XAPP590 (v1.0) July 27, 2012

High Bit Rate Media Transport over IP Networks with Forward Error Correction

Author: Gilbert Magnaye, Josh Poh, Myo Tun Aung, and Tom Sun

Summary

This application note covers the design considerations of a video over IP networks system using the performance features of the LogiCORE™ IP SMPTE 2022-5/6 video over IP transmitter and receiver cores [Ref 1]. The design focuses on high bit rate native media transport over 10 Gb/s Ethernet with a built-in forward error correction (FEC) engine. The design is able to support up to 3x SD/HD/3G-SDI streams.

The entire reference design is composed of two platforms, the transmitter platform and the receiver platform. The transmitter platform design uses three LogiCORE IP triple-rate SDI cores to receive the incoming SDI video streams. The received SDI streams are multiplexed and encapsulated into fixed sized datagrams by the SMPTE 2022-5/6 video over IP transmitter core and sent out through the LogiCORE IP Ten Gigabit (10G) Ethernet MAC. The 10G link is supported by LogiCORE IP XAUI using a CX4 cable connected to the receiver end. On the receiver platform, the Ethernet datagrams are collected at the 10G Ethernet MAC. The SMPTE 2022-5/6 video over IP receiver core filters the datagrams, de-encapsulates and de-multiplexes them into individual streams, and outputs the SDI videos through the triple-rate SDI cores. The Ethernet datagrams are buffered in a DDR3 SDRAM for both the transmitter and receiver. The DDR traffic passes through the Advanced Microcontroller Bus Architecture (AMBA® protocol) Advanced eXtensible Interface (AXI) interconnect to the AXI memory controller on the Virtex-6 FPGA. A MicroBlaze™ processor is included in the design to initialize the cores and read the status.

The reference design is targeted for the Virtex-6 FPGA broadcast connectivity kit using the Virtex-6 XC6VLX240TFF1156-1 FPGA [Ref 2].

Included Systems

The reference design is created and built using the ISE® Design Suite: System Edition, version 14.1. Part of the design is created using the Xilinx Platform Studio (XPS) tool. The design also includes software built using the Xilinx Software Development Kit (SDK). The software runs on the MicroBlaze processor subsystem and implements control and status functions. Complete ISE software, XPS, and SDK project files are provided with this application note to allow the user to examine and rebuild the design or to use it as a template for starting a new design.

Introduction

The reference design implements the SMPTE 2022-5/6 video over IP cores as modules for broadcast applications that require bridging between broadcast connectivity standards (standard-definition serial digital interface (SD-SDI)/high-definition serial digital interface (HD-SDI)/3 Gb/s serial digital interface (3G-SDI)) and 10GbE networks. The cores are used for developing internet protocol (IP)-based systems to reduce overall cost in broadcast facilities for distribution and routing of audio and video data. The SDI data to be transported is mapped into media datagram payloads as per SMPTE 2022-6. The systematically generated redundant FEC datagrams are formatted according to SMPTE 2022-5. IP, user datagram protocol (UDP), and real-time transport protocol (RTP) provide standard headers in transporting the media and FEC datagrams over the IP network.

To support the system functions correctly, the bandwidth available in the network always meets or exceeds the bandwidth required by the stream generated by the system. The overhead due

to media datagram generation is approximately 5% due to IP/UDP/RTP and SMPTE 2022-6 headers.

The reference design is built around the SMPTE 2022-5/6 video over IP transmitter and receiver cores and leverages existing Xilinx IP cores to form the complete system. The input and output of the system are SDI video streams. The system contains two platforms—the transmitter core resides in one platform while the receiver core resides in the other. A CX4 cable connects the two platforms simulating an IP network, as shown in Figure 1.

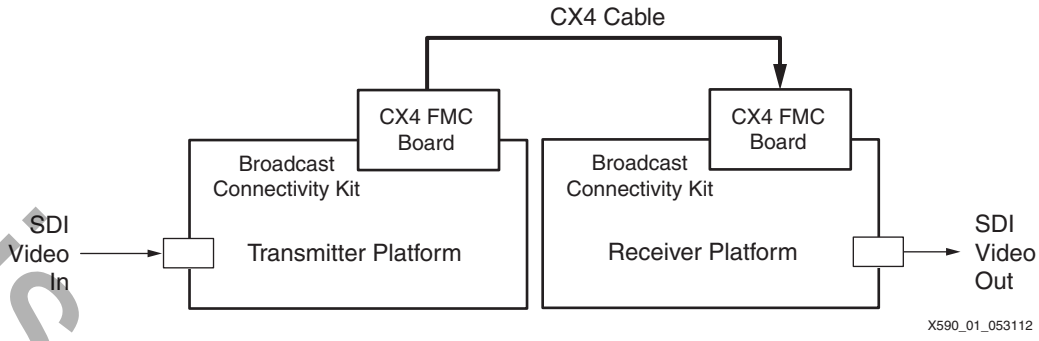

Figure 1: **Block diagram of the Video over IP Receiver FPGA**

The triple-rate SDI core helps the video over IP cores receive and transmit SDI streams while the 10G Ethernet MAC and XAUI enable the video over IP cores to transfer SDI data in the Ethernet medium (see Figure 2 and Figure 3).

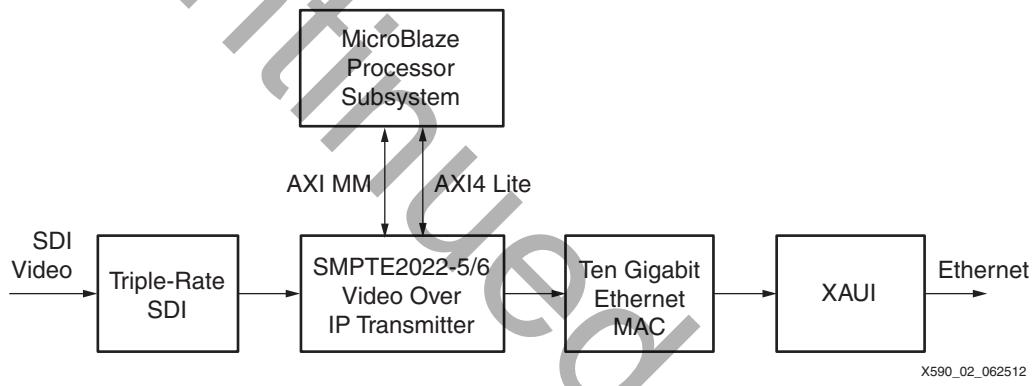

Figure 2: **Block Diagram of the Video over IP Transmitter FPGA**

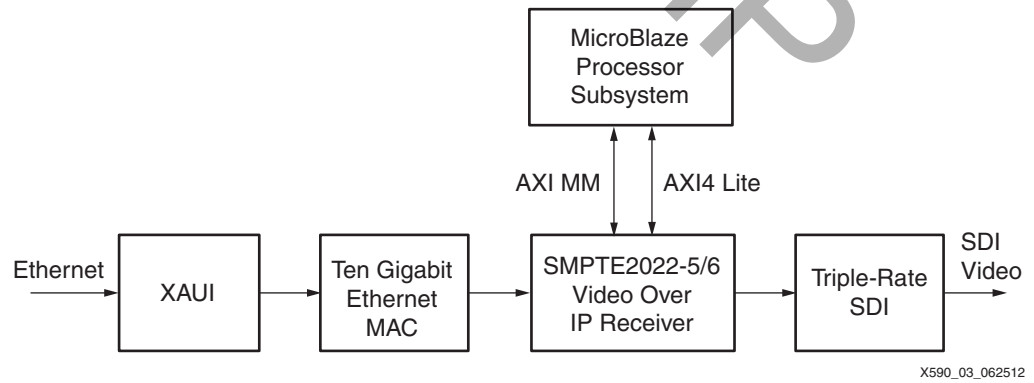

Figure 3: **Block Diagram of the Video over IP Receiver FPGA**

Other than managing the SDI streams, encapsulation and de-encapsulation, the transmitter and receiver cores include FEC protection features. FEC protects the video stream during transport of high-quality video over IP networks. With FEC, the transmitter adds systematically generated redundant data to its video. This carefully designed redundancy allows the receiver to detect and correct a limited number of packet errors occurring anywhere in the video without asking the transmitter for additional video data. These errors in the form of lost video packets are caused by a number of reasons such as thermal noise, 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. This feature can be enabled via the core's registers.

High-level control of the system is provided by a simplified embedded MicroBlaze processor subsystem containing I/O peripherals and processor support IP. A clock generator and processor system reset block supply clocks and resets for the system, respectively. An AXI interconnect and an AXI MIG are instantiated in the subsystem to allow the video over IP cores access to the DDR memory. Figure 4 shows a block diagram of the MicroBlaze processor subsystem.

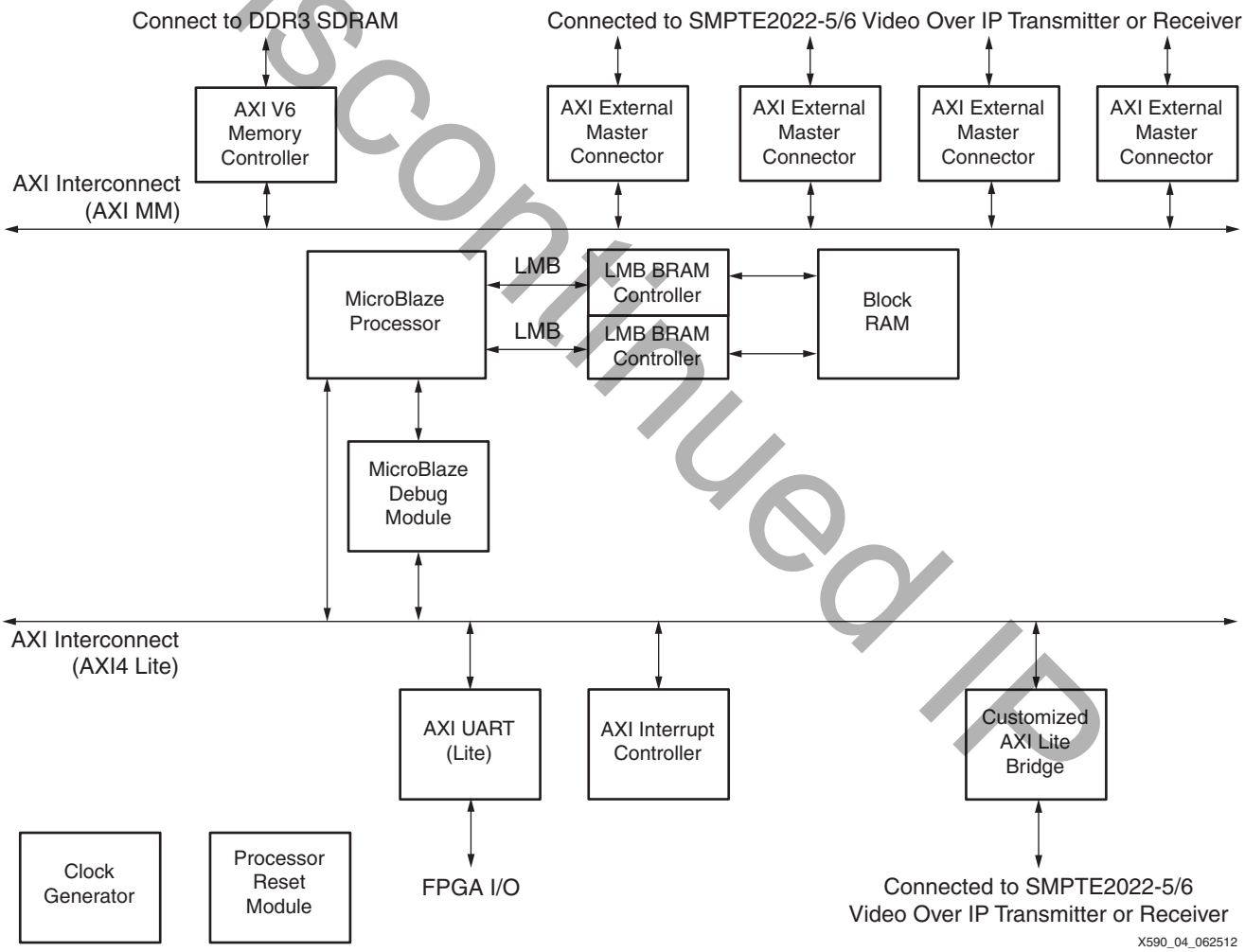

Figure 4: **Block Diagram of the Microblaze Processor Subsystem Built Using XPS**

Table 1 shows the address map of the MicroBlaze processor subsystem.

Table 1: **Microblaze Processor Subsystem Address Map**

Peripheral	Instance	Base Address	High Address
lmb_bram_if_cntlr	microblaze_0_i_bram_ctrl	0x00000000	0x0001ffff
lmb_bram_if_cntlr	microblaze_0_d_bram_ctrl	0x00000000	0x0001ffff
Mdm	debug_module_0	0x7e200000	0x7e20ffff
axi_v6_ddrx	axi_v6_ddrx_0	0xe0000000	0xefffffff
axi_uartlite	RS232_Uart_1	0x40600000	0x4060ffff
axilite_bridge	axilite_bridge_0	0x70e00000	0x70e0ffff

Hardware Requirements

The hardware requirements for the reference design are:

- Two Virtex-6 FPGA broadcast connectivity kits (each comprising an ML605 board, a CTXIL671 FMC mezzanine card, and two CTSLCM1 clock modules)
- Two HiTech Global FMC dual CX4 modules (HTG-FMC-X2CX4)
- Two extra CTSLCM1 clock modules
- CX4 cable
- Xilinx Platform Studio 14.1
- ISE Design Suite 14.1
- SDK 14.1

Reference Design Specifics

In addition to the SMPTE 2022-5/6 video over IP transmitter and receiver cores, the reference design includes these cores:

- AXI Interconnect
- MicroBlaze Processor
- MicroBlaze Debug Module
- Local Memory Bus
- LMB BRAM Controller
- Block RAM
- AXI External Master Connector
- Clock Generator
- Processor System Reset Module
- AXI UART (lite)
- Customized AXI Lite Bridge (pcore)
- AXI V6 Memory Controller (DDR2/DDR3)
- Triple-Rate SDI
- 10 Gigabit Ethernet MAC
- XAUI

Hardware System Specifics

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

SMPTE 2022-5/6 Video over IP Transmitter

The SMPTE 2022-5/6 video over IP transmitter in the reference design is configured to accept three channels of SDI inputs from the triple-rate SDI receiver. The SMPTE 2022-5/6 video over IP transmitter connects to the 10G Ethernet MAC via the AXI4-stream data interface. It connects to a customized pcore in the MicroBlaze processor subsystem via the AXI4-Lite control interface. The core does not have native EDK support. Thus, a customized pcore called `axilite_bridge` is created for register access. The core uses two AXI external master connectors to access the DDR memory via an AXI interconnect. The memory map address range is fixed at `0xE0000000` to `0xEFFFFFFF`.

The transmitter source MAC address is set to `0x000000000000AA`. The transmitter source IP address is set to 192.168.1.100, and the destination IP address for all channels is set to 192.168.1.101. The UDP ports are configured as shown in Table 2.

Table 2: **UDP ports values for the three SDI channels**

BNC connector	Channel	Source UDP port	Destination UDP port
RX1	0	0x10	0x10
RX2	1	0x20	0x20
RX3	2	0x30	0x30

The FEC matrix sizes set for the channels are in Table 3. These parameters are configurable through the registers.

Table 3: **FEC matrix size values for the three SDI channels**

BNC connector	Channel	L	D
RX1	0	77	77
RX2	1	77	77
RX3	2	77	77

The SMPTE 2022-5/6 video over IP transmitter core contains an AXI4-Lite interface that allows users to dynamically control the parameters within the core from a processor. For more information about the registers, see *LogiCORE IP SMPTE 2022-5/6 Video over IP Transmitter v1.0 Product Guide* [Ref 5].

The registers are categorized into two sections, the general space and channel space. The parameters in the general space apply to all the channels. The parameters in the channel space apply to individual channel.

For the general registers, normal address read and write access is applied. For the channel registers, these steps must be followed to update the registers:

1. Set the channel to be configured at the register address base_addr + `0x030`.
2. Configure the channel-specific register.
3. Pulse bit 1 of the register address base_addr + `0x000` to commit the channel registers change.
4. Repeat step 1 to step 3 for another channel or registers (see Figure 5).

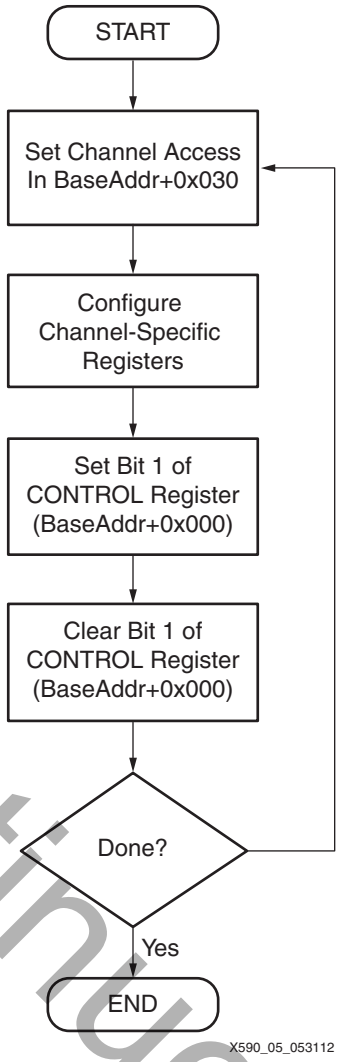

Figure 5: **Channel Register Configuration Flow Chart**

SMPTE 2022-5/6 Video over IP Receiver

The SMPTE 2022-5/6 video over IP receiver in the reference design is configured to stream three channels of SDI outputs to the triple-rate SDI transmitters. The SMPTE 2022-5/6 video over IP receiver connects to a 10G Ethernet MAC via the AXI4-stream data interface. It connects to a customized pcore in the MicroBlaze processor subsystem via an AXI4-Lite control interface. The core does not have native EDK support. Thus, a customized pcore called `axilite_bridge` is created for register access. The core uses three AXI external master connectors to access the DDR memory via an AXI interconnect. The memory map address range is fixed at `0xE0000000` to `0xEFFFFFFF`.

The incoming media packets are filtered based on the UDP destination ports, as shown in Table 4. The IP source address and synchronization source (SSRC) are not used as criteria.

Table 4: **UDP ports values for the three SDI channels**

BNC Connector	Channel	Destination UDP Port
TX1	0	`0x10`
TX2	1	`0x20`
TX3	2	`0x30`

The amount of media packets to be buffered before starting the SDI output per channel is set at 14,000.

The SMPTE 2022-5/6 video over IP receiver contains an AXI4-Lite interface that allows users to dynamically control the parameters within the core from a processor. For more information about the registers, see *LogiCORE IP SMPTE 2022-5/6 Video over IP Receiver v1.0 Product Guide* [Ref 6].

The registers are categorized into two sections, the general space and channel space. The parameters in the general space apply to all the channels. The parameters in the channel space apply to individual channels.

For the general registers, normal address read and write access is applied. For the channel registers, these steps must be followed to update the registers:

1. Set the channel to be configured at the register address base_addr + `0x030`.
2. Configure all the channel registers of interest for the particular channel from step 1.
3. Pulse bit 1 of the register address base_addr + `0x000` to commit the channel registers change.
4. Repeat step 1 to step 3 for another channel (see Figure 5, page 6).

Triple-Rate SDI

The Triple-Rate SDI core provides transmitter and receiver interfaces for the SMPTE SD-SDI, HD-SDI and 3G-SDI standards. The core is connected to the Virtex-6 FPGA GTX transceiver for serialization and deserialization of the SDI bitstream. The Triple-Rate SDI receiver uses a 148.5 MHz GTX 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. For the Triple-Rate SDI transmitter, the receiver requires two different GTX reference clock frequencies for its supported SDI bit rates. In the reference design, 148.5 MHz and 148.35 MHz are used. 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 port to configure the GTX transmitter appropriately for each SDI mode. See *LogiCORE IP Virtex-6 FPGA Triple-Rate SDI v1.0 User Guide* for more information [Ref 7].

10 Gigabit Ethernet MAC

The 10 Gigabit Ethernet MAC instance at the transmitter side has its AXI4-Stream transmit interface connected to the output of the SMPTE 2022-5/6 video over IP transmitter. The 10 Gigabit Ethernet MAC instance at the receiver side has its AXI4-Stream receive interface fed to the input of the SMPTE 2022-5/6 video over IP receiver. A 64-bit SDR PHY port is configured in the 10 Gigabit Ethernet MAC to interface to the XAUI. No flow control is used. See *LogiCORE IP 10-Gigabit Ethernet MAC v11.1 User Guide* for more information [Ref 8].

XAUI

The eXtended Attachment Unit Interface (XAUI) creates a 10 Gb/s copper link between the video over IP transmitter and receiver. It uses four transceivers at a 3.125 Gb/s line rate to achieve a 10 Gb/s data rate. A CX4 cable is connected between the CX4 FMC boards on both sides. The XAUI standard is fully specified in clauses 47 and 48 of the 10 Gigabit Ethernet *IEEE 802.3-2008* specification. See *LogiCORE IP XAUI v10.3 User Guide* for more information [Ref 9].

AXI Interconnect (AXI_MM)

This AXI interconnect instance provides the high F_{MAX} and throughput for the design by having a 256-bit core data width and running at 200 MHz. The AXI interconnect core data width and

clock frequency match the capabilities of the attached AXI MIG so that width and clock converters between them are not needed. Sizing the AXI interconnect core data width and clock frequency below the native width and clock frequency of the memory controller creates a system bandwidth bottleneck in the system. To help meet the timing requirements of a 256-bit AXI interface at 200 MHz, a rank of register slices are enabled between AXI_MM and AXI MIG. Together, AXI_MM and AXI MIG form a four-port AXI multi-port memory controller (MPMC) connected to four AXI external master connectors. The configuration of this AXI interconnect is consistent with the system performance optimization recommendations for an AXI MPMC based system as described in the *AXI Reference Guide* [Ref 10].

AXI V6 Memory Controller (DDR2/DDR3)

The single slave connected to the AXI interconnect is the AXI V6 memory controller (a block that integrates the MIG tool into XPS). The memory controller's AXI interface is 256 bits wide running at 200 MHz, and disables narrow burst support for optimal throughput and timing. This configuration matches the native AXI interface clock and width corresponding to a 64-bit DDR3 DIMM at 400 MHz memory clock, which is the maximum performance of the memory controller for a Virtex-6 device with a -1 speed grade. Register slices are enabled to ensure that the interface meets timing at 200 MHz. These settings help ensure that a high degree of transaction pipelining is active to improve system throughput. See the *Virtex-6 FPGAs Memory Interface Solutions User Guide* [Ref 11] for more information about the memory controller.

AXI Interconnect (AXI4-Lite)

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 F_{MAX} and throughput. Therefore, they are connected to a slower F_{MAX} 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, area can be optimized over performance on this interconnect block. Also, this interconnect is clocked at 50 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 MDM, AXI UART (lite), and a customized pcore to the SMPTE 2022-5/6 video over IP transmitter or receiver core.

Software Applications

The individual software applications initialize the video over IP TX and RX systems, respectively. After the initialization, the user can select commands from the menu in the UART display.

Application-level software and the drivers for controlling the system are written in C. Alternatively, drivers and application software can be written to use the IP control registers directly.

The software configures the register values as shown in Table 5 and Table 6.

Table 5: **Initialized Video over IP-Video over IP TX Register Values**

Offset	Register Name	Value
General Space		
0x060	Src_mac_low_addr	0x000000AA
0x064	Src_mac_high_addr	0x00000000
0x068	Src_IP_addr	0xC0A80064

Table 5: **Initialized Video over IP-Video over IP TX Register Values *(Cont'd)***

Offset	Register Name	Value		
Channel Space		**Ch1**	**Ch2**	**Ch3**
0x104	FEC_config	0x7	0x7	0x7
0x10C	FEC_L	0x4D	0x4D	0x4D
0x110	FEC_D	0x4D	0x4D	0x4D
0x128	dest_ip_addr	0xC0A80032	0xC0A80032	0xC0A80032
0x138	src_udp_port	0x10	0x20	0x30
0x13C	dest_udp_port	0x10	0x20	0x30
0x140	SSRC	0x12345600	0x12345610	0x12345620

Table 6: **Initialized Video over IP-Video over IP RX Register Values**

Offset	Register Name	Value		
General Space				
0x060	mac_low_addr	0x000000BB		
0x064	mac_high_addr	0x00000000		
0x068	IP_host_addr	0xC0A80032		
Channel Space		**Ch1**	**Ch2**	**Ch3**
0x100	chan_en	0x1	0x1	0x1
0x104	reserved	0x3	0x3	0x3
0x110	firewall_sel	0x0	0x0	0x0
0x114	dest_port	0x10	0x20	0x30
0x118	SSRC	0x12345600	0x12345610	0x12345620
0x11C	src_ip_host_addr	0xC0A80064	0xC0A80064	0xC0A80064
0x12C	start_buffer_size	0x36B0	0x36B0	0x36B0

After the initial setup sequence, the user can choose commands from the multi-layered menu list in the UART display.

Executing the Reference Design in Hardware

This section provides instructions to execute the reference design in hardware. This reference design runs on the ML605, CTXIL671, and HTG-FMC-2CX4 boards shown in Figure 6 and Figure 7.

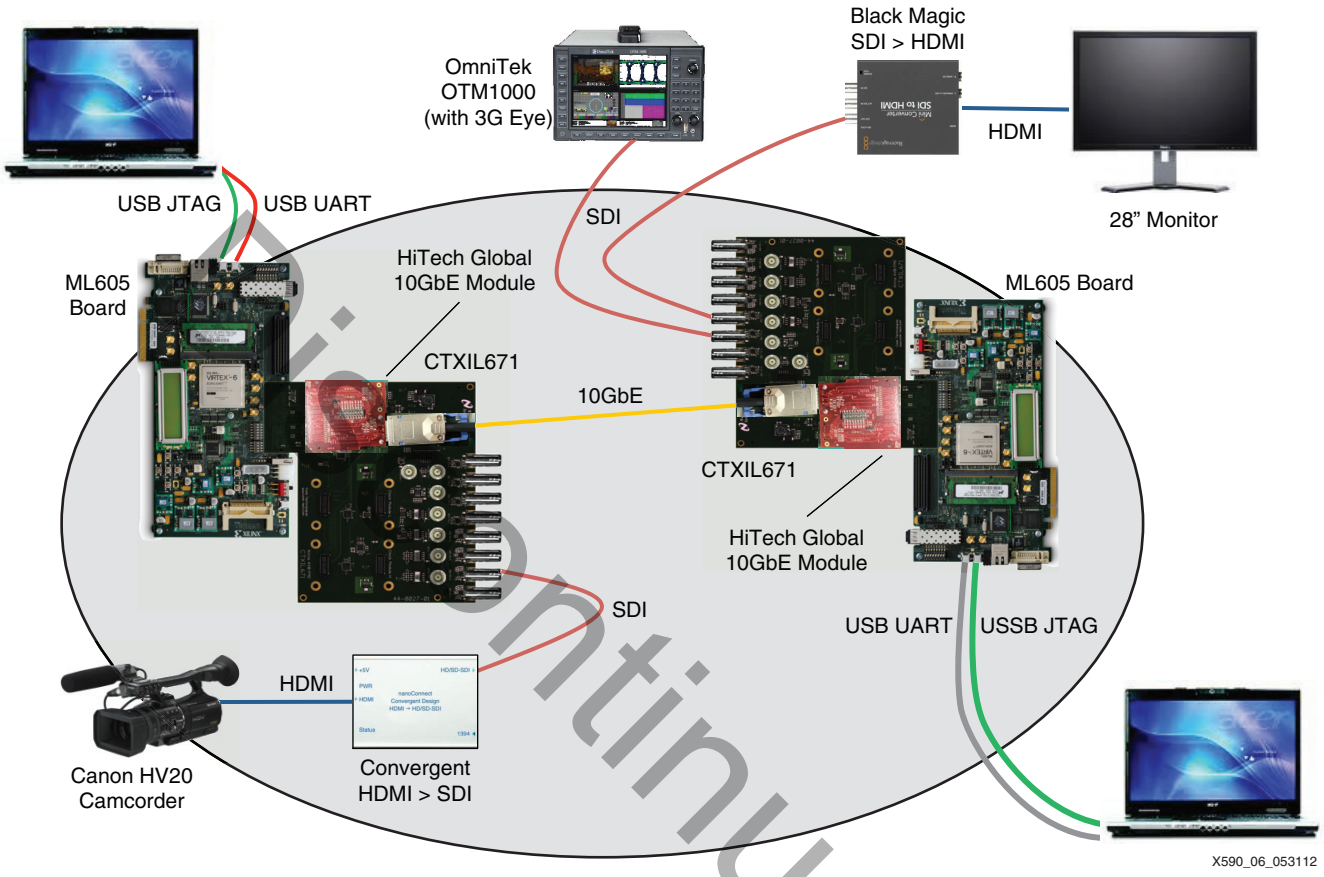

Figure 6: **Video over IP System Setup**

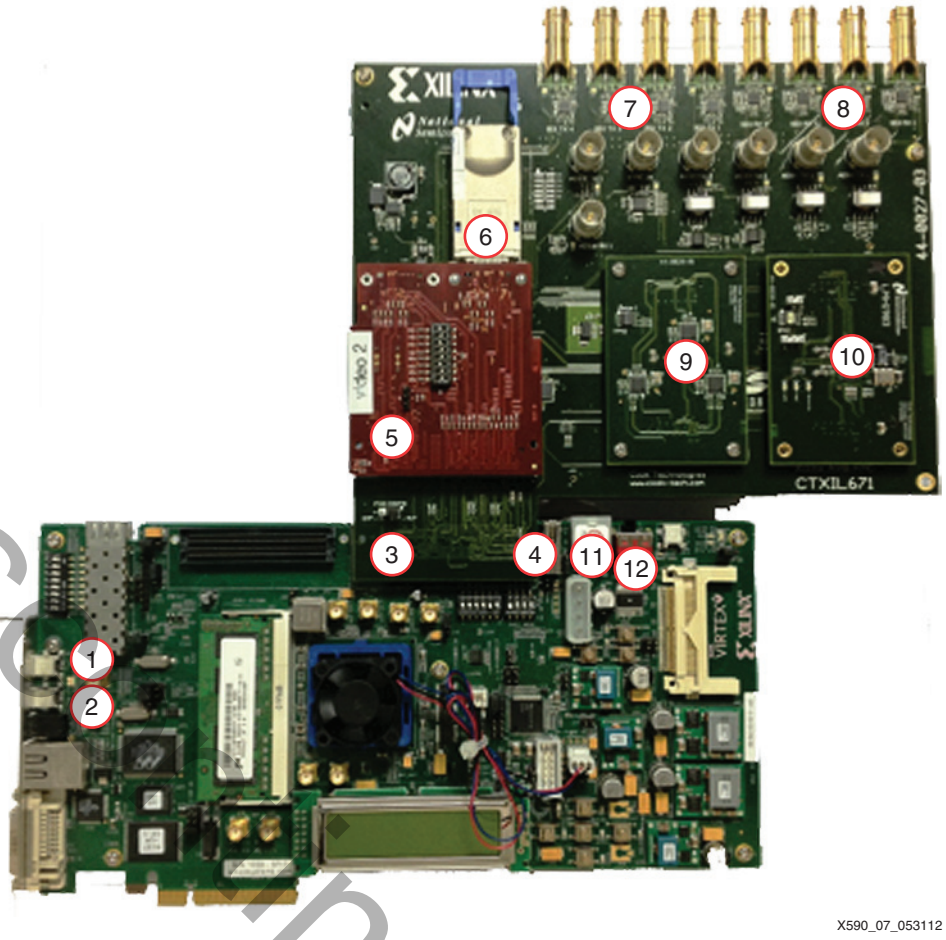

Figure 7: **ML605, CTLXIL671, and HTG-FMC-2CX4 Boards**

In these instructions, numbers in parentheses correspond to callout numbers in Figure 7.

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 Requirements, page 4 have been installed.
3. Connect the CTXIL671 board to the HPC-FMC of the ML605 board (3).
4. Set jumper J17 (4) to **INC HPC**.
5. Connect the HTG-FMC-2CX4 board to the FMC of the CTXIL671 board (5).
6. Connect one end of the CX4 cable (6) to the HTG-FMC-2CX4 of the video over IP-video over IP transmitter board, and the other end to the HTG-FMC-2CX4 of the video over IP receiver board.
7. If the ML605 board is the video over IP receiver, connect the SDI TX ports 1 to 3 (7) to the SDI video monitor, otherwise leave them unconnected.
8. If the ML605 board is the video over IP transmitter, connect the SDI RX ports 1 to 3 (8) to the SDI video generator, otherwise leave them unconnected.
9. Make sure that the CTSLCM1 board is connected to CML (9) of the CTXIL671 board.
10. Connect the CTSLCM1 board to the CMH (10) of the CTXIL671 board if the ML605 board is the video over IP receiver.
11. Connect the power supply extension that comes with the CTXIL671 board to the power supply and J25 (11).

12. Switch on the ML605 board.
13. Start a terminal program (e.g., HyperTerminal) on the host PC with these settings:
    - Baud Rate: **115200**
    - Data Bits: **8**
    - Parity: **None**
    - Stop Bits: **1**
    - Flow Control: **None**

Executing the Reference System Using the Pre-Built Bitstream and Compiled Software Application

This section contains the steps to execute the system using files in the `ready_for_download` directory:

VoIP_TX: `<unzip_dir>/ml605_smpte2022_56_3ch_tx/`

VoIP_RX: `<unzip_dir>/ml605_smpte2022_56_3ch_rx/`

Note: The XMD tool is used to type the commands in this section.

1. In the **ISE Design Suite 64 Bit Command Prompt**, change the directories to the `ready_for_download` directory:

   VoIP_TX:

   ```
   >cd <unzip_dir>/ml605_smpte2022_56_3ch_tx/ready_for_download
   ```

   VoIP_RX:

   ```
   >cd <unzip_dir>/ml605_smpte2022_56_3ch_rx/ready_for_download
   ```

2. Invoke the Xilinx Microprocessor Debugger (XMD) tool:

   ```
   >xmd
   ```

3. Download the bitstream inside XMD:

   ```
   XMD% fpga -f download.bit -cable type xilinx_platformusb port
   USB<port #> frequency 6000000 -debugdevice devicenr 4
   ```

Note: The software application starts immediately after the completion of FPGA configuration. The executable file (`.elf`) is embedded in the configuration file (`download.bit`). The USB port number usually starts with “21” and increments by 1 as the number of boards increases.

Results from Running Hardware and Software

HyperTerminal screens of the video over IP TX and RX output displays are shown in Figure 8 and Figure 11, respectively.

Figure 8: **VoIP_TX HyperTerminal Output**

The user can choose one of the five options displayed on the HyperTerminal screen (Figure 8):

- **1**: Reset core
- **2**: Initialize core general space registers
- **3**: Configure channel (opens “Select Channel” submenu)
- **q**: Exit software application
- **?**: Display current menu

X590_09_053112

Figure 9: **VoIP_TX Select Channel HyperTerminal Output**

The user can choose one of the three channels or go back to the main menu (Figure 9):

- **1**: Channel 1
- **2**: Channel 2
- **3**: Channel 3
- **m**: Main menu

After selecting any of the channels, the **Select Option** submenu is displayed (Figure 10).

X590_10_053112

Figure 10: **VoIP_TX Select Option HyperTerminal Output**

The user can choose one of the five options in the menu list:

- **1**: Channel Init (configure target channel registers)
- **2**: FEC On/OFF (toggle FEC engine)
- **3**: Probe Status
- **m**: Main menu
- **c**: Channel Select

The video over IP RX output display is shown in Figure 11. The user can choose one of the five options displayed on the HyperTerminal screen:

- **1**: Reset core
- **2**: Initialize core general space registers
- **3**: Configure channel (opens “Select Channel” submenu)
- **q**: Exit software application
- **?**: Display current menu

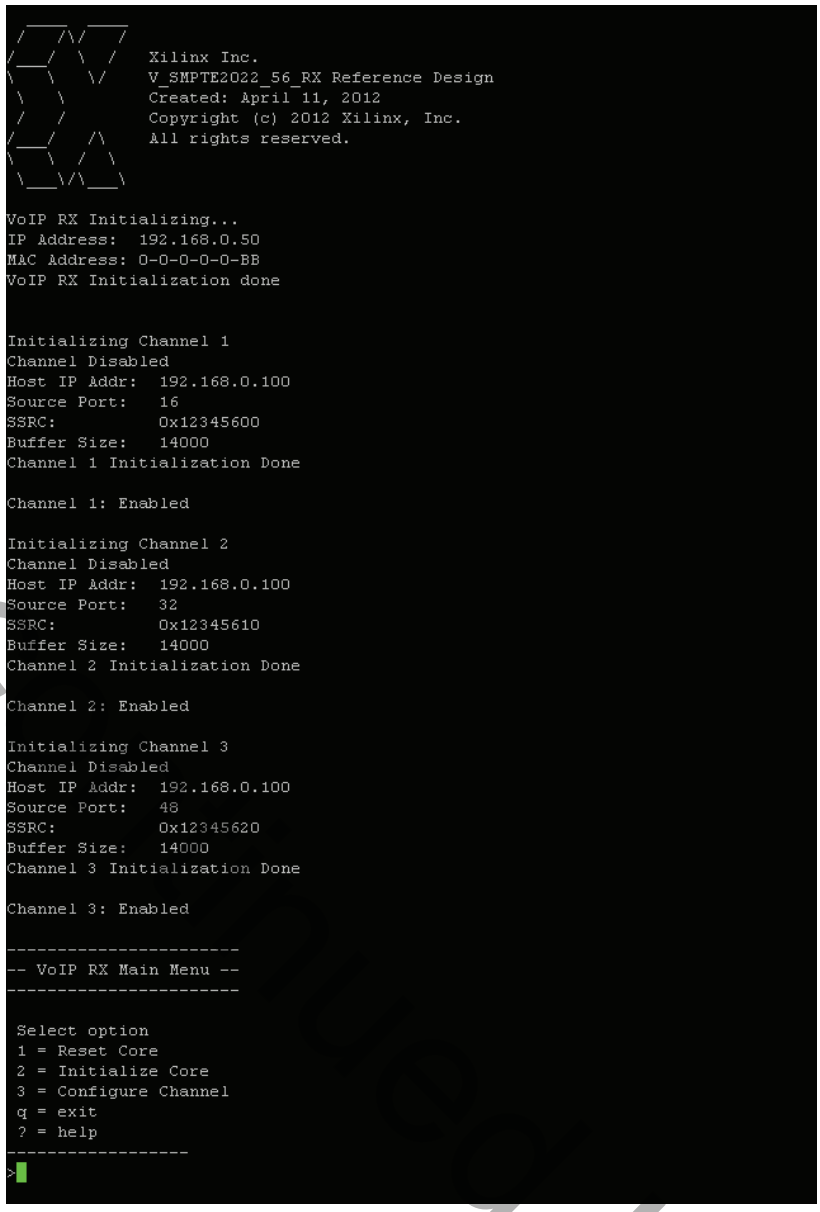

Figure 11: **VoIP_RX HyperTerminal Output**

The user can choose one of the three channels or go back to the main menu (Figure 12):

- **1**: Channel 1
- **2**: Channel 2
- **3**: Channel 3
- **m**: Main menu

Figure 12: **VoIP_RX Select Channel HyperTerminal Output**

After selecting any of the channels, the **Select Option** submenu is displayed (Figure 13). The user can choose one of the five options in the menu list:

- **1**: Channel Init (configure target channel registers)
- **2**: Channel Enable/Disable
- **3**: Probe Status
- **m**: Main menu
- **c**: Channel Select

Figure 13: **VoIP_RX Select Option HyperTerminal Output**

Building Hardware

This section covers rebuilding the hardware design. Before rebuilding the project, the user must ensure that the licenses for the SMPTE 2022-5/6 video over IP transmitter and receiver cores and 10 Gigabit Ethernet MAC are installed.

Generating Programming File in ISE Software

1. Open the XISE project file in the ISE software:

   VoIP_TX:

   <unzip dir>\ml605_smpte2022_56_3ch_tx\HW\VoIP_TX_10G_ISE\VoIP_TX_ISE.xise

   VoIP_RX:

   <unzip dir>\ml605_smpte2022_56_3ch_rx\HW\VoIP_TX_10G_ISE\VoIP_RX_ISE.xise
2. Click **Generate Programming File**.

Compiling Software in SDK

1. Click **system_basic_i** in the Hierarchy view of the ISE software.
2. Double click **Export Hardware Design To SDK with Bitstream**.
3. In the workspace launcher, select the workspace:

   VoIP_TX:

   <unzip dir>\ml605_smpte2022_56_3ch_tx\SW\SDK_workspace

   VoIP_RX:

   <unzip dir>\ml605_smpte2022_56_3ch_rx\SW\SDK_workspace

4. Click **OK**.
5. Import the board support package (BSP), hardware platform, and software applications by selecting **File > Import > General > Existing Projects** into the workspace.
6. Click **Next**, then browse:

   VoIP_TX:

   <unzip dir>\ml605_smpte2022_56_3ch_tx\SW

   VoIP_RX:

   <unzip dir>\ml605_smpte2022_56_3ch_rx\SW
7. Click **OK**.
8. Ensure that all checkboxes are selected.
9. Click **Finish**.

The BSP and software applications compile at this step. The process takes 2 to 5 minutes. The user can now modify existing software applications and create new software applications in SDK.

Update Bitstream in XPS to include ELF File

1. Open the XMP project file in XPS:

   VoIP_TX:

   <unzip dir>\ml605_smpte2022_56_3ch_tx\HW\TOP\peripheral\system_basic\system_basic.xmp

   VoIP_RX:

   <unzip dir>\ml605_smpte2022_56_3ch_rx\HW\TOP\peripheral\system_basic\system_basic.xmp
2. Add the ELF file from SDK_Workspace into the XPS project. Select **Project > Select Elf File…**. In **Choose Implementation Elf File**, select the ELF file accordingly:

   VoIP_TX:

   <unzip dir>\ml605_smpte2022_56_3ch_tx\SW\SDK_workspace\voip_tx\Debug\voip_tx.elf

   VoIP_RX:

   <unzip dir>\ml605_smpte2022_56_3ch_rx\SW\SDK_workspace\voip_rx\Debug\voip_rx.elf
3. Select **Device Configuration > Update Bitstream** to initialize the block RAM with a bootloop program. This ensures that the processor boots up with a stable program in memory.

The updated bitstream is in:

VoIP_TX:

```
<unzip dir>\ml605_smpte2022_56_3ch_tx\HW\TOP\peripheral\system_basic\
implementation\download.bit
```

VoIP_RX:

```
<unzip dir>\ml605_smpte2022_56_3ch_rx\HW\TOP\peripheral\system_basic\
implementation\download.bit
```

Running the Hardware and Software through SDK

1. Open JTAG configuration by selecting **Xilinx Tools > Configure JTAG Settings**.
2. Select **Xilinx USB Cable** in the Type field.
3. Set **Port** accordingly.
4. Click on **Manual Configuration of JTAG Chain**.
5. Copy the JTAG chain configuration shown in Figure 14.

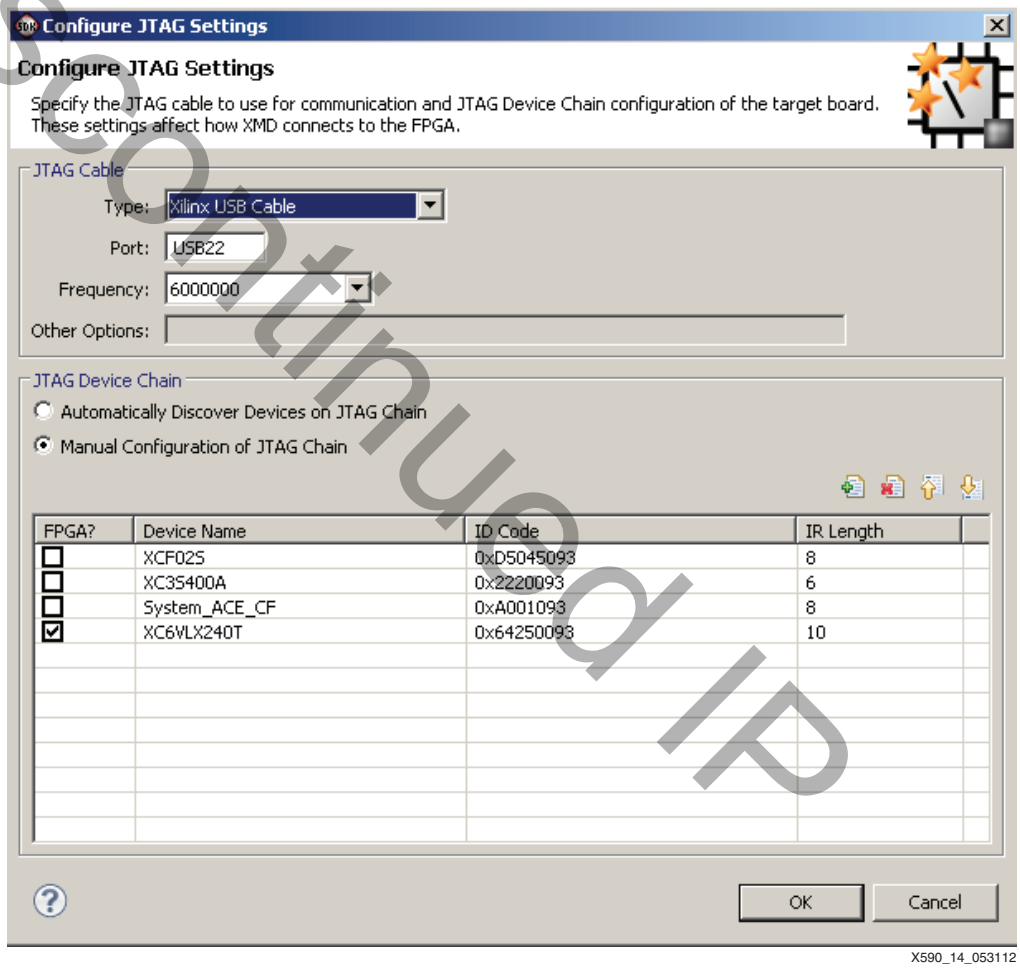

Figure 14: **Video over IP JTAG Configuration Settings**

6. Select **Xilinx Tools > Program FPGA**.

   Note: Ensure bootloop is used for microblaze_0.
7. Click **Program**.
8. In the Project Explorer window, right click and select:

VoIP_TX:
voip_tx_main > Run As > Launch on Hardware

VoIP_RX:
voip_rx_main > Run As > Launch on Hardware

Reference Design

The reference design files for this application note can be downloaded from:

http://www.xilinx.com/member/smpte2022/index.htm

Table 7 shows the reference design checklist.

Table 7: **Reference Design Checklist**

Parameter	Description
General	
Developer name	Gilbert Magnaye, Josh Poh, Myo Tun Aung, and Tom Sun
Target devices (stepping level, ES, production, speed grades)	Virtex-6 FPGAs
Source code provided	Yes
Source code format	VHDL (some sources encrypted)
Design uses code/IP from existing Xilinx application note/reference designs, CORE Generator™ software, or third party	Cores generated from EDK and from CORE Generator software
Simulation	
Functional simulation performed	N/A
Timing simulation performed	N/A
Test bench used for functional and timing simulations	N/A
Test bench format	N/A
Simulator software/version used	N/A
SPICE/IBIS simulations	N/A
Implementation	
Synthesis software tools/version used	XST 14.1
Implementation software tools/versions used	ISE Design Suite: System Edition 14.1
Static timing analysis performed	Yes (Passing timing in PAR/TRCE)
Hardware Verification	
Hardware verified	Yes
Hardware platform used for verification	Virtex-6 FPGA broadcast connectivity kit

Design Characteristics

The reference design is implemented in a Virtex-6 FPGA (XC6VLX240T-1FF1156) using the ISE Design Suite: System Edition 14.1. The resources used for the video over IP TX and RX platforms per the summary report are summarized in Table 8.

Table 8: **Resource Utilization from Summary Report**

Platform	LUTs	I/Os	RAMB36E1s	RAMB18E1s
TX	34,117 out of 150,720 (22%)	146 out of 600 (24%)	88 out of 416 (21%)	15 out of 832 (1%)
RX	35,773 out of 150,720 (23%)	160 out of 600 (26%)	104 out of 416 (25%)	19 out of 832 (2%)

Notes:

1. Device resource utilization results depend on the implementation tool versions. Exact results can vary. These numbers should be used as a guideline.

Utilization and Performance

Table 9 and Table 10 are extracted from module utilization reports of the design summary in the ISE software.

Table 9: **VoIP_TX Device Utilization**

Module Name	Slices	Slice Registers	LUTs	LUTRAM	Block RAM/ FIFOs	DSP48E1 Slices	BUFG	BUFIO	BUFR	MMCM_ADV
VoIP_Framework_TX_IOB	0	0	0	0	0	0	1	0	1	0
AVBFMC	52	84	139	24	1	0	0	0	0	0
CMLCTRL	58	96	143	24	1	0	0	0	0	0
VOIP	11,133	24,347	23,354	316	69	0	8	0	0	0
SDI_RXTX_1	753	1,118	1,803	14	1	0	2	0	0	0
SDI_RXTX_2	784	1,118	1,794	14	1	0	2	0	0	0
SDI_RXTX_3	730	1,117	1,791	15	1	0	2	0	0	0
v_smpte2022_56_tx	7,415	18,129	14,971	145	66	0	1	0	0	0
ten_gig_block	1,431	2,784	2,935	128	0	0	0	0	0	0
xaui_block	339	696	667	0	0	0	0	0	0	0
xgmac_block	1,092	2,088	2,268	128	0	0	0	0	0	0
VoIP_Framework_IOB	21	12	17	0	0	0	0	0	0	0
system_basic	6,935	13,483	10,452	1883	32	3	5	0	2	3
RS232_UART	59	81	93	18	0	0	0	0	0	0
axi4lite	136	80	165	0	0	0	0	0	0	0
axi_interconnect	1,137	3,686	1,710	6	0	0	0	0	0	0
axi_v6_ddrx	4,103	8,032	6,424	1,500	0	0	0	0	2	0
axilite_bridge	5	0	10	0	0	0	0	0	0	0
clock_generator	0	0	0	0	0	0	4	0	0	3
debug_module	93	130	120	23	0	0	3	0	0	0
microblaze	1,370	1,433	1,902	334	0	3	0	0	0	0
microblaze_bram_block	0	0	0	0	32	0	0	0	0	0
microblaze_d_bram_ctrl	6	2	4	0	0	0	0	0	0	0
microblaze_dlmb	1	1	0	0	0	0	0	0	0	0
microblaze_i_bram_ctrl	4	2	2	0	0	0	0	0	0	0
microblaze_ilmb	1	1	0	0	0	0	0	0	0	0
proc_sys_reset	20	35	22	2	0	0	0	0	0	0
Total	18,199	38,022	34,105	2,247	103	3	14	0	3	3

Table 10: **VoIP_RX Device Utilization**

Module Name	Slices	Slice Registers	LUTs	LUTRAM	Block RAM/ FIFOs	DSP48E1 Slices	BUFG	BUFIO	BUFR	MMCM_ADV
VoIP_Framework_RX_IOB	9	16	29	0	0	0	1	0	1	0
AVBMFC	69	99	157	24	1	0	0	0	0	0
CMHCTRL	53	82	141	24	1	0	0	0	0	0
CMLCTRL	56	82	142	24	1	0	0	0	0	0
VoIP_Framework_IOB	24	12	18	0	0	0	0	0	0	0
VoIP_Framework_RX	1,1640	26,161	24,331	652	88	0	8	0	0	0
SDI_RXTX_1	758	1,022	1,792	15	0	0	2	0	0	0
SDI_RXTX_2	749	1,022	1,796	15	0	0	2	0	0	0
SDI_RXTX_3	754	1,021	1,766	16	0	0	2	0	0	0
clock_recovery_top	996	2,014	2,279	147	0	0	0	0	0	0
v_smpte2022_56_rx	7,052	18,300	13,864	292	88	0	1	0	0	0
ten_gig_block	1,304	2,700	2,769	167	0	0	0	0	0	0
xaui_block	340	693	647	0	0	0	0	0	0	0
xgmac_block	964	2,007	2,122	167	0	0	0	0	0	0
genlock_control	26	56	66	2	0	0	0	0	0	0
system_basic	7,283	14,805	10,889	1,883	32	3	5	0	2	3
RS232_UART	60	81	89	18	0	0	0	0	0	0
axi4lite	128	80	173	0	0	0	0	0	0	0
axi_interconnect	1,432	4,988	2,141	6	0	0	0	0	0	0
axi_v6_ddrx	4,156	8,052	6,434	1,500	0	0	0	0	2	0
axilite_bridge	5	0	10	0	0	0	0	0	0	0
clock_generator	0	0	0	0	0	0	4	0	0	3
debug_module	92	130	122	23	0	0	1	0	0	0
microblaze	1,378	1,433	1,892	334	0	3	0	0	0	0
microblaze_bram_block	0	0	0	0	32	0	0	0	0	0
microblaze_d_bram_ctrl	5	2	4	0	0	0	0	0	0	0
microblaze_dlmb	1	1	0	0	0	0	0	0	0	0
microblaze_i_bram_ctrl	4	2	2	0	0	0	0	0	0	0
microblaze_ilmb	1	1	0	0	0	0	0	0	0	0
proc_sys_reset	21	25	22	2	0	0	0	0	0	0
Total	19,160	41,313	35,773	2,609	123	3	14	0	3	3

Conclusion

This application note describes a video over IP network system using various Xilinx IP cores. It demonstrates the ability of the SMPTE 2022-5/6 video over IP cores to encapsulate and de-encapsulate multiple SDI streams and transport these streams through a 10 Gb/s Ethernet pipe. The utilization of the Ethernet bandwidth is over 90% with three 3G-SDI videos. The design can perform recovery of a limited number of Ethernet packets when impairment is introduced into the network with the FEC engine turned on.

References

This application note uses the following references:

1. SMPTE2022-5/6 Video Over IP Core
   http://www.xilinx.com/products/intellectual-property/EF-DI-SMPTE2022-56.htm
2. *Virtex-6 FPGA Broadcast Connectivity Kit*
   www.xilinx.com/support/documentation/virtex-6_broadcast_connectivity_kit.htm
3. *AMBA AXI4 Specifications*
   infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.amba/index.html
4. UG683, *EDK Concepts, Tools, and Techniques: A Hands-On Guide to Effective Embedded System Design* (v14.1)
5. PG032, *LogiCORE IP SMPTE 2022-5/6 Video over IP Transmitter v1.0 Product Guide*
6. PG033, *LogiCORE IP SMPTE 2022-5/6 Video over IP Receiver v1.0 Product Guide*
7. UG823, *LogiCORE IP Virtex-6 FPGA Triple-Rate SDI v1.0 User Guide*
8. UG773, *LogiCORE IP 10-Gigabit Ethernet MAC v11.1 User Guide*
9. UG150, *LogiCORE IP XAUI v10.3 User Guide*
10. UG761, *AXI Reference Guide*
11. UG406, *Virtex-6 FPGAs Memory Interface Solutions User Guide*
12. SMPTE 2022-5 *Forward Error Correction for High Bit Rate Media Transport over IP Networks*
    www.smpte.org
13. SMPTE 2022-6 *High Bit Rate Media Transport over IP Networks*
    www.smpte.org

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
07/27/12	1.0	Initial Xilinx release.

Notice of Disclaimer

written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; 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 Critical Applications: http://www.xilinx.com/warranty.htm#critapps.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.