HDMI 1.4/2.0 Receiver Subsystem v1.0

LogiCORE IP Product Guide

Vivado Design Suite

PG236 (v1.0) November 18, 2015

XILINX
ALL PROGRAMMABLE™

Table of Contents

IP Facts

Chapter 1: Overview

Applications 5
Unsupported Features 5
Licensing and Ordering Information 6

Chapter 2: Product Specification

Standards 8
Performance 8
Resource Utilization 9
Port Descriptions 9
Clocks and Resets 14

Chapter 3: Designing with the Subsystem

General Design Guidelines 15
Clocking 21
Resets 22

Chapter 4: Design Flow Steps

Customizing and Generating the Subsystem 23
Constraining the Subsystem 26
Simulation 27
Synthesis and Implementation 27

Appendix A: Verification, Compliance, and Interoperability

Interoperability 28
Hardware Testing 28
Video Resolutions 29

Appendix B: Debugging

Finding Help on Xilinx.com 31
Debug Tools 32
Hardware Debug 33

Interface Debug 34

Appendix C: Application Software Development

Device Drivers 35

Appendix D: Additional Resources and Legal Notices

Xilinx Resources 37
References 37
Revision History 38
Please Read: Important Legal Notices 38

Introduction

The HDMI 1.4/2.0 Receiver Subsystem is a hierarchical IP that bundles a collection of HDMI RX IP sub-cores and outputs them as a single IP. It is an out-of-the-box ready-to-use HDMI 1.4/2.0 Receiver Subsystem and avoids the need to assemble sub-cores to create a working HDMI RX system.

Features

- HDMI 2.0 and 1.4a compatible
- 1, 2, or 4 symbol/pixel per clock input
- Supports resolutions up to 3840 x 2160 @ 60 fps
- 8, 10, 12 & 16 bit Deep-color support
- Support color space for RGB, YUV 4:4:4, YUV 4:2:2, YUV 4:2:0
- Audio support for up to 8 channels
- Info frames
- Data Display Channel (DDC)
- Hot-Plug Detection

LogiCORE™ IP Facts Table	
Subsystem Specifics	
Supported Device Family(1)	UltraScale™ Architecture Zynq-7000 All Programmable SoC 7 Series
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	Performance and Resource Utilization web page
Provided with Subsystem	
Design Files	RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	XDC
Simulation Model	Not Provided
Supported S/W Driver(2)	Standalone
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. Standalone driver details can be found in the SDK directory (*<install_directory>*/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from the Xilinx Wiki page.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The HDMI 1.4/2.0 Receiver Subsystem is a feature-rich soft IP incorporating all the necessary logic to properly interface with PHY layers and provide HDMI decoding functionality. The subsystem is a hierarchical IP that bundles a collection of HDMI RX-related IP sub-cores and outputs them as a single IP. The subsystem receives the captured native video from the video PHY layer. It then extracts the video and audio streams from the HDMI stream and converts it to AXI video and audio streams.

The subsystem can be configured at design time through a single interface in the Vivado® Integrated Design Environment (IDE) for performance and quality.

Applications

HDMI is a common interface used to transport video and audio and is seen in almost all consumer video equipment such as DVD and media players, digital televisions, camcorders, mobile tablets and phones. The omnipresence of the interface has also spread to most professional equipment such as professional cameras, video switchers, converters, monitors and large displays used in video walls and public display signs.

For tested video resolutions for the subsystem see Appendix A, Verification, Compliance, and Interoperability.

Unsupported Features

The following features are not supported in this subsystem:

- Lip sync
- CEC
- HEAC
- HDMI 2.0 dual view
- HDMI 2.0 multi stream audio
- HDCP
- 3D

Licensing and Ordering Information

License Checkers

If the IP requires a license key, the key must be verified. The Vivado® design tools have several license checkpoints for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with error. License checkpoints are enforced by the following tools:

- Vivado synthesis
- Vivado implementation
- write_bitstream (Tcl command)

IMPORTANT: *IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.*

License Type

This Xilinx LogiCORE™ IP module is provided under the terms of the Xilinx Core License Agreement. The module is shipped as part of the Vivado® Design Suite. For full access to all subsystem functionalities in simulation and in hardware, you must purchase a license for the subsystem. Contact your local Xilinx sales representative for information about pricing and availability.

For more information, visit the Xilinx HDMI web page.

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

Product Specification

This chapter includes a description of the subsystem and details about the performance and resource utilization.

A high-level block diagram of the HDMI 1.4/2.0 Receiver Subsystem is shown in Figure 2-1.

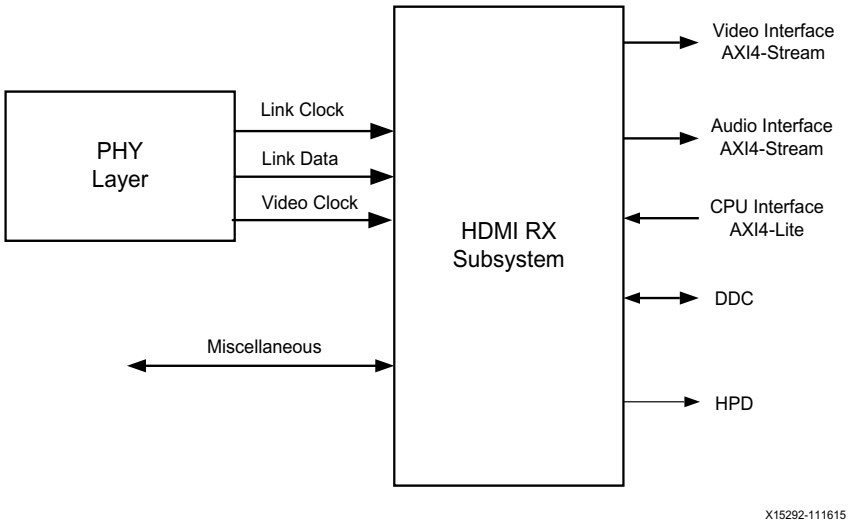

Figure 2-1: **Subsystem Block Diagram**

The subsystem has three AXI interfaces, the video interface, the audio interface and the CPU interface.

The video interface is an AXI4-Stream master bus. The subsystem converts the captured native video data to AXI streaming video and outputs the video data on this interface.

The audio interface is a 32-bit AXI4-Stream master bus. The subsystem converts the captured audio to a multiple channel AXI audio stream and outputs the audio data on this interface.

The CPU interface is an AXI4-Lite bus interface, which allows a processor, such as MicroBlaze, to control the subsystem by accessing its registers. This AXI4-Lite slave interface supports single beat read and write data transfers (no burst transfers).

The subsystem also supports the features described in the following sections.

Audio Clock Regeneration Signals

The subsystem can output Audio Clock Regeneration (ACR) signals that allow receiver audio peripherals to regenerate the audio clock.

The audio clock regeneration architecture is not part of the HDMI RX subsystem. You must provide an audio clock to the application. This can be achieved by using an internal PLL or external clock source, depending on the audio clock requirements, audio sample frequency and jitter.

Display Data Channel (DDC)

The subsystem allows the end-user to build an HDMI sink device, which negotiates with the targeted HDMI source device for supported features and capabilities. The communication between the source device(s) and the sink device is implemented through the DDC lines, which is an I2C bus included on the HDMI cable.

Hot Plug Detect

The subsystem supports the Hot Plug Detect (HPD) feature, which is a communication mechanism between HDMI source and HDMI sink devices. For example, when an HDMI cable is inserted between the HDMI source and HDMI sink devices, the cable-detect signal is asserted. The subsystem then outputs a `hpd` signal, which triggers the start of a communication between the source device and sink device.

Standards

The HDMI 1.4/2.0 Receiver Subsystem is compliant with the AXI4-Stream Video Protocol and AXI4-Lite interconnect standards. See the *Vivado AXI Reference Guide* (UG1037) [Ref 1] for additional information.

Performance

For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.

Maximum Frequencies

See the *Kintex/Virtex Ultrascale Architecture Data sheets* [Ref 2], [Ref 3] and *7 Series Architecture Data sheet: DC and AC Switching Characteristics* [Ref 4], [Ref 5], [Ref 6]. The frequency ranges specified in these documents must be adhered to for proper transceiver and subsystem operation.

Latency

Video Latency

The video latency is measured from the first active video data on the link (LINK_DATA*_IN) to the start of the active video data on the output (VIDEO_OUT). For example, the receiver video path has a latency of 3.85 µs when measuring a 1080p video signal with a 148.5 MHz pixel clock.

Audio Latency

The audio latency is measured from the first active audio data on the link (LINK_DATA*_IN) to the start of the active audio data on the output (AUDIO_OUT). For example, the receiver audio path has a latency of 2.91 µs when measuring a 1080p video signal with a 148.5 MHz pixel clock.

Resource Utilization

For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.

Port Descriptions

Figure 2-2 shows the HDMI 1.4/2.0 Receiver Subsystem ports. The subsystem has three default AXI interfaces:

- AXI-Lite control interface (S_AXI_CPU_IN)
- AXI4-Stream streaming video output (VIDEO_OUT)
- AXI4-Stream streaming audio output (AUDIO_OUT)

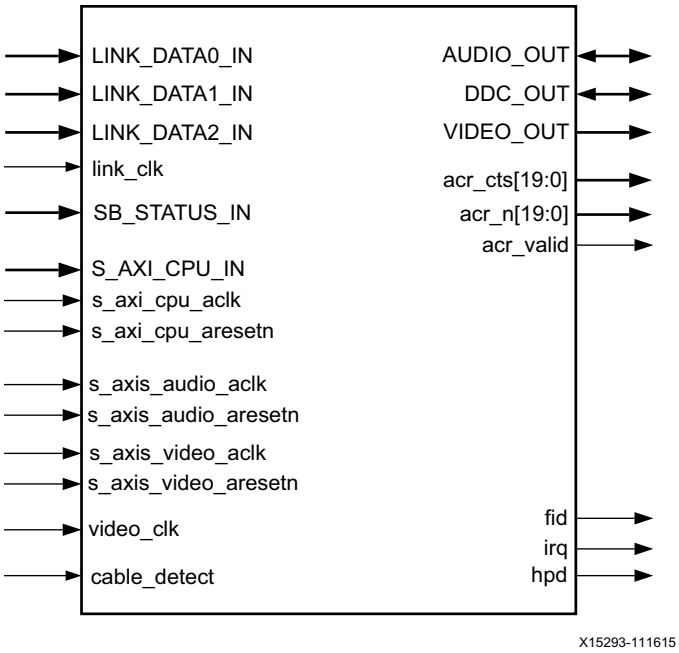

Figure 2-2: **Subsystem Pinout**

CPU Interface

Table 2-1 shows the AXI4-Lite control interface signals. This interface is an AXI4-Lite interface and runs at the `s_axi_cpu_aclk` clock rate. Control of the subsystem is only supported through the subsystem driver.

Table 2-1: **CPU Interface Ports**

Name	Direction	Width	Description
s_axi_cpu_aresetn	Input	1	Reset (Active-Low)
s_axi_cpu_aclk	Input	1	Clock for AXI4-Lite control interface
S_AXI_CPU_IN_awaddr	Input	19	Write address
S_AXI_CPU_IN_awprot	Input	3	Write address protection
S_AXI_CPU_IN_awvalid	Input	1	Write address valid
S_AXI_CPU_IN_awready	Output	1	Write address ready
S_AXI_CPU_IN_wdata	Input	32	Write data
S_AXI_CPU_IN_wstrb	Input	4	Write data strobe
S_AXI_CPU_IN_wvalid	Input	1	Write data valid
S_AXI_CPU_IN_wready	Output	1	Write data ready
S_AXI_CPU_IN_bresp	Output	2	Write response
S_AXI_CPU_IN_bvalid	Output	1	Write response valid
S_AXI_CPU_IN_bready	Input	1	Write response ready
S_AXI_CPU_IN_araddr	Input	19	Read address

Table 2-1: **CPU Interface Ports *(Cont'd)***

Name	Direction	Width	Description
S_AXI_CPU_IN_arprot	Input	3	Read address protection
S_AXI_CPU_IN_arvalid	Input	1	Read address valid
S_AXI_CPU_IN_aready	Output	1	Read address ready
S_AXI_CPU_IN_rdata	Output	32	Read data
S_AXI_CPU_IN_rresp	Output	2	Read data response
S_AXI_CPU_IN_rvalid	Output	1	Read data valid
S_AXI_CPU_IN_rready	Input	1	Read data ready

Video Output Stream Interface

Table 2-2 shows the signals for AXI4-Stream video output streaming interface. This interface is an AXI4-Stream master interface and runs at the `s_axis_video_aclk` clock rate. The data width is user-configurable in the Vivado IDE by setting **Max bits per component** (BPC) and **Number of pixels per clock on AXI4-S output** (PPC).

Table 2-2: **Video Output Stream Interface**

Name	Direction	Width	Description
s_axis_video_aclk	Input	1	AXI4-Stream clock
s_axis_video_aresetn	Input	1	Reset (Active-Low)
VIDEO_OUT_tdata	Output	3*BPC*PPC	Data
VIDEO_OUT_tlast	Output	1	End of line
VIDEO_OUT_tready	Input	1	Ready
VIDEO_OUT_tuser	Output	1	Start of frame
VIDEO_OUT_tvalid	Output	1	Valid

Audio Output Stream Interface

Table 2-3 shows the signals for AXI4-Stream audio output streaming interfaces. The audio interface transports 24-bits audio samples in the IEC 60958 format. A maximum of eight channels are supported. The audio interface is a 32-bit AXI4-Stream master interface and runs at the `s_axis_audio_aclk` clock rate.

Table 2-3: **Audio Output Stream Interface**

Name	Direction	Width	Description
s_axis_audio_aclk	Input	1	Clock (The audio streaming clock must be greater than or equal or greater than 128 times the audio sample frequency)
s_axis_audio_aresetn	Input	1	Reset (Active-Low)

Table 2-3: **Audio Output Stream Interface *(Cont'd)***

Name	Direction	Width	Description
AUDIO_OUT_tdata	Output	32	Data [31] P (Parity) [30] C (Channel status) [29] U (User bit) [28] V (Validity bit) [27:4] Audio sample word [3:0] Preamble code 4'b0001 Subframe 1/start of audio block 4'b0010 Subframe 1 4'b0011 Subframe 2
AUDIO_OUT_tid	Output	3	Channel ID
AUDIO_OUT_tready	Input	1	Ready
AUDIO_OUT_tvalid	Output	1	Valid

Audio Clock Regeneration Interface

The audio clock regeneration (ACR) interface has a Cycle Time Stamp (CTS) parameter vector and an Audio Clock Regeneration Value (N) parameter vector. Both vectors are 20 bits wide. The valid signal is driven High when the CTS and N parameters are stable. For more information, see Chapter 7 of the HDMI 1.4 specification [Ref 7].

On the rising edge of the valid signal, the RX reads the CTS and N parameters from the ACR input interface and transmits an audio clock regeneration packet. Therefore, the subsystem should set up the CTS and N parameters before asserting the valid signal.

Table 2-4 shows the Audio Clock Regeneration (ACR) interface signals. This interface runs at the `s_axis_audio_aclk` clock rate.

Table 2-4: **Audio Clock Regeneration (ACR) Interface**

Name	Direction	Width	Description
acr_cts	Output	20	CTS
acr_n	Output	20	N
acr_valid	Output	1	Valid

HDMI Link Input Interface

Table 2-5 shows the HDMI Link Input interface signals. This interface runs at the `link_clk` clock rate.

Table 2-5: **HDMI Link Input Interface**

Name	Direction	Width	Description
link_clk	Input	1	Link clock
LINK_DATA0_IN_tdata	Input	40	Link data 0
LINK_DATA0_IN_tvalid	Input	1	Link Data 0 Valid
LINK_DATA1_IN_tdata	Input	40	Link data 1
LINK_DATA1_IN_tvalid	Input	1	Link Data 1 Valid
LINK_DATA2_IN_tdata	Input	40	Link data 2
LINK_DATA2_IN_tvalid	Input	1	Link Data 2 Valid

Data Display Channel Interface

Table 2-6 shows the Data Display Channel interface signals.

Table 2-6: **Data Display Channel (DDC) Interface**

Name	Direction	Width	Description
ddc_scl_i	Input	1	DDC serial clock in
ddc_scl_o	Output	1	DDC serial clock out
ddc_scl_t	Output	1	DDC serial clock tri-state
ddc_sda_i	Input	1	DDC serial data in
ddc_sda_o	Output	1	DDC serial data out
ddc_sda_t	Output	1	DDC serial data tri-state

Miscellaneous Signals

Table 2-7 shows the miscellaneous signals.

Table 2-7: **Miscellaneous Signals**

Name	Direction	Width	Description
cable_detect	Input	1	This input is used by the subSystem to detect if a HDMI source is connected. When it is not used it can be hardwired to one. 1 - cable connected 0 - cable disconnected
hpd	Output	1	Hot plug detect
irq	Output	1	Interrupt request for CPU. Active-High.

Table 2-7: **Miscellaneous Signals *(Cont'd)***

Name	Direction	Width	Description
video_clk	Input	1	Reference Native Video clock
SB_STATUS_IN_tdata	Input	2	Side Band Status input signals Bit 0: link_rdy Bit 1: video_rdy
SB_STATUS_IN_tvalid	Input	1	Side Band Status input valid
fid	Output	1	Field ID for AXI4-Stream bus. Used only for interlaced video. 0 - even field 1 - odd field For progress video the output is always Low.

Clocks and Resets

Table 2-8 provides an overview of the clocks and resets. See Clocking for more information.

Table 2-8: **Clocks and Resets**

Name	Direction	Width	Description
s_axi_cpu_aclk	Input	1	AXI4-Lite CPU control interface clock.
s_axi_cpu_aresetn	Input	1	Reset, associated with s_axi_cpu_aclk (active-Low). The s_axi_cpu_aresetn signal resets the entire subsystem including the data path and AXI4-Lite registers.
s_axis_video_aclk	Input	1	AXI4-Stream video output clock.
s_axis_video_aresetn	Input	1	Reset, associated with s_axis_video_aclk (active-Low). Resets the AXI4-Stream data path for the video output.
s_axis_audio_aclk	Input	1	AXI4-Stream Audio output clock. (The audio streaming clock must be greater than or equal to 128 times the audio sample frequency)
s_axis_audio_aresetn	Input	1	Reset, associated with s_axis_audio_aclk (active-Low). Resets the AXI4-Stream data path for the audio output.
link_clk	Input	1	HDMI Link data output clock.
video_clk	Input	1	Clock for the native video interface.

Designing with the Subsystem

This chapter includes guidelines and additional information to facilitate designing with the subsystem.

General Design Guidelines

The subsystem connects to other hardware components to construct a complete HDMI RX system. These hardware components usually are different from device to device. For example, Kintex®-7 devices have a different PLL architecture from UltraScale™ devices. Therefore, you need to fully understand the system and adjust the subsystem parameters accordingly. Appendix C, Application Software Development describes how to integrate the subsystem API into a software application.

Audio Data Stream

An AXI4-Stream audio cycle is illustrated in Figure 3-1. The data is valid when both the valid (TVLD) and ready (TRDY) signals are asserted. The HDMI 1.4/2.0 Receiver Subsystem sends out adjacent channels in sequential order (CH0, CH1, etc). Usually, the audio stream receiver also expects the channels in sequential order. If the channel data is not in order, the channel data might be mapped into other channel sample slots.

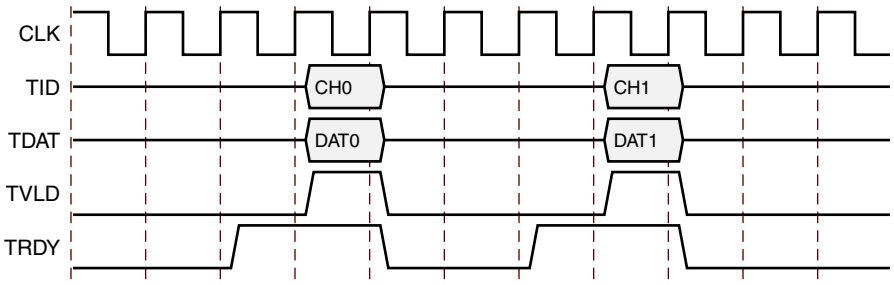

Figure 3-1: **Audio Cycle**

Video Output Stream Interface

The AXI4-Stream video interface supports single, dual or quad pixels per clock with 8 bits, 10 bits, 12 bits and 16 bits per component for RGB and YUV444 color spaces. The color depth in YUV422 color space is always 12-bits per pixel.

When the parameter, **Max Bits Per Component**, is set to 16, Figure 3-2 shows the data format for quad pixels per clock to be fully compliant with the AXI4-Stream video protocol. A data format for a fully compliant AXI4-Stream video protocol dual pixels per clock is illustrated in Figure 3-3. Similarly, a data format for a fully compliant AXI4-Stream video protocol single pixel per clock is illustrated in Figure 3-4.

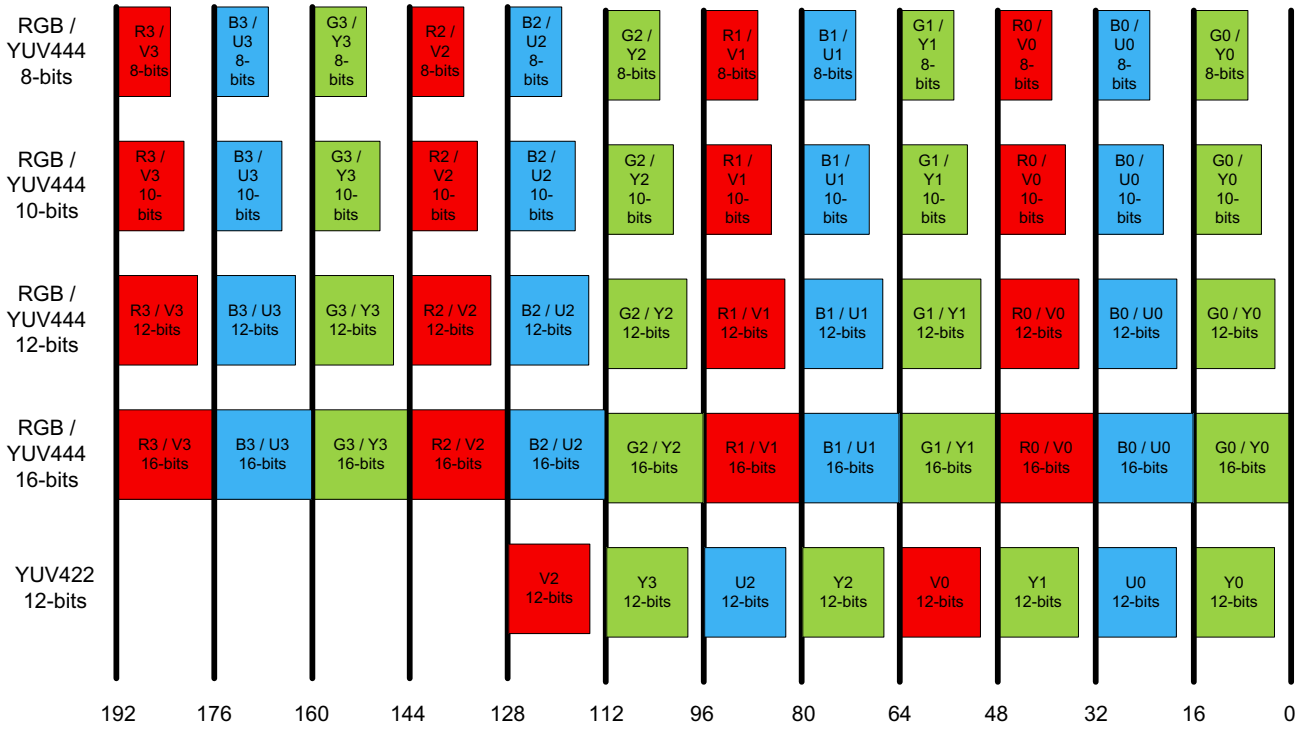

Figure 3-2: **Quad Pixels Data Format (Max Bits Per Component=16)**

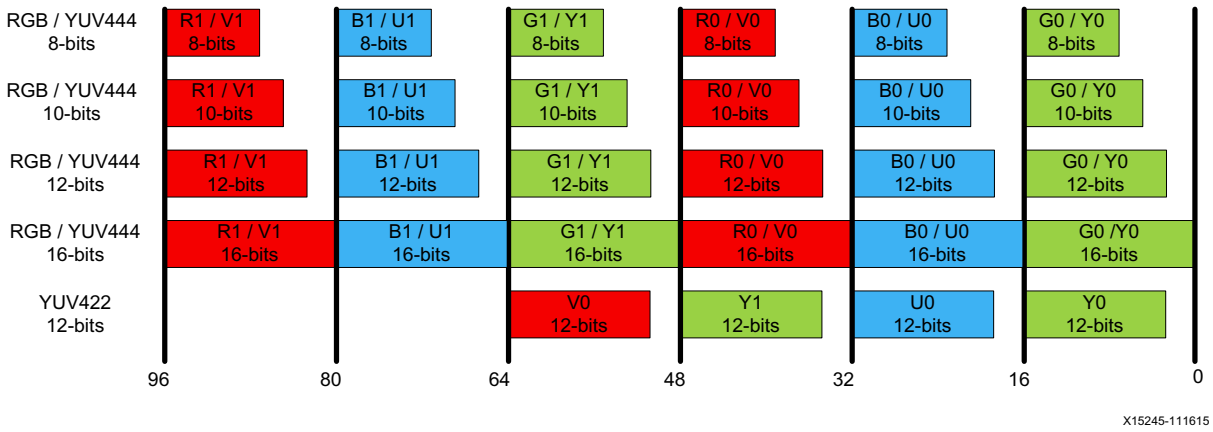

Figure 3-3: **Dual Pixels Data Format (Max Bits Per Component=16)**

Figure 3-4: **Single Pixel Data Format (Max Bits Per Component=16)**

When the parameter, **Max Bits Per Component**, is set to 12, video formats with actual bits per component larger than 12 is truncated to the Max Bits Per Component. The remaining least significant bits are discarded. If the actual bits per component is smaller than Max Bits Per Component set in the Vivado IDE, all bits are transported with the MSB aligned and the remaining LSB bits are padded with 0. This applies to all **Max Bits Per Component** settings.

Table 3-1: **Max Bits Per Component Support**

Max Bits Per Component	Actual Bits Per Component	Bits Transported by Hardware
16	8	[7:0]
	10	[9:0]
	12	[11:0]
	16	[15:0]
12	8	[7:0]
	10	[9:0]
	12	[11:0]
	16	[15:4]
10	8	[7:0]
	10	[9:0]
	12	[11:2]
	16	[15:6]
8	8	[7:0]
	10	[9:2]
	12	[11:4]
	16	[15:8]

As an illustration, when **Max Bits Per Component** is set to 12, Figure 3-5 shows the data format for quad pixels per clock to be fully compliant with the AXI4-Stream video protocol. A data format for a fully compliant AXI4-Stream video protocol with dual pixels per clock is illustrated in Figure 3-6. Similarly, a data format for a fully compliant AXI4-Stream video protocol with a single pixel per clock is illustrated in Figure 3-7. When single pixel per clock is selected, the YUV422 video format is not supported by the subsystem.

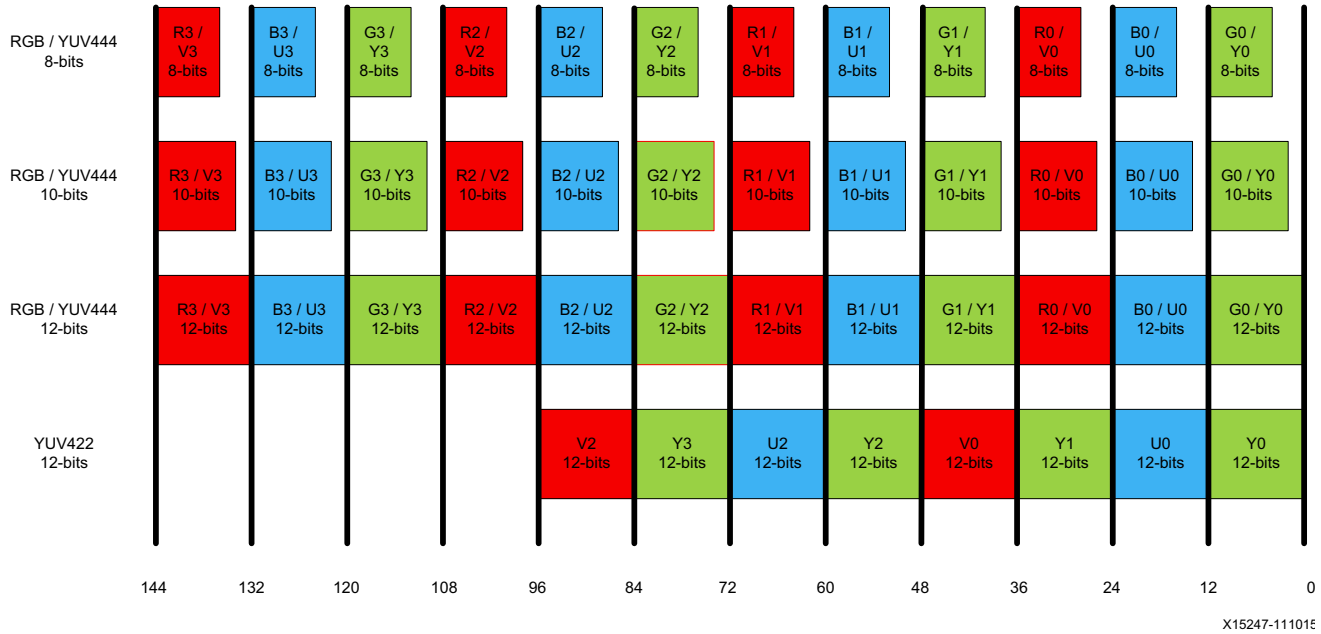

Figure 3-5: **Quad Pixels Data Format (Max Bits Per Component=12)**

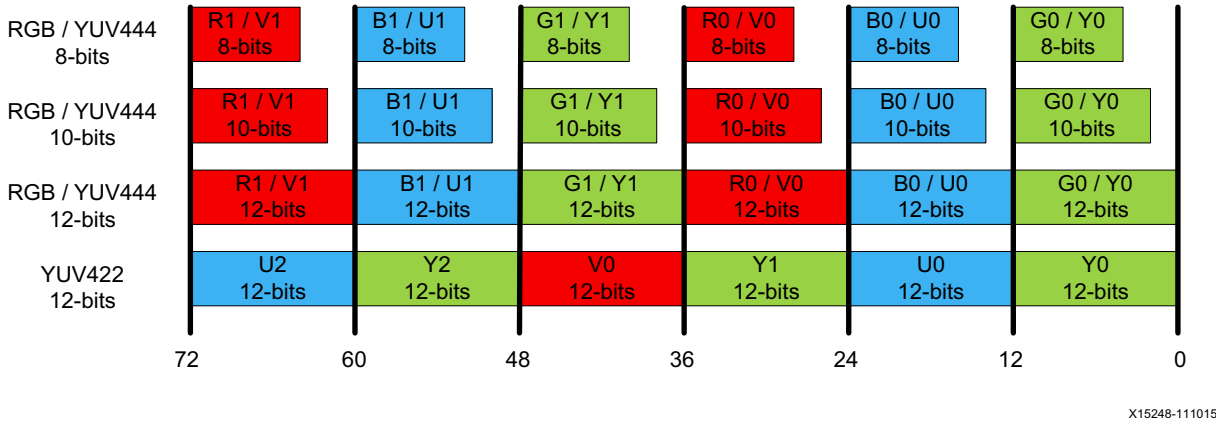

Figure 3-6: **Dual Pixels Data Format (Max Bits Per Component=12)**

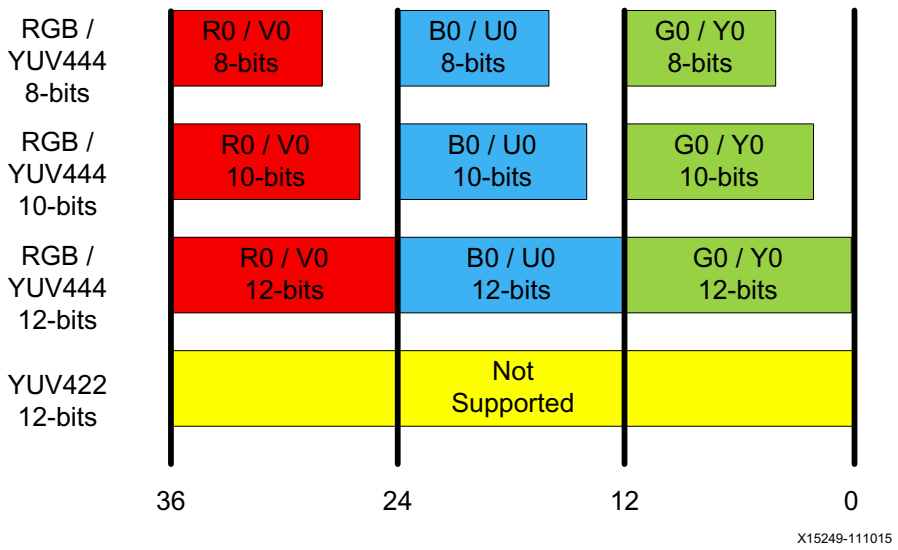

Figure 3-7: **Single Pixels Data Format (Max Bits Per Component=12)**

The video interface can also transport quad and dual pixels in the YUV420 color space. However the current data format is not complaint with the AXI4-Stream video protocol. Figure 3-8 and Figure 3-9 show the data format for quad and dual pixels formats.

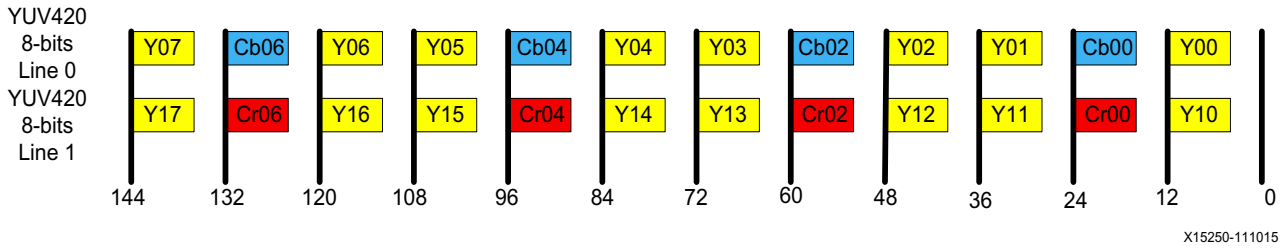

Figure 3-8: **YUV420 Color Space Quad Pixels Data Format**

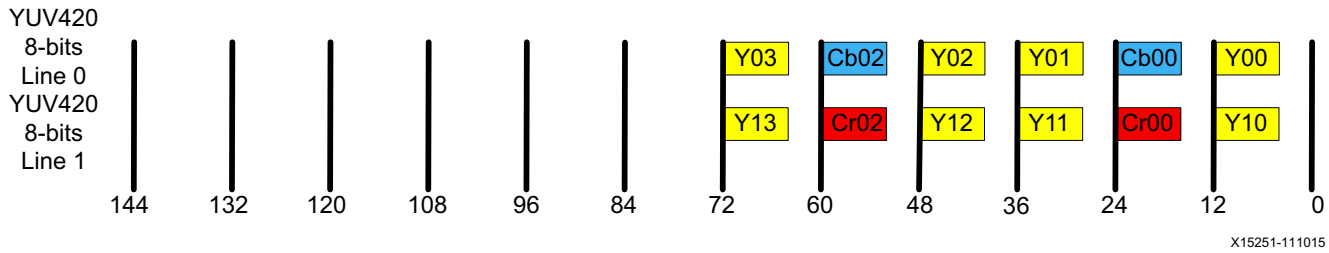

Figure 3-9: **YUV420 Color Space Dual Pixels Data Format**

The subsystem provides full flexibility to construct a system using the configuration parameters, Max bits per component and Number of pixels per clock. Set these parameters so that the video clock and link clock are supported by the targeted device. For example, timing violations can occur when selecting dual pixels per clock for high video resolutions (ultra HD). Therefore the quad pixels per clock data mapping is recommended for these higher video resolutions.

Some video resolutions (for example, 720p50 and wxga+p60) have horizontal timing parameters which are not a multiple of 4. In this case the dual pixels per clock data mapping must be chosen.

For more information on the video AXI4-Stream interface and video data format, see the *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 8].

Clocking

The S_AXI_CPU_IN, VIDEO_IN, and AUDIO_IN can be run at their own clock rate. The HDMI link interfaces and native video interface also run at their own clock rate. Therefore, five separate clock interfaces are provided called `s_axi_cpu_aclk`, `s_axis_video_aclk`, `s_axis_audio_aclk`, `link_clk`, and `video_clk` respectively.

The audio streaming clock must be greater than or equal or greater than 128 times the audio sample frequency. Because audio clock regeneration is not part of the HDMI RX subsystem, you must provide an audio clock to the application. This can be achieved by using an internal PLL or external clock source.

The HDMI clock structure is illustrated in Figure 3-10 and Table 3-2.

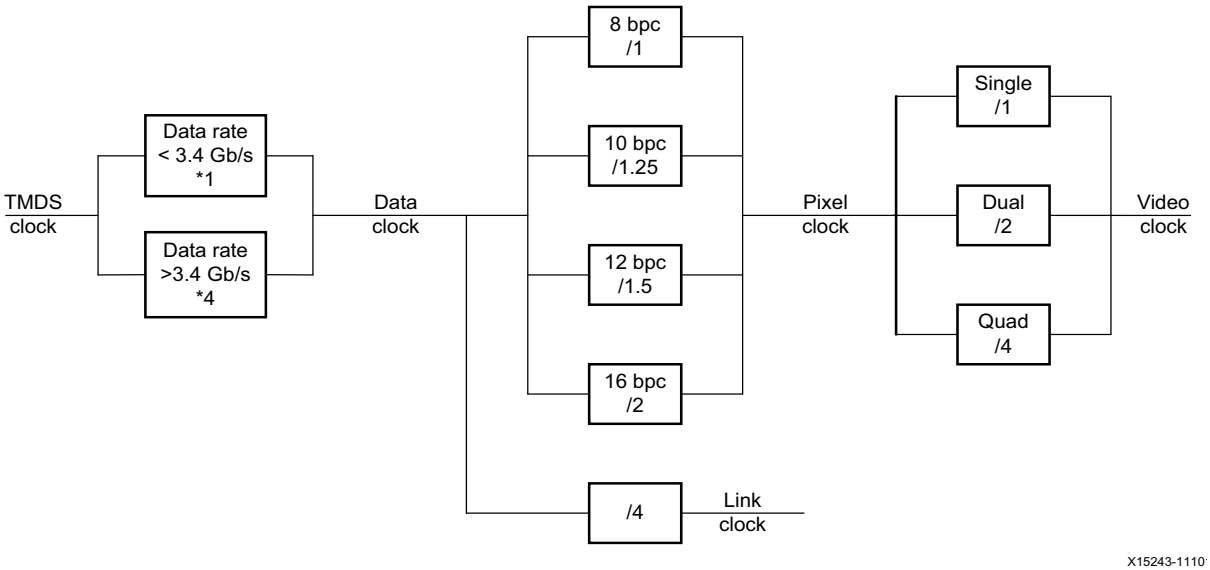

Figure 3-10: **HDMI Clocking Structure**

Table 3-2: **Clocking**

HDMI clocking			
Clock	**Function**	**Freq/Rate**	**Example**
TMDS clock	Source synchronous clock to HDMI interface (This is the actual clock on the HDMI cable).	= 1/10 data rate (for data rates < 3.4 Gb/s) = 1/40 data rate (for data rates > 3.4 Gb/s)	Data rate = 2.97 Gb/s TMDS clock = 2.97/10 = 297 MHz Data rate = 5.94 Gb/s TMDS clock = 5.94/40 = 148.5 MHz
Data clock	This is the actual data rate clock. This clock is not used in the system. It is only listed to illustrate the clock relations.	= TMDS clock (for data rates < 3.4 Gb/s) = TMDS clock * 4 (for data rates > 3.4 Gb/s)	Data rate = 2.97 Gb/s Data clock = TMDS clock * 1 = 297 MHz Data rate = 5.94 Gb/s Data clock = TMDS clock * 4 = 594 MHz TMDS clock = 148.5 MHz
Link clock	Clock used for data interface between HDMI PHY Layer Module and subsystem	= 1/4 of data clock	TMDS clock = 297 MHz Data clock = 297 MHz Link clock = 297 MHz/4 = 74.25 MHz Data clock = 594 MHz Link clock = 594 MHz/4 = 148.5 MHz
Pixel clock	This is the internal pixel clock. This clock is not used in the system. It is only listed to illustrate the clock relations.	for 8 bpc pixel clock = data clock for 10bpc pixel clock = data clock/1.25 for 12bpc pixel clock = data clock/1.5 for 16bpc pixel clock = data clock/2	
Video clock	Clock used for video interface	for single pixel video clock = pixel clock for dual pixel video clock = pixel clock/2 for quad pixel video clock = pixel clock/4	297 MHz for single pixel wide interface 297 MHz/2 = 148.5 MHz for dual pixel wide interface 297 MHz/4 = 74.25 MHz for quad pixel wide interface

Resets

Each AXI input interface has its own reset signal. The reset signals, `s_axi_cpu_aresetn`, `s_axis_video_aresetn` and `s_axis_audio_aresetn` are for `S_AXI_CPU_IN`, `VIDEO_IN`, and `AUDIO_IN` respectively. These three reset signals are active-Low. Because the reset signal is used across multiple sub-blocks in the subsystem, keep the system in the reset state until all the clocks are stabilized. You can use the `locked` signal from the clock generation block as a reset signal.

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 9]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 10]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 11]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 12]

Customizing and Generating the Subsystem

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

The HDMI 1.4/2.0 Receiver Subsystem can be added to a Vivado IP integrator block design in the Vivado Design Suite and can be customized using IP catalog. For more detailed information on customizing and generating the subsystem in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 9]. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl Console.

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

1. In the **Flow Navigator**, click on **Create Block Diagram** or **Open Block Design** under the IP Integrator heading.
2. Right click in the diagram and select **Add IP**.

   A searchable IP catalog opens. You can also add IP by clicking on the Add IP button on the left side of the IP Integrator Block Design canvas.

3. Click on the IP name and press the Enter key on your keyboard or double click on the IP name.
4. Double-click the selected IP block or select the **Customize Block** command from the right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 10] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 11].

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). The layout depicted here might vary from the current version.

Top Level Tab

The Toplevel tab is shown in Figure 4-1.

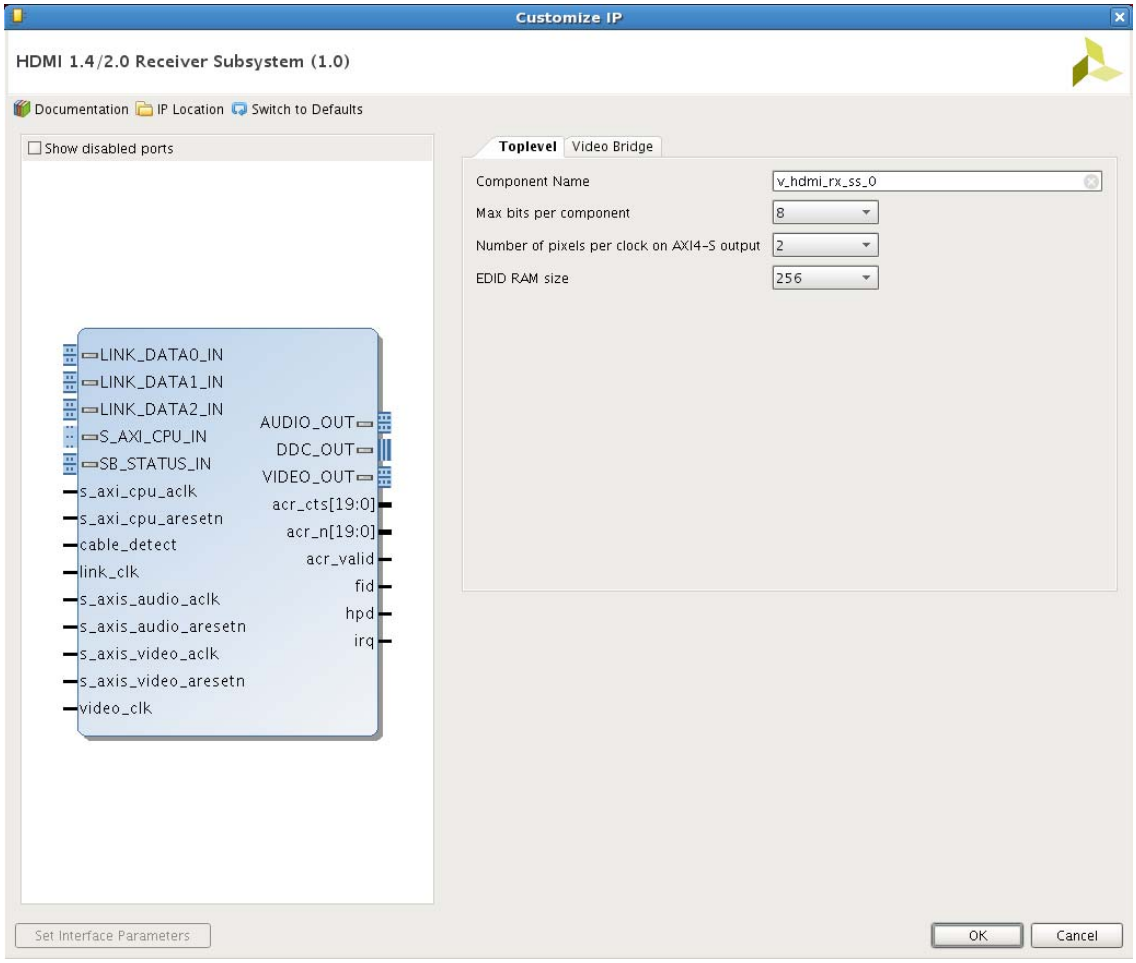

Figure 4-1: **Toplevel Tab**

The parameters on the Toplevel tab are as follows:

Component Name: The component name is set automatically by IP Integrator.

Max bits per component: This option selects the maximum bits per component. The allowable options are, 8, 10, 12 or 16 bits. This parameter is to set the maximum "allowed" bits per component, and the actual bits per component can be set from the software API to a different value. However, the actual bits per component is bounded by the **Max bits per component**. For example, if the **Max bits per component** is set to 16, the user can set the actual bits per component from the software API to any of the values, 8, 10, 12 or 16. But if the **Max bits per component** is set to 8, you can only set the actual bits per component to 8.

Number of pixels per clock on AXI4-S output: This option selects the number of pixels per clock. The allowable options are, 1, 2, or 4 pixels.

EDID RAM size: The allowable options are, 256, 512, 1024 or 4096.

Video Bridge Tab

The Video Bridge tab is shown in Figure 4-2.

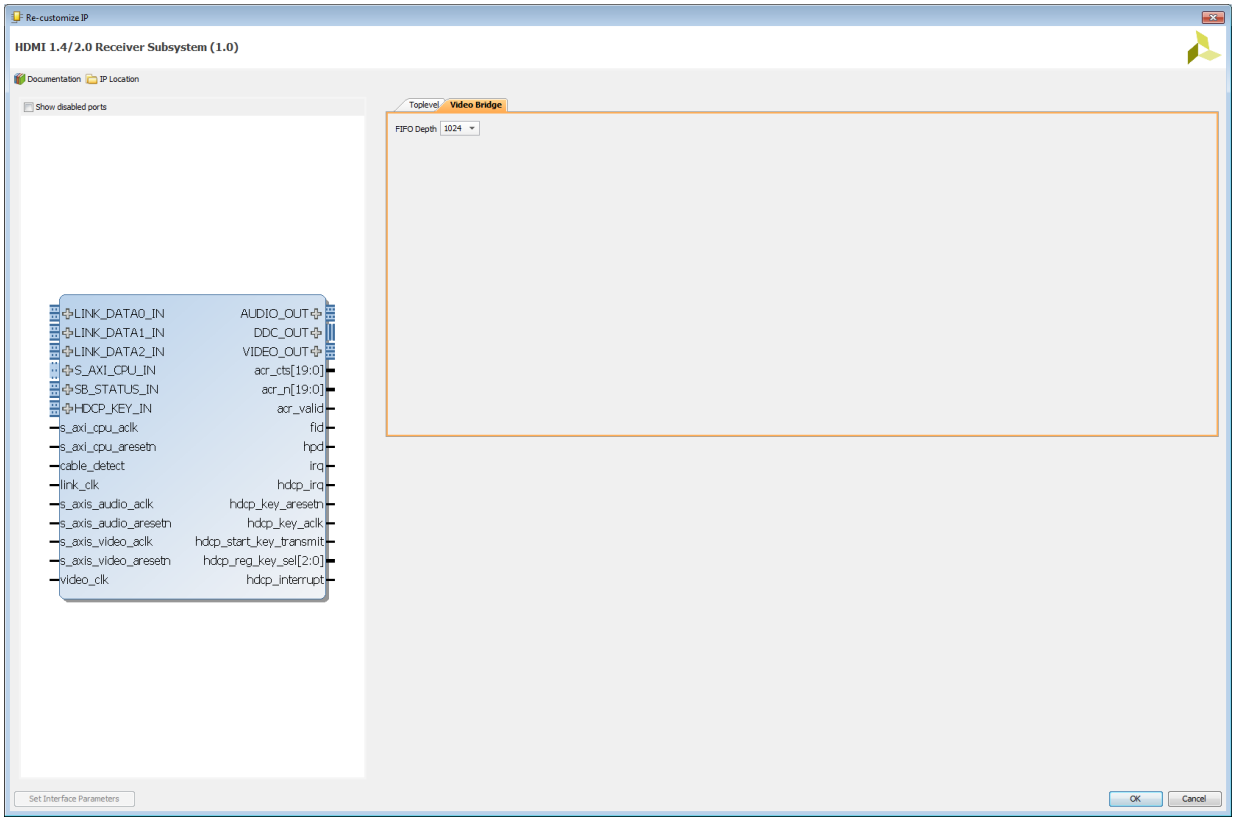

Figure 4-2: **Video Bridge Tab**

The parameters on the Video Bridge tab are as follows:

FIFO Depth: Specifies the number of locations in the input FIFO. The allowable values are 32, 1024, 2048, 4096, and 8192.

User Parameters

Table shows the relationship between the fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl Console).

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter	User Parameter	Default Value
Toplevel		
Max bits per component	C_MAX_BITS_PER_COMPONENT	8
Number of pixels per clock on AXI4-S output	C_INPUT_PIXELS_PER_CLOCK	2
EDID RAM size	C_EDID_RAM_SIZE	256
Video Bridge		
FIFO Depth	C_ADDR_WIDTH	1024

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 10].

Constraining the Subsystem

This section contains information about constraining the subsystem in the Vivado Design Suite.

Required Constraints

There are clock frequency constraints for the `s_axi_cpu_aclk`, `s_axis_video_aclk`, `s_axis_audio_aclk`, `link_clk`, and `video_clk`. For example,

```
create_clock -name s_axi_cpu_aclk -period 10.0 [get_ports s_axi_cpu_aclk]
create_clock -name s_axis_audio_aclk -period 10.0 [get_ports s_axis_audio_aclk]
create_clock -name link_clk -period 13.468 [get_ports link_clk]
create_clock -name video_clk -period 6.734 [get_ports video_clk]
create_clock -name s_axis_video_aclk -period 5.0 [get_ports s_axis_video_aclk]
```

These constraints are only for illustration purpose. When using this subsystem in the Vivado® Design Suite flow with supporting GT transceiver modules, all these clocks are generated from the GT transceiver. Therefore, the clock constraints will be set to the GT transceiver clock constraints instead of these generated clocks.

Device, Package, and Speed Grade Selections

This section is not applicable for this IP subsystem.

Clock Frequencies

This section is not applicable for this IP subsystem.

Clock Management

This section is not applicable for this IP subsystem.

Clock Placement

This section is not applicable for this IP subsystem.

Banking

This section is not applicable for this IP subsystem.

Transceiver Placement

This section is not applicable for this IP subsystem.

I/O Standard and Placement

This section is not applicable for this IP subsystem.

Simulation

Simulation of the subsystem is not supported.

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 10].

Appendix A

Verification, Compliance, and Interoperability

Interoperability

The HDMI 1.4/2.0 Receiver Subsystem has been verified with functional simulation for all related internal sub-cores and extensive hardware testing. Interoperability tests for the HDMI 1.4/2.0 Receiver Subsystem are in progress.

Hardware Testing

The HDMI 1.4/2.0 Receiver Subsystem has been validated using

- Kintex®-7 FPGA Evaluation Kit (KC705)
- Kintex® UltraScale™ FPGA Evaluation Kit (KCU105)
- Inrevium Artix-7 FPGA ACDC A7 Evaluation Board

This release is tested with the following source devices:

- Quantum Data 980B
- Quantum Data 780B
- Apple TV
- Android M8 media player
- Apple MacBook Pro
- Google Chromecast
- Open Hour media box
- Dell Latitude laptop (E7240)
- Intel HD Graphics 4000
- Nvidia GTX970 graphics card
- UGOOS media box

Video Resolutions

Table A-1, Table A-2, and Table A-3 show the video resolutions that were tested as part of the release for different video formats.

Table A-1: **Tested Video Resolutions for RGB 4:4:4 and YCbCr 4:4:4**

Resolution	Horizontal		Vertical		Frame Rate (Hz)
	Total	Active	Total	Active	
480i60	858	720	525	480	60
576i50	864	720	625	576	50
1080i50	2640	1920	1125	1080	50
1080i60	2200	1920	1125	1080	60
480p60	858	720	525	480	60
576p50	864	720	625	576	50
720p50	1980	1280	750	720	50
720p60	1650	1280	750	720	60
1080p24	2750	1920	1125	1080	24
1080p25	2640	1920	1125	1080	25
1080p30	2200	1920	1125	1080	30
1080p50	2640	1920	1125	1080	50
1080p60	2200	1920	1125	1080	60
2160p24	5500	3840	2250	2160	24
2160p25	5280	3840	2250	2160	25
2160p30	4400	3840	2250	2160	30
2160p60	4400	3840	2250	2160	60
vgap60	800	640	525	480	60
svgap60	1056	800	628	600	60
xgap60	1344	1024	806	768	60
sxgap60	1688	1280	1066	1024	60
wxgap60	1440	1280	790	768	60
wxga+p60	1792	1366	798	768	60
uxgap60	2160	1600	1250	1200	60
wuxgap60	2592	1920	1245	1200	60
wsxgap60	2240	1680	1089	1050	60

Notes:

1. In this release, UXGA 60 Hz is supported in the HDMI 1.4/2.0 Receiver Subsystem for 8, 10, and 12 bits per component only.

Table A-2: **Tested Video Resolutions for YCbCr 4:2:2 at 12 Bits/component**

Resolution	Horizontal		Vertical		Frame Rate (Hz)
	Total	Active	Total	Active	
1080i50	2640	1920	1125	1080	50
1080i60	2200	1920	1125	1080	60
480p60	858	720	525	480	60
576p50	864	720	625	576	50
720p50	1980	1280	750	720	50
720p60	1650	1280	750	720	60
1080p24	2750	1920	1125	1080	24
1080p25	2640	1920	1125	1080	25
1080p30	2200	1920	1125	1080	30
1080p50	2640	1920	1125	1080	50
1080p60	2200	1920	1125	1080	60
2160p24	5500	3840	2250	2160	24
2160p25	5280	3840	2250	2160	25
2160p30	4400	3840	2250	2160	30
vgap60	800	640	525	480	60
svgap60	1056	800	628	600	60
wxgap60	1440	1280	790	768	60
wxga+p60	1792	1366	798	768	60
uxgap60	2160	1600	1250	1200	60
wuxgap60	2592	1920	1245	1200	60
wsxgap60	2240	1680	1089	1050	60

Table A-3: **Tested Video Resolutions for YCbCr 4:2:0 at 8 Bits/Component**

Resolution	Horizontal		Vertical		Frame Rate (Hz)
	Total	Active	Total	Active	
2160p60	4400	3840	2250	2160	60

Debugging

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

TIP: *If the IP generation halts with an error, there might be a license issue. See License Checkers in Chapter 1 for more details.*

Finding Help on Xilinx.com

To help in the design and debug process when using the HDMI 1.4/2.0 Receiver Subsystem, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the HDMI 1.4/2.0 Receiver Subsystem. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

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

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

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

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

Master Answer Record for the HDMI 1.4/2.0 Receiver Subsystem

AR: 54546

Technical Support

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

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

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Debug Tools

Tools are available to address HDMI 1.4/2.0 Receiver Subsystem design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

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

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

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

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 14].

Reference Boards

Various Xilinx development boards support the HDMI 1.4/2.0 Receiver Subsystem. These boards can be used to prototype designs and establish that the subsystem can communicate with the system.

- 7 series FPGA evaluation boards
  - KC705
- UltraScale FPGA evaluation board
  - KCU105
- Inrevium Artix-7 FPGA ACDC A7 Evaluation Board

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado debug feature is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the debug feature for debugging the specific problems.

General Checks

Ensure that all the timing constraints and all other constraints were met during implementation.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.
- If your outputs go to 0, check your licensing.

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Output `s_axi_arready` asserts when the read address is valid, and output `s_axi_rvalid` asserts when the read data/response is valid. If the interface is unresponsive, ensure that the following conditions are met:

- The `s_axi_aclk` and `aclk` inputs are connected and toggling.
- The interface is not being held in reset, and `s_axi_areset` is an active-Low reset.
- The interface is enabled, and `s_axi_aclken` is active-High (if used).
- The main subsystem clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or a debug feature capture that the waveform is correct for accessing the AXI4-Lite interface.

AXI4-Stream Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit `<interface_name>_tready` is stuck Low following the `<interface_name>_tvalid` input being asserted, the subsystem cannot send data.
- If the receive `<interface_name>_tvalid` is stuck Low, the subsystem is not receiving data.
- Check that the `aclk` inputs are connected and toggling.
- Check that the AXI4-Stream waveforms are being followed.
- Check subsystem configuration.

Application Software Development

Device Drivers

The HDMI 1.4/2.0 Receiver Subsystem driver abstracts the included supporting elements and presents it as a black-box. It shields you from the internal workings of the sub-cores thereby providing an out-of-the-box solution.

The subsystem driver is a bare-metal driver, which provides an abstracted view of the feature set provided by each sub-core. It dynamically manages the data and control flow through the processing elements, based on the input/output stream configuration set at run time. Internally, it relies on sub-core layer 1 drivers to configure the sub-core IP blocks.

Dependencies

A video common driver is delivered as part of the Xilinx Software Development Kit (SDK). Most video IPs have master/slave AXI4-Stream interfaces so the concept of "Video Streams" as an interface parameter for the drivers for such cores was introduced.

The `video_common_v2_0` driver defines the following features:

- Enumerations for video specific details such as color format, color depth, and frame rate.
- Video stream, video timing and video window data types.
- Video mode table with pre-defined resolutions and their timing details.
- Utility APIs that can access data from the mode table and work with stream data types.

Architecture

The subsystem driver provides an easy-to-use, well-defined API to help integrate the subsystem in an application without having to understand the underlying complexity of configuring each and every sub-core.

The subsystem driver consists of the following:

- **Subsystem layer**: Controls the data/control flow of AXI4-Stream data through the processing cores. It uses support functions to examine the input and output stream properties and make appropriate decisions to implement the determined use case.

- **Sub-core drivers**: Every included sub-core has a driver associated with it and provides two layers of abstraction
  - **Layer 1**: Implements APIs to peek/poke IP registers at HW level
  - **Layer 2**: Implements feature set to abstract core functionality

Figure C-1 shows the HDMI 1.4/2.0 Receiver Subsystem architecture.

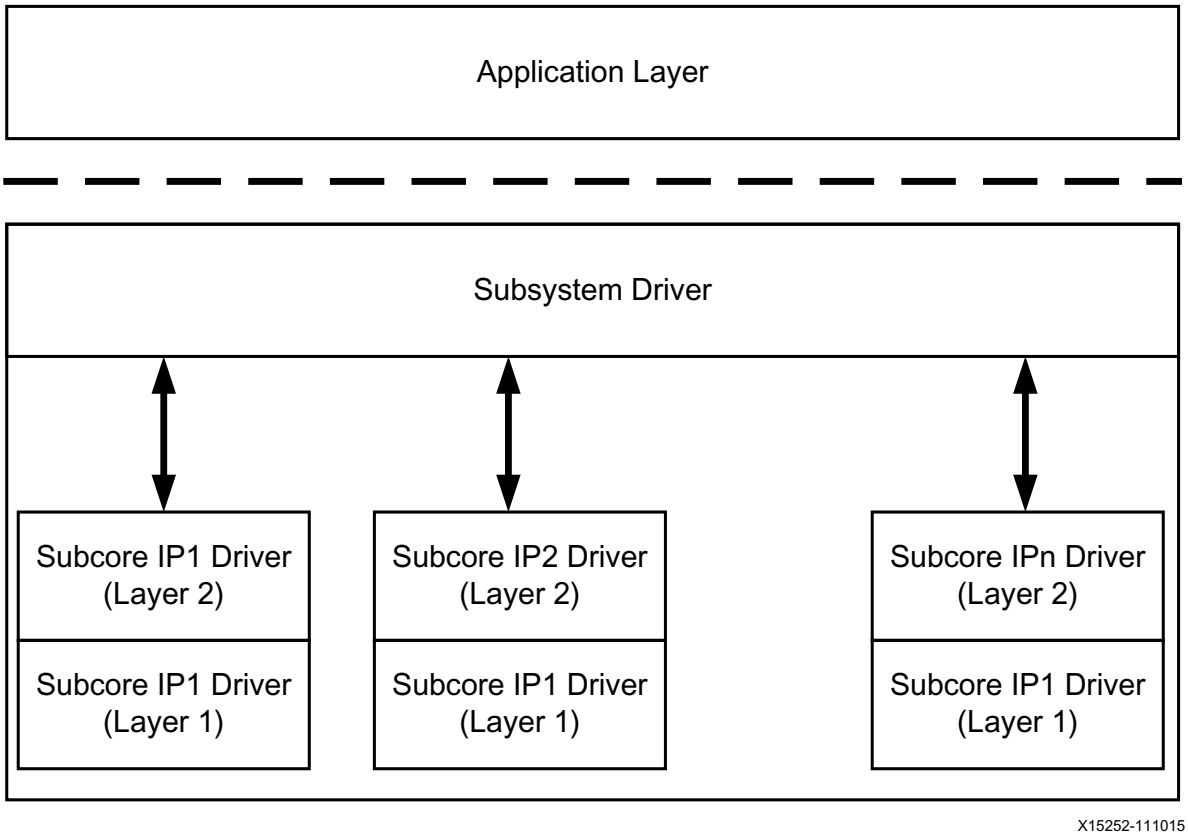

Figure C-1: **Subsystem Architecture**

Driver Use

The subsystem driver itself is not an active driver and relies on application software to make use of the APIs provided to configure it. Application software is responsible for monitoring the system for external inputs and communicating changes to input/output stream properties to the subsystem through APIs, and triggering the subsystem auto-reconfiguration process.

Additional Resources and Legal Notices

Xilinx Resources

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

References

These documents provide supplemental material useful with this product guide:

1. *Xilinx Vivado AXI Reference Guide* (UG1037)
2. *Kintex UltraScale FPGAs Data Sheet: DC and AC Switching Characteristics* (DS892)
3. *Virtex UltraScale FPGAs Data Sheet: DC and AC Switching Characteristics* (DS893)
4. *Kintex-7 FPGAs Data Sheet: DC and AC Switching Characteristics* (DS182)
5. *Virtex-7 FPGAs Data Sheet: DC and AC Switching Characteristics* (DS183)
6. *Artix-7 FPGAs Data Sheet: DC and AC Switching Characteristics* (DS181)
7. HDMI specifications (www.hdmi.org/manufacturer/specification.aspx)
8. AXI4-Stream Video IP and System Design Guide (UG934)
9. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
10. *Vivado Design Suite User Guide: Designing with IP* (UG896)
11. *Vivado Design Suite User Guide: Getting Started* (UG910)
12. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
13. *ISE to Vivado Design Suite Migration Guide* (UG911)
14. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
15. *Vivado Design Suite User Guide: Implementation* (UG904)
16. *LogiCORE IP AXI Interconnect Product Guide* (PG059)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/18/2015	1.0	Initial Xilinx release.

Please Read: Important Legal Notices