frequency to receive its supported SDI bit rates. The receiver automatically determines the incoming SDI bit rate and configures itself and the GTX transceiver appropriately for that SDI mode. The SMPTE SDI transmitter requires two different GTX transceiver reference clock frequencies for its supported SDI bit rates. 148.5 MHz and 148.35 MHz are used in the design. The clock multiplexer built into the GTX transceiver switches between these two reference clocks. A port dynamically controls the operating SDI mode for the transmitter. The transmitter, in turn, controls the GTX transmitter through the DRP to provide the appropriate configuration for each SDI mode. See the *SMPTE SD/HD/3G-SDI Product Guide* (PG071) [Ref 6] for more information.

Ten Gigabit Ethernet MAC Subsystem

The 10-Gigabit Ethernet MAC Subsystem instances on the transmitter side has the AXI4-Stream transmit interfaces connected to the output of the VoIP Modular Transmitter Subsystem Framer module. The 10-Gigabit Ethernet MAC Subsystem instances on the receiver side has the AXI4-Stream receive interfaces connected to the input of the VoIP Modular Receiver Subsystem Decap module. See the *10 Gigabit Ethernet Subsystem Product Guide* (PG157) [Ref 7] for more information.

AXI Interconnect (AXI-MM Interface)

This AXI4 interconnect instance provides the high FMAX and throughput for the design with a 256-bit core data width and a 200 MHz clock frequency (System Clock of the Reference Design). The AXI4 interconnect core data width and clock frequency match the capabilities of the attached AXI4 MIG so that width and clock converters are not required between them. Setting the AXI4 interconnect core data width and clock frequency below the native width and clock frequency of the memory controller creates a bandwidth bottleneck within the system. To help meet the timing requirements of a 512-bit AXI4 interface at 200 MHz, a rank of register slices are enabled between the AXI_MM interconnect and the AXI4 MIG. Together, the AXI4 interconnect and AXI4 MIG form a 4-port AXI4 MPMC connected to four AXI4 external master connectors. The configuration of this AXI4 interconnect is consistent with the system performance optimization recommendations for an AXI4 MPMC-based system as described in the *AXI Reference Guide* (UG1037) [Ref 8]. Also see the *AXI Interconnect LogiCORE IP Product Guide* (PG059) [Ref 17].

Memory Interface Generator

The Memory Interface Generator (MIG) forms the single slave connected to the AXI4 Interconnect. The MIG AXI4 interface is 512-bits wide, runs at 133.25 MHz, and disabled narrow burst support for optimal throughput and timing. This configuration matches the native AXI4 interface clock and width corresponding to a 64-bit DDR3 DIMM with 533.33 MHz memory clock which is the nominal performance of the memory controller for a Kintex-7 device with a -2 speed grade. Register slices are enabled to ensure that the interface meets timing at 133.25 MHz. These settings help ensure that a high degree of transaction pipelining is active to improve system throughput. See the *7 Series FPGAs Memory Interface Solutions User Guide* (UG586) [Ref 9] for more information about the memory controller.

AXI Interconnect (AXI4-Lite Interface)

The MicroBlaze processor data peripheral (DP) interface master writes and reads to all AXI4-Lite slave registers in the design for control and status information. These interconnects are 32 bits and do not require high FMAX and throughput. Therefore, they are connected to a slower FMAX portion of the design by a separate AXI Interconnect. The AXI4-Lite Interconnect block is configured for shared-access mode because high throughput is not required in this portion of the design. Therefore, the area can be optimized over performance on this interconnect block. Also, this interconnect is clocked at 100 MHz to ensure that synchronous integer ratio clock converters in the AXI Interconnect can be used, which offer lower latency and less area than asynchronous clock converters. The slaves on the AXI4-Lite Interconnect are AXI UART Lite, AXI Interrupt Controller, SDI2AXIS Adapter (SDI Front End Interface) and VoIP TX Subsystem, VoIP RX Subsystem and AXIS2SDI Adapter (SDI Back End Interface). Also see the *AXI Interconnect LogiCORE IP Product Guide* (PG059) [Ref 17].

System Module Register Space

In the reference design, there are multiple modules that have an AXI4-Lite interface that allows dynamic control of the parameters within the modules from a processor. This section describes the details for each of the module parameters. The register map for modules that are not described in the following sections can be found in the *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241) [Ref 14].

SDI2AXIS Adapter Register Map

Table 8: **SDI2AXIS Register Map**

Address (HEX)	Register Name	Access Type	Clear with Soft Reset	Default Value (HEX)	Description	
					Bit Range	Register Description
0x0000	control	R/W	N	0x00000000	**Control**	
					31:1	Reserved
					0	**Soft Reset**
						1 Reset all pcore registers.
						0 Unset all pcore registers.
0x0004	Status	R	N	0x00000000	**Status**	
					31:2	Reserved
					1	Video_locked
						0 No video is locked in hardware.
						1 Video is locked in hardware.
					0	Reserved
0x0008	video_lock_ window	R/W	N	0x00000080	**Video Lock Window**	
					31:16	Reserved
					15:0	Number of sdi clk cycles used to lock the video after first start of frame detected (SOF).
0x000C	version	R	No	0x01000000	**Hardware Version**	
					31:24	Version Major
					23:16	Version Minor
					15:12	Version Revision
					11:8	Patch ID
					7:0	Revision Number

Table 8: **SDI2AXIS Register Map *(Cont'd)***

Address (HEX)	Register Name	Access Type	Clear with Soft Reset	Default Value (HEX)	Description	
					Bit Range	Register Description
0x0010	module_ctrl	R/W	N	0x00010000	**Module Control**	
					31:3	Reserved
					2	Lossless mode
						0 Normal mode
						1 Lossless mode
					1	Transmit Enable
						0 Transmission disabled
						1 Transmission enabled
					0	Module Enable
						0 Module disabled
						1 Module enabled
0x0014	video_format	R	Yes	0x00000000	**Video Format**	
					31:4	Reserved
					3	**RX Bit Rate**
						1 148.5/1.001 MHz (3G and SD) or 74.25/1.001 MHz (HD)
						0 148.5 MHz (3G and SD) or 74.25 MHz (HD)
					2	**3G Level B**
						1 3G-Level B
						0 3G-Level A
					1:0	**SDI Mode**
						00 HD-SDI
						01 SD-SDI
						10 3G-SDI
						11 Invalid
0x0018	vid_src_fmt	R	Yes	0x00000000	**Video Source Format**	
					31:28	MAP
					27:20	FRAME
					19:12	FRATE
					11:8	SAMPLE
					7:0	Reserved
0x001C	frame_cnt	R	Yes	0x00000000	**SDI Frame Counter**	
					31:0	Incoming SDI frame counter

Table 8: **SDI2AXIS Register Map *(Cont'd)***

Address (HEX)	Register Name	Access Type	Clear with Soft Reset	Default Value (HEX)	Description	
					Bit Range	Register Description
0x0020	stat_reset	W	Yes	0x00000000	**Statistic Reset**	
					31:1	Reserved
					0	SDI Frame Counter Reset (Self Clear)
					1	Reset SDI Frame Counter
0x0024	buffer_reset	W	Yes	0x00000000	**SDI2AXIS Adapter Buffer Reset**	
					31:1	Reserved
					0	Module Buffer Reset (Self Clear)
					1	Reset module buffer
0x0040	interrupt_status	R	Yes	0x00000000	**Interrupt Status**	
					31:3	Reserved
					2	SDI2AXIS Adapter FIFO Full
					1	Interrupt is still valid
					0	Interrupt is cleared
					1	SDI Unlock
					1	Interrupt is still valid
					0	Interrupt is cleared
					0	SDI Lock
					1	Interrupt is still valid
					0	Interrupt is cleared
0x0044	interrupt_mask	R/W	Yes	0x00000000	**Interrupt Mask**	
					31:3	Reserved
					2	SDI2AXIS Adapter FIFO Full
					1	Do not mask
					0	Mask
					1	SDI Unlock
					1	Do not mask
					0	Mask
					0	SDI Lock
					1	Do not mask
					0	Mask

Table 8: **SDI2AXIS Register Map *(Cont'd)***

Address (HEX)	Register Name	Access Type	Clear with Soft Reset	Default Value (HEX)	Description	
					Bit Range	Register Description
0x0048	interrupt_clear	W	Yes	0x00000000	**Interrupt Clear**	
					31:3	Reserved
					2	SDI2AXIS Adapter FIFO Full (Self Clear)
					1	Clear Interrupt Status for corresponding bit
					1	SDI Unlock (Self Clear)
					1	Clear Interrupt Status for corresponding bit
					0	SDI Lock (Self Clear)
					1	Clear Interrupt Status for corresponding bit

control (0x0000)

Bit 0 is a soft reset that is used to reset all other registers.

Status (0x0004)

Bit 0 is reserved.

Bit 1 indicates the hardware video locked status, which is used by software for decision making if multiple interrupts seen at the same time by software due to the delay of software processing.

Bit 2 and Bit 3 indicate the module packet buffer status.

Video_lock_window (0x0008)

This is the number of `sdi_clk` cycles for the video to be asserted as locked when first line is detected. It is used to filter out an unstable lock signal from the SDI core. During the locking window, if either the `rx_mode_locked` or `rx_t_locked` signal goes down, the locking process will start over again.

module_ctrl (0x0010)

Bit 0 is used to enable the module during system initialization or system enabling.

- 0 - Mutes the module and drops video data at the input port; no video detection, no interrupt will be fired.
- 1 - Activate the module and let it start detecting the video format and buffering incoming video data.

Bit 1 is used to enable the transmission of the module.

- 0 - Silently drops video data at input and does not buffer video data into an internal buffer.
- 1 - Buffers video data and transmits to downstream module.

Bit 2 is normal mode or the lossless mode selection bit.

- 0 - Normal mode. Upon video lock, adapter does self-disable. Self-disable means self disable the transmit. In this mode, module enable should be enabled during the initialization phase. Transmit enable is enabled by software after the video lock interrupt is received.
- 1 - Lossless mode. Upon video lock, adapter does not self-disable. It starts buffer data when start of frame (SOF) is detected. In this case, module enable and transmit enable should be both enabled during the initialization phase because you do not want to waste any video data in lossless mode.

video_format (0x0014)

This video format information comes directly from the SDI core and is latched into this register for user information.

vid_src_fmt (0x0018)

This is the media header second word "map_frame_frate_sampe_fmt". Refer to *ST 2022-6:2012 - Transport of High Bit Rate Media Signals over IP Networks (HBRMT)* [Ref 10]. When the `vpid` signal is valid from the SDI core, the `vid_src_fmt` is derived from `vpid` (`sdi_rx_a_vpid`) from the SDI core. When the `vpid` signal is invalid from the SDI Core, the `vid_src_fmt` is derived from `t_*` signals (`sdi_rx_t_family`, `sdi_rx_t_rate`, `sdi_rx_tscan`, `sdi_rx_bit_rate`...) from the SDI core.

frame_cnt (0x001C)

This statistics register shows the number of start of frames detected at the input of the module when the video is locked. The statistics are active only when the module is enabled but independent of transmit enable settings.

stat_reset (0x0020)

Each bit of this register resets the corresponding statistics registers to zero. This stat_reset register is self-clearing; you only need to program once for statistics reset. Bit 0 is used to reset the frame count statistics register to zero

Note: Module disable also resets the frame count register to zero).

Data Pull Out

Table 9: **Data Pull Out Registers**

Address Offset (HEX)	Register Name	Access Type	Default Value (Hex)	Description	
				Bit Range	Value
General					
0x0000	control	R/W	0x00000000	**Control**	
				31:2	Reserved
				1	Channel Update Allow configured registers to take effect for the channel setting in offset 0x08.
				0	Reserved
0x0008	channel_access	R/W	0x00000000	**Channel Access**	
				31:3	Reserved
				2:0	The channel to access for the registers in channel space
0x000C	sys_config	R	Propagated from the GUI Number of Channel Settings	**System Configuration**	
				31:3	Reserved
				2:0	Number of channels supported
0x0010	version	R	0x01000000	**Hardware Version**	
				31:24	Version major
				23:16	Version minor
				15:12	Version revision
				11:8	Patch ID
				7:0	Revision number
Channel					
0x0080	chan_conf	R/W	0x00000000	**Channel configuration**	
				31:1	Reserved
				0	Channel enable
					1 Enable channel
					0 Disable channel
0x0084	chan_buf_thres	R/W	0x00000000	**Channel Buffer Threshold**	
				31:0	Sets the threshold for the channel request to kick start

Table 9: **Data Pull Out Registers**

Address Offset (HEX)	Register Name	Access Type	Default Value (Hex)	Description	
				Bit Range	Value
0x0088	chan_buf_depth	R	0x00000000	**Channel Buffer Depth**	
				31:0	Stores current event buffer depth for the channel
0x008C	chan_pend_req	R	0x00000000	**Channel Pending Request**	
				31:0	Stores current pending requests for the channel

control (0x0000)

Bit 1 is used to update the channel at offset 0x08 with the register settings in channel space.

channel access (0x0008)

Sets the channel to program. Currently limited to 8 channels. Works with the module's MAX_CHANNELS generic in the GUI.

sys_config (0x00C)

Reflects the module's MAX_CHANNELS generic in the GUI. Indicates the number of channels the module can support in the hardware.

version (0x0010)

Module current version.

chan_config (0x0080)

Bit 0 enables the channel for operation. It is also used to reset the channel when disabled.

chan_buf_thres (0x0084)

Sets the number of packets to accumulate in the FEC RX module before requesting for packets. When the targeted threshold is reached, the module signals ready to the ST 2022-6 Depacketizer module and the request process begins.

chan_buf_depth (0x0088)

Shows the current channel buffer depth extracted from the status event coming from the FEC RX. This value is compared with Chan_buf_thres to start the request process.

chan_pend_req (0x8C)

Shows the pending requests from the ST 2022-6 Depacketizer module that are still not sent out to the FEC RX module.

AXIS2SDI Adapter

Table 10: **AXIS2SDI Adapter Register Map**

Address offset	Register Name	Access Type	Cleared with SOFT reset	Default Value	Description	
					Bit Range	Register Description
0x0000	control	R/W	N	0x00000000	**Control**	
					31:1	Reserved
					0	Soft reset
						1 Reset all pcore registers
						0 Unreset the pcore registers
0x000C	version	R	N	0x00000000	**Hardware version**	
					31:24	Version major
					23:16	Version minor
					15:12	Version revision
					11:8	Patch ID
					7:0	Revision number
0x0010	module_ctrl	R/W	Y	0x00000000	**Module Control**	
					31:2	Reserved
					1	Output Enable
						0 Output is disabled and outputs zero data.
						1 Output is enabled.
					0	Module Enable
						0 Module is disabled.
						1 Module is enabled.

Table 10: **AXIS2SDI Adapter Register Map *(Cont'd)***

Address offset	Register Name	Access Type	Cleared with SOFT reset	Default Value	Description	
					Bit Range	Register Description
0x0014	video_format	R/W	Y	0x00000000	**Video Format**	
					31	Module Video format and video source format Selection
						0: Take video format and video source format from SDI over AXIS interface Tuser bus.
						1: Take video format and video source format from programmed register value.
					30:4	Reserved
					3	tx_bit_rate is used to generate tx_bit_rate output
						0: Not divided by 1.001.
						1: Divided by 1.001.
					2	3G Level A/B is used to generate the level_b_3g output
						0: 3G Level A
						1: 3G Level B
					1:0	video_mode is used to generate video_mode output:
						00: HD
						01: SD
						10: 3G
						11: None.
0x0018	frame_cnt	R	Y	0x00000000	**Number Of Frames Received At Input**	
					31:0	Number of frames received. Adapter starts buffer video frames only when the module is enabled.
0x001C	stat_reset	W	Y	0x00000000	**Clear Statistics Registers**	
					31:1	Reserved
					0	Clear frame_cnt statistics registers:
						0: No clear
						1: Clear

Table 10: **AXIS2SDI Adapter Register Map *(Cont'd)***

Address offset	Register Name	Access Type	Cleared with SOFT reset	Default Value	Description	
					Bit Range	Register Description
0x0020	buffer_reset	W	Y	0x00000000	**Adapter Packet Buffer Clear**	
					31:1	Reserved
					0	Buffer reset
						0 Do not reset.
						1 Reset packet buffer to empty.

control (0x0000)

Bit 0 is used to reset all pcore registers.

module_ctrl (0x0010)

Bit 0 is used to enable or disable the module. When the module gets enabled, it starts to accept and process incoming data. Bit 1 is used to enable transmission of the module.

video_format (0x0014

Bit 31 is the video format selection bit. When bit 31 is set to 0, the video format used by the module is taken from the AXI4-Stream interface Tuser bus. When bit 31 is set 1, the video format used by the module is taken from the video_format register.

frame_cnt (0x0018)

This statistics register records the number of frames received at the `sdi_axis` slave interface when the module is enabled. It is reset by the stat_reset register.

stat_reset (0x001C)

Each bit of this register resets the corresponding statistics registers to zero. This stat_reset register is self-clearing; you only need to program once for statistics reset.

buffer_reset (0x0020)

Bit 0 is used to reset the FIFO inside the module when the software wants to clear the FIFO in order to prepare for a new video stream. The register is a self-clearing register; you only need to program once.

Software Application

The software application performs module initialization, core configuration, and the command selection process for the system. Application-level software and drivers for controlling the modular system are written in C.

Software Initialization

Software initializes the SDI Front End Interface and Video over IP (VoIP) Transmitter Subsystem for the Modular SMPTE2022-56 Transmitter and Video over IP (VoIP) Receiver and the SDI Back End Interface for Modular SMPTE2022-56 Receiver. Peripheral modules, UART Lite and Interrupt Controller, are initialized for both Modular VoIP SMPTE2022-56 TX and RX.

Software Initialization is shown in Figure 21 for the Modular SMPTE2022-56 TX and Figure 22 for the Modular SMPTE2022-56 RX.

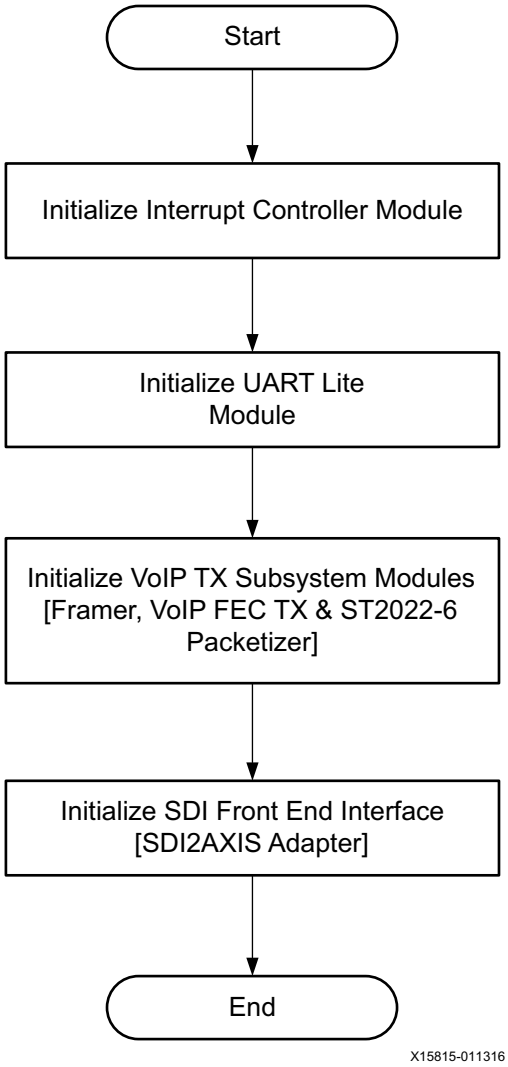

Figure 21: **Modular SMPTE2022-56 TX Initialization Process**

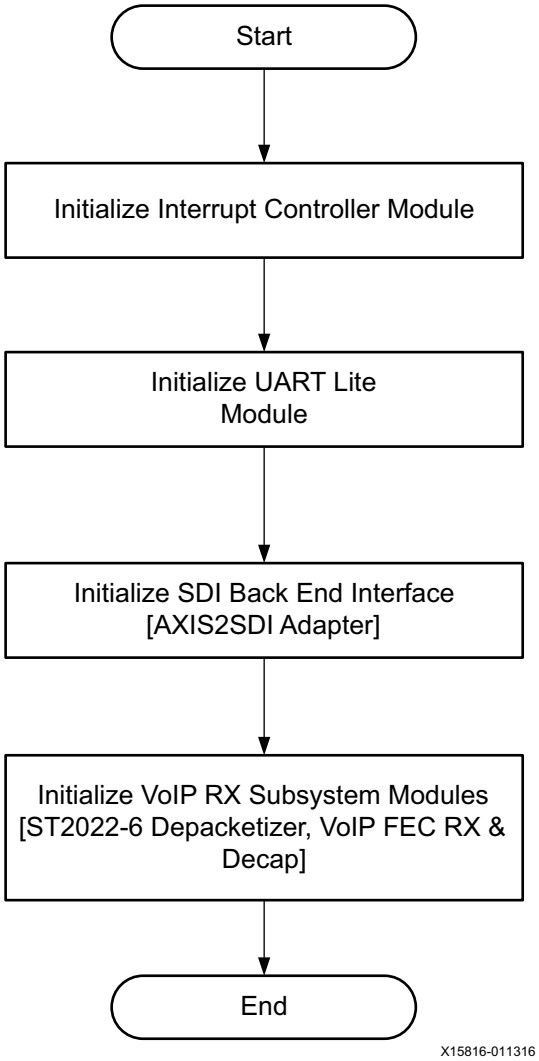

Figure 22: **Modular SMPTE2022-56 RX Initialization Process**

The initialization process performs base address mapping for each module and clears the entire register space for each module.

Modules/Core Configuration

After the initialization, the software application performs modules/core configuration for the system. The core/modules configuration flow is shown in Figure 23 for Modular SMPTE2022-56 TX and Figure 24 for Modular SMPTE2022-56 RX.

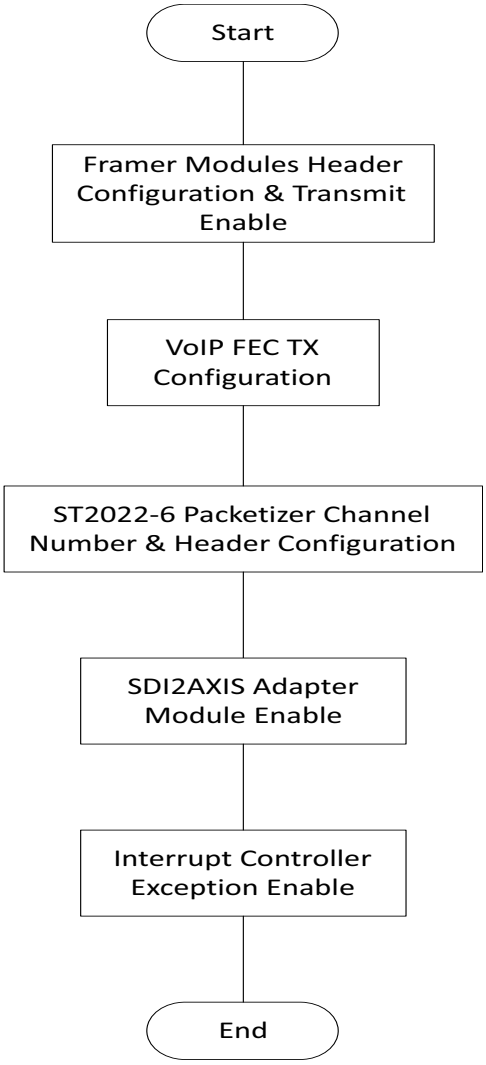

Figure 23: **Modular SMPTE2022-56 TX Configuration Flow**

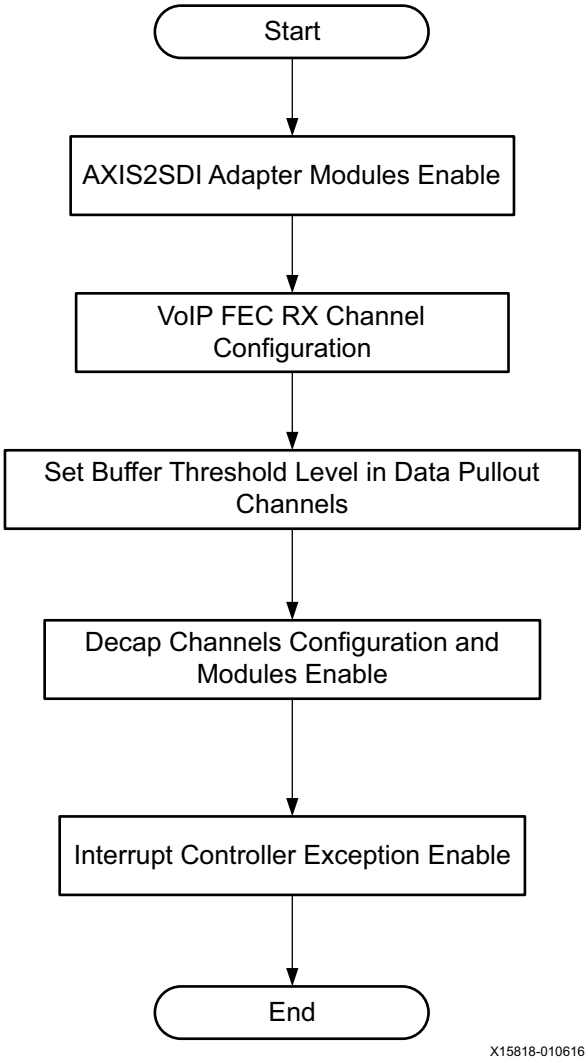

Figure 24: **Modular SMPTE2022-56 RX Configuration Flow**

Software Process Flow

The entire software flow for Modular SMPTE2022-56 TX and Modular SMPTE2022-56 RX is shown in Figure 25.

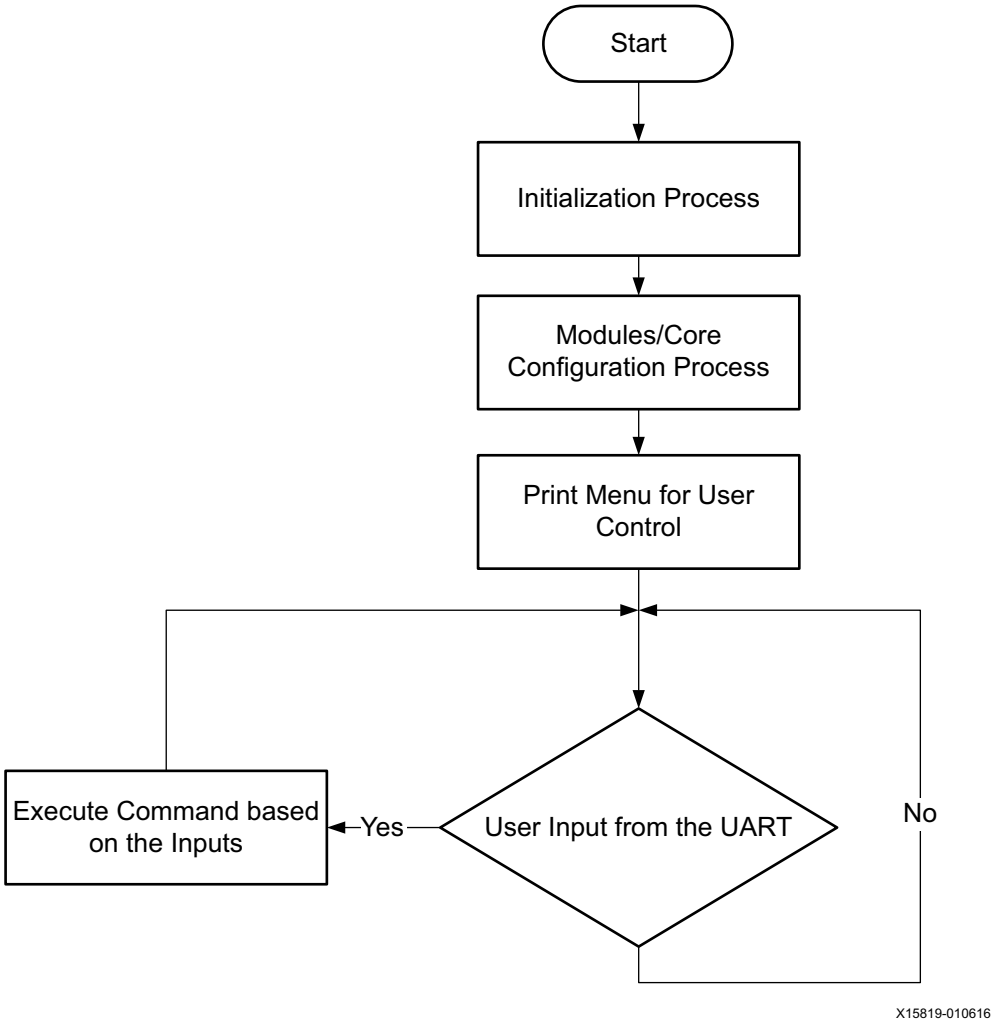

Figure 25: **Modular SMPTE2022-56 Transmitter & Receiver Software Flow**

Modular SMPTE2022-56 System Reset and System Enable

To perform system reset and system enable while the core has been configured and operating, follow the steps as shown in Figure 26 and Figure 27 for Modular ST 2022-6 TX and Figure 28 for Modular ST 2022-6 RX.

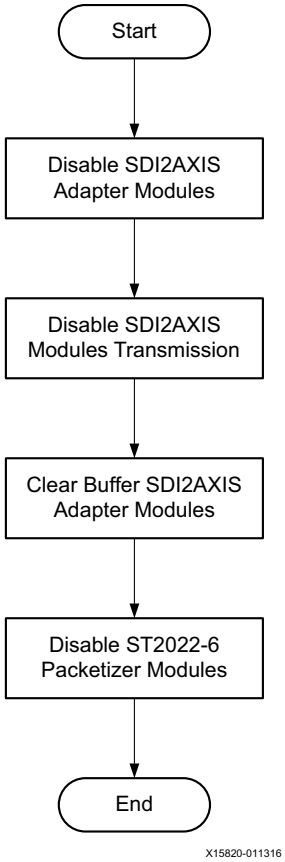

Figure 26: **Modular SMPTE2022-56 Transmitter System Reset Process**

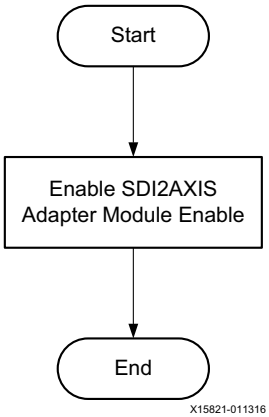

Figure 27: **Modular SMPTE2022-56 TX System Enable Process**

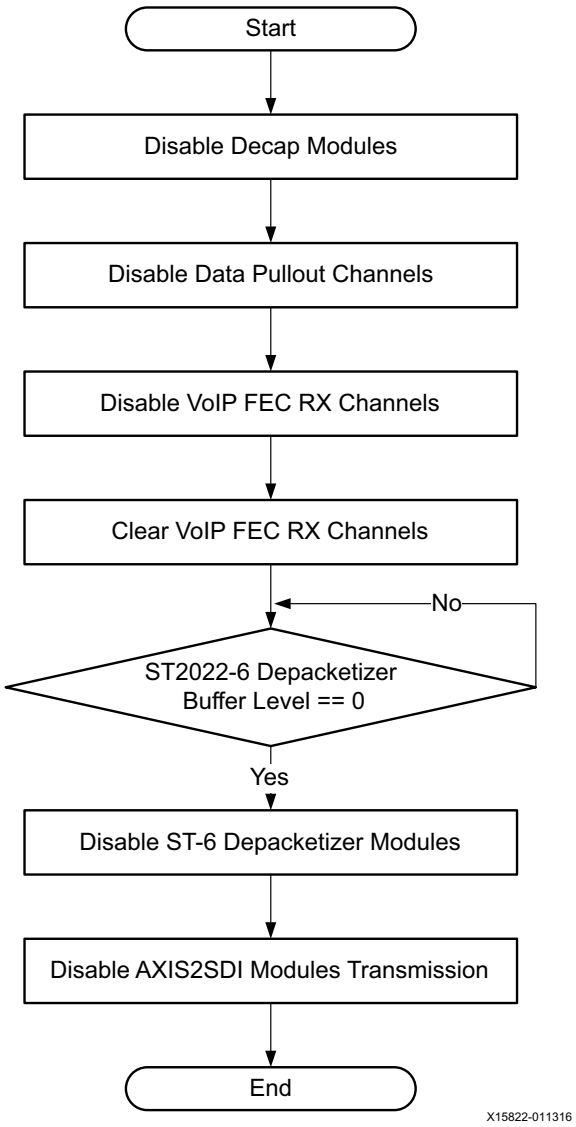

Figure 28: **Modular SMPTE2022-56 Receiver System Reset Process**

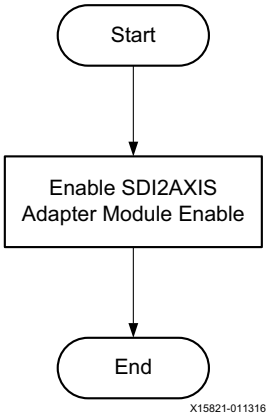

Figure 28: **Modular SMPTE2022-56 System Enable Process**

Software Interrupt Flow

Modular SMPTE2022-56 Transmitter

The Software Interrupt flow for the Modular SMPTE2022-56 Transmitter has two flows: during SDI Lock and SDI Unlock. The software flow during SDI Lock is shown in Figure 29 and SDI Unlock shown in Figure 30.

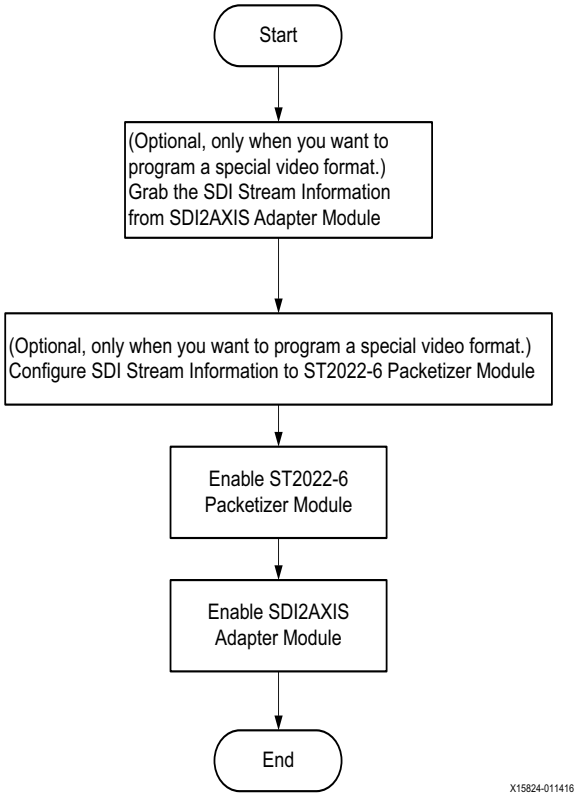

Figure 29: **SDI Lock Software Flow**

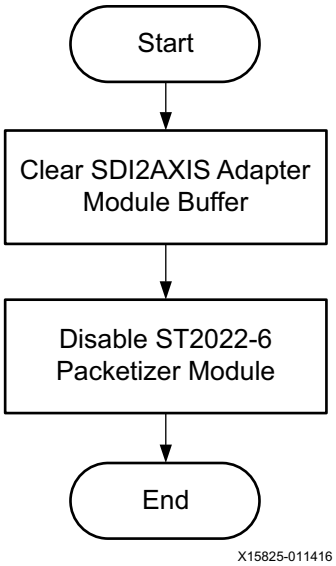

Figure 30: **SDI Unlock Software Flow**

Modular SMPTE2022-56 Receiver

The Software Interrupt flow for the Modular SMPTE2022-56 Receiver has two flows: during SMPTE2022-6 Packet Lock, SMPTE2022-6 Packet Unlock and ST 2022-6 Depacketizer Empty. The software flow during SMPTE2022-6 Packet Lock is shown in Figure 31. SMPTE2022-6 Packet Unlock is shown in Figure 32 and ST 2022-6 Depacketizer Empty is shown in Figure 33.

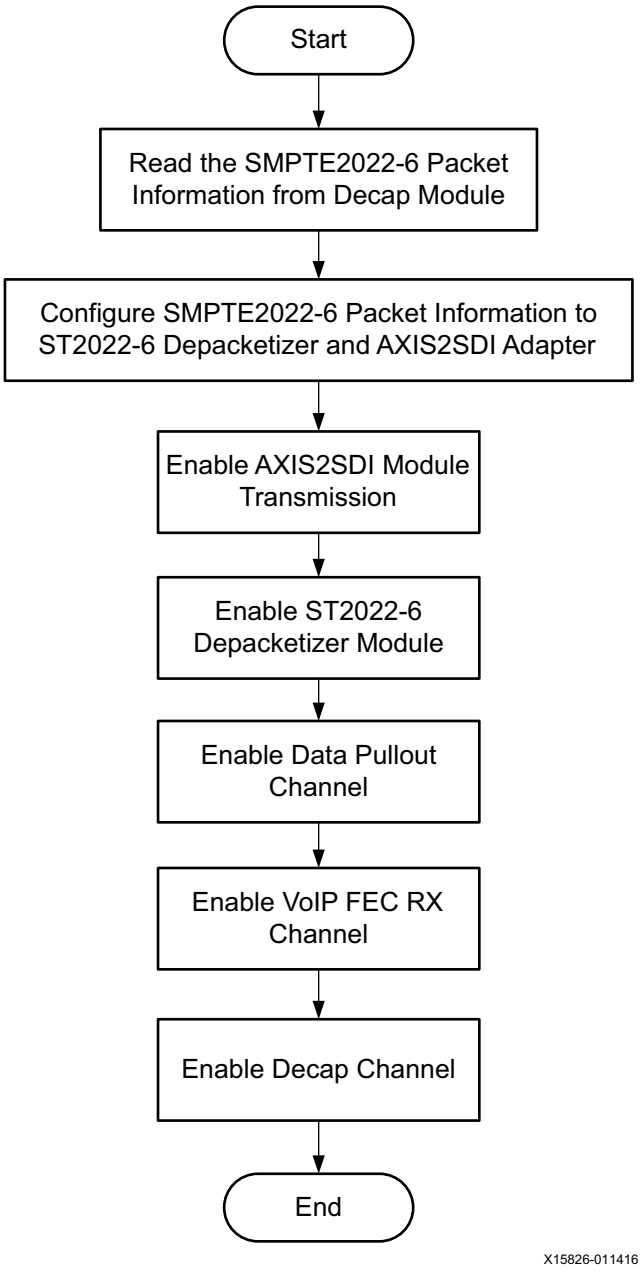

Figure 31: **SMPTE2022-6 Packet Lock Software Flow**

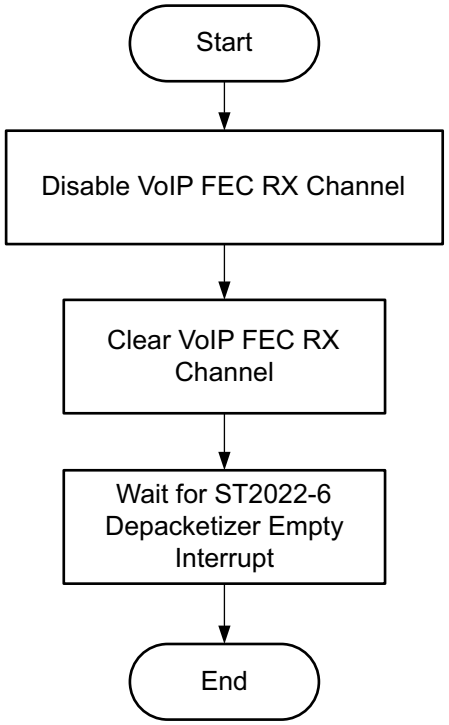

Figure 32: **SMPTE2022-6 Packet Unlock Software Flow**

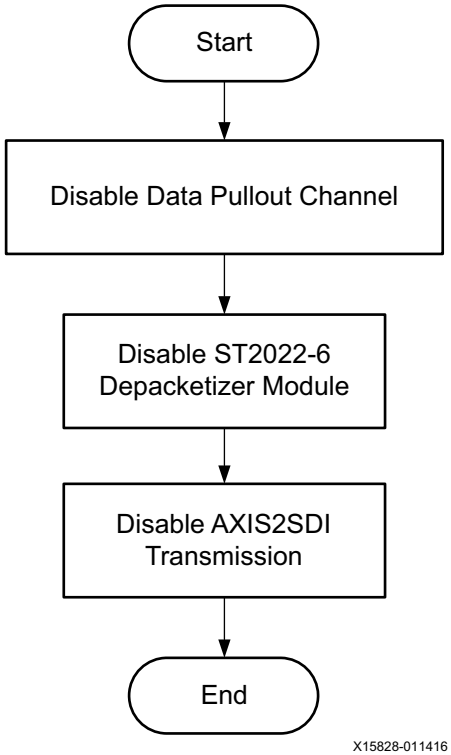

Figure 33: **ST 2022-6 Depacketizer Empty Software Flow**

Tool Flow and Verification

The following checklist indicates the tool flow and verification procedures used for the provided reference design.

Table 11: **Reference Design Checklist**

Parameter	Description
General	
Developer Name	Ilias Ibrahim, Josh Poh, Cunhua Xue
Target Devices	Kintex-7 FPGA
Source code provided?	No
Source code format (if provided)	IP Integrator Design
Design uses code or IP from existing reference design, application note, 3rd party or Vivado software? If yes, list.	Cores generated from the Vivado IP Catalog
Simulation	
Functional simulation performed	N/A
Timing simulation performed?	N/A
Test bench provided for functional and timing simulation?	N/A
Test bench format	N/A
Simulator software and version	N/A
SPICE/IBIS simulations	N/A
Implementation	
Synthesis software tools/versions used	Vivado 2016.1
Implementation software tool(s) and version	Vivado 2016.1
Static timing analysis performed?	Yes
Hardware Verification	
Hardware verified?	Yes
Platform used for verification	Xilinx Kintex-7 Series FPGA KC705 Evaluation Kit

Table 12 provides the specific versions for documents referenced in this application note.

Table 12: **Specific Versions of Referenced Documents**

Document and Version
Modular Media over IP Infrastructure Suite v1.0 LogiCORE IP Product Guide (PG241)
Video over IP Transmitter v1.0 LogiCORE IP Product Guide (PG206)
Video Over IP FEC Receiver v1.0 LogiCORE IP Product Guide (PG207)
SMPTE SD/HD/3G-SDI v3.0 Product Guide (PG071)
10 Gigabit Ethernet Subsystem v3.0 Product Guide (PG157)
7 Series FPGAs Memory Interface Solutions v2.0 User Guide (UG586)
AXI4-Stream Interconnect v1.1 LogiCORE IP Product Guide (PG035)
AXI Interconnect v2.1 LogiCORE IP Product Guide (PG059)

Requirements

Hardware

The hardware requirements for this reference system are:

- Two Xilinx Kintex-7 FPGA KC705 Evaluation Kits
- Two Inrevium 3G-SDI boards (TB-FMCH-3GSDI2A)
- Two Faster Technology Quad SFP/SFP+ Transceivers (FM-S14)
- Two SFP+ optical transceivers
- Optical cable

Software

This section includes any software requirements:

- Vivado Design Suite 2016.1
- SDK 2016.1
- Software terminals (for example, Tera Term, HyperTerminal or PuTTY)

Reference Design Files

The reference design files for this application note can be downloaded from the SMPTE2022 Reference Design Lounge (registration required).

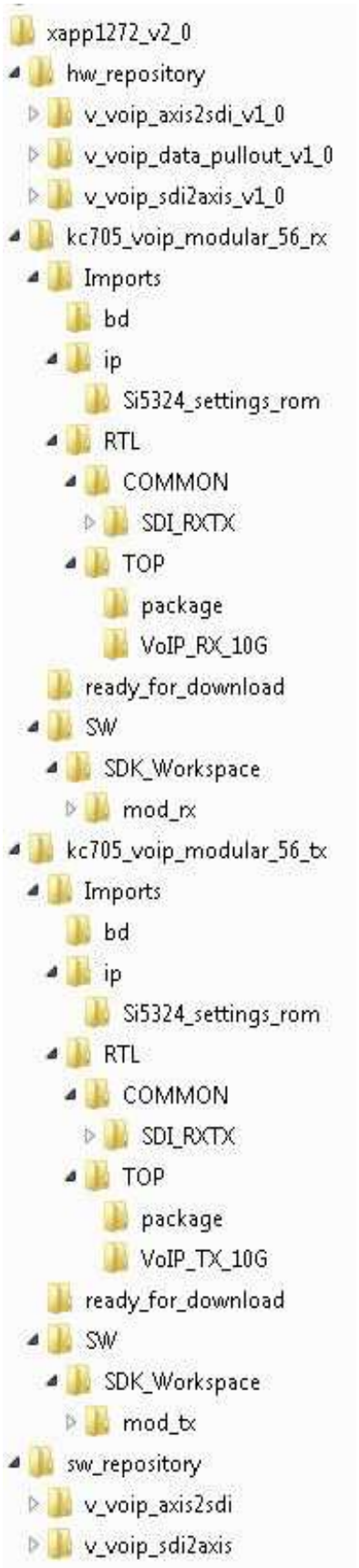

Figure 34: **Reference Design Directory Structure**

The two main directories in XAPP1272.zip are `kc705_voip_modular_56_tx` and `kc705_voip_modular_56_rx`. Both have the same directory structure which is described below.

```
hw_repository
     |\v_voip_sdi2axis_v1_0:                 User IP which acts as a bridge for SDI stream to AXI4-Stream
     |\v_voip_axis2sdi_v1_0:                 User IP which acts as a bridge for AXI4-Stream and SDI Stream
     |\v_voip_data_pullout_v1_0:             User IP which Control the request generator to FEC RX based
                                             on ST2022-6 Depacketizer Status
     |
kc705_voip_modular_56_rx
     | add_hdl.tcl:                          TCL for importing package RTL into the project
     | all.tcl:                              TCL for building and compiling reference design package
     | hw_bldr_utils.tcl:                    TCL which contains utility functions for building the project
     | proj.tcl:                             Creates Vivado project and adds "repository" folder into
                                             project local repository
     | rsb.tcl:                              TCl for constructing IPI subsystem by calling system_basic.TCL
                                             in Imports\bd folder and Generates Output Product of the IPI
     | sdk_build.tcl                         Generate HDF File and placed at the \SW\SDK_Workspace folder
     | sdk_init.tcl                         Generate ELF File based on Generated HDF using SDK Batch Command
     |
    \Imports
     |
     |-------\bd:                            Contains files for IPI subsystem construction
     |-------\RTL:
     |-------------\TOP:
     |-------------------\VoIP_RX_10G:       Contains the top level HDL and constraint files.
         |-------------------\package:       Contains local variable library for VHDL
         |-------------\COMMON:
         |-------------------\SDI_RXTX:      Contains Design Files for SDI
     |
    \ready_for_download:                     Contains "download.bit" file of the system.
     |
     |
    \SW
     |
     |-------\SDK_Workspace:                Contains source code of VoIP Transmitter Application.
         |-------------\mod_tx:              Contains the Top Level Application
     |
kc705_voip_modular_56_tx
     | add_hdl.tcl:                          TCL for importing package RTL into the project
     | all.tcl:                              TCL for building and compiling reference design package
     | hw_bldr_utils.tcl:                    TCL which contains utility functions for building the project
     | proj.tcl:                             Creates Vivado project and adds "repository" folder into
                                             project local repository
     | rsb.tcl:                              TCl for constructing IPI subsystem by calling system_basic.TCL
                                             in Imports\bd folder and Generates Output Product of the IPI
     | sdk_build.tcl                         Generate HDF File and placed at the \SW\SDK_Workspace folder
     | sdk_init.tcl                         Generate ELF File based on Generated HDF using SDK Batch Command
     |
    \Imports
     |
     |-------\bd:                            Contains files for IPI subsystem construction
     |-------\RTL:
     |-------------\TOP:
         |-------------\VoIP_TX_10G:         Contains the top level HDL and constraint files.
             |-------------\package:         Contains local variable library for VHDL
         |-------------\COMMON:
             |-------------\SDI_RXTX:        Contains Design Files for SDI
     |
    \ready_for_download:                     Contains "download.bit" file of the system.
     |
     |
    \SW
     |
     |-------\SDK_Workspace:                Contains source code of VoIP transmitter Application.
         |-------------\mod_tx:              Contains the Top Level Application
     |
sw_repository
     |\v_voip_axis2sdi:                      User Software Driver for VoIP AXIS2SDI
     |\v_voip_sdi2axis:                      User Software Driver for VoIP SDI2AXIS
```

Licensing

Ensure that the licenses for the Video over IP FEC Transmitter and Receiver cores, Modular Media over IP Infrastructure, and the 10G Ethernet MAC Subsystem are installed.

Reference Design Steps

Setup

This reference design runs on the Kintex-7 Evaluation board (KC705) using the TB-FMCH-3GSDI2A Mezzanine boards and FM-S14 Quad SFP/SFP+ Transceiver FMC Boards shown in Figure 35.

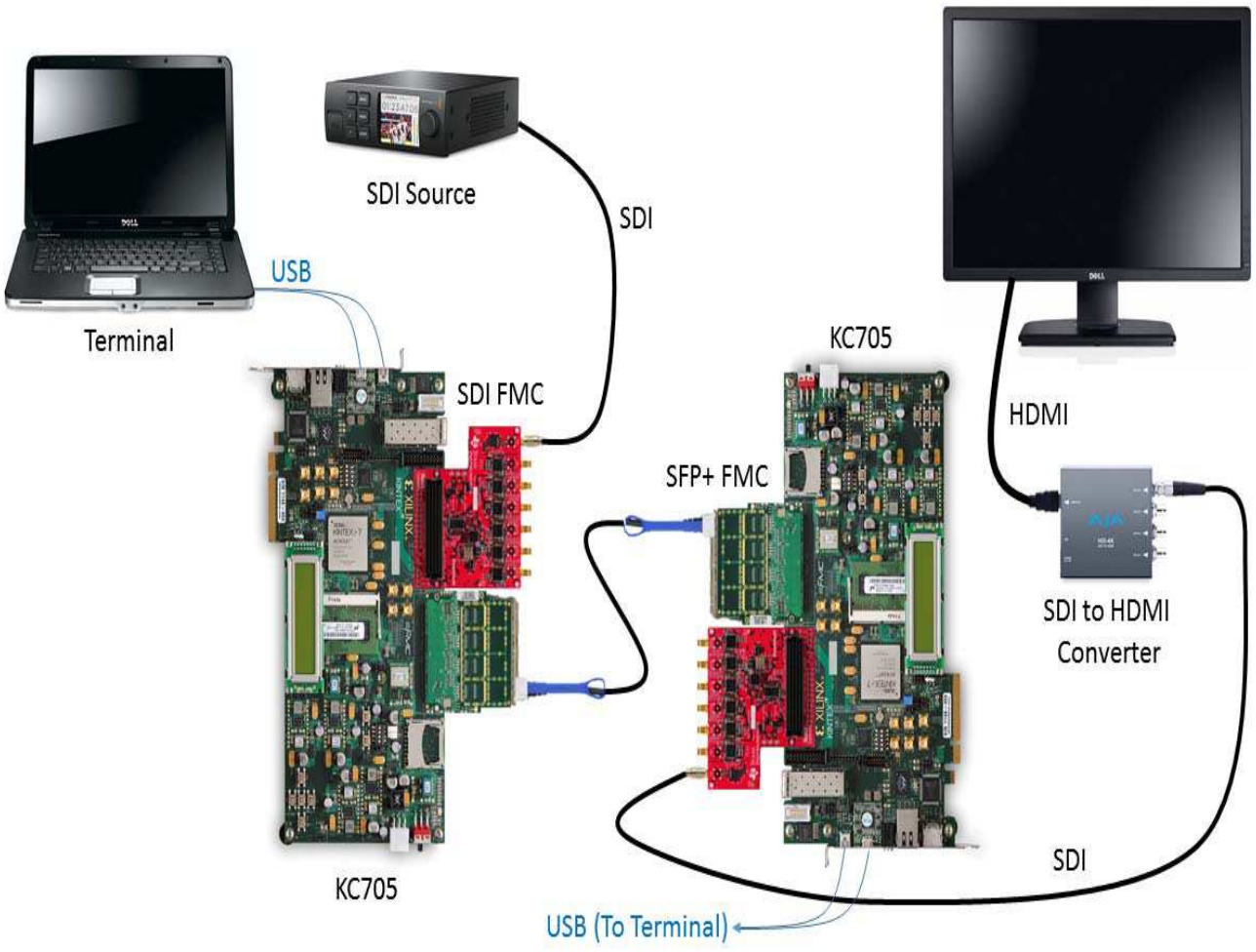

Figure 35: **Modular Video over IP ST 2022-56 System Setup**

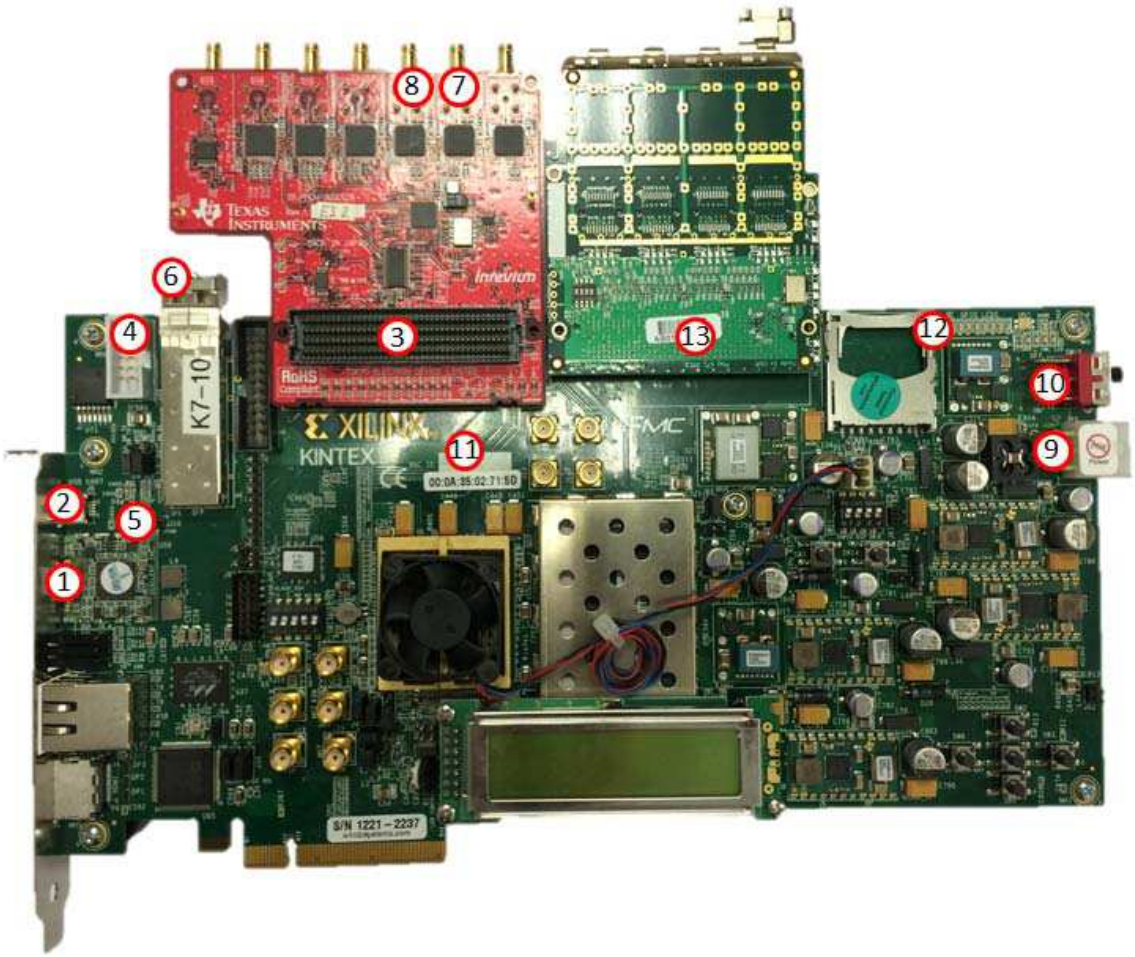

Figure 36: **KC705, and TB-FMCH-3GSDI2A Boards**

In these instructions, the numbers in parentheses correspond to the callout numbers in Figure 36.

1. Connect a USB cable from the host PC to the USB JTAG port (1). Ensure the appropriate device drivers are installed.
2. Connect a second USB cable from the host PC to the USB UART port (2). Ensure that the USB UART drivers described in Hardware have been installed.
3. Connect the TB-FMCH-3GSDI2A board to the HPC-FMC of KC705 (3).
4. Ensure that the SMPTE 352/Payload ID is enabled in the SDI Stream connected to the HPC-FMC of KC705.
5. Connect an SFP+ optical transceiver to the SFP slot (4).
6. Connect a jumper to J4 (5) to enable the SFP+ transmitter.
7. Connect one end of the optical cable (6) to the SFP+ on the VoIP transmitter board, the other end to the SFP+ on the VoIP receiver board.

8. Connect the CH0-TX, CH1-TX, and CH2 ports of TB-FMCH-3GSDI2A (7) to the SDI video monitor if the KC705 board is the VoIP receiver.
9. Connect the CH0-RX, CH1-RX, and CH2 ports of TB-FMCH-3GSDI2A (8) to the SDI video generator if the KC705 board is the VoIP transmitter.
10. Connect the KC705 board to a power supply slot J49 (9).
11. Switch on the KC705 board (10).
12. Make sure that the HW-KC705 board revision (11) is the same for the VoIP TX and RX platforms.
13. Connect one end of the optical cable to the SFP+ of the VoIP TX FM-S14 (13) board, the other end to the SFP+ of the VoIP RX FM-S14 (13) board.
14. Ensure LEDs 0, 1, 4, 5, 6, and 7are lit on (refer to Debugging for details)
15. Start a terminal program (for example, HyperTerminal) on the host PC with these settings:
    - Baud Rate: 115,200
    - Data Bits: 8
    - Parity: None
    - Stop Bits: 1
    - Flow Control: None

Running the Reference Design

This section details the steps necessary to execute the system using the files in the `ready_for_download` directory.

1. Launch the Xilinx Microprocessor Debugger by selecting **Start > All Programs > SDK 2016.1 > Xilinx Software Command Line Tool 2016.1**.
2. Execute the Xilinx Microprocessor Debug, by typing in the Xilinx Software command line:

   xmd
3. In the Xilinx command shell window, change to the `ready_for_download` directory:

   VoIP_TX: >**cd** <unzip_dir>**/kc705_voip_modular_56_tx/ready_for_download**
   VoIP_RX: >**cd** <unzip_dir>**/kc705_voip_modular_56_rx/ready_for_download**
4. Download the bitstream to the FPGA:

   XMD% **fpga -f download.bit**
5. Exit the XMD command prompt:

   XMD% **exit**

Note: The software application starts immediately after the completion of FPGA configuration. The executable file (.elf) is embedded in the configuration file (`download.bit`).

Building Hardware

This section covers rebuilding the hardware design. Before rebuilding the project, ensure that the licenses for the 10-Gigabit Ethernet MAC Subsystem and VoIP FEC Transmitter and Receiver are installed.

Note: To ensure that no compilation errors occur due to long file paths, unzip the project files as close to the root directory as possible. For example, with a typical Windows installation, unzip the files at C:\.

To generate a programming file in the Vivado Design Suite 2016.1, perform the following steps.

1. Open the Vivado Design Suite.
2. At the Tcl Console, change to the workspace directory by typing:
   VoIP_TX: > **cd** <unzip dir>**\kc705_voip_modular_56_tx**
   VoIP_RX: > **cd** <unzip dir>**\kc705_voip_modular_56_rx**
3. Run the all.tcl script to create, compile, and generate the project bitstream.

   > source all.tcl

Compiling Software in SDK

The SDK environment for the ST 2022-56 Modular Media over IP is compiled during execution of the all.tcl script. If you are required to build the SDK environment from the beginning, follow these steps

1. Export hardware: In Vivado 2016.1 select **File > Export > Export Hardware**.
   a. In the Export Hardware popup window, Check the **Include bitstream** option.
   b. Set the **Export to**: field correspondingly

      VoIP_TX: <unzip dir>**\kc705_voip_modular_56_tx\SW\SDK_Workspace**

      VoIP_RX: <unzip dir>**\kc705_voip_modular_56_rx\SW\SDK_Workspace**
2. Launch Xilinx SDK 2016.1vfrom Vivado 2016.1 by selecting **File > Launch SDK**.
   a. In the launch SDK popup window, set the **Exported Location** field and **Workspace** field accordingly:

      VoIP _TX: <unzip dir>**\kc705_voip_modular_56_tx\SW\SDK_Workspace**

      VoIP _RX: <unzip dir>**\kc705_voip_modular_56_rx\SW\SDK_Workspace**

   b. Create the local repository in SDK, perform the following steps: (Figure 37)

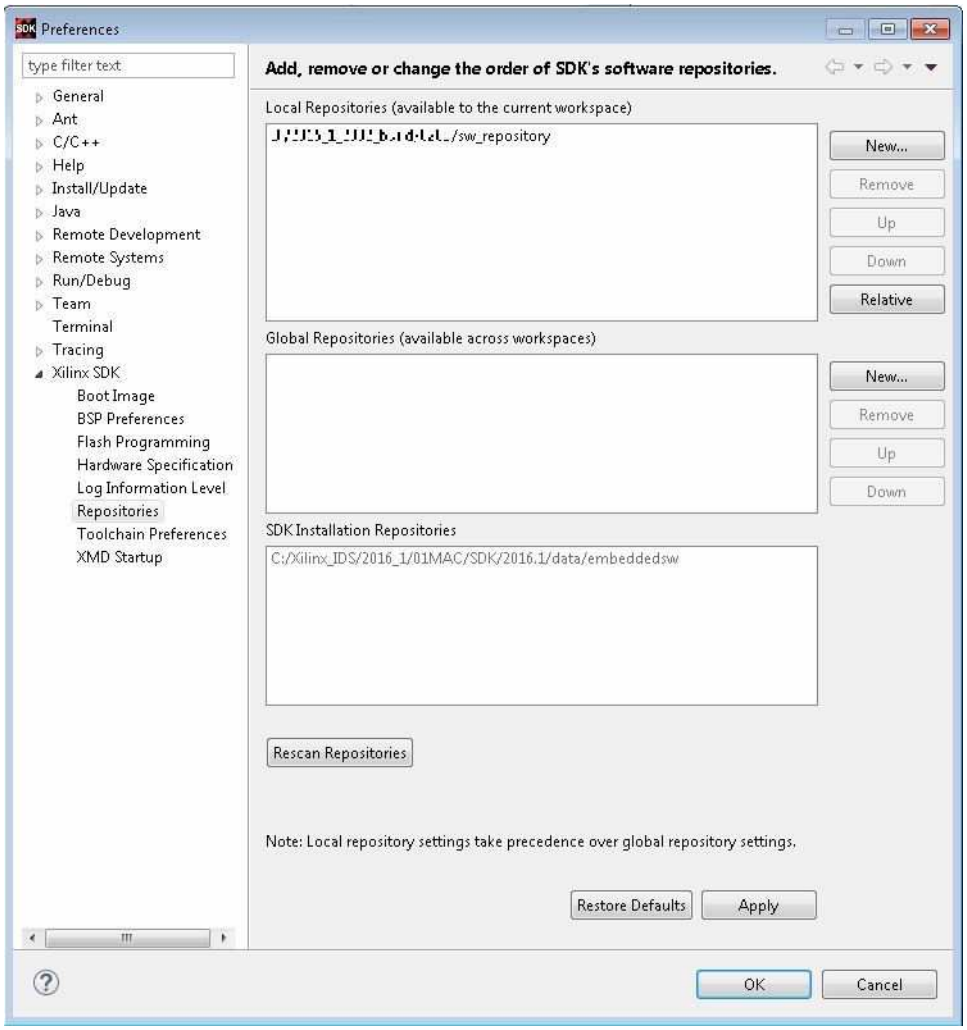

Figure 37: **Local Repository**

i. Select **Xilinx Tools > Repositories** in SDK. A new window pops up.

ii. Click the **New** tab, for **Local Repositories (available to the current workspace)**.

iii. Add the local repository path:

   VoIP_TX: <unzip dir>sw_repository

   VoIP_RX: <unzip dir>sw_repository

iv. Click the **Rescan Repositories** tab. Note that execution takes around 1 to 2 minutes.

v. Click **Apply** tab.

vi. Click the **OK** tab.

Note: The software local repository contains the low-level drivers for <v_voip_sdi2axis> for VoIP Modular 56 Transmitter and <v_voip_axis2sdi> for VoIP Modular 56 Receiver, which is portable to the Board Support Package in SDK 2016.1.

c. Create a new Board Support Package; **File > New > Board Support Package**.

d. Key in **mod_bsp** in the **Project name** field of **In the New Board Support Package Project** popup window.

e. Click **Finish**, then click **OK**.

The board support package (BSP) and software applications compile at this step. The process takes two to five minutes. The existing software applications can now be modified and new software applications can be created in the SDK.

3. Import SDK Sources: In SDK 2016.1 select **File > Import**.

   a. In the Import popup window, select **General > Existing Projects into the Workspace**

   b. Click **Next**.

   c. Click **Browse...** and make sure that it points to the corresponding folders.

      VoIP_TX: <unzip dir>**\kc705_voip_modular_56_tx\SW\SDK_Workspace**

      VoIP_RX: <unzip dir>**\kc705_voip_modular_56_rx\SW\SDK_Workspace**

   d. Click **OK**.

   e. Make sure **mod_tx/rx** are checked.

   f. Click **Finish**.

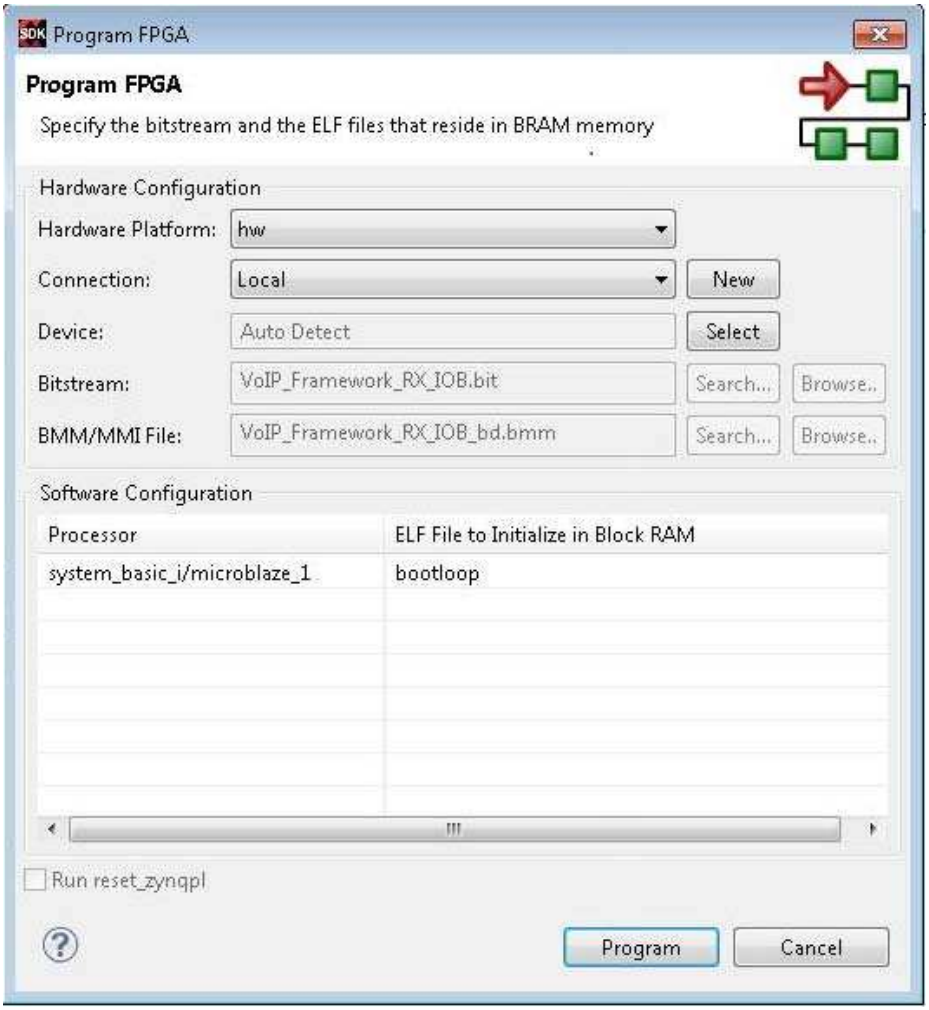

Figure 38: **Program FPGA**

Results

The HyperTerminal screen of the VoIP TX and RX displays the output shown in Figure 39 through Figure 44 respectively.

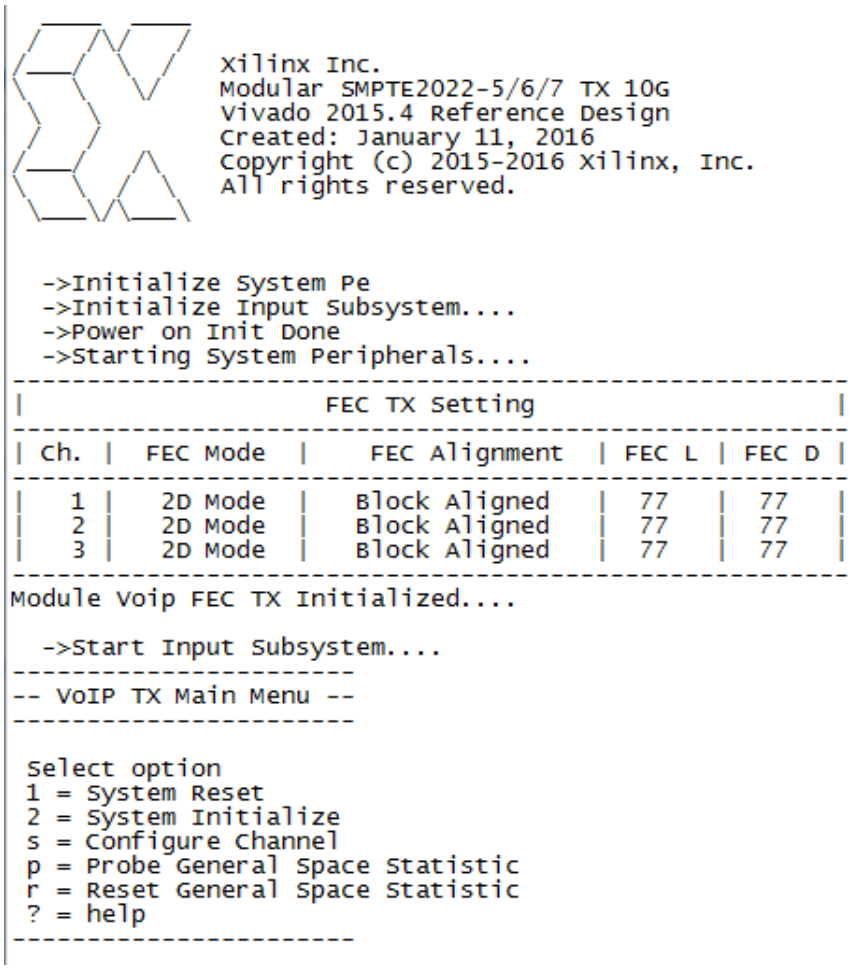

Figure 39: **Modular VoIP 56 Transmitter Main Menu**

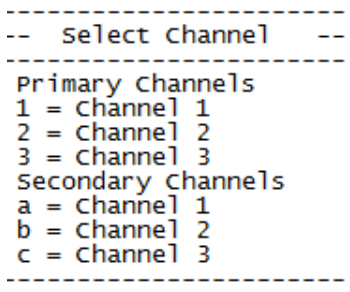

Figure 40: **Modular VoIP 56 Transmitter Channel Selection Menu**

```
----------------------
--   Select Option   --
----------------------
 1 = Channel Reset
 2 = Channel Initialize
 a = FEC On/Off
 b = Toggle FEC Level (1D/2D)
 c = Set Column FEC
 d = Set Row FEC
 e = Toggle Block Alignment
 t = Transmit Enable/Disable
 r = Transmit Packet Count Stat. Reset
 p = Probe Status
 m = Main Menu
 s = Channel Select
----------------------
```

Figure 41: **Modular VoIP 56 Transmitter Channel Menu**

```
Xilinx Inc.
Modular SMPTE2022-5/6/7 RX 10G
Vivado 2015.4 Reference Design
Created: January 16, 2016
Copyright (c) 2015-2016 Xilinx, Inc.
All rights reserved.

  ->Initialize System Pe  ->Initialize Input Subsystem....
  ->VoIP Subsystem Power on Init Done
  ->Start Input Subsystem Initialization..
  ->Starting System Peripherals....
  ->Start SMPTE 2022-56 VoIP RX Application....

-----------------------
-- VoIP RX Main Menu --
-----------------------

 Select option
 1 = System Reset
 2 = System Initialize
 s = Configure Channel
 p = Probe Current Settings
 r = Reset General Space Stat Counters
-----------------------
```

Figure 42: **Modular VoIP 56 Receiver Main Menu**

```
-----------------------
--  Select Channel   --
-----------------------
 1 = Channel 1
 2 = Channel 2
 3 = Channel 3
-----------------------
```

Figure 43: **Modular VoIP 56 Receiver Channel Selection Menu**

```
----------------------
--   Select Option   --
----------------------
 1 = Channel Reset
 2 = Channel Initialize
 p = Probe Status
 r = Channel Statistics Reset
 m = Main Menu
 s = Channel Select
----------------------
```

Figure 44: **Modular VoIP 56 Receiver Channel Menu**

Debugging

On-board general purpose I/O (GPIO) LEDs can be used for quick troubleshooting. During normal operation all LEDs should asynchronously turn on within five seconds after bit stream configuration. The LED representations are shown in Table 13.

Table 13: **KC705 GPIO LED Designations for Transmitter and Receiver**

GPIO_LED	Designation
0	Primary Link PCS Locked
1	Secondary Link PCS Locked
2	Unused
3	Unused
4	Si5324 Locked
5	100 MHz Locked
6	DDR MMCM Locked
7	DDR Memory Initialization Done

GPIO_LED 7: When this LED is low, it means that the memory subsystem did not successfully initialize. Sometimes switching development boards helps to get around this problem.

After the system is brought up, IP (Video over IP FEC TX and RX and Modular Media over IP Infrastructure) debug needs to be performed. This can be found under the Core Debug Section in the *Video over IP FEC Transmitter LogiCORE IP Product Guide* (PG206) [Ref 4], the *Video over IP FEC Receiver LogiCORE IP Product Guide* (PG207) [Ref 5] and the Modular Media over IP Infrastructure v1.0 LogiCORE IP Product Guide (PG241) [Ref 14].

References

This application note uses the following references:

1. Inrevium 3G/HD/SD-SDI FMC
2. Kintex-7 FPGA KC705 Evaluation Kit product page
3. Faster Technology FM-S14 Quad SFP/SFP+ transceiver FMC
4. *Video Over IP FEC Transmitter LogiCORE IP Product Guide* (PG206)
5. *Video Over IP FEC Receiver LogiCORE IP Product Guide* (PG207)
6. *SMPTE SD/HD/3G-SDI Product Guide* (PG071)
7. *10 Gigabit Ethernet Subsystem Product Guide* (PG157)
8. *AXI Reference Guide* (UG1037)
9. *7 Series FPGAs Memory Interface Solutions User Guide* (UG586)
10. ST 2022-6:2012 - Transport of High Bit Rate Media Signals over IP Networks (HBRMT)

    Note: Only registered users can access.
11. NUMERICAL INDEX OF SMPTE STANDARDS
12. *7 Series FPGAs Configuration User Guide* (UG470)
13. *Tri-Mode Ethernet MAC LogiCORE IP Product Guide* (PG051)
14. *Modular Media over IP Infrastructure Suite LogiCORE IP Product Guide* (PG241)
15. *AXI4-Stream Infrastructure IP Suite LogiCORE IP Product Guide* (PG085)
16. *AXI4-Stream Interconnect LogiCORE IP Product Guide* (PG035)
17. *AXI Interconnect LogiCORE IP Product Guide* (PG059)

Revision History

The following table shows the revision history for this document.

Date	Version	Changes
05//05/2016	2.0	• Replaced “SMPTE2020” in title with “SMPTE ST” • Replaced “Video over IP SMPTE2022-56” with “Modular ST 2022-56 Media over IP” in text and in graphics. • Updated Figures 2, 3, and 4. • Replaced “SMPTE2022” with “ST 2022” throughout. • Replaced Table 3 with two new tables (Table 3 and Table 4) • Updated Figure 4. • Beginning after new Table 4 removed text and Figure 9. • Removed the following sections: ST2022-6 Packetizer Module Register Space, Frame Module Register Space, Decapsulator Module Register Map, and ST2022-6 Depacketizer Module, • Changed 2015.4 to 2016.1 throughout • Updated old Figure 21 (now Figure 20). • Updated Figure 35 and Figure 38. • Updated directory structure description.
01/25/2016	1.0	Initial Xilinx release.

Please Read: Important Legal Notices