# Video Over IP User Guide

**UG463 (v2.0) January 20, 2009**

XILINX®



# Revision History

The following table shows the revision history for this document.

| Date | Version | Revision |
|---|---|---|
| 2/22/08 | 1.0 | Initial Xilinx release. |
| 5/2/08 | 1.1 | Updated to Reflect V1.01 Hardware Platform |
| 5/27/08 | 1.2 | Made content edits in Chapter 4; updated figure 4_2. |
| 1/20/09 | 2.0 | Migrated to EDK tools; updated from MPMC2 to MPMC version4; added IGMP/Multicast. |

# Table of Contents

## Chapter 3: IP to ASI Bridge (Receiver)

## Chapter 4: Video Over IP Testing

## Chapter 5: Demonstrating the Video Over IP System

# About This Guide

## Guide Contents

This user guide contains the following chapters:

Chapter 1, “Video Over IP Overview”

Chapter 2, “ASI to IP Bridge (Transmitter)”

Chapter 3, “IP to ASI Bridge (Receiver),”

Chapter 4, “Video Over IP Testing,”

Chapter 5, “Demonstrating the Video Over IP System”

## Additional Resources

To request access to the reference systems associated with this user guide, visit:

https://secure.xilinx.com/webreg/register.do?group=video_ip.

To find additional documentation, see the Xilinx website at:

http://www.xilinx.com/literature.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

http://www.xilinx.com/support.

## Conventions

This document uses the following conventions. An example illustrates each convention.

### Typographical

The following typographical conventions are used in this document:

| Convention | Meaning or Use | Example |
|---|---|---|
| `Courier font` | Messages, prompts, and program files that the system displays | `speed grade: - 100` |
| **`Courier bold`** | Literal commands that you enter in a syntactical statement | **`ngdbuild`** *design_name* |

| Convention | Meaning or Use | Example |
|---|---|---|
| **Helvetica bold** | Commands that you select from a menu | **File → Open** |
| | Keyboard shortcuts | **Ctrl+C** |
| Italic font | Variables in a syntax statement for which you must supply values | **ngdbuild** *design_name* |
| | References to other manuals | See the *Development System Reference Guide* for more information. |
| | Emphasis in text | If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected. |
| Square brackets [ ] | An optional entry or parameter. However, in bus specifications, such as **bus[7:0]**, they are required. | **ngdbuild** [*option_name*] *design_name* |
| Braces { } | A list of items from which you must choose one or more | **lowpwr ={on\|off}** |
| Vertical bar \| | Separates items in a list of choices | **lowpwr ={on\|off}** |
| Vertical ellipsis<br>.<br>.<br>. | Repetitive material that has been omitted | IOB #1: Name = QOUT'<br>IOB #2: Name = CLKIN'<br>.<br>.<br>. |
| Horizontal ellipsis ... | Repetitive material that has been omitted | **allow block** *block_name loc1 loc2 ... locn;* |

## Online Document

The following conventions are used in this document:

| Convention | Meaning or Use | Example |
|---|---|---|
| Blue text | Cross-reference link to a location in the current document | See the section "Additional Resources" for details.<br>Refer to "Title Formats" in Chapter 1 for details. |
| Red text | Cross-reference link to a location in another document | See Figure 2-5 in the *Virtex-II Platform FPGA User Guide.* |
| Blue, underlined text | Hyperlink to a website (URL) | Go to http://www.xilinx.com for the latest speed files. |

Chapter 1

# Video Over IP Overview

## Introduction

This chapter provides an overview of the Xilinx Video Over IP system.

## Video Over IP Fundamentals

This section describes the fundamentals of Video Over IP independent of the Xilinx solution. It is not intended to be a complete tutorial, but solely an introduction to Video Over IP.

Video Over IP is a system for transporting video and audio, usually compressed, from an input source to a destination using IP protocol over Ethernet. UDP is the protocol used to transport video over an IP network. RTP protocol on top of UDP may also be used and is required when FEC (Forward Error Correction) standards are implemented.

Many Video Over IP systems provide a bridging function from one physical interface, such as ASI or SDI, to IP over Ethernet or from IP over Ethernet to another physical interface.

The data to be transported is called a transport stream which consists of packets of 188 or 204 bytes. A number of transport stream packets are encapsulated in IP packets to be transported over Ethernet. These packets are transported continuously and unidirectionally without any back channel for feedback or control.

### Network Protocols

The following network protocols are used in some manner for Video Over IP.

#### UDP (User Datagram Protocol)

UDP is a low overhead connectionless transport protocol on top of IP protocol. Because packets are sent to the Receiver without any confirmation that they were received, there is no reliability mechanism built into the protocol.

#### RTP (Real-time Transport Protocol)

A protocol on top of UDP, used for video, which contains timestamps to help with synchronization. RTP is needed when FEC is employed.

#### ARP (Address Resolution Protocol)

A protocol to determine the MAC address of a device on an IP network. This protocol can be used such that the sender of data only needs to know the IP address of the destination, but not the MAC address.

### IGMP (Internet Group Management Protocol)

A protocol to support multicast which allows a receiver of video to send a message to a router to join a group. The router then sends multicast video data to all members of the group so that the transmitter does not have to replicate the data for each receiver.

### SNMP (Simple Network Management Protocol)

A protocol used to exchange management information between network devices.

## Video Terms

### Transport Stream

A transport stream refers to an MPEG transport stream which is a protocol for audio, video, and data. The protocol is specified in the MPEG-2 specification. Data files for testing of the Video Over IP system are typically called transport stream files.

When a transport stream is carried over IP, the packets of the transport stream are put into User Datagram Packet (UDP) format together with an RTP header. From one to seven packets from the transport stream are placed into the payload.

### FEC (Forward Error Correction)

When video data is transported over longer distances, data can be lost. FEC sends extra data with the media so that errors can be corrected by the Receiver.

### SMPTE 2022

SMPTE 2022-1-2007, Forward Error Correction for Real-Time Video/Audio Transport Over IP Networks and SMPTE 2022-2-2007, Unidirectional Transport of Constant Bit Rate MPEG-2 Transport Stream on IP network, together define packetizing video streams and adding forward error correction..

### ASI (Asynchronous Serial Interface)

ASI, sometimes referred to as DVB-ASI, is a digital video interface used for carrying compressed video and audio streams. When the transport stream is on its physical link, it is serialized and 8B/10B coded for a 270 Mbps link but the maximum payload is 216 Mbps. This link contains live transport stream data, but since the actual data link is usually running at < 216 Mbps, most of the physical layer contains null bytes.

### SDI (Serial Digital Interface)

SDI is a digital video interface used for uncompressed video.

### PCR (Program Clock Reference)

The PCR is a field contained in the MPEG data to assist the decoder in presenting programs on time, at the right speed, and with synchronization.

### CC (Continuity Counter)

The continuity counter is a field contained in the MPEG data to allow the detection of lost data.

# EDK

The ASI to IP (Transmitter) and IP to ASI (Receiver) are built using the Xilinx Embedded Development Kit (EDK). Either project is opened through EDK by navigating to the top level directory and selecting the XMP file for that project. Once opened in EDK either project can be configured to change the number of channels or to include/exclude Forward Error Correction. For more information see the “Getting Started with EDK” document available at www.xilinx.com

# Multiport Memory Controller (MPMC)

The ASI to IP (Transmitter) and IP to ASI (Receiver) both use version 4 of the MPMC from the Xilinx Embedded Development Kit (EDK). Each uses a slightly different configuration of the MPMC with a different number of ports and different port memory interface, BRAM or SRL. Documentation for the MPMC can be found at http://www.xilinx.com, and is also available through EDK.

## Port Configuration

The ASI to IP (Transmitter) requires a single NPI port on the MPMC for use with FEC. The NPI port is configured to use BRAM for the packet FIFOs. The MPMC used by the ASI to IP has a 64-bit data interface to the DDR and a 64-bit data interface to the NPI port. The ASI to IP uses the DDR for storing FEC data.

The IP to ASI (Receiver) requires 3 NPI ports on the MPMC. All NPI ports are configured to use SRL FIFOs to help minimize the size of the MPMC. The MPMC used by the IP to ASI has a 64-bit data interface to the DDR and a 64-bit data interface to each NPI port. FEC packets and media packets are stored in DDR by the IP to ASI.

## Configuring the MPMC

Should the MPMC need to be re-configured, for example to add ports for a processor interface, it is important to keep the ports as defined above for the Transmitter and Receiver. This is due to potential latency differences between BRAM and SRL.

The MPMC is configured from within the EDK by right clicking on the DDR_SDRAM in the System Assembly View tab.

XILINX®

Chapter 2

# ASI to IP Bridge (Transmitter)

## Introduction

This chapter describes the ASI to IP Bridge, also referred to as the Transmitter. This reference system takes from 1 - 8 ASI inputs and encapsulates each transport stream data into a single IP stream output on gigabit Ethernet.

To request access to the Transmitter reference system, visit:

https://secure.xilinx.com/webreg/register.do?group=video_ip

Decision to grant access will be made on an individual basis.

## Features

The Transmitter reference system has the following features.

- 1 to 8 channels of ASI input, full bandwidth up to 1 Gigabit aggregate
- IP output on gigabit Ethernet using the hard TEMAC
- SMPTE 2022-1-2007 and SMPTE 2022-2-2007
- Multi Channel FEC support
- ARP protocol support
- IGMP version 2 support
- No processor is required or included
- Serial control and diagnostics
- Virtex®-5 LXT Platform support
- ASI inputs verified to be operational for a range of 2.5 - 208 Mbps per channel

## Functional Description

The ASI to IP Bridge (Transmitter) takes transport stream data from up to 8 ASI inputs and bridges the data to a gigabit Ethernet interface. It recovers the 10-bit ASI data for each ASI input, decodes it to 8-bit transport stream data, writes the transport stream data into memory, encapsulates up to 7 transport stream packets into an IP packet with UDP and RTP headers, and sends the data out the gigabit Ethernet interface.

## Version

This document applies to version 2.00 of the hardware.

# Overview

The key functional blocks of the design are shown in Figure 2-1.

# Hardware Platform

This reference system runs on the Xilinx ML505 board which is available from Xilinx.

http://www.xilinx.com/ml505

It also requires an 8 channel ASI daughter card from Cook Technologies which is referred to as the CT8ASI 8 Channel DVB-ASI Daughter Card.

http://www.cook-tech.com/ct8asi.html

The daughter card has 8 input BNC connectors on it labelled RX1 to RX8 and 8 output BNC connectors on it labelled TX1 to TX8.

In the HDL design of the Transmitter and a lot of the documentation, channels are numbered from 0 to 7 which can be confusing since the daughter card connectors are numbered from 1 to 8.

# Network Configuration

## IP Addresses

The Transmitter defaults to 192.168.1.100 as its source IP address. For channels 0 - 3, it sends data to IP address 192.168.1.102. For channels 4 - 7, it sends data to 192.168.1.101.

## UDP Ports

The Transmitter defaults to port 1234 as its source UDP port. Table 2-1 contains the destination UDP ports for each channel.

*Table 2-1:* **Channel Destination UDP Ports**

| Daughter Board BNC Connector | Channel | Destination UDP Port |
|---|---|---|
| RX1 | 0 | 1000 |
| RX2 | 1 | 2000 |
| RX3 | 2 | 3000 |
| RX4 | 3 | 4000 |
| RX5 | 4 | 5000 |
| RX6 | 5 | 6000 |
| RX7 | 6 | 7000 |
| RX8 | 7 | 8000 |

## MAC Addresses

The Transmitter defaults to `0xD00DBEEFD00D` for its source MAC address. By default it uses Address Resolution Protocol (ARP) to determine the MAC address of the destination IP address. If

the destination is a multicast address, the destination MAC address will be set as per the Internet Group Management Protocol (IGMP) specification. In lieu of using the MAC address provided by ARP, the destination MAC addresses can also be overridden by writing the desired MAC address in the Destination MAC Address Register and configuring the Control Register appropriately.

# RS-232 Commands

The Transmitter design contains a set of registers which can be read and written, or both, using the RS-232 interface. The registers control the operation of the device and indicate the status of the device. The UART is configured for **57600** baud, **1** stop bit, **8** data bits, and parity of **None**. A dumb terminal can be used to communicate with the device.

## Identify Command: “?”

The ‘?’ character causes the Transmitter to output an ID prompt to the RS-232 interface. The ID prompt indicates the Transmitter and the version of the hardware.

## Read Command: “rAA”

The ‘r’ character followed by two additional address characters cause the device to output the register contents for the hex address specified by AA.

## Write Command: “wAADD”

The ‘w’ character followed by two additional address characters and two additional data characters cause the device to write the value specified by DD to the hex address specified by AA.

## Get Command: “g”

The ‘g’ character causes the Transmitter to dump the contents of all the registers from address 0 to FF.

# Channel Specific Registers Definition

The following text describes the registers which are contained in the Transmitter.

Each register at each address is a byte wide with the most significant bit being bit 7 and least significant bit being bit 0 (i.e. [7:0]).

Offsets for each register are specified rather than an address. The offset must be added to the base address of the channel to get the actual register address for a channel.

## Base Address Map For Each Channel

*Table 2-2:* **Channel Base Address**

| Channel | Base Address |
|---|---|
| 0 | 0x00 |
| 1 | 0x20 |
| 2 | 0x40 |

*Table 2-2:* **Channel Base Address *(Cont'd)***

| Channel | Base Address |
|---|---|
| 3 | 0x60 |
| 4 | 0x80 |
| 5 | 0xA0 |
| 6 | 0xC0 |
| 7 | 0xE0 |

## Offset 0x00, Control Register

This register controls many of the primary functions of the Transmitter.

Non-block aligned FEC is currently not supported. Transport Stream Packet Size defines the input ASI packet size. Destination MAC Address can be resolved by using ARP or the value written to destination MAC address register can be used. The number of TS packets per IP packet must be the same for all channels.

*Table 2-3:* **Control Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 | FEC Alignment | Read/Write | 1 | 1 = block aligned<br>0 = non-block aligned |
| 6 | Transport Stream Packet Size | Read/Write | 1 | 1 = 188 bytes<br>0 = 204 bytes |
| 5 | Destination MAC Address | Read/Write | 0 | 1 = from register<br>0 = from ARP |
| 4 | Row FEC | Read/Write | 1 | 1 = enabled<br>0 = disabled |
| 3 | Column FEC | Read/Write | 1 | 1 = enabled<br>0 = disabled |
| 2 - 0 | TS Packets per IP Packet | Read/Write | 7 | 1 - 7 |

## Offset 0x01: L Value Register

The hardware checks the L value and ensures that 1<=L<=20. If L is set to less than 1, the hardware will use 1 for the L value. If L value is set to greater than 20, the hardware will use 20 for the L value. The product of LxD can not exceed 100.

*Table 2-4:* **L Value Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 5 | Unused | Read/Write | 0 | Unused |
| 4 - 0 | FEC L Value | Read/Write | 4 | The L value controls the number of column packets generated for FEC. |

## Offset 0x02: D Value Register

The hardware checks D value and makes sure that 4<=D<=20. If D is set to less than 4, the hardware will use 4 for the D value. If D is set to greater than 20, the hardware will use 20 for the D value. The product of LxD can not exceed 100.

*Table 2-5:* **D Value Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 5 | Unused | Read/Write | 0 | Unused |
| 4 - 0 | FEC D Value | Read/Write | 4 | The D value controls the number of row packets generated for FEC. |

## Offset 0x03 - 0x06: SSRC Register

Four consecutive byte registers together form a 32-bit register containing the SSRC value that is inserted into the RTP header of each packet.

*Table 2-6:* **SSRC Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x03 | 7 - 0 | SSRC[31:24] | Read/Write | 0x67 | The first byte (most significant) of the SSRC value. |
| 0x04 | 7 - 0 | SSRC[23:16] | Read/Write | 0x45 | The second byte of the SSRC value. |
| 0x05 | 7 - 0 | SSRC[15:8] | Read/Write | 0x23 | The third byte of the SSRC value. |
| 0x06 | 7 - 0 | SSRC[7:0] | Read/Write | 0x01 | The fourth byte of the SSRC value. |

## Offset 0x07 - 0x08: Destination UDP Port Register

Two consecutive byte registers together form a 16-bit register containing the UDP destination port that is inserted into the UDP header of each packet.The default value for destination port 1 is 1000, for destination port 2 is 2000, etc.

*Table 2-7:* **Destination UDP Port Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x07 | 7 - 0 | Destination UDP Port[15:8] | Read/Write | See register description above. | The most significant byte of the destination UDP port. |
| 0x08 | 7 - 0 | Destination UDP Port[7:0] | Read/Write | See register description above. | The least significant byte of the destination UDP port. |

## Offset 0x09 - 0x0C: Destination IP Address Register

Four consecutive byte registers together form a 32-bit register containing the Destination IP Address that is that is inserted into the IP header of each packet.

*Table 2-8:* **Destination IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x09 | 7 - 0 | Destination IP Address[31:24] | Read/Write | 0xC0 | The first (most significant) byte of the Destination IP Address. |
| 0x0A | 7 - 0 | Destination IP Address[23:16] | Read/Write | 0xA8 | The second byte of the Destination IP Address. |
| 0x0B | 7 - 0 | Destination IP Address[15:8] | Read/Write | 0x01 | The third byte of the Destination IP Address. |
| 0x0C | 7 - 0 | Destination IP Address[7:0] | Read/Write | 0x66 for channels 0 - 3, 0x65 for channels 4 - 7 | The fourth byte of the Destination IP Address. |

The default destination IP address for channels 0 - 3 is 192.168.1.102 and for channels 4 - 7 is 192.168.1.101. The purpose of these destination IP addresses is to better facilitate demonstrating the Transmitter together with the Receiver and a PC.

## Offset 0x0D - 0x12: Destination MAC Address Register

Six consecutive byte registers together form a 48-bit register containing the destination MAC address that is inserted into the Ethernet header of each packet. This register value is only used when the Control Register bit 5 is 1 and the destination IP address is not a multicast address.

If the Control register bit 5 is 0, writing to the Destination MAC Address Register does not change the MAC address used for transmitting packets. For unicast destination IP addresses, ARP protocol is used to determine the MAC address. If the destination is a multicast IP address, the destination MAC address will be set as per the IGMP v.2 specification.

If the Control register bit 5 is 1, the MAC used for transmitting packets is read from this register unless the destination IP address is a multicast address, in which case the MAC address used for transmitting is set as per the IGMP v.2 specification.

Reading the Destination MAC Address Register always returns the actual MAC used for transmitting packets.

*Table 2-9:* **Destination MAC Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x0D | 7 - 0 | Destination MAC Address[47:40] | Read/Write | 0xC0 | The first byte of the Destination MAC Address. |
| 0x0E | 7 - 0 | Destination MAC Address[39:32] | Read/Write | 0xED | The second byte of the Destination MAC Address. |
| 0x0F | 7 - 0 | Destination MAC Address[31:24] | Read/Write | 0xBE | The third byte of the Destination MAC Address. |
| 0x10 | 7 - 0 | Destination MAC Address[23:16] | Read/Write | 0xEF | The fourth byte of the Destination MAC Address |
| 0x11 | 7 - 0 | Destination MAC Address[15:8] | Read/Write | 0xCA | The fifth byte of the Destination MAC Address |
| 0x12 | 7 - 0 | Destination MAC Address[7:0] | Read/Write | 0xFE | The sixth byte of the Destination MAC Address |

## Offset 0x13: ASI Input Statistics Register

This register helps expand the number of registers in the memory map by providing a level of indirection through the Stats Read Address register.

*Table 2-10:* **ASI Input Statistics Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 0 | ASI Input Statistics | Read Only | X | The value in this registers depends on the Stat Read Address register. |

## Offset 0x14: Status/Clear Register

This register contains status information when read and allows other information to be cleared when written to such that the read and write definitions are different.

*Table 2-11:* **Status/Clear Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 4 | Unused | | 0 | Unused |
| 3 | Transport Stream Alarm | Read Only | 0 | 1 = no valid transport stream data for ~8 ms<br>0 = a valid transport stream is being received |

*Table 2-11:* **Status/Clear Register *(Cont'd)***

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 2 | Transport Stream RAM Overflow | Read Only | 0 | 1 = an overflow occurred<br>0 = no overflow |
| 1 | Sync Byte Locked | Read Only | 0 | 1 = a sync byte (0x47) has been received<br>0 = no sync byte has been received |
| 0 | Carrier Detect | Read Only | 0 | 1 = an input signal is being received by the channel<br>0 - no input signal |

The write definition of the register is shown in Table 2-12. Each bit in this register is self-clearing after it is set.

*Table 2-12:* **Register Write Definition**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 3 | Unused | | 0 | Unused |
| 2 | Transport Stream RAM Overflow Status Clear | Write Only | 0 | Writing a 1 to this bit clears the overflow status |
| 1 | ASI Recovery Error Count Clear | Write Only | 0 | Writing a 1 to this bit clears the error count |
| 0 | Sync Lost Byte Counter Clear | Write Only | 0 | Writing a 1 to this bit clears the lost byte count |

## Offset 0x15: Packet Drop Register

This register controls the packet drop function of the Transmitter.

*Table 2-13:* **Packet Drop Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 3 | Unused | | 0 | Unused |
| 2 | Channel Disable | Read/ Write | 0 | 1 = Disable transmission from this ASI input channel<br>0 = Transmision is enabled from this ASI input channel |
| 1 | FEC Packet Drop Enable | Read/ Write | 0 | 1 = Drop FEC packets along with TS packets<br>0 = Do not drop any FEC packets<br>This bit is only used when Packet Drop Enable bit is set to 0 |
| 0 | Packet Drop Enable | Read/ Write | 0 | 1 = Enable packet dropping in general and drop TS packets<br>0 = Do not drop any packets |

### Offset 0x16 - 0x17: Packet Drop Interval Register

Two consecutive byte registers together form a 16-bit register that controls the interval between each packet drop. The interval is measured in number of packets.

*Table 2-14:* **Packet Drop Interval Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x16 | 7 - 0 | Interval[15:8] | Read/Write | 0x00 | The first byte (most significant) of the interval value. |
| 0x17 | 7 - 0 | Interval[7:0] | Read/Write | 0x0A | The last byte (least significant) of the interval value. |

## Common Registers Definitions

The following definitions are for the registers which are not channel specific and apply to all the channels. The address for each register is the actual address rather than an offset from a base address as with the channel specific registers.

### Addresses 0x1A - 0x1F: Source MAC Address Register

Six consecutive byte registers together form a 48-bit register containing the source MAC address that is inserted into the Ethernet header of each packet.

*Table 2-15:* **Source MAC Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x1A | 7 - 0 | Source MAC Address[47:40] | Read/Write | 0xD0 | The first byte of the Source MAC Address. |
| 0x1B | 7 - 0 | Source MAC Address[39:32] | Read/Write | 0x0D | The second byte of the Source MAC Address. |
| 0x1C | 7 - 0 | Source MAC Address[31:24] | Read/Write | 0xBE | The third byte of the Source MAC Address. |
| 0x1D | 7 - 0 | Source MAC Address[23:16] | Read/Write | 0xEF | The fourth byte of the Source MAC Address |
| 0x1E | 7 - 0 | Source MAC Address[15:8] | Read/Write | 0xD0 | The fifth byte of the Source MAC Address |
| 0x1F | 7 - 0 | Source MAC Address[7:0] | Read/Write | 0x0D | The sixth byte of the Source MAC Address |

## Addresses 0x3A - 0x3D: Source IP Address Registers

Four consecutive byte registers together form a 32-bit register containing the source IP address that is that is inserted into the IP header of each packet. The default source IP address is 192.168.1.100.

*Table 2-16:* **Source IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x3A | 7 - 0 | Source IP Address[31:24] | Read/Write | 0xC0 | The first byte of the Source IP Address. |
| 0x3B | 7 - 0 | Source IP Address[23:16] | Read/Write | 0xA8 | The second byte of the Source IP Address. |
| 0x3C | 7 - 0 | Source IP Address[15:8] | Read/Write | 0x01 | The third byte of the Source IP Address. |
| 0x3D | 7 - 0 | Source IP Address[7:0] | Read/Write | 0x64 | The fourth byte of the Source IP Address. |

## Addresses 0x3E - 0x3F: Source UDP Port Register

Two consecutive byte registers together form a 16-bit register containing the UDP source port that is inserted into the UDP header of each packet. The default UDP source port is 1234.

*Table 2-17:* **Source UDP Port Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x3E | 7 - 0 | Source UDP Port[15:8] | Read/Write | 0x04 | The most significant byte of the UDP port. |
| 0x3F | 7 - 0 | Source UDP Port[7:0] | Read/Write | 0xD2 | The least significant byte of the UDP port. |

## Address 0xFC: Statistics Read Address Register

This register controls which statistics data is returned in the ASI Input Statistics Register.

*Table 2-18:* **Statistics Read Address Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 0 | Stats Read Address | Read/Write | 0 | This address controls which statistics register is read. |

## Addresses 0xFD - 0xFF: Hardware Version Register

*Table 2-19:* **Hardware Version Registers**

| Address | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0xFD | 7 - 0 | Bug Fix Version Letter | Read Only | Bug Fix Version | A letter used to track bug fixes for a major and minor version. This register will read a 0x00 which is version “a” a read of 0x01 is version “b” etc. |
| 0xFE | 7 - 0 | Major Version Number | Read Only | 0x02 | The major version number only changes when there are large changes to the hardware. |
| 0xFF | 7 - 0 | Minor Version Number | Read Only | 0x00 | The minor version changes whenever there are small changes such as an added feature.<br>Writing a 0xAA to this location will reset the system.<br>The minor version is always read back regardless of what has been written to this location. |

The following register descriptions are statistic registers that are accessed indirectly through the Statistics Read Address Register. The statistic address to be read is written to the Statistics Read Address Register then the data is read from ASI Input Statistics Register.

## Statistic Address 0x04 - 0x06: Transport Stream Packets Count Register

*Table 2-20:* **Transport Stream Packets Count Register**

| Address | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x04 | 7 - 2 | Unused | Read Only | 0 | Unused |
| | 1 - 0 | Valid Transport Stream Packets / Second[17:16] | Read Only | 0 | The most significant two bits of the counter |
| 0x05 | 7 - 0 | Valid Transport Stream Packets per Second[15:8] | Read Only | 0 | The second byte of the counter |
| 0x06 | 7 - 0 | Valid Transport Stream Packets per Second[7:0] | Read Only | 0 | The third byte of the counter |

## Statistic Address 0x08 - 0x09: Sync Byte Lost Count Register

*Table 2-21:* **Sync Byte Lost Count Register**

| Address | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x08 | 7 - 0 | Sync Byte Lost Count[15:8] | Read Only | 0 | The most significant byte of the counter |
| 0x09 | 7 - 0 | Sync Byte Lost Count[7:0] | Read Only | 0 | The least significant byte of the counter |

## Statistic Address 0x0A: Data Status Register

*Table 2-22:* **Data Status Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7-4,2 | Unused | Read Only | 0 | Unused |
| 3 | Sync Byte Locked | Read Only | 0 | 1 = sync byte locked<br>0 = no sync byte |
| 1 - 0 | Transport Stream Format | Read Only | 0 | 00 = 204 byte packets<br>01 = 188 byte packets |

### Statistic Address 0x0C - 0x0E: ASI Recovery Error Count Register

*Table 2-23:* **ASI Recovery Error Count Register**

| Address | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x0C | 7 - 0 | Error Count[23:16] | Read Only | 0 | The first byte (most significant) of the counter |
| 0x0D | 7 - 0 | Error Count[15:8] | Read Only | 0 | The second byte of the counter |
| 0x0E | 7 - 0 | Error Count[7:0] | Read Only | 0 | The third byte of the counter |

# Design Parameters

Certain features are parameterizable to allow the user to generate a Transmitter that is tailored for their system. This allows the user to create a design that only utilizes the resources required by their system and runs at the best possible performance. The features that are parameterizable in the design are shown in Table 2-24.

*Table 2-24:* **Design Parameters**

| Feature/Description | Parameter Name | Allowable Values | Default Values | VHDL Type |
|---|---|---|---|---|
| **Top Level** | | | | |
| FPGA Architecture | C_FAMILY | “Virtex5 | “Virtex5” | string |
| Number of ASI channels | C_ASI_CHANNELS | 1 - 8 | 8 | integer |
| FEC Control | C_INCLUDE_TX_FEC | 0 - 1 | 1 | integer |

## Detailed Parameter Descriptions

### C_FAMILY

This parameter indicates for which FPGA architecture the design is being built. It is currently set to Virtex5 which is the only value supported.

### C_ASI_CHANNELS

This parameter specifies the number of input channels (1 - 8) to build into the hardware. One channel is the quickest build when making system changes.

### C_INCLUDE_TX_FEC

This parameter enables FEC. If this parameter is set to 0, for example, if FEC is not included, the MPMC instance must also be removed from the project as DDR is no longer used.

For more details about removing MPMC, see the `readme.txt` file in the top-level project.

# External DDR Memory

When FEC is enabled, the Transmitter design requires external DDR memory to store the generated FEC IP packets. If FEC is not enabled, DDR is not used and the MPMC memory controller must be removed from the project. The Transmitter uses 128 Kb per channel for data storage. The MPMC is used to interface between the ASI to IP and the external DDR memory. The MPMC NPI port is used for all reads and writes to the DDR. The MPMC NPI address map is defined as follows:

NPI Addr (23 : 17) -> Channel Number

NPI Addr (16) -> 0

NPI Addr(15 : 11) -> Row or Column FEC indicator

11111 => Row FEC

00000 to 10011 => Column L FEC

NPI Addr(10 : 7) -> FEC packet memory, 2KB memory used per IP packet

NPI Addr(6 : 0) -> FEC packet memory, single burst counter, 16 double words.

## Memory Bandwidth

The DDR to MPMC memory interface is 64 bits wide and runs at 250MHz. The NPI PIM latency and throughput are specified in the MPMC data sheet. All reads and writes are done using bursts with a size of 128 bytes (16 double words). The complete packet including all of the IP/UDP/RTP/FEC headers is written into memory. This results in 1390 bytes for each packet assuming 7 TS packets are contained in one IP packet. This requires 11 bursts to write one complete packet into memory. This packet contains 1316 bytes of transport stream data (7*188). This results in approximately 7% overhead due to IP headers and burst alignment. A transport stream requires approximately 4.28x memory bandwidth, composed of 1.07x for row FEC writes, 1.07x for row FEC reads, 1.07x for column FEC writes, and 1.07x for column FEC reads. This calculation does not include addressing, arbitration or refresh times, all of which reduce the available memory bandwidth.

# Building Hardware

The hardware is built using Xilinx EDK development platform.

To execute the system using EDK, follow these steps:

1. Open `.xmp` inside EDK.
2. In the System Assembly View tab, right click on **Configure** to configure the transmitter.
3. Use **Hardware** → **Generate Bitstream** to generate a bitstream for the system.
4. Download the bitstream to the board with **Device Configuration** → **Download Bitstream**.

## Directory Structure

The following paragraphs describe the primary directories for the design.

### /

The top level directory contains system project files. This includes the `.xmp` project file and the `.mhs` file. The `readme.txt` file details revision history with a list of known issues at the time of release.

### /data

This directory contains the constraint file for the system.

### /implementation

This directory contains the implementation of the design including the bit file to be downloaded to the FPGA. This directory is removed when a **Hardware → Clean Hardware** command is issued from EDK. The `.mcs` PROM file is not automatically created and must be generated using the Xilinx Impact tool.

### /pcores

This directory contains the HDL source files, in VHDL, for the design.

### /simulation

This directory contains the simulation files for the system.

## Source Files

All of the design is provided as VHDL source code.

## Libraries

The project contains multiple VHDL libraries to help partition the design into logical pieces. Each library is a subdirectory in the `/pcores` directory.

### /pcores/tx_ancillary_v2_00_a

This library contains the clock generation for the project.

### /pcores/tx_asi_intf_v2_00_a

This library contains all the ASI input and output interfaces.

### /pcores/tx_ts_input_v2_00_a

This library contains all the transport stream specific processing such as synchronization.

### /pcores/tx_rs232_57600_v2_00_a

This library contains the UART and the command processing to read and write the registers.

### /pcores/tx_pro_mpeg_cop3_v2_00_a

This library contains the processing to do IP, UDP, and RTP encapsulation of the transport stream and the FEC generation.

### /pcores/tx_gb_intf_v2_00_a

This library contains the gigabit Ethernet interface which uses the hard TEMAC and the ARP processing.

### /pcores/tx_top_v2_00_a

This library contains the VHDL package that defines common types used throughout the project. It also contains some structures to help generate some of the CORE Generator components used in the design.

### /pcores/asitoip_v2_00_a

This library contains the top level file for the project.

## Typical Modifications

### Number Of Channels

The number of channels can be configured by right-clicking on the Transmitter instance in the **System Assembly View** tab in EDK.

### Include FEC

FEC generation can be included or excluded by configuring the Transmitter in the **System Assembly View** tab in EDK. If FEC is excluded, the MPMC memory controller must also be removed from the project.

For more details about removing MPMC, see the `readme.txt` fle in the top-level project.

### Register Defaults Including MAC Addresses & IP Addresses

The source file, `/pcores/tx_top_v2_00_a/hdl/vhdl/asitoip_pkg.vhd` is a VHDL package file that contains the constants used by the Transmitter. The following constants can be modified to customize the system at build time:

- Source IP address
- Source MAC address
- Source UDP Port address
- Debug Mode
- Default Gateway IP address. Set this when the destination IP address for any channel lies outside of the Transmitter's subnet
- Subnet Mask

## EDK Version

EDK 10.1.3 has been used to build the Transmitter.

# Block Diagrams

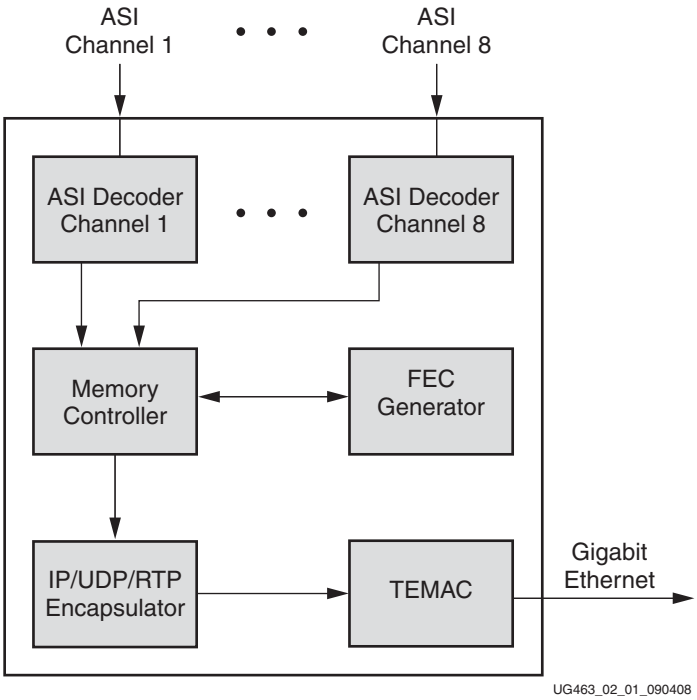


*Figure 2-1:* **ASI to IP Out Bridge Transmitter**

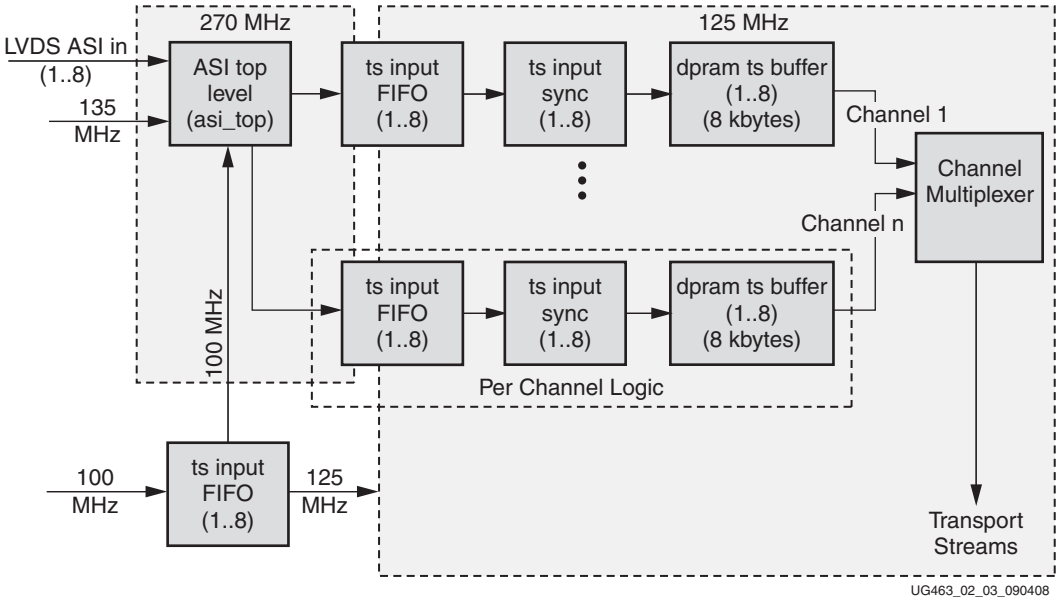


*Figure 2-2:* **Top Level Library - Part 1**

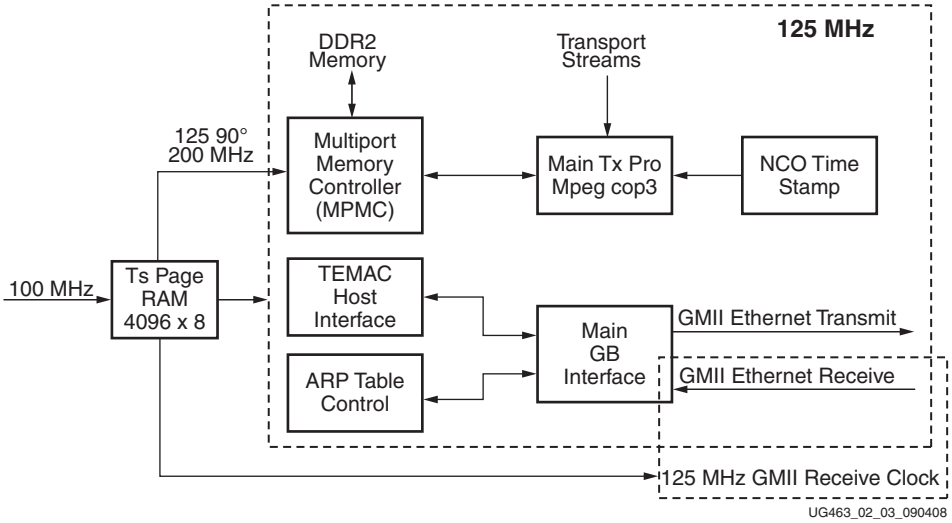


*Figure 2-3:* **Top Level Library - Part 2**

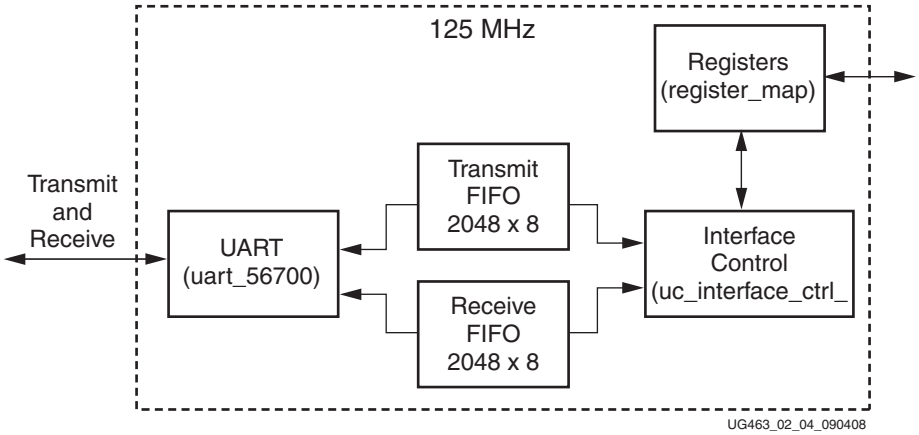


*Figure 2-4:* **Top Level Library - Part 3**

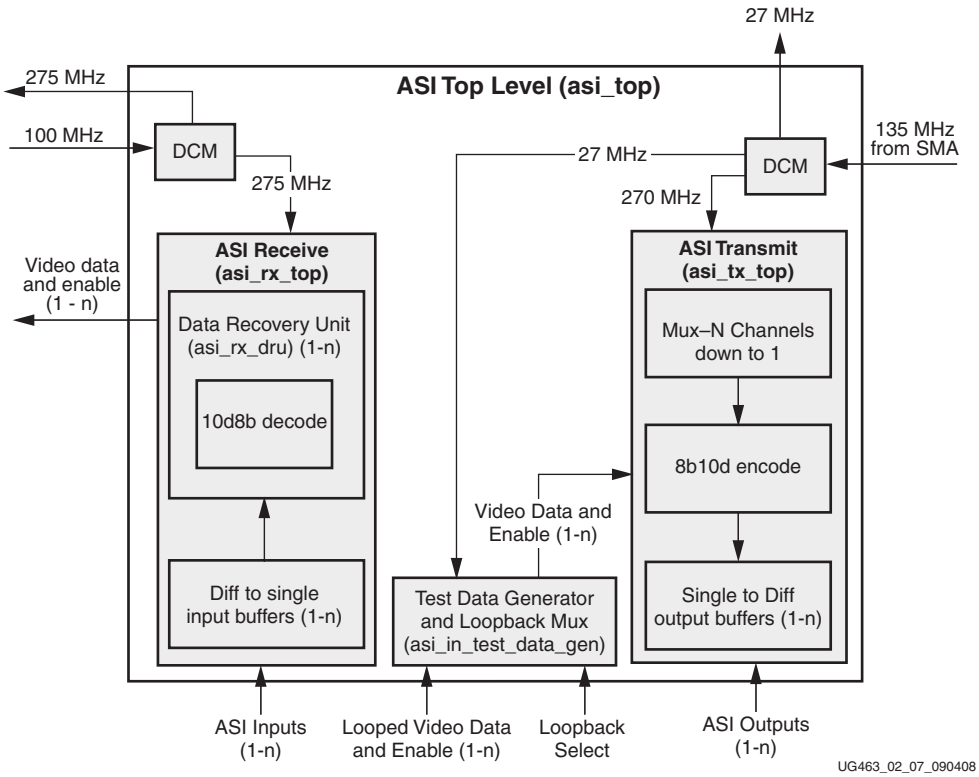


*Figure 2-5:* **Pro-MPEG Transmitter - tx_asi_intf Library**

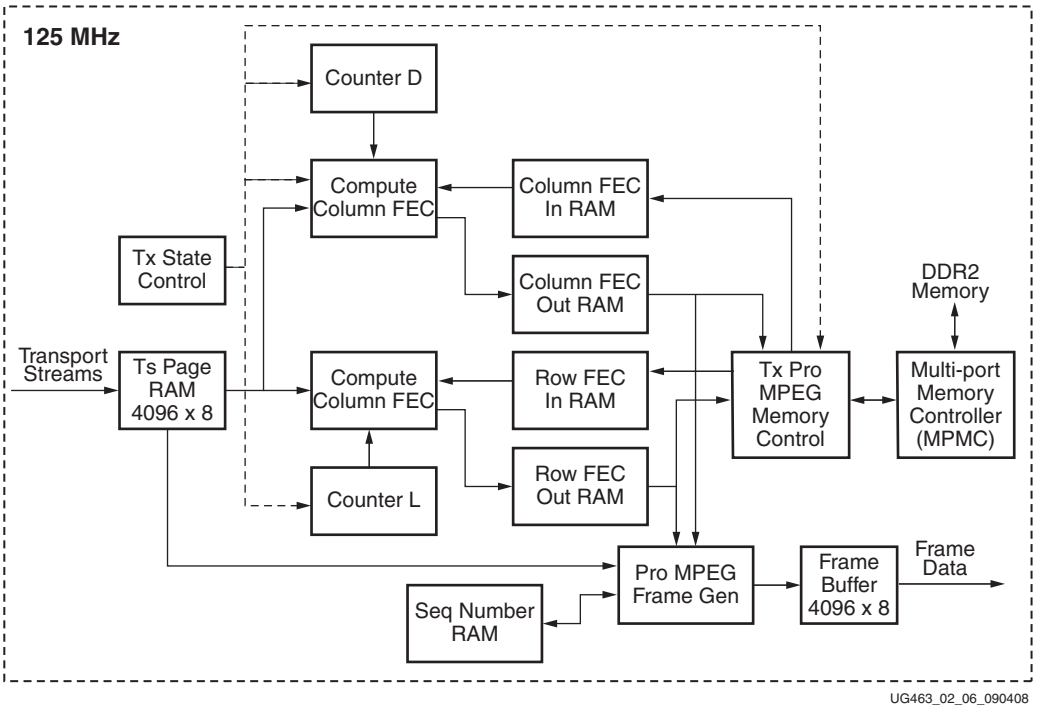


*Figure 2-6:* **tx_pro_mpeg_cop3 Library**

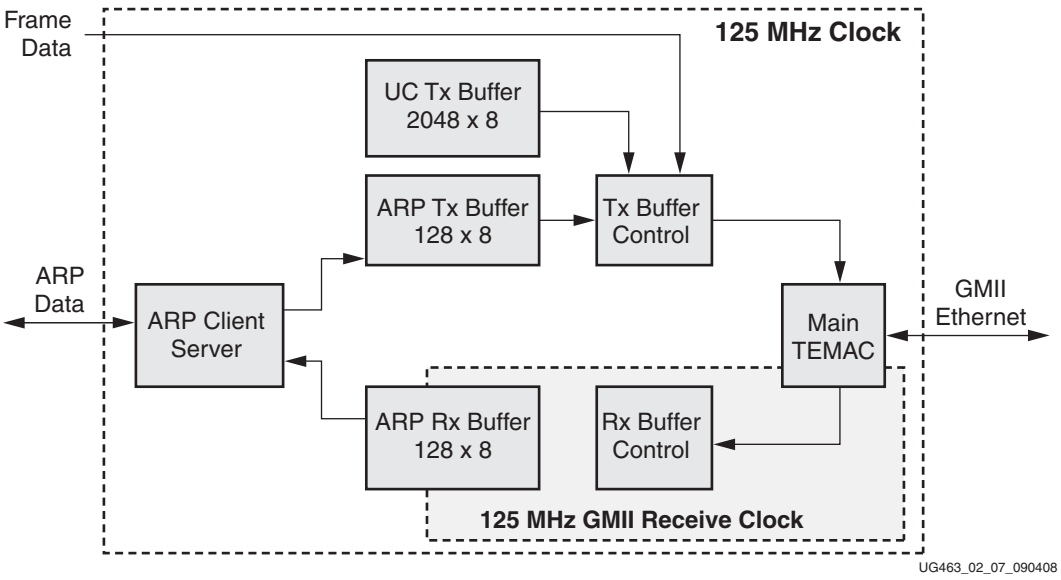


*Figure 2-7:* **tx_gb_intf Library**

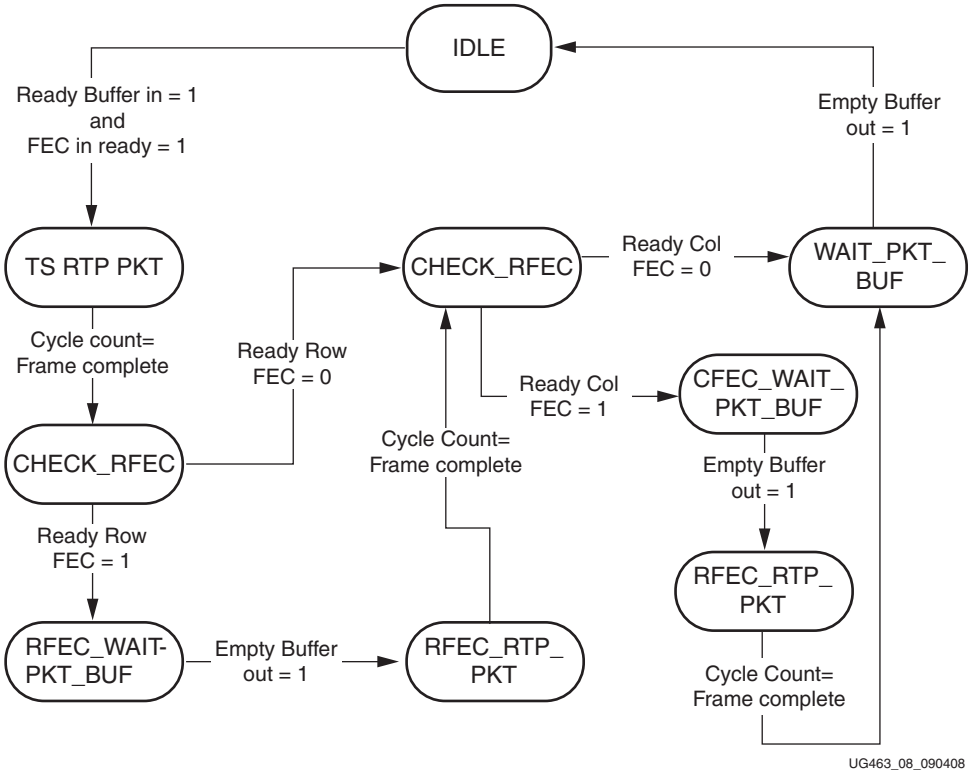


*Figure 2-8:* **Tx State Control**

# Test Modes

The design includes a loopback which connects the receive channels to the transmit channels such that a single ASI source can be used to drive all the ASI input channels.

The data received from each receive channel is transmitted out to the next higher channel (RX1 to TX2, RX2 to TX3, and so forth) except RX8 which is sent to TX1. The TX channels must still be connected to an RX connector with a cable to utilize the data.

The SMA cables that connect the daughter board clock to the ML505 clock input are only needed for this test mode.

# Known Issues

For all known issues, please read the `readme.txt` file in the top level directory of the Transmitter reference system.

# Limitations

The following items document the known limitations of the Transmitter reference system.

1. It only supports 1 gigabit over Ethernet. It will not autonegotiate to 10/100 as the design needs the 125 MHz PHY clock of the 1 Gigabit rate. It would require some redesign to work around this. This has not been a major issue since a network switch can be used to connect to 10/100 Ethernet devices.
2. The Transmitter does not fragment IP packets such that any device receiving the IP data over Ethernet must be able to accept large (up to 1500 byte) Ethernet packets.
3. Non Block Aligned FEC is not supported.
4. The number of TS packets per IP packet must be the same for all channels.

# FPGA Resources

The following resources are for the Virtex-5 architecture.

*Table 2-25:* **Virtex-5 Architecture FPGA Resources**

| C_ASI_CHANNELS | C_INCLUDE_TX_FEC | LUTs | Flip Flops | BRAMs | DCMs | Hard TEMACs |
|---|---|---|---|---|---|---|
| 1 | 0 | 2699 | 2144 | 9 | 5 | 1 |
| 1 | 1 | 5305 | 5099 | 26 | 5 | 1 |
| 8 | 0 | 8034 | 8257 | 27 | 5 | 1 |
| 8 | 1 | 10845 | 11299 | 44 | 5 | 1 |

Chapter 3

# IP to ASI Bridge (Receiver)

## Introduction

This chapter describes the IP to ASI Bridge, also referred to as the Receiver. This reference system takes a single IP stream input on gigabit Ethernet and bridges the transport streams of eight ports onto eight ASI outputs.

To request access to the Receiver reference system, visit:

https://secure.xilinx.com/webreg/register.do?group=video_ip

Decision to grant access will be made on an individual basis.

## Features

The Receiver reference system has the following features.

- a single IP input on gigabit Ethernet using the hard TEMAC
- 1 to 8 channels of ASI output, full bandwidth up to 1 gigabit aggregate
- ASI bitrates from 2.5 Mbps to 208 Mbps
- SMPTE 2022-1-2007 and SMPTE 2022-2-2007
- Multi Channel FEC support
- ARP protocol support
- IGMP version 2 support
- No processor is required or included
- Serial control and diagnostics
- Virtex-5 LXT Platform support

## Functional Description

The IP to ASI Bridge (Receiver) takes transport stream data from up to 8 UDP ports of the IP input on a single gigabit Ethernet interface and bridges the data onto up to 8 ASI outputs. It writes the transport stream data into memory, de-encapsulates the transport stream packets from the IP packet with UDP and RTP data, recovers lost packets using FEC, generates the ASI output rate from the RTP data of the packets, and sends the data out the ASI output based on the UDP port from which it came.

## Version

This document applies to version 2.00 of the hardware.

# Overview

The block diagram shown in Figure 3-1 illustrates the key functional blocks of the design.

# Hardware Platform

This reference system runs on the Xilinx ML505 board which is available from Xilinx.

http://www.xilinx.com/ml505

It also requires an 8 channel ASI daughter card from Cook Technologies which is referred to as the CT8ASI 8 Channel DVB-ASI Daughter Card.

http://www.cook-tech.com/ct8asi.html

The daughter card has 8 input BNC connectors on it labelled RX1 to RX8 and 8 output BNC connectors on it labelled TX1 to TX8.

In the HDL design of the Receiver and a lot of the documentation, channels are numbered from 0 to 7 which can be confusing since the daughter card connectors are numbered from 1 to 8.

# Network Configuration

## IP Addresses

The Receiver defaults to 192.168.1.101 as its board IP address.

## UDP Ports

Table 3-1 contains the UDP ports for each channel. The Receiver defaults to 0x2345 as it board UDP port. The board UDP port is not currently used by the IP to ASI system.

*Table 3-1:* **Channel UDP Input Ports**

| Daughter Board BNC Connector | Channel | Destination UDP Port |
|---|---|---|
| TX1 | 0 | 1000 |
| TX2 | 1 | 2000 |
| TX3 | 2 | 3000 |
| TX4 | 3 | 4000 |
| TX5 | 4 | 5000 |
| TX6 | 5 | 6000 |
| TX7 | 6 | 7000 |
| TX8 | 7 | 8000 |

## MAC Addresses

The Receiver defaults to `0xC0EDBEEFCAFE` for its MAC address. It will send out ARP responses when it receives ARP requests.

### IGMP

The Receiver defaults to `0.0.0.0` for its IGMP address. Upon writing a valid multicast address to the Multicast IP address register the receiver will join the session. Once the receiver has joined a session, and if the multicast address is then changed, the receiver will first leave the current session and then join the new session.

## RS-232 Commands

The Receiver design contains a set of registers which can be read and written, or both, using the RS-232 interface. The registers control the operation of the device and indicate the status of the device. The UART contained in the system is configured for **57600** baud, **1** stop bit, **8** data bits, and parity of **None**. A dumb terminal can be used to communicate with the device.

### Identify Command: “?”

The ‘?’ character causes the Receiver to output an ID prompt to the RS-232 interface. The ID prompt indicates it is the Receiver and the version of the hardware.

### Read Command: “rAA”

The ‘r’ character followed by two additional address characters cause the device to output the register contents for the hex address specified by AA.

### Write Command: “wAADD”

The ‘w’ character followed by two additional address characters and two additional data characters cause the device to write the value specified by DD to the hex address specified by AA.

### Get Command: “g”

The ‘g’ character causes the Receiver to dump the contents of all the registers from address 0 to FF.

## Channel Specific Registers Definition

The following text describes the registers which are contained in the Receiver.

Each register at each address is byte wide with most significant bit being bit 7 and least significant bit being bit 0 (for example [7:0]).

Offsets for the register are specified rather than an address. The offset must be added to the base address of the channel to get the actual register address for a channel.

### Base Address Map For Each Channel

*Table 3-2:* **Channel Base Address**

| Channel | Base Address |
|---|---|
| 0 | 0x00 |
| 1 | 0x20 |

*Table 3-2:* **Channel Base Address *(Cont'd)***

| Channel | Base Address |
|---|---|
| 2 | 0x40 |
| 3 | 0x60 |
| 4 | 0x80 |
| 5 | 0xA0 |
| 6 | 0xC0 |
| 7 | 0xE0 |

## Offset 0x00: Control Register

This register controls the operation of the channel.

The FIFO metering mode can not be used with FEC to recover lost packets. FIFO metering mode is primarily used as a test mode to allow the IP to ASI to receive packets with improper RTP time stamps or a system with to much jitter.

.

*Table 3-3:* **Control Register**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 3 | Unused | Read/Write | 0 | Unused |
| 2 - 1 | Firewall Select | Read/Write | 0 | '11' = The firewall checks the source IP address and the SSRC<br>'10' = The firewall checks the SSRC only<br>'01' = The firewall checks the source IP address only<br>'00' = The firewall does not check either the source IP address or the SSRC |
| 0 | ASI Playout Rate Basis | Read/Write | 0 | 1= Use FIFO metering (more flexible)<br>0 = Use the RTP timestamp (lower jitter) |

## Offset 0x01: Status Register 1

*Table 3-4:* **Status Register 1**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 | Transport Stream Packet Size | Read | 0 | 1 = 188 byte transport stream packets being received<br>0 = 204 byte transport stream packets being received |
| 6 | Unused | Read | 0 | Unused |
| 5 | FEC Loss Of Sync | Read | 0 | 1 = FEC has loss of synchronization with input<br>0 = FEC is synchronized |
| 4 - 0 | FEC L Value | Read | 0 | The L value being received in RTP packets for the channel. |

## Offset 0x02: Status Register 2

*Table 3-5:* **Status Register 2**

| Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 - 5 | Transport Stream Packets / IP Packet | Read | 0 | The number of transport stream packets per IP packet (1 - 7) |
| 4 - 0 | FEC D Value | Read | 0 | The D value being received in RTP packets for the channel. |

## Offset 0x03 - 0x06: SSRC Firewall Register

Four consecutive byte registers together form a 32-bit register containing the SSRC. This value is used when the SSRC firewall is enabled and will prevent packets from being processed if the transmitted SSRC does not match this value.

*Table 3-6:* **SSRC Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x03 | 7 - 0 | SSRC[31:24] | Read/Write | 0x67 | The first byte (most significant) of the SSRC value. |
| 0x04 | 7 - 0 | SSRC[23:16] | Read/Write | 0x45 | The second byte of the SSRC value. |
| 0x05 | 7 - 0 | SSRC[15:8] | Read/Write | 0x23 | The third byte of the SSRC value. |
| 0x06 | 7 - 0 | SSRC[7:0] | Read/Write | 0x01 | The fourth byte of the SSRC value. |

## Offset 0x07 - 0x08: Source UDP Port Register

Two consecutive byte registers together form a 16-bit register containing the UDP source port that is monitored for TS packets for this particular channel. The default UDP source port value for channel 1 is 1000, for channel 2 is 2000, etc., as defined in Table 3-1.

*Table 3-7:* **Source UDP Port Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x07 | 7 - 0 | Source UDP Port[15:8] | Read/Write | See description above | The most significant byte of the UDP port. |
| 0x08 | 7 - 0 | Source UDP Port[7:0] | Read/Write | See description above | The least significant byte of the UDP port. |

## Offset 0x09 - 0x0C: Source IP Address Firewall Register

Four consecutive byte registers together form a 32-bit register containing the IP address of the source that is expected to send data to the Receiver. This value is used when the source IP address firewall is enabled and will prevent packets from being processed if the transmitted source IP address does not match this value. The default value for this register is 192.168.1.100.

*Table 3-8:* **Destination IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x09 | 7 - 0 | Destination IP Address[31:24] | Read/Write | 0xC0 | The fourth byte of the Destination IP Address. |
| 0x0A | 7 - 0 | Destination IP Address[23:16] | Read/Write | 0xA8 | The third byte of the Destination IP Address. |
| 0x0B | 7 - 0 | Destination IP Address[15:8] | Read/Write | 0x01 | The second byte of the Destination IP Address. |
| 0x0C | 7 - 0 | Destination IP Address[7:0] | Read/Write | 0x64 | The first (least significant) byte of the Destination IP Address. |

## Offset 0x0D - 0x0F: IP Packet Count Received Register

Three consecutive byte registers together form a 24-bit register containing the number of IP packets received for the channel. A write to channel 0 control register resets this register for all channels.

*Table 3-9:* **IP Packet Count Received Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x0D | 7 - 0 | Counter[23:16] | Read | 0x00 | The first byte (most significant) of the counter. |
| 0x0E | 7 - 0 | Counter[15:8] | Read | 0x00 | The second byte of the counter. |
| 0x0F | 7 - 0 | Counter[7:0] | Read | 0x00 | The third byte (least significant) of the counter. |

## Offset 0x10 - 0x12: ASI Output Rate Register

Three consecutive byte registers together form a 24-bit register containing the bit rate of the ASI output for the channel. The value in this register converted from hex to decimal and multiplied by 32 is the ASI output rate in hertz.

*Table 3-10:* **ASI Output Rate Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x10 | 7 - 0 | Rate[23:16] | Read | 0x00 | The first byte (most significant) of the rate. |
| 0x11 | 7 - 0 | Rate[15:8] | Read | 0x00 | The second byte of the rate. |
| 0x12 | 7 - 0 | Rate[7:0] | Read | 0x00 | The third byte (least significant) of the rate. |

## Offset 0x13 - 0x15: Missing Media Packet Count Register

Three consecutive byte registers together form a 24-bit register containing the number of missing media packets. A write to channel 0 control register resets this register for all channels.

*Table 3-11:* **Missing Media Packet Count Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x13 | 7 - 0 | Counter[23:16] | Read | 0x00 | The first byte (most significant) of the counter. |
| 0x14 | 7 - 0 | Counter[15:8] | Read | 0x00 | The second byte of the counter. |
| 0x15 | 7 - 0 | Counter[7:0] | Read | 0x00 | The third byte (least significant) of the counter. |

## Offset 0x16 - 0x17: Program Identifier (PID) Control Register

Two consecutive byte registers together form a 16-bit register containing the PID which is monitored in the transport stream. If the PID has not been written to, the first PID that is found in the data is used for monitoring CC errors. Reading this register when an IP input stream is not present for this channel will report 0x00.

*Table 3-12:* **PID Control Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x16 | 7 - 0 | PID[15:8] | Read/Write | 0x00 | The first byte (most significant) of the PID. |
| 0x17 | 7 - 0 | PID[7:0] | Read/Write | 0x00 | The second byte of the PID. |

## Offset 0x18 - 0x19: Continuity Count (CC) Error Count Register

Two consecutive byte registers together form a 16-bit register containing the number of CC errors in the transport stream for the PID specified by the PID Control Register. A write to channel 0 control

register resets this register for all channels. Each discontinuity in the continuity counter of the transport stream is a single error in this register.

*Table 3-13:* **Continuity Count Error Count Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x18 | 7 - 0 | Counter[15:8] | Read | 0x00 | The first byte (most significant) of the counter. |
| 0x19 | 7 - 0 | Counter[7:0] | Read | 0x00 | The second byte of the counter. |

### Offset 0x1A - 0x1C: Packets Recovered with FEC Register

Three consecutive byte registers together form a 24-bit register containing the number of packets recovered using FEC. A write to channel 0 control register resets this register for all channels.

*Table 3-14:* **Packets Recovered with FEC Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x1A | 7 - 0 | Counter[23:16] | Read | 0x00 | The first byte (most significant) of the counter. |
| 0x1B | 7 - 0 | Counter[15:8] | Read | 0x00 | The second byte of the counter. |
| 0x1C | 7 - 0 | Counter[7:0] | Read | 0x00 | The third byte (least significant) of the counter |

## Common Registers Definitions

The following definitions are for the registers which are not channel specific and apply to all the channels. The addresses for each register are the actual address rather than an offset from a base address as with the channel specific registers.

### Board MAC Address Register

The following six non-consecutive byte registers together form a 48-bit register containing the board MAC address. The default board MAC Address is 0xC0EDBEEFCAFE

*Table 3-15:* **Source MAC Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x1E | 7 - 0 | Board MAC Address[47:40] | Read/Write | 0xC0 | The first byte (most significant) of the board MAC Address. |
| 0x1F | 7 - 0 | Board MAC Address[39:32] | Read/Write | 0xED | The second byte of the board MAC Address. |
| 0x3E | 7 - 0 | Board MAC Address[31:24] | Read/Write | 0xBE | The third byte of the board MAC Address. |
| 0x3F | 7 - 0 | Board MAC Address[23:16] | Read/Write | 0xEF | The fourth byte of the board MAC Address |

*Table 3-15:* **Source MAC Address Register *(Cont'd)***

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x5E | 7 - 0 | Board MAC Address[15:8] | Read/Write | 0xCA | The fifth byte of the board MAC Address |
| 0x5F | 7 - 0 | Board MAC Address[7:0] | Read/Write | 0xFE | The sixth byte (least significant) of the board MAC Address |

## Board IP Address Register

Four non-consecutive byte registers together form a 32-bit register containing the board IP address. The default board IP address is 192.168.1.101.

*Table 3-16:* **Source IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0x7E | 7 - 0 | Board IP Address[31:24] | Read/Write | 0xC0 | The first byte of the board IP Address. |
| 0x7F | 7 - 0 | Board IP Address[23:16] | Read/Write | 0xA8 | The second byte of the board IP Address. |
| 0x9E | 7 - 0 | Board IP Address[15:8] | Read/Write | 0x01 | The third byte of the board IP Address. |
| 0x9F | 7 - 0 | Board IP Address[7:0] | Read/Write | 0x65 | The fourth byte of the board IP Address. |

## Multicast IP Address Register

Four non-consecutive byte registers together form a 32-bit register containing the board multicast IP address. The default board multicast IP address is 0.0.0.0.

*Table 3-17:* **Source IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0xBE | 7 - 0 | Multicast IP Address[31:24] | Read/Write | 0x00 | The first byte of the board IP Address. |
| 0xBF | 7 - 0 | Multicast IP Address[23:16] | Read/Write | 0x00 | The second byte of the board IP Address. |
| 0xDE | 7 - 0 | Multicast IP Address[15:8] | Read/Write | 0x00 | The third byte of the board IP Address. |
| 0xDF | 7 - 0 | Multicast IP Address[7:0] | Read/Write | 0x00 | The fourth byte of the board IP Address. |

## Monitored UDP Port Register

Two consecutive byte registers together form a 16-bit register containing the UDP Port that is monitored for control traffic. The default value of the monitored UDP control port is 0x2345. Currently, this register is not used.

*Table 3-18:* **Source IP Address Register**

| Offset | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0xBD | 7 - 0 | Monitored UDP port[15:8] | Read/Write | 0x23 | The first byte of the monitored UPD port. |
| 0xDD | 7 - 0 | Monitored UDP Port[7:0] | Read/Write | 0x45 | The second byte of the monitored UPD port. |

## Hardware Version Register

*Table 3-19:* **Hardware Version Registers**

| Address | Bit(s) | Name | Access | Reset Value | Description |
|---|---|---|---|---|---|
| 0xFD | 7 - 0 | Bug Fix Version Number | Read Only | Bug Fix Version | A letter used to track bug fixes for a major and minor version. This register will read a 0x00 which is version “a” a read of 0x01 is version “b” etc. |
| 0xFE | 7 - 0 | Major Version Number | Read Only | 0x02 | The major version number only changes when there are large changes to the hardware. |
| 0xFF | 7 - 0 | Minor Version Number | Read/Write | 0x00 | The minor version changes whenever there are small changes such as an added feature.<br>Writing a 0xAA to this location will reset the system.<br>The minor version is always read back regardless of what has been written to this location. |

# Design Parameters

To allow the user to generate a IP to ASI receiver that is tailored for their system, certain features are parameterizable. This allows the user to have a design that only utilizes the resources required by their system and runs at the best possible performance. The features that are parameterizable in the design are shown in Table 3-20

*Table 3-20:* **Design Parameters**

| Feature/Description | Parameter Name | Allowable Values | Default Values | VHDL Type |
|---|---|---|---|---|
| **Top Level** | | | | |
| FPGA Architecture | C_FAMILY | “Virtex5 | “Virtex5” | string |
| Number of ASI channels | C_NUM_ASI_CHANNELS | 1 - 8 | 8 | integer |
| FEC Control | C_INCLUDE_RX_FEC | 0 - 1 | 1 | integer |

## Detailed Parameter Descriptions

### C_FAMILY

This parameter indicates for which FPGA architecture the design is being built. It is currently set to Virtex5 which is the only value supported.

### C_NUM_ASI_CHANNELS

This parameter specifies the number of input channels (1 - 8) to build into the hardware. One channel is the quickest build when making system changes.

### C_INCLUDE_RX_FEC

This parameter enables Forward Error Correction (FEC).

# External DDR Memory

The Receiver design requires external DDR memory to store the received IP packets. It uses 4 MByte per channel for storage of the IP data. The MPMC is used to interface between the IP to ASI and the external DDR memory. The MPMC NPI port is used for all reads and writes to the DDR. The MPMC NPI address map is defined as follows:

NPI Addr (24 : 22) -> Channel Number(2 : 0)

NPI Addr(21 : 20) -> Type of Frame

00 => Media

01 => Column FEC

10 => Row FEC

11 => Unused

NPI Addr(19 : 11) -> RTP Sequence Number (8 : 0)

NPI Addr(10 : 7) -> IP packet memory, 2KB memory used per IP packet

NPI Addr(6 : 0) -> IP packet memory, single burst counter, 16 double words

## Memory Bandwidth

The DDR to MPMC memory interface is 64 bits wide and runs at 250MHz. The IP to ASI uses the NPI PIM as the interface for all three ports. The NPI PIM latency and throughput are specified in the

MPMC data sheet. All reads and writes are done using bursts with a size of 128 bytes (16 double words). The complete packet including all of the IP/UDP/RTP header is written into memory. This results in 1374 bytes for each media packet assuming 7 TS packets are contained in one IP packet. This requires 11 bursts to write one complete packet into memory. This packet contains 1316 bytes of transport stream data (7*188). This results in approximately 7% overhead due to IP headers and burst alignment. A transport stream bit rate requires approximately 2.14x memory bandwidth, 1.07x for writes and 1.07x for reads.

Forward Error Correction packets are also stored in DDR memory and require memory bandwidth. An FEC packet is 1390 bytes in length for a packet containing 7 TS packets per IP packet. The FEC packets are only read when a lost packet can be recovered. The FEC packets will require approximately 1.07x memory bandwidth for a write. The number of FEC packets is dependent upon the size of LxD matrix. The FEC packet overhead is (L + D)/(L*D).

The FEC must access DDR to recover lost packets. Each lost packet requires either L or D packets to be read depending upon whether a row or column will be used to recover the lost packet. The recovered packet is written back into DDR.

# Algorithms

## NCO Adjustment

In RTP mode, the Receiver uses RTP timestamps to determine the rate of the ASI output. Based on the difference between timestamps in multiple packets, the Receiver uses an NCO (numerically controlled oscillator) to generate the ASI output rate. A critical aspect of the ASI output is the amount of jitter and RTP timestamps allow the jitter to be minimized.

The RTP timestamp contained in the received packets has 90 KHz resolution. In the current reference system, the Transmitter generates the RTP timestamps in the packets on the IP network. The Transmitter and Receiver oscillators are nominally the same frequency, but the Transmitter and Receiver are not synchronized.

The NCO also performs timestamp processing to help minimize the amount of jitter. It goes through a learning phase in which it determines the delta between the timestamp of a number of packets. After a number of packets have been received, referred to as the learning phase, the output rate is updated every $2^{21}$ packets.

Each channel of the Receiver uses a fixed amount of external DDR memory to buffer the incoming IP network packets before they are sent over the ASI output corresponding to the UDP port. The amount of memory is large enough to provide elasticity for when the input and output rates do not match exactly, but over enough time, the NCO must be adjusted to prevent an underrun or an overrun.

The Receiver incorporates an NCO adjustment which monitors the number of packets which are buffered for each channel and then adjusts the NCO to better match the output rate to the input rate. The NCO adjustment uses watermarks around the center point of the DDR buffer to determine when the NCO should be faster or slower. It also incorporates a filter such that it will try to better match the output rate to the input rate over time rather than continually adjusting the NCO.

The NCO uses an increment value that has 22 bits of resolution together with an accumulator and overflow values that are 32 bits. The smallest change to the increment value has been found to cause an amount of jitter that is larger than desired. Prototyping has shown that the increment value can be increased in resolution along with the accumulator and overflow values. The extra precision could allow the NCO to be adjusted in smaller increments such that less jitter is caused.

### FIFO Metering

FIFO metering is a special mode in the Receiver that allows it to receive RTP data without using the RTP timestamps. This mode works well for demonstrations when a PC is used to send video data to the Receiver as many video players on the PC do not create good timestamps. This mode is not intended for a production mode as the jitter is not as good as when using the RTP timestamps.

## Building Hardware

The hardware is built using Xilinx EDK development platform.

To build the system using EDK, follow these steps:

1. Open `.xmp` inside EDK.
2. Configure the IP to ASI by right-clicking on it in the System Assembly View tab
3. Use **Hardware** → **Generate Bitstream** to generate a bitstream for the system.
4. Download the bitstream to the board with **Device Configuration** → **Download Bitstream**.

## Directory Structure

The following paragraphs describe the primary directories for the design.

### /

The top level directory contains system project files. This includes the `.xmp` project file and the `.mhs` file. The `readme.txt` file details revision history with a list of known issues at the time of release.

### /data

This directory contains the constraint file for the system.

### /implementation

This directory contains the implementation of the design including the bit file to be downloaded to the FPGA. This directory is removed when a **Hardware → Clean Hardware** command is issued from EDK. The `.mcs` PROM file is not automatically created and must be generated using the Xilinx Impact tool.

### /pcores

This directory contains the VHDL libraries for the design.

### /simulation

This directory contains the simulation files for the system.

## Source Files

All of the design is provided as VHDL source code.

## Libraries

The project contains multiple VHDL libraries to help partition the design into logical pieces. Each library is a subdirectory in the pcores directory.

### pcores/iptoasi_v2_00_a/hdl/vhdl

This directory contains the top level file.

### pcores/iptoasi_v2_00_a/data

This directory contains the top level EDK project files.

### pcores/iptoasi_v2_00_a/netlist

This directory contains the Chipscope ILA and ICON for use when debug is enabled in the IP to ASI.

### pcores/rx_ancillary_v2_00_a/hdl/vhdl

This library contains the clock generation and the package file for the project. The package file contains common types used throughout the project, together with constants, such as the hardware version numbers.

### pcores/rx_asi_out_v2_00_a/hdl/vhdl

This library contains all the ASI output processing.

### pcores/rx_fec_v2_00_a/hdl/vhdl

This library contains the forward error correction (FEC) specific processing files.

### pcores/rx_gb_intf_v2_00_a/hdl/vhdl

This library contains the interface to the hard TEMAC. ARP and IGMP processing along with initialization of the TEMAC are contained here.

### pcores/rx_packetizer_v2_00_a/hdl/vhdl

This library contains all the transport stream specific processing such as synchronization. The NCO processing and interfacing to the MPMC are contained here.

### pcores/rx_rs232_57600_v2_00_a/hdl/vhdl

This library contains the UART and the command processing to read and write the registers.

## Typical Modifications

### Number Of Channels

The number of channels can be changed by configuring the iptoasi in the System Assembly View tab.

### Include FEC

To include or exclude Forward Error Correction (FEC) configure the iptoasi in the System Assembly View tab.

### Register Defaults Including MAC Addresses & IP Addresses

The source file, `/pcores/rx_ancillary_v2_00_a/hdl/vhdl/iptoasi_pkg.vhd` is a VHDL package file that contains the constants used by the receiver. For example the version number is a constant in the package file. Debug mode is also enabled in the package file and is used to connect signals to chipscope.

The following default values also located in the package file can be changed via the RS-232 interface. However, if the system needs to power up with a different value the constant can be change in the package file.

Board IP Address

Board MAC Address

Board UDP port

Board Multicast MAC Address

Board Multicast IP Address

Group Multicast Address

### EDK Version

EDK 10.1.3 has been used to build the IP to ASI.

## Block Diagrams

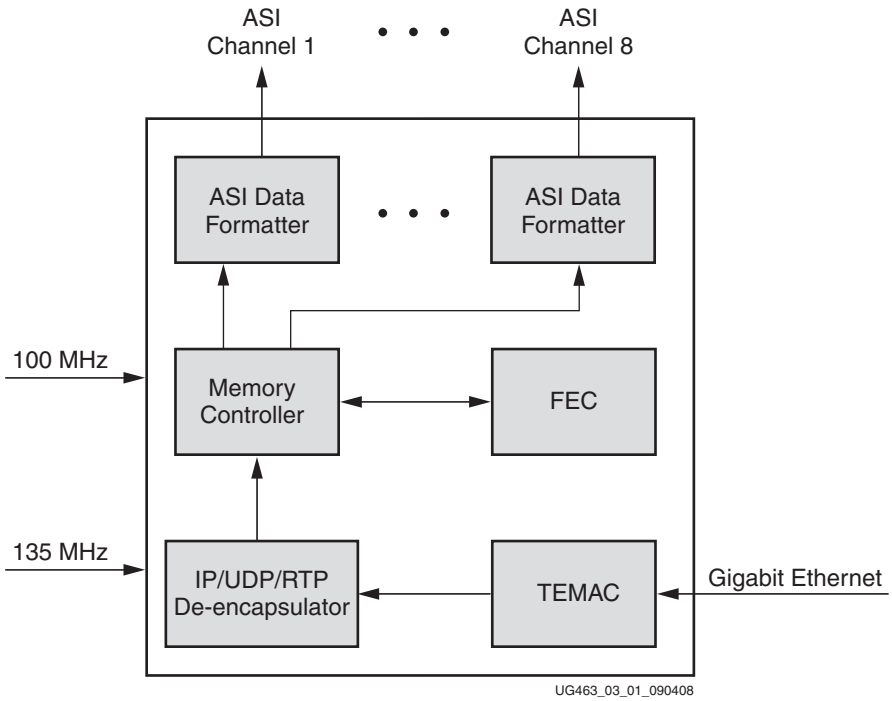


*Figure 3-1:* **IP to ASI Bridge (Receiver)**

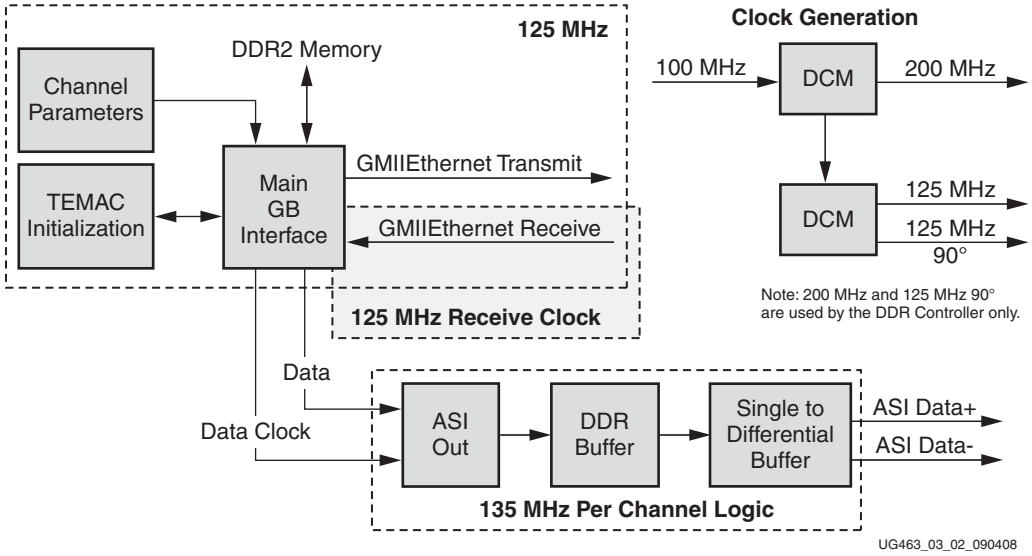


*Figure 3-2:* **rx_top Library**

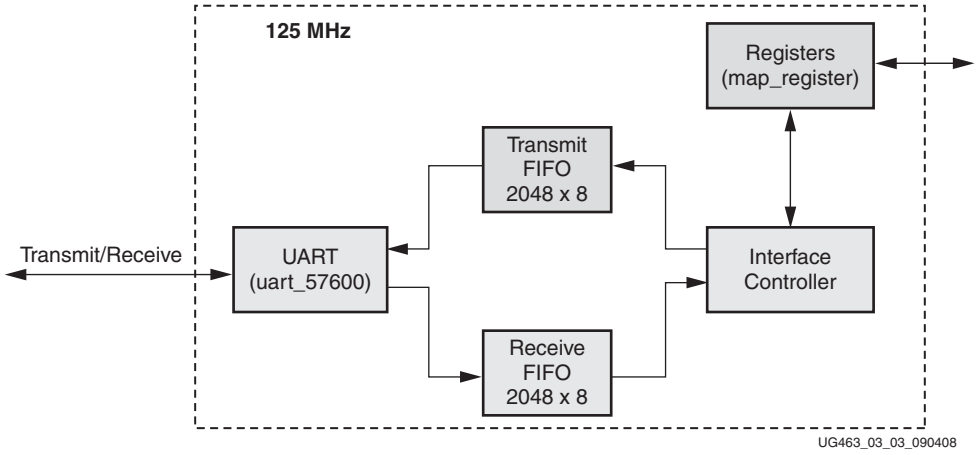


*Figure 3-3:* **rx_top Instantiations**

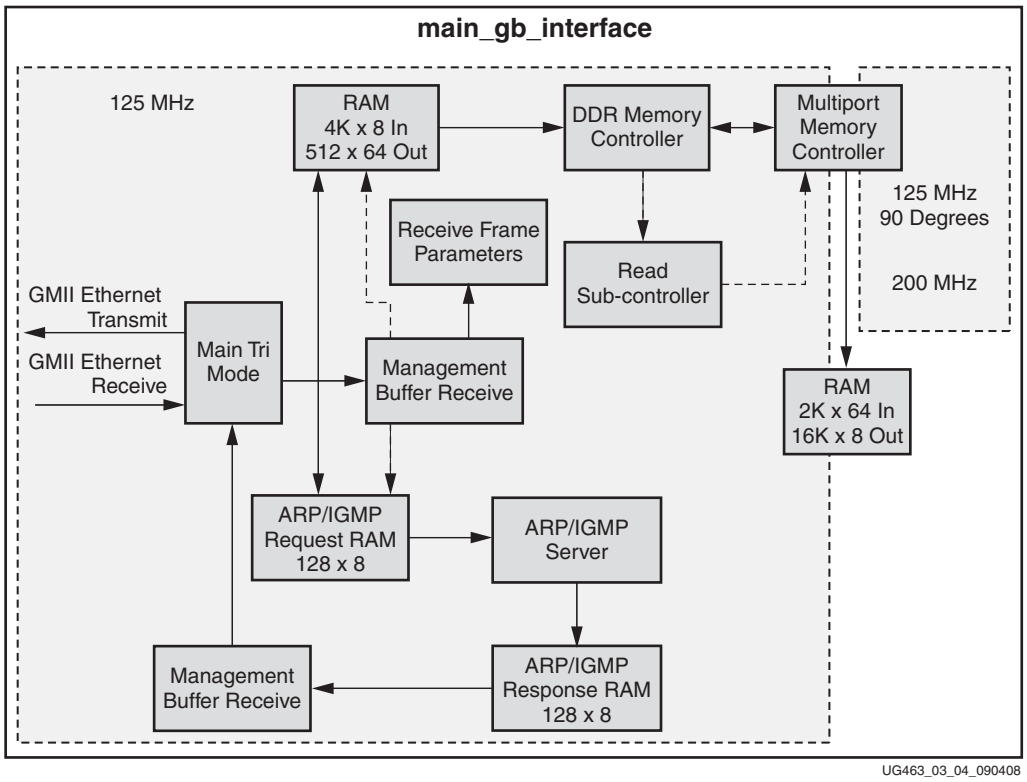


*Figure 3-4:* **rx_packetizer Library (part 1)**

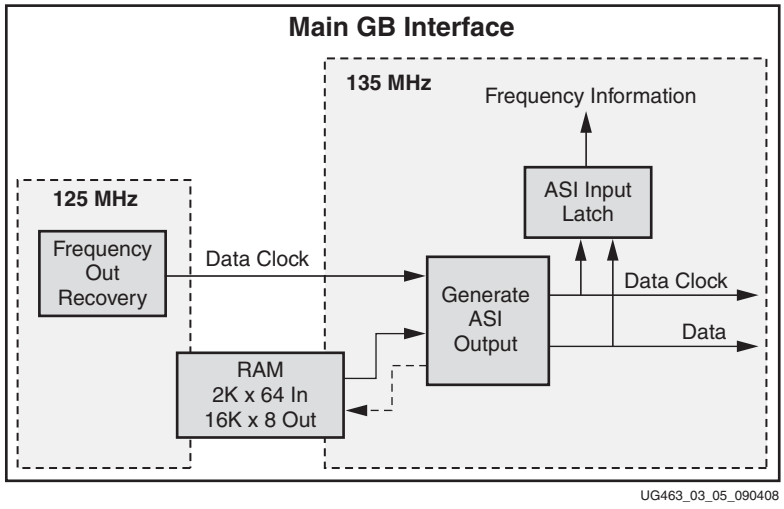


*Figure 3-5:* **rx_packetizer Library (part 2)**

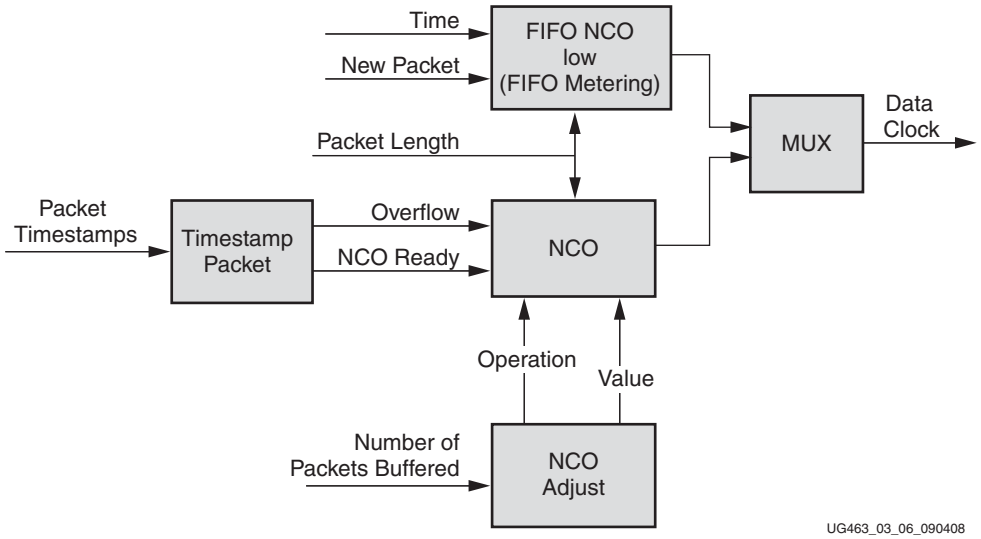


*Figure 3-6:* **rx_packetizer Library (part 3)**

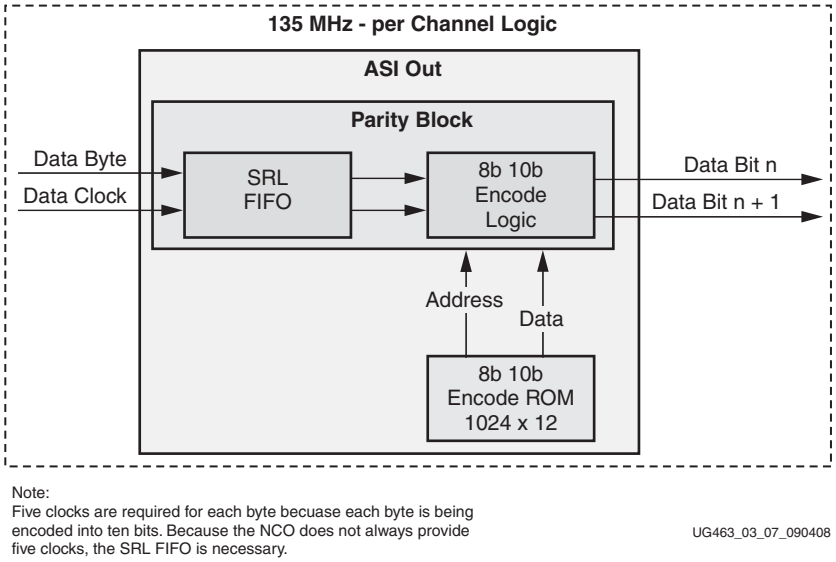


*Figure 3-7:* **rx_asi_out Library**

# Known Issues

For all known issues, please read the `readme.txt` file in the top level directory of the IP to ASI reference system.

# Limitations

The following items document known limitations of the IP to ASI reference system.

1. This reference system only supports 1 gigabit over Ethernet. It will not autonegotiate to 10/100 as the design needs the 125 MHz PHY clock of the 1 Gigabit rate. It would require some redesign to work around this. This has not been a major issue since a network switch can be used to connect to 10/100 Ethernet devices.
2. The Receiver does not work with fragmented IP packets such that any device sending the IP data over Ethernet must be able to send large (1500 bytes) Ethernet packets without fragmentation.
3. Because of test equipment limitations, out of order packets are not tested.
4. Non block aligned FEC is not supported.
5. To maintain the memory buffer at the appropriate level, the NCO is occasionally adjusted. The amount of jitter caused by that adjustment can be larger than 500 ns. Please see NCO adjustment for more details.

# FPGA Resources

The resources shown in Table 3-21 are for the Virtex-5 architecture.

*Table 3-21:* **Virtex-5 Architecture FPGA Resources**

| C_NUM_ASI_CHANNELS | C_INCLUDE_RX_FEC | LUTs | Flip Flops | BRAMs | DCMs | Hard TEMACs |
|---|---|---|---|---|---|---|
| 1 | 0 | 6618 | 6398 | 11 | 3 | 1 |
| 1 | 1 | 7503 | 7450 | 13 | 3 | 1 |
| 8 | 0 | 18007 | 14868 | 45 | 3 | 1 |
| 8 | 1 | 19817 | 17326 | 47 | 3 | 1 |

Chapter 4

# Video Over IP Testing

## Introduction

This chapter describes the testing that has been completed on the Video Over IP reference system.

## Test Equipment

The following test equipment is used during development of the Video Over IP reference system.

### Hardware

#### Tektronix MTS430 MPEG Test System

This test equipment allows the user to play transport streams over an ASI or IP network and also allows transport streams from an ASI or IP source to be analyzed for errors. It also has the ability to update the CC and PCR in the transport stream so that looping of data files works.

This device has some throughput limitations with play out of transport streams and analysis. It can play out and analyze at the same time up to at rates of about 100 Mbps.

#### Tektronix MTX100 MPEG Player

This test equipment allows the user to play transport streams over an ASI interface. It also has the ability to update the CC and PCR in the transport stream such that the looping of data files works.

#### Ineoquest Singulus Ethernet Analyzer

This test equipment is primarily used with the IP network for testing. It is the primary IP analysis tool used to verify that there are no CC errors in the IP transport stream and the bit rate of the transport stream is complete. This device also allows the user to drop packets from an IP link so that FEC can be tested.

#### Fast PC

A fast PC is necessary if the user wants to view higher bandwidth video (> 100 Mb) on a PC using VLC. A more advanced Network Interface Card (NIC) and an advanced video card can help to allow higher bit rates to be displayed more reliably.

## Software

### Video Lan Client (VLC)

This free software application is available via the Internet. It can receive video from an IP network and is capable of decoding and displaying the video. It also has the ability to stream out video from a file onto an IP network. This software provides a useful check, but is not considered to be the final test. Multiple instances of the software can execute simultaneously such that multiple channels of video can be viewed and streamed simultaneously or both.

### Wireshark

This free software application is an Ethernet analyzer that is available via the Internet. It allows the user to gather packets from an IP network for analysis. This is not as powerful as a hardware device, but very useful for quick analysis of packets on a network.

# Network Switches

Network switches can be a cause of lost packets such that they are not used during the testing of the Video Over IP reference system. They are used for convenience during demonstration of the system as the effects are generally not visible to the user.

# Transport Stream Files

There are a number of transport stream files that are publicly available, but it may be difficult to locate them for higher bit rates.

Access to certain transport stream files can be requested by visiting:

https://secure.xilinx.com/webreg/register.do?group=video_ip

Decision to grant access will be made on an individual basis.

# Testing

The Video Over IP reference system is tested with the goal of minimizing issues for customers who use it. Due to the large number of system variables, the testing performed does not guarantee the system to be free of defects.

## Primary Test Modes

With the large number of modes in the system, it is difficult to test all permutations. The following modes are the primary modes tested with other modes being secondary and having limited testing. Many of the primary test modes are the default modes for the system and are listed below.

204 byte transport stream packet testing was done with a 188 byte transport stream padded to 204 bytes. Due to test equipment limitations packet re-ordering has not been tested.

- 188 byte transport stream packets
- 1, 4, and 7 transport stream packets per IP packet
- RTP protocol
- a single ASI input source replicated for multiple channels
- ARP for MAC address resolution

- single channel test using channel 1 or multiple channel test using all 8 channels
- IGMP Version 2

## Test Success Criteria

The Ineoquest Ethernet/MPEG Analyzer is used for most testing to verify the integrity of the transport stream after going through the system. It has the ability to monitor multiple channels (UDP ports) simultaneously for Continuity Count (CC) errors. It also displays the bit rate for each channel. CC errors are the primary test success criteria together with a visual check of the bit rate.

VLC is used on a PC to verify a visual display for at least a single channel. The display is not monitored 100% of the time. Higher bit rates (> 100 Mb) are challenging for the PC to consistently display, but it is still useful data. The Transmitter in the test that is sending to the PC is typically setup so that only the single channel being displayed is sent to the PC IP address and the other channels are set for a different IP address such that the PC is not overwhelmed with network traffic. In the higher bit rate data files that are used for testing, there are multiple programs. The program that is the easiest to view and see any problems is typically viewed in VLC.

## Test Setup 1 - Basic Functionality and Data Throughput

The test setup shown in Figure 4-1 is used to test the basic functionality for verifying that data is passed through the system successfully and with multiple data rates and different numbers of channels.

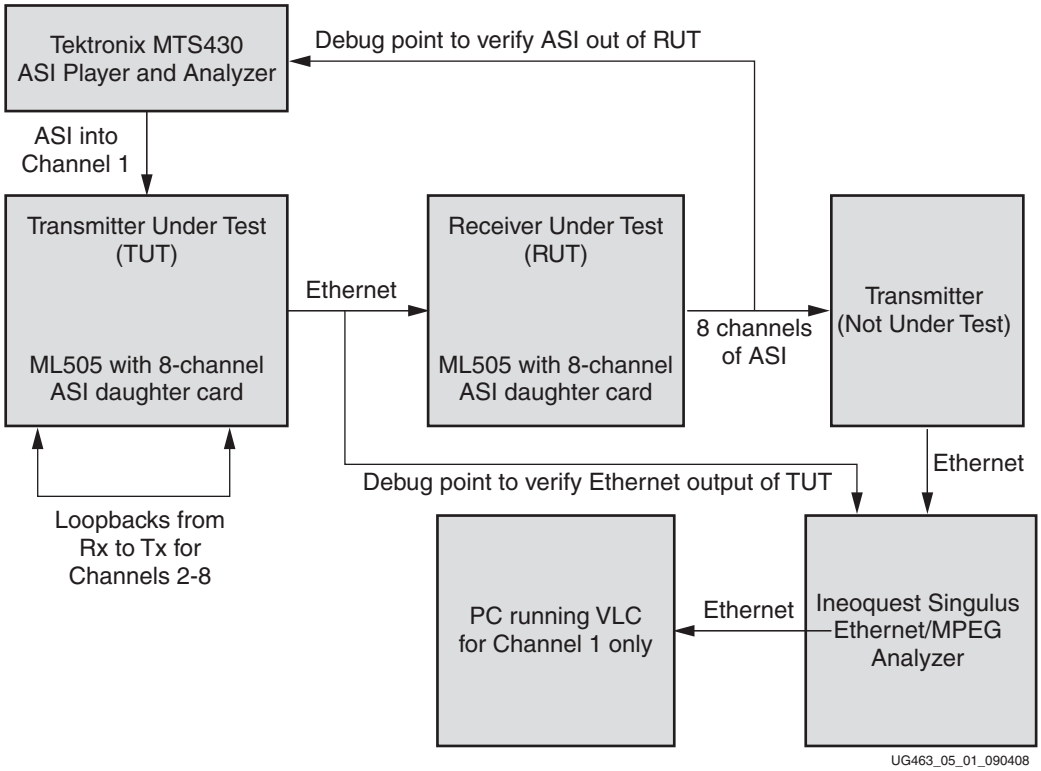


*Figure 4-1:* **Data Throughput Test Setup**

## Robustness Testing

The following testing was performed to verify the robustness of the Transmitter and Receiver as a system. Each test was run for approximately 12 hours to ensure that the system is robust.

### Single Channel Testing

A single channel was tested for approximately 12 hours with the following data rates: 2.5, 10, 75, 125, and 200 Mbps.

### Multiple Channel Testing

Multiple channels were tested for approximately 12 hours with the following data rates: 2.5, 10, 75, 100 and 200 Mbps. A single ASI input source was used with loopbacks on the Transmitter to replicate the input ASI data on the other 7 channels.

### Multiple Channel Multiple ASI Source Test

Two ASI input sources were used to verify that the independent rates on the channels function. One ASI input source was used for channels 1 - 4 with a data rate of 2.5 Mbps. Another ASI input source was used for channels 5 - 8 with a data rate of 50 Mbps.

## Jitter Testing

The Tektronix MTS430 is used to measure ASI jitter on the ASI output of the Receiver. It only handles a single channel at one time. Table 4-1 lists the jitter test results found for RTP mode of the Receiver.

*Table 4-1:* **RTP Mode Receiver Jitter**

| Data Rate (MB) | Negative Jitter | Positive Jitter | Notes |
|---|---|---|---|
| 2.5 | .2 us | .12 us | |
| 10 | .06 us | .08 us | |
| 100 | .1 us | .15 us | |
| 200 | .05 us | .06 us | |
| 200 | 1.06 us | .43 us | When the ASI output rate changes to maintain the buffer |

Chapter 5

# Demonstrating the Video Over IP System

## Introduction

The Video Over IP system is used for moving video and audio files, from an input source to a destination, using an IP protocol on Ethernet. IP protocol, together with UDP and RTP, provide *connectionless* packet delivery to a specified host address. The advantage of the IP protocol is that it delivers data independent of the underlying network technology. A transport stream consists of packets of 188 or 204 bytes. These packets are transported continuously and unidirectional without any back channel for feedback or control. When the transport stream is on its physical link, it is serialized and 8B/10B-coded on a 270 Mbps link called an Asynchronous Serial Interface (ASI). This link contains live transport stream data, but because the actual data link is running at much less than 270 Mbps, most of the physical layer contains null bytes. When a transport stream is carried over IP, the packets of the transport stream are put into UDP format together with an RTP header. From one to seven packets from the transport stream are placed into the payload.

This chapter may be used as a standalone document that provides instructions for demonstrating the Video Over IP System.

## System Features

- Up to 8 channels of full bandwidth ASI (< 1 Gigabit aggregate bandwidth)
- Forward Error Correction (FEC)
- Gigabit Ethernet with full bandwidth
- ARP protocol support
- IGMP version 2 support
- No processor is required or included
- External DDR memory required
- Serial port for control and diagnostics
- Virtex- 5 LXT Platform support on the ML505 platform

# Reference System Specifics

The reference system is made up of two reference systems as shown in Figure 5-1: the transmitter reference system and the receiver reference system.

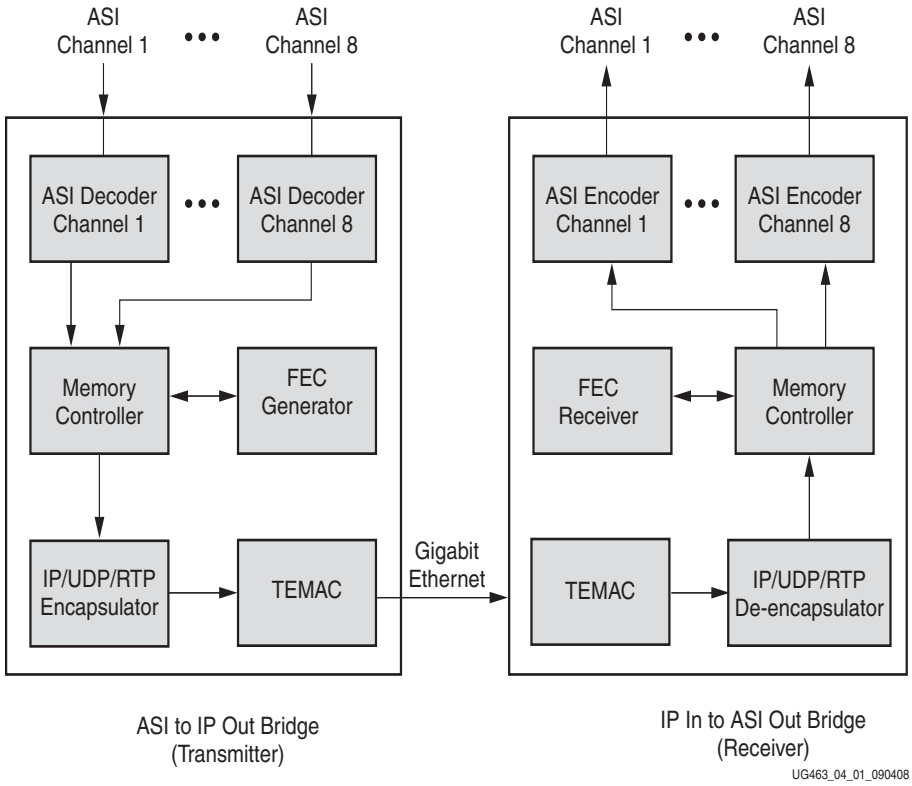


*Figure 5-1:* **Video Over IP Reference System**

# Demonstrating the Reference System

The equipment that is required to demonstrate the reference system is listed in Table 5-1.

*Table 5-1:* **Required Equipment and Software for Demonstrating**

| |
|---|
| Xilinx ML505 boards, qty 2 |
| Xilinx XGI 8Ch SDI video modules, qty 2 |
| BNC to BNC cable to connect the ASI output of the receiver ML505 to the ASI input of the transmitter ML505, qty 1 for single channel demo, qty 3 for FEC demo |
| SMA to SMA cables to connect the differential clock output of the daughter card to the ML505 differential clock input, qty 4 |
| Ethernet cables, qty 3 |
| Gigabit Ethernet switch, 3 ports are required, qty 1 |
| Video LAN Client (VLC) Software |

## Xilinx - XGI - 8 Ch SDI Video Module Setup

The Xilinx XGI 8 Ch SDI Video Module (daughter card) is also necessary for this system and is available from Cook Technologies at http://www.cook-tech.com/ct8asi.html. This card mates to the ML505 board and requires SMA cables to connect from the clock output SMA connectors on the module to the ML505 differential clock input SMA connectors. A BNC cable should connect the ASI output of the ASI Video Source to ASI input (RX1) on the daughter card.

## The Demonstrations

This chapter discusses two separate demonstrations. The first demonstration is a simple set up which demonstrates a single channel transmit/receive scenario only. The second demonstration shows how to use multiple channels and how to utilize FEC to recover dropped packets. Because the multi-channel FEC demonstration builds on the single channel demonstration, it is recommended that the single channel demonstration be set up first and use it as a starting platform for the more complex demonstration.

The Video Over IP reference system consists of two reference designs, a receiver which receives IP input and generates ASI output, and a transmitter which receives the ASI input and generates IP output. The purpose of these demonstrations is to show the Video Over IP reference system in a working capacity.

Because the receiver does not work with fragmented IP packets, any device sending the IP data over Ethernet must be able to send large (1500 bytes) Ethernet packets without fragmentation.

In the single channel demonstration, video will be streamed from a PC to the receiver over an ethernet connection. The receiver will use the IP input and generate an ASI output, which is connected to the ASI input of the transmitter. The transmitter will use the ASI input and generate IP data packets which are transmitted on the ethernet back to the PC.

In the FEC/multi-channel demonstration, video will be streamed from a PC to the receiver over an Ethernet connection. The receiver will use the IP input and generate an ASI output, which is connected to the first ASI input channel of the transmitter. The transmitter will use the ASI input and replicate the ASI stream (using loopback) on the second channel and then generate IP data and FEC packets which are transmitted over Ethernet to the receiver, on port 2000. The transmitter will drop packets prior to sending the data out to the receiver. The receiver will use the FEC data to recover the lost packets and generate an ASI stream, with the lost packets recovered, which is connected to the third ASI input of the transmitter. The transmitter will use the third ASI input and generate IP packets which are transmitted across Ethernet back to the PC, showing that all lost packets are recovered.

## Physical Connections

Make the physical connections as shown in Figure 5-2, and described in steps 1-8 below.

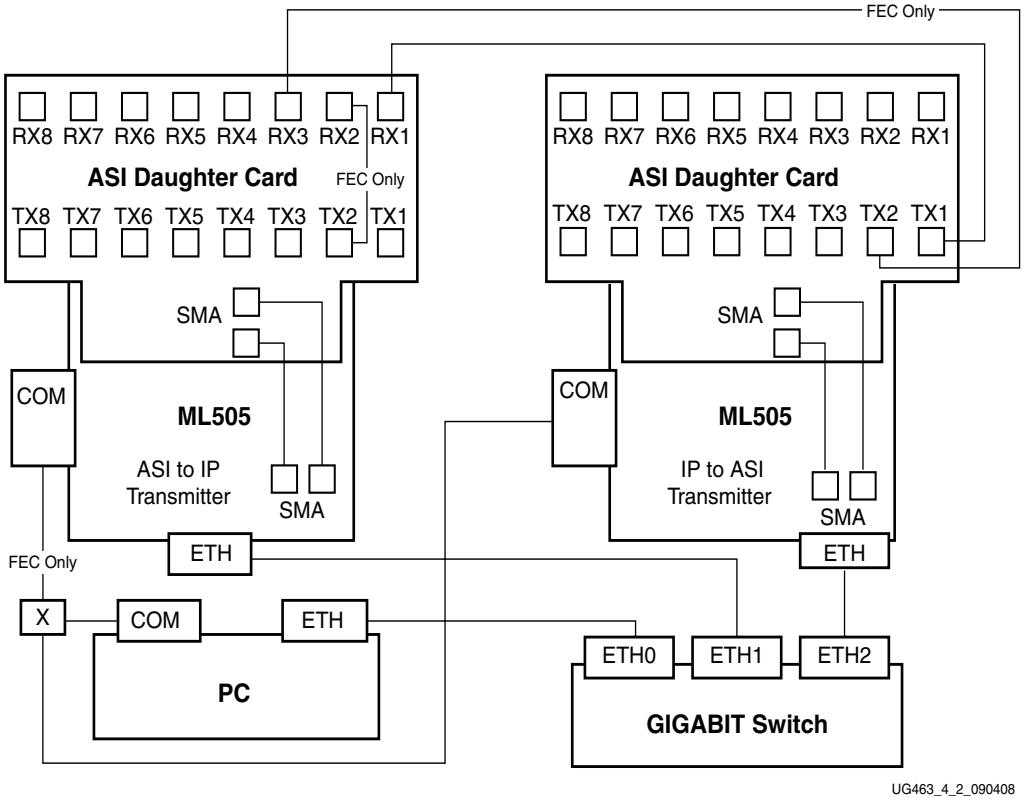


*Figure 5-2:* **Demonstration System Block Diagram**

1. Connect the PC and both the ML505 boards to the Gigabit Ethernet switch.
2. Connect the SMA cables from the daughter card differential clock outputs to the ML505 differential clock inputs. Because there are no polarity restrictions, either SMA connector on the daughter card can be connected to either SMA connector on the ML505.
3. Connect the BNC cable from the Tx1 BNC connector on the daughter card of the receiver ML505 board to the Rx1 BNC connector on the daughter card of the transmitter ML505 board.
4. Connect a RS-232 cable between the PC and the receiver ML505 board. Use **`57600`** for the baud rate, **`8`** bits for data, **`no`** parity, **`1`** stop bit and **`no`** flow control for the terminal settings. Figure 5-3 illustrates the terminal settings when using HyperTerminal.
5. For the FEC demo connect another short BNC cable from the Tx2 on the transmitter daughter card to the Rx2 connector on the same board. This replicates the ASI stream received on Rx1 to the Rx2 input on the transmitter.
6. For the FEC demo connect a third BNC cable from the Tx2 BNC connector on the daughter card of the receiver ML505 board to the Rx3 BNC connector on the daughter card of the transmitter ML505 board.

7. For the FEC demo the RS-232 cable must be connected between the PC and the transmitter ML505 board after enabling FIFO metering on the receiver. Use **57600** for the baud rate, **8** bits for data, **no** parity, **1** stop bit and **no** flow control for the terminal settings. Figure 5-3 illustrates the terminal settings when using HyperTerminal.

## Performing the Demonstration

A PC is required to send (to stream) video to the Video Over IP reference system and receive and display video from the reference system. Perform the following steps to set up the PC to send video to the system and to receive video from the system.

### Enable FIFO Metering on the Receiver

FIFO metering is used for demonstration purposes. Streaming video from a PC does not generate stable enough RTP time stamps for the NCO to reliably lock. FIFO metering does induce more jitter and is not recommended for use with a source that generates reliable RTP time stamps.

1. Using a terminal, such as HyperTerminal, set up the first channel on the receiver to use FIFO metering by typing **w0001** in the terminal. Typing **r00** in the terminal window will verify the write occurred by reading the data at address `0x00`. The value returned by the read should be `0x01`. Set the Port Settings values as follows: Bits per second: **57600**; Data bits: **8**; Parity: None; Stop bits: **1**; Flow Control: **None**, as shown in Figure 5-3.

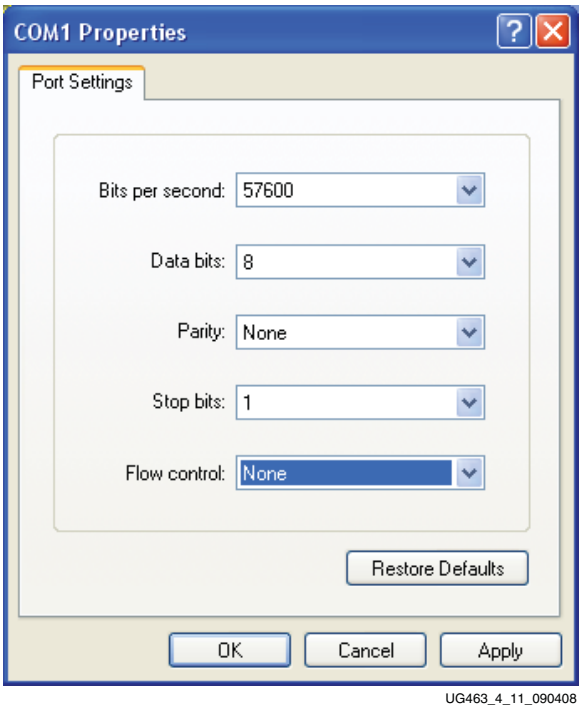


*Figure 5-3:* **HyperTerminal Settings**

## TCP/IP Address Setup

The reference system uses a static IP address, therefore the computer must be set up for a static IP address also. To set up the static IP address for the computer, use the following steps.

1. From the properties of the LAN Connection as shown in Figure 5-4, deselect all protocols except **Internet Protocol (TCP/IP)**, so that additional Ethernet data will not be sent to the ML505 board.
2. Click on **Properties**.

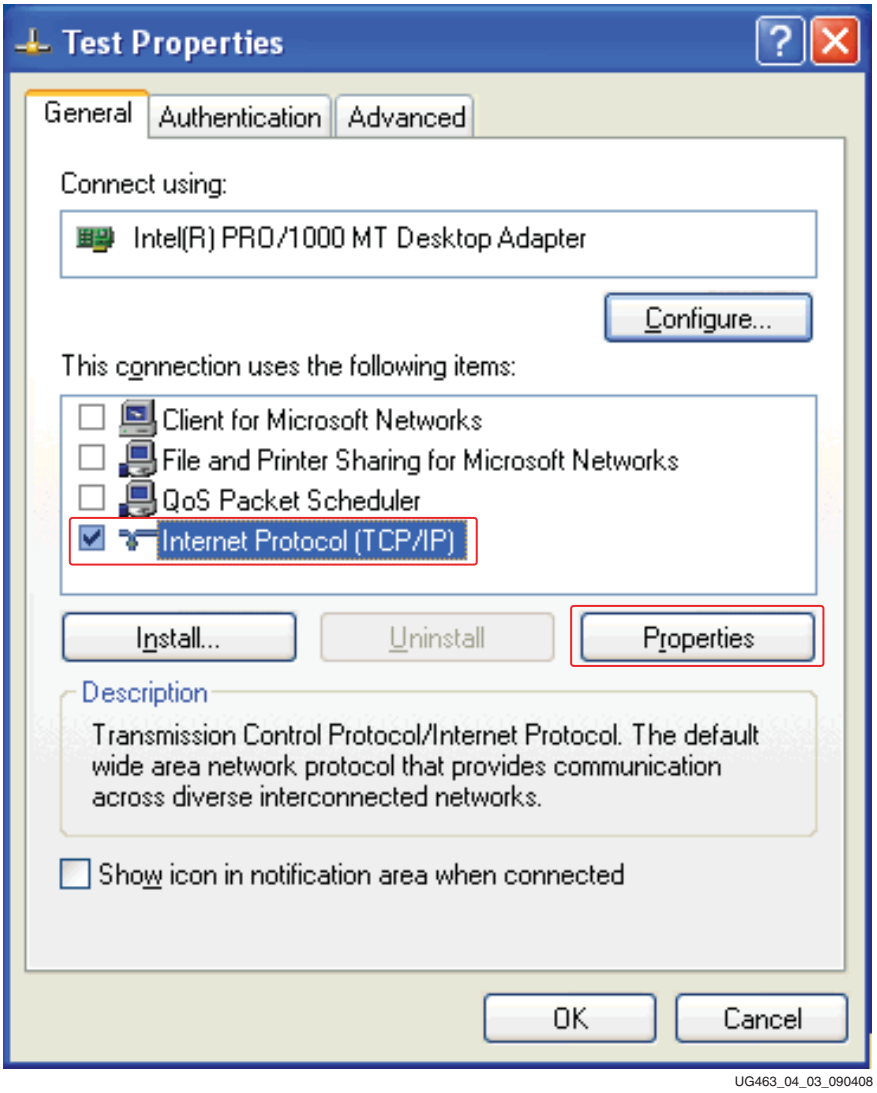


*Figure 5-4:* **LAN Connection Properties**

In the Internet Protocol (TCP/IP) Properties window, change the properties of the Internet Protocol as shown in Figure 5-5 to assign a static IP address to the PC which will be communicating with the ML505 embedded platforms.

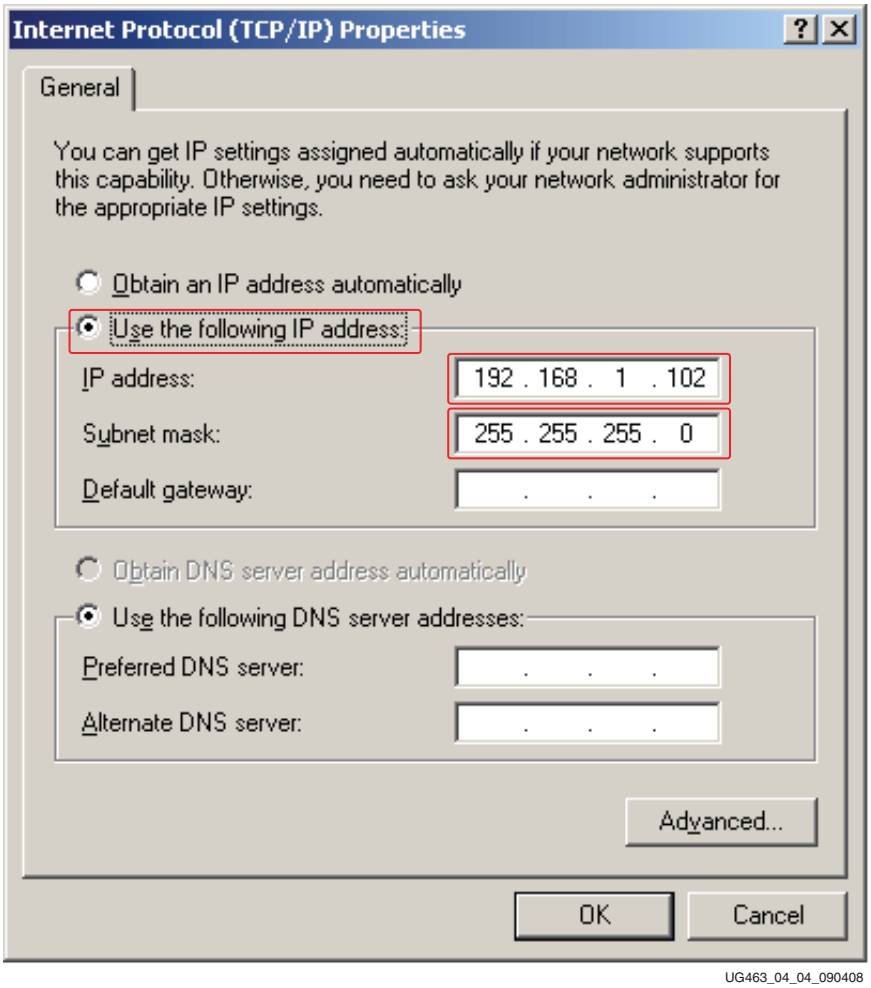


*Figure 5-5:* **TCP/IP Properties**

## Video Tool Setup

VLC is a multimedia player that is capable of sending video data to a network connection, accepting video data from a network connection, and displaying the video.

1. Download the VLC from http://www.videolan.org, then install it on the host computer.

   Two instances of VLC will be running on the PC for the single channel demo: one to send the video data to the receiver reference system, and another to receive and display the video data from the transmitter reference system. For the FEC demo a third instance of VLC will be running to view the recovered packet media stream.

   ***Note:*** This description for VLC is based on version 0.8. Later versions may have different menu selections.

2. Start VLC on the PC.

3. From the VLC media player window shown in Figure 5-6, use **File** → **Open Network Stream**.

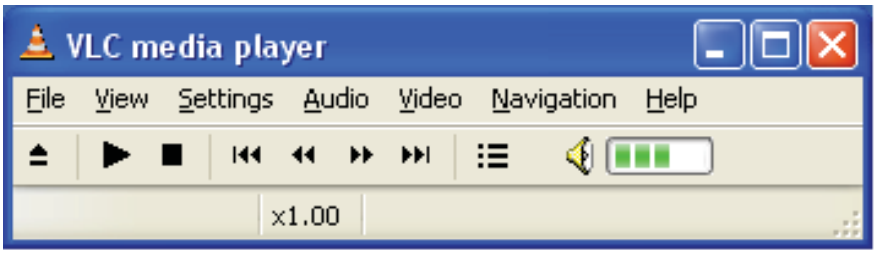


*Figure 5-6:* **VLC Started**

4. In the Open... window shown in Figure 5-7, from the Network tab, enter **1000** for the port of the UDP/RTP protocol, then click **OK**. VLC is now ready to accept video data and display it.

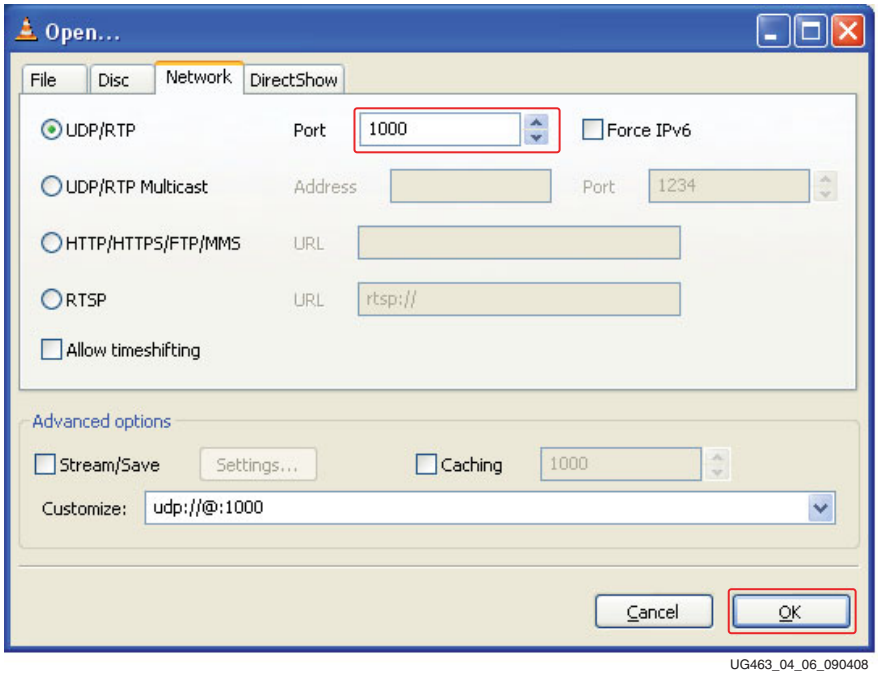


*Figure 5-7:* **Open Network Stream**

5. Start the second instance of VLC running. In the Open... window, select **File** → **Open Network Stream**.

6. In the Network tab, under the Advanced options, select the **Stream/Save** check box as shown in Figure 5-8, then click on **Settings**.

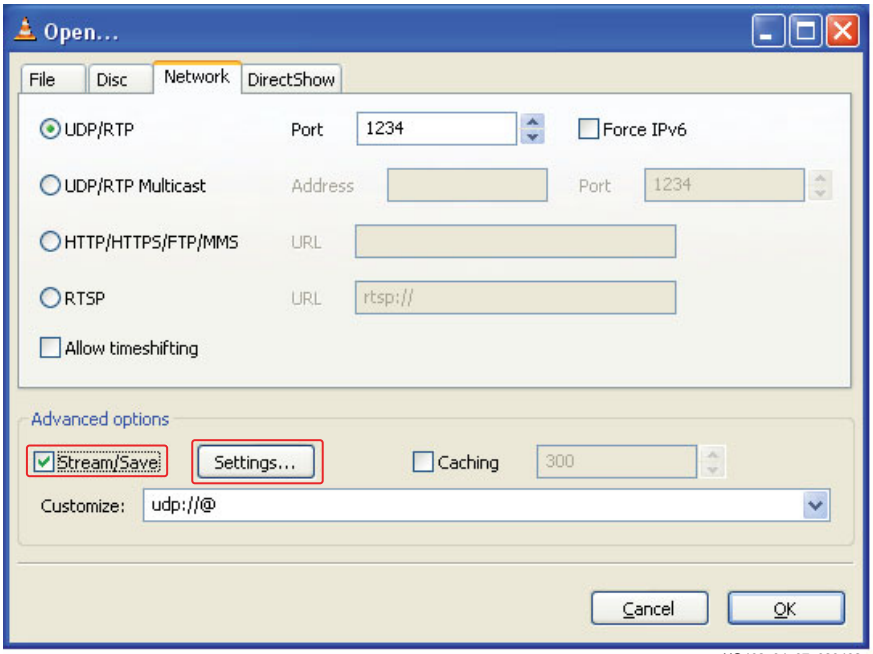


*Figure 5-8:* **Advanced Options**

7. In the Stream Output window, select the RTP checkbox, then enter the IP address and Port information as shown in Figure 5-9. Entering this information at this point causes the information in the Target data entry box to be generated so that the user does not need to enter it. Click **OK**.

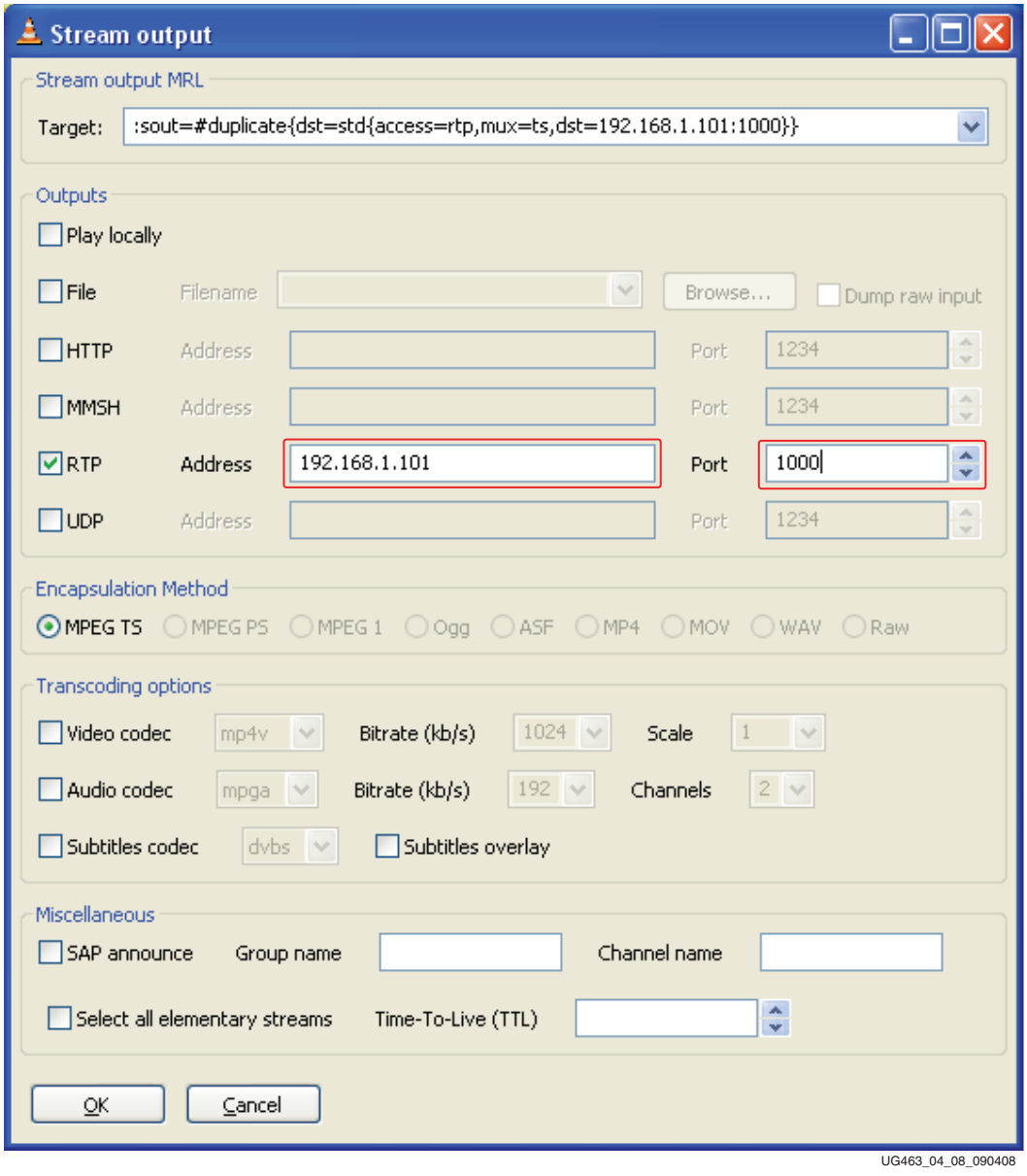


*Figure 5-9:* **Stream Output**

8. In the Open window shown in Figure 5-10, in the Network tab, confirm that **Stream/Save** is checked. Click on Browse, then browse to locate the video file to be sent through the system. Click **OK**.

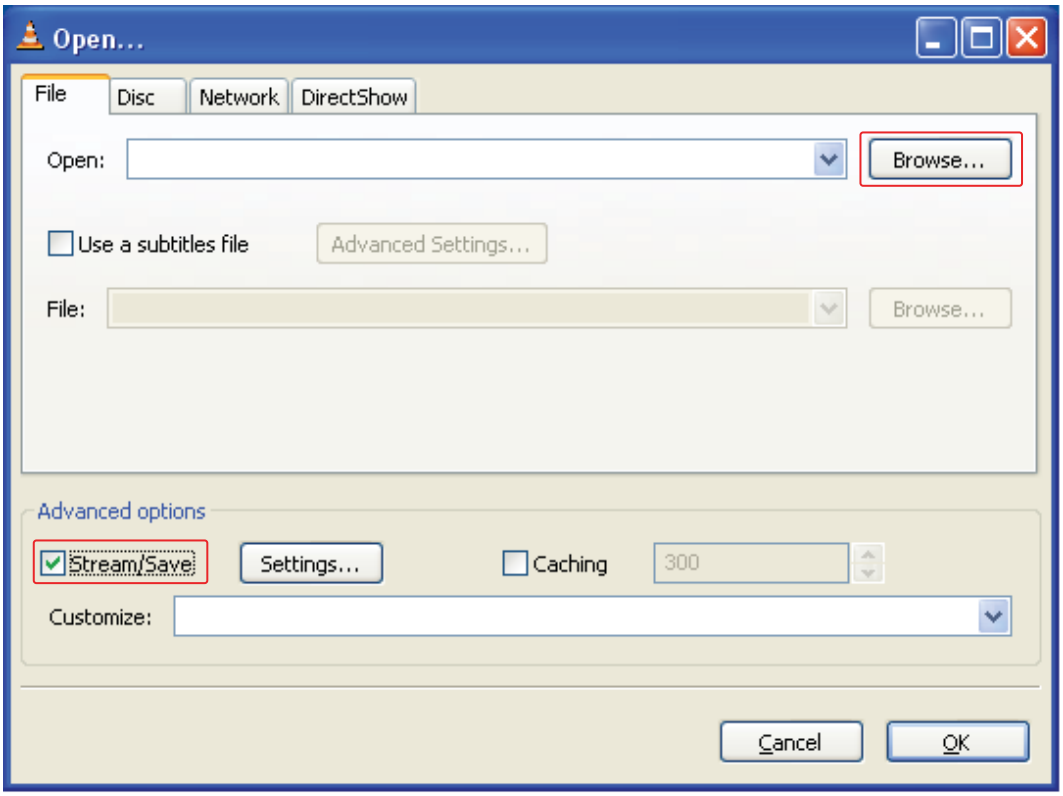


*Figure 5-10:* **File Open**

In the VLC media player window, the video file should start being sent as shown in Figure 5-11 and the received video should be displayed by the instance of VLC setup to receive video.

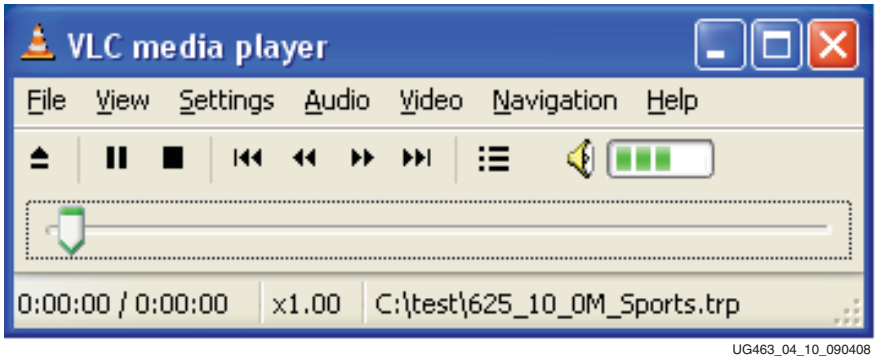


*Figure 5-11:* **Sending Video Data**

The following additional steps are required to demonstrate FEC.

9. Make sure the Chapter 5, "Physical Connections" described in steps 4-6 have been performed. Switch the RS-232 cable connection from the receiver board to the transmitter board.
10. Using a terminal, set up transmitter port 2000 to send data to the receiver IP address by typing **`w2C65`** in the terminal.
11. Launch a third instance of VLC and set it to receive data on port 3000. Video received on port 1000 and port 3000 should now be playing on VLC.
12. To drop packets the following commands must be issued on the terminal to the transmitter board.

    **`w3764`** – This sets up the transmitter packet drop interval to 100 on port 2000. One packet dropped, then 100 packets transmitted, then one packet dropped...

    **`w3501`** – This enables the packet drop function on port 2000. The stream displayed by VLC on port 3000 shows the stream after the missing packets have all been recovered using FEC.

    **`w20C7`** – This disables FEC on transmitter port 2000. The stream displayed by VLC on port 3000 should now be corrupted.

    **`w20DF`** – This enables the FEC out of the transmitter on port 2000. The stream displayed on port 3000 should be recovering packets.

This reference system provides a working demonstration which bridges between ASI interfaces containing MPEG-2 transport streams and Ethernet devices using the UDP/IP/RTP protocol. Contact a Xilinx FAE for support with this system.

# Debugging The System

There are a number of issues that can arise when using the system. The purpose of this section is to help determine why the system is not functioning. The items listed are based on experience derived from demonstrating the system.

## Ethernet NIC Issues

Video data being sent over an IP network and into the receiver should not be fragmented.

Sometimes an Ethernet NIC can cause fragmentation of the IP packets when it is streaming video from the PC. This is difficult to detect and usually requires a software Ethernet analyzer such as Wireshark to gather packets from the interface and to verify they are not fragmented. Registry changes to the PC may be necessary to increase the size of the network MTU when this issue occurs.

The following instructions can help set the MTU size in Windows. Be aware that this is modifying the registry which can be dangerous.

1. Click **Start**, then click on **Control Panel**.
2. Double-click **Network** and **Internet Connections**, and then click to open the Network Connections folder.
3. Do the following if more than one network connection is listed. For each connection, double-click the connection, then click the Support tab of the Status interface that opens. The connection that shows a Default Gateway entry is probably the network connection that is used to connect to the Internet. Note the name of the connection (for example, "Local Area Connection 2").
4. Start Registry Editor by using the command: **`Regedit.exe`**.
5. Under the HKEY_LOCAL_MACHINE tree, go to the following key:

SYSTEM\CurrentControlSet\Control\Network\{4D36E972-E325-11CE-BFC1-08002BE10318}\

6. Under that key are one or more keys that have numeric identifiers. Each of these keys has a connection subkey. Examine each key that look like:

   ID_for_Adapter\Connection

   The Name value in the Connection subkey provides the network connection name that is used in the Network Connections folder. When the network connection is found that matches the name that was found in step 3, note the ID_for_Adapter under which the network connection name is.

7. Return to HKEY_LOCAL_MACHINE, then locate the following key:

   SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\ID_for_Adapter, where ID_for_Adapter is the number that was noted in step 6. When this key is highlighted, several values appear on the right side of the screen, including DefaultGateway and EnableDHCP.

8. Right-click the right side of the screen, click **New**, then click **DWORD Value**. Name the value **MTU**.
9. Double-click the value so that it can be edited. Change **Base** to **Decimal**, then enter the largest acceptable MTU size, which is the size that was identified by using the Ping tests.
10. Quit Registry Editor.

## Defective SMA Cables

The transmitter does not require the SMA cables if the loopback of channels is not being used. The receiver requires the SMA cables. Loopback is used for the FEC/multi-channel demo.

The SMA cables that connect the daughter card to the main board are susceptible to breakage. The inside pin on the cable may be missing or pushed into the cable housing.

To determine if the SMA cable is defective, observe the LEDs on the ASI daughter board near the RX BNC connectors when the receiver is connected to the transmitter. If the LEDs are not lighted or are flickering, this may indicate that the receiver does not have a good clock source from the ASI daughter card. Replace the SMA cable.

## Receiver board LED indicators

The receiver board uses the GPIO LEDs 1 though 8 to indicate IP data is being received on ports 1000 - 8000. The receiver board uses the LEDs on the ASI daughter board near the Tx BNC to indicate that the NCO is ready on that channel and the receiver is transmitting ASI data.

## Streaming Video Quality Issues

VLC is not a robust streaming source and some video degradation may result that is visible. This does not indicate a problem with the reference system but rather with the quality of the stream produced by the video streamer. Also while displaying the corrupted stream VLC is likely to crash, the longer the corrupted stream is displayed the more likely VLC will crash.