Debug Bridge v2.0

LogiCORE IP Product Guide

Vivado Design Suite

PG245 April 28, 2017

XILINX
ALL PROGRAMMABLE™

Table of Contents

Appendix D: Additional Resources and Legal Notices

Introduction

The Xilinx® LogiCORE™ IP Debug Bridge core is a controller which provides a mechanism to establish a communication channel for debug cores with runtime software. The Debug Bridge usage can be classified into two categories: Tandem with Field Updates and Xilinx Virtual Cable (XVC). These two categories provide the means for communicating with the debug IP (including Memory IP) that is in the design.

The Tandem with Field Updates flow allows you to download new functionality into a device over the PCIe® link after the device is initially configured through the Tandem PROM/PCIe. The XVC flow allows you to use debug cores and debug the design over non-JTAG interface (for example, Ethernet/PCIe).

Note: The current version is for the Tandem with Field Updates solution, Partial Reconfiguration design debugging, and XVC based designs.

Features

There are two broad classification of Debug Bridge IP functionality, which are supported using six different modes.

- **Tandem with Field Updates and Partial Reconfiguration Solution** – User selectable mode **From_BSCAN_to_Debug** is used to add a Debug Bridge instance in each Reconfigurable Module which would connect to debug cores like ILA, VIO, Memory IP, and JTAG2AXI
- **Xilinx Virtual Cable (XVC) Solution** – Five modes are supported:
    - User selectable mode **From_AXI_to_BSCAN** is used to add a Debug Bridge instance in the design with an Ethernet/PCIe master. This mode is a slave to Ethernet/PCIe master while connecting to debug cores like ILA, VIO, Memory IP, and JTAG2AXI in the same chip.
    - User selectable mode **From_AXI_to_JTAG** is used to add a Debug Bridge instance in the design with an Ethernet/PCIe master. This mode is a slave to Ethernet/PCIe master while bringing out the JTAG pins out of the FPGA through I/O pins. This mode is mainly used to debug design on another board over XVC.
    - User selectable mode **From_JTAG_to_BSCAN** is used to add a Debug Bridge instance to debug the designs over soft Test Access Port (TAP) controller.
    - The following modes are only available for UltraScale+™ and UltraScale™ device architectures:
        - User selectable mode **From_PCIE_to_BSCAN** is used to add a Debug Bridge instance in the design with a PCIe master. This mode is a slave connected on the Extended Config interface to a PCIe master while connecting to debug cores like ILA, VIO, Memory IP, and JTAG2AXI in the same chip.
        - User selectable mode **From_PCIE_to_JTAG** is used to add a Debug Bridge instance in the design with a PCIe master. This mode is a slave connected on the Extended Config interface to a PCIe master while bringing out the JTAG pins out of the FPGA through I/O pins. This mode is mainly used to debug design on another board over XVC.

LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family[(1)]	UltraScale+™, UltraScale™, 7 Series
Supported User Interfaces	IEEE Standard 1149.1 – JTAG
Resources	Performance and Resource Utilization web page
Provided with Core	
Design Files	Register Transfer Level (RTL)
Example Design	Verilog
Test Bench	Not Provided
Constraints File	Xilinx Design Constraints (XDC)
Simulation Model	Not Provided
Supported S/W Driver	Not Provided
Tested Design Flows[(2)]	
Design Entry	Vivado® Design Suite, Verilog, VHDL
Simulation	Not Provided
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. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The Xilinx® Debug Bridge IP core establishes the communication channel between the host machine and debug cores inside a Reconfigurable Module (RM) region through a Static region. The debug bridge needs to be instantiated in the RM with a BSCAN interface defined at the Partial Reconfiguration (PR) boundary.

The Xilinx Debug Bridge IP core establishes the communication channel between the host machine and debug cores in both Tandem with Field Updates, Partial Reconfiguration based and Xilinx Virtual Cable (XVC) based designs. The following six modes describe the usage of the Debug Bridge for various designs.

Tandem with Field Updates and Partial Reconfiguration Solution

The **From_BSCAN_to_DebugHub** mode is used to create a Debug Bridge instance that must be placed in each Reconfigurable Module. This IP connects to debug cores like ILA, VIO, Memory IP, and JTAG2AXI. As of Vivado® Design Suite 2017.1, this connectivity between debug components in static and reconfigurable parts of the design is done automatically. Direct instantiation of this core should only be done as a backup methodology in the case that the Debug Hub inference methodology cannot be used.

For further information on use of the Debug Bridge Partial Reconfiguration designs, see the *Vivado Design Suite User Guide: Partial Reconfiguration* (UG909) [Ref 2].

Figure 1-1 and Figure 1-2 show the mode used to add a Debug Bridge instance in each Reconfigurable Module.

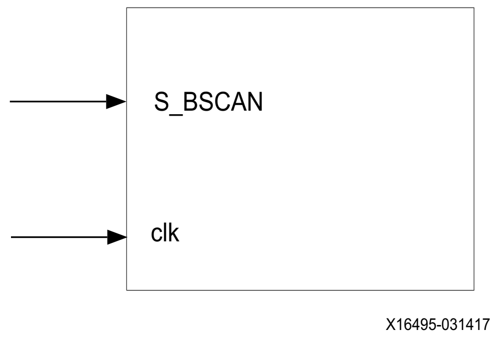


Figure 1-1: **From_BSCAN_to_DebugHub Mode Port Diagram**

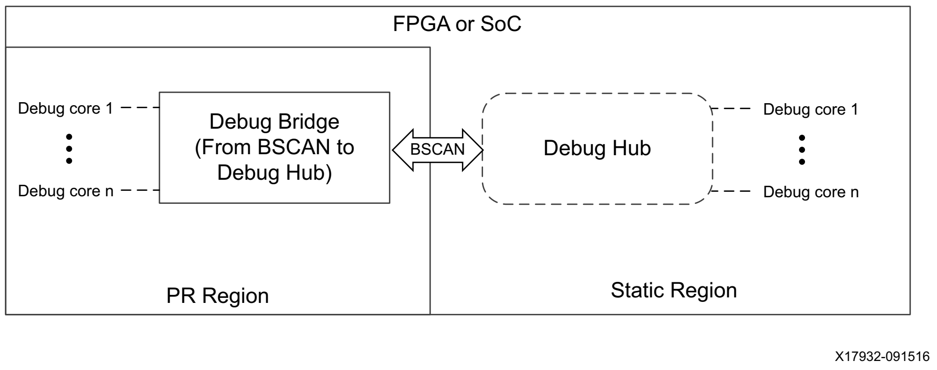


Figure 1-2: **Debug Bridge Configured with From_BSCAN_to_DebugHub Mode**

Xilinx Virtual Cable

The **From_AXI_to_BSCAN** mode is used to add a Debug Bridge instance in the design with an Ethernet/PCIe master. This mode of Debug Bridge is a slave to Ethernet/PCIe master while connecting to debug cores like ILA, VIO, Memory IP, and JTAG2AXI in the same chip.

Figure 1-3 and Figure 1-4 show the **From_AXI_to_BSCAN** mode in the XVC use case.

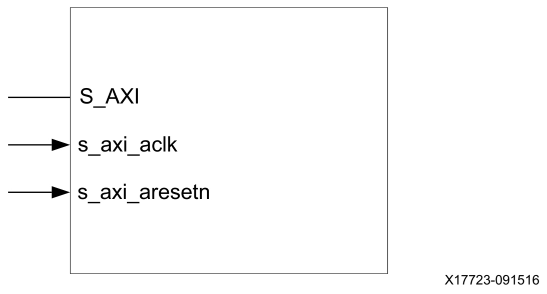


Figure 1-3: **From_AXI_to_BSCAN Mode Port Diagram**

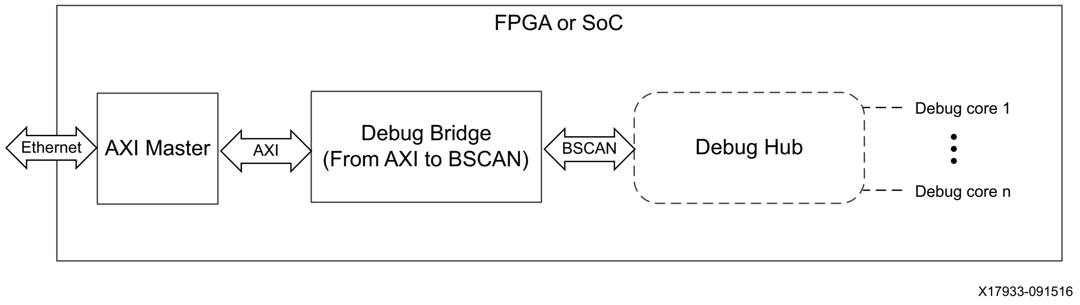


Figure 1-4: **Debug Bridge Configured From_AXI_to_BSCAN Mode**

The **From_AXI_to_JTAG** mode is used to add a Debug Bridge instance in the design with an Ethernet/PCIe master. This mode of Debug Bridge is a slave to Ethernet/PCIe master while bringing out the JTAG pins out of the FPGA through I/O pins. This mode is mainly used to debug design on another board over XVC.

Figure 1-5 and Figure 1-6 show the **From_AXI_to_JTAG** mode in the XVC use case.

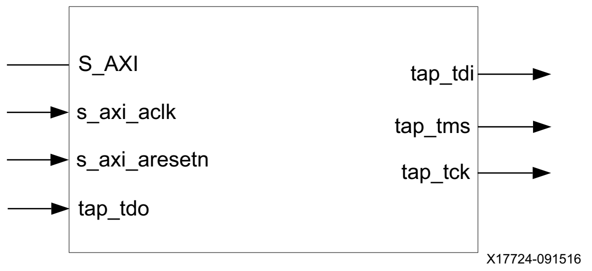


Figure 1-5: **From_AXI_to_JTAG Mode Port Diagram**

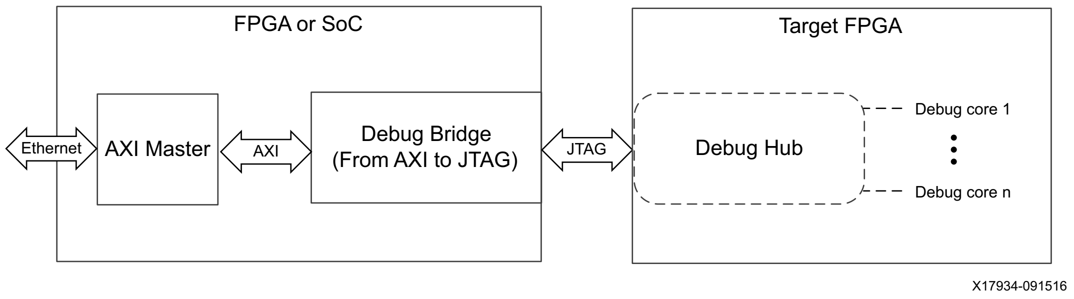


Figure 1-6: **Debug Bridge Configured From_AXI_to_JTAG Mode**

The **From_JTAG_to_BSCAN** mode is used to add a Debug Bridge instance to debug the designs over soft Test Access Port (TAP) controller.

Figure 1-7 and Figure 1-8 show the **From_JTAG_to_BSCAN** mode in the XVC use case.

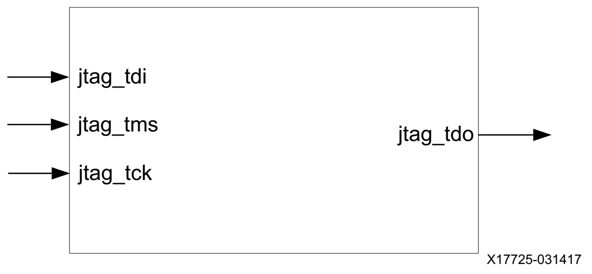


Figure 1-7: **From_JTAG_to_BSCAN Mode Port Diagram**

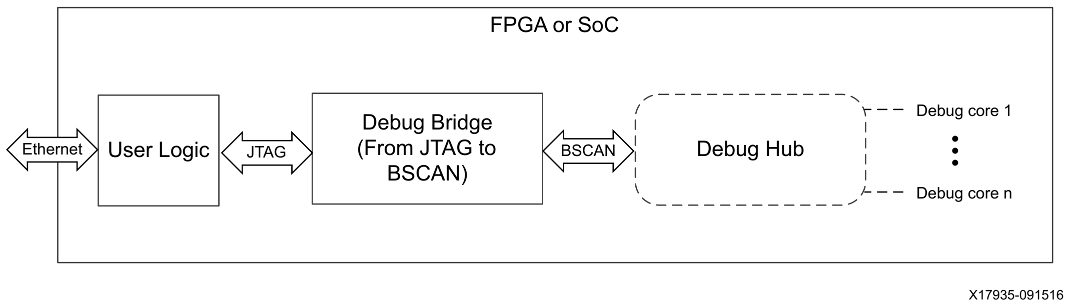


Figure 1-8: **Debug Bridge Configured From_JTAG_to_BSCAN Mode**

The **From_PCIE_to_BSCAN** mode is used to add a Debug Bridge instance in the design with a PCIe master. This Debug Bridge mode is a slave connected on the Extended Config interface to a PCIe master to debug cores like ILA, VIO, Memory IP, and JTAG2AXI in the same chip.

Figure 1-9 and Figure 1-10 show the **From_PCIE_to_BSCAN** mode in the XVC use case.

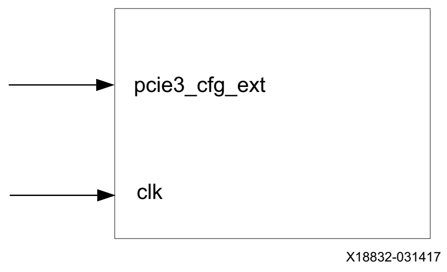


Figure 1-9: **From_PCIE_to_BSCAN Mode Port Diagram**

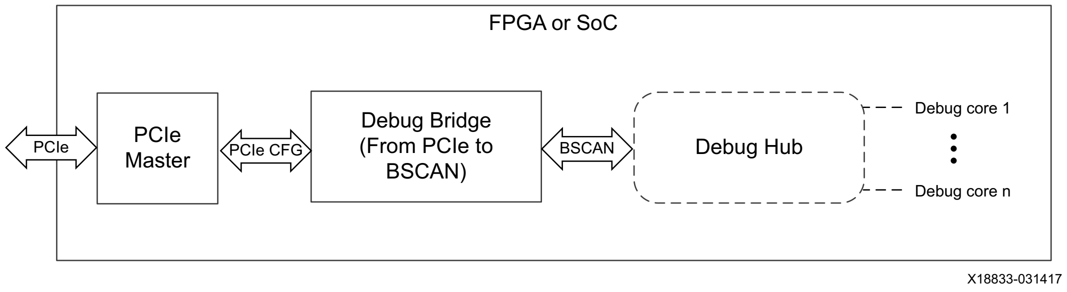


Figure 1-10: **Debug Bridge Configured From_PCIE_to_BSCAN Mode**

The **From_PCIE_to_JTAG** mode is used to add a Debug Bridge instance in the design with a PCIe master. This mode is a slave connected on the Extended Config interface to a PCIe master while bringing out the JTAG pins out of the FPGA through I/O pins. This mode is mainly used to debug design on another board over XVC.

Figure 1-11 and Figure 1-12 show the **From_PCIE_to_JTAG** mode in the XVC use case.

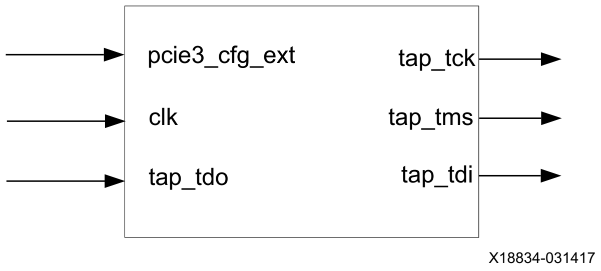


Figure 1-11: **From_PCIE_to_JTAG Mode Port Diagram**

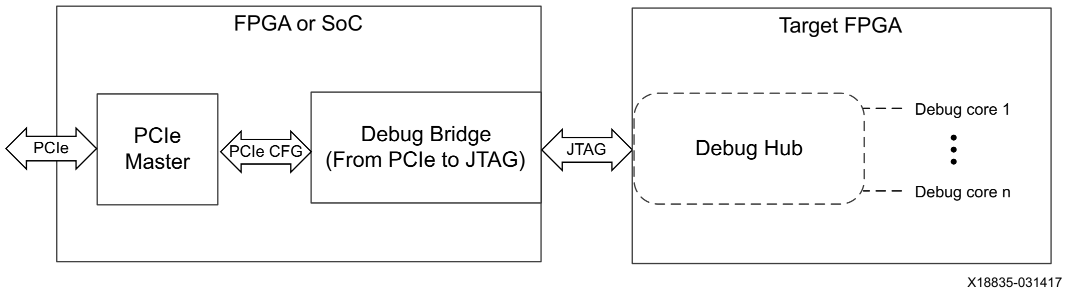


Figure 1-12: **Debug Bridge Configured From_PCIE_to_JTAG Mode**

Licensing and Ordering

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

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

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.*

Product Specification

Performance

The Debug Bridge core can be configured in two modes which caters the needs for debugging the designs for Tandem with Field Updates.

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

Resource Utilization

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

Port Descriptions

Table 2-1 and Table 2-4 show the signals for the different bridge type modes.

Table 2-1: **From BSCAN to DebugHub Mode**

Signal	I/O	Description
clk	Input clock to drive DebugHub logic	This port needs to connect to a free running clock available in the design.
S_BSCAN	BSCAN slave interface (with BSCANID_EN)	This slave interface needs to connect to the BSCAN interface of a BSCAN master. (When C_BSCANID_VEC = 0).
S_BSCAN_VEC	BSCAN slave interface (with BSCANID as 32-bit)	This slave interface needs to connect to the BSCAN interface of a BSCAN master. (When C_BSCANID_VEC = 1).

Table 2-2: **From AXI4 to BSCAN Mode**

Signal	I/O	Description
S_AXI	AXI4-Lite slave interface	This interface needs to connect to an AXI4.
s_axi_aclk	I	This port needs to connect to an AXI4 clock pin.
s_axi_aresetn	I	This port needs to connect to an AXI4 reset pin.

Table 2-3: **From AXI4 to JTAG Mode**

Signal	I/O	Description
S_AXI	AXI4-Lite slave interface	This interface needs to connect to an AXI4.
s_axi_aclk	I	This port needs to connect to the AXI4 clk pin.
s_axi_aresetn	I	This port needs to connect to the AXI4 reset pin.
tap_tdo	I	This port needs to connect to TDO pin of JTAG slave.
tap_tdi	O	This port needs to connect to TDI pin of JTAG slave.
tap_tck	O	This port needs to connect to TCK pin of JTAG slave.
tap_tms	O	This port needs to connect to TMS pin of JTAG slave.

Table 2-4: **From JTAG to BSCAN Mode**

Signal	I/O	Description
jtag_tdo	O	This port needs to connect to TDO pin of JTAG master.
jtag_tdi	I	This port needs to connect to TDI pin of JTAG master.
jtag_tck	I	This port needs to connect to TCK pin of JTAG master.
jtag_tms	I	This port needs to connect to TMS pin of JTAG master.

Table 2-5: **From PCIe to BSCAN Mode**

Signal	I/O	Description
clk	Input clock to drive DebugHub logic	This port needs to connect to a free running clock available in the design.
pcie3_cfg_ext	PCIe Extended Config slave interface	This slave interface needs to connect to PCIe extended config master interface.

Table 2-6: **From PCIe to JTAG Mode**

Signal	I/O	Description
clk	Input clock to drive DebugHub logic	This port needs to connect to a free running clock available in the design.
pcie3_cfg_ext	PCIE Extended Config slave interface	This slave interface needs to connect to PCIe extended config master interface.
tap_tdi	O	This port needs to connect to TDI pin of JTAG slave.
tap_tdo	I	This port needs to connect to TDO pin of JTAG slave.
tap_tck	O	This port needs to connect to TCK pin of JTAG slave.
tap_tms	O	This port needs to connect to TMS pin of JTAG slave.

Designing with the Core

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

Clocking

The `clk` input port in the **From_BSCAN_to_DebugHub** mode needs to be connected to a free running clock available in the design. This clock is used by Debug Hub.

Resets

There are no resets for this IP core.

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 3]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 4]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 5]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 6]

Customizing and Generating the Core

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

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 3] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

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

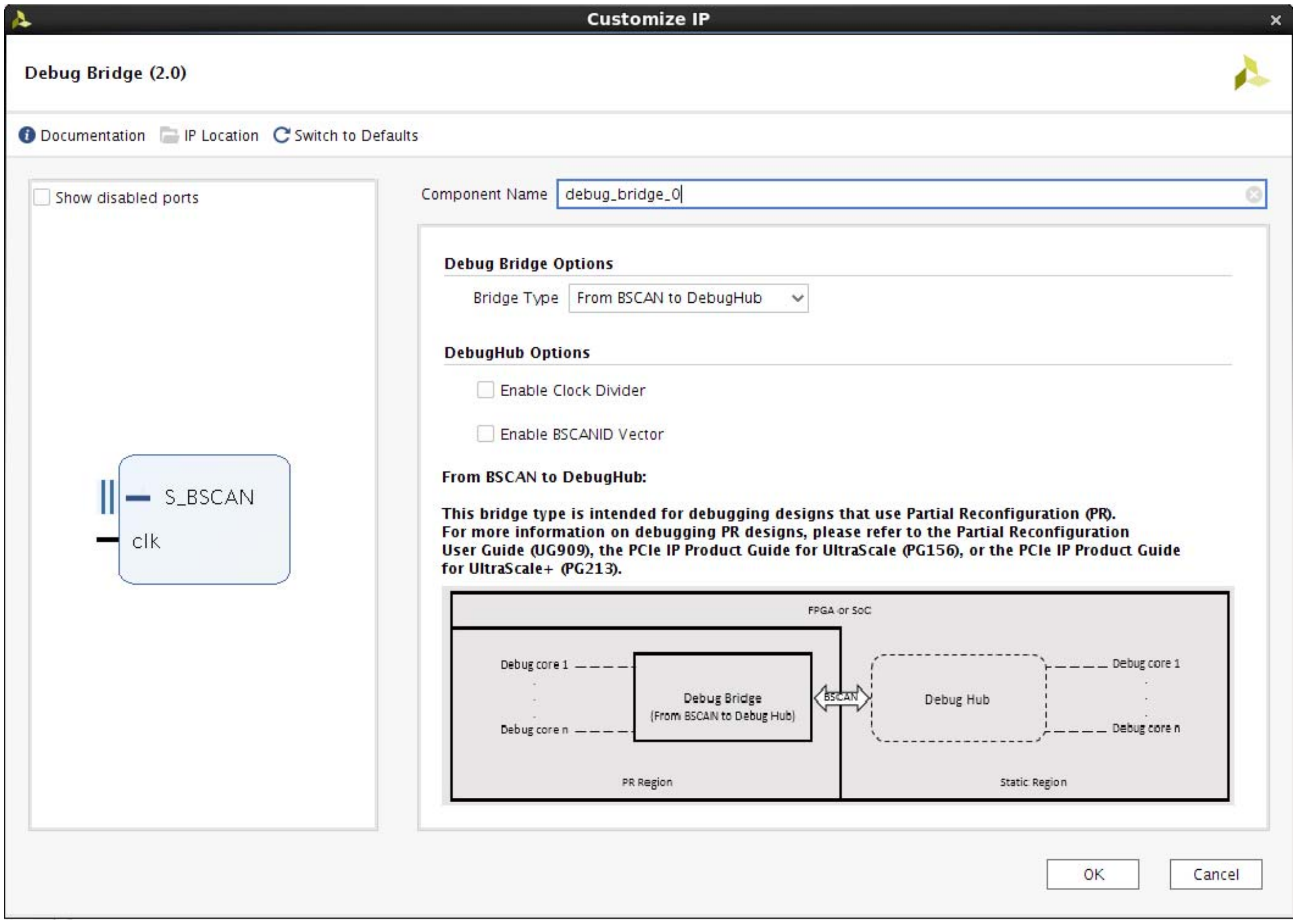
1. Open a project by selecting **File > Open Project** or create a new project by selecting **File > New Project** in Vivado.
2. Select the **Debug Bridge** IP from the **Debug & Verification > Debug > Debug Bridge** in Vivado IP catalog.
3. Double-click the selected IP or select the **Customize IP** command from the toolbar or right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 4] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 5].

Note: Figure in this chapter is an illustration of the Vivado Integrated Design Environment (IDE). The layout depicted here might vary from the current version.

General Options Panel

Figure 4-1 shows the Debug Bridge Vivado IDE main configuration screen.



Debug Bridge (2.0)

Component Name: debug_bridge_0

Debug Bridge Options

Bridge Type: From BSCAN to DebugHub

DebugHub Options

☐ Enable Clock Divider

☐ Enable BSCANID Vector

From BSCAN to DebugHub:

This bridge type is intended for debugging designs that use Partial Reconfiguration (PR). For more information on debugging PR designs, please refer to the Partial Reconfiguration User Guide (UG909), the PCIe IP Product Guide for UltraScale (PG156), or the PCIe IP Product Guide for UltraScale+ (PG213).

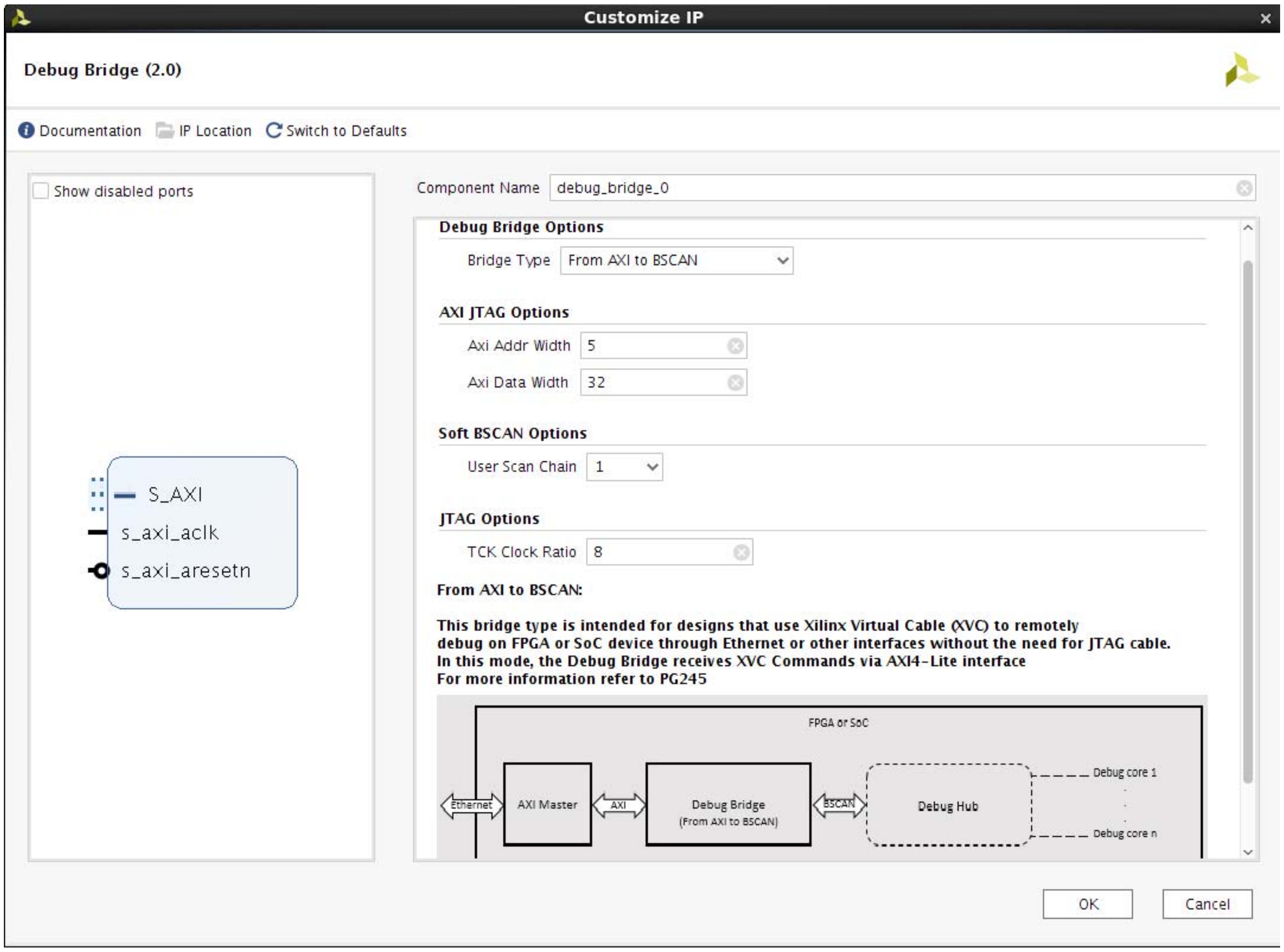
Figure 4-1: **Debug Bridge Customize IP – From BSCAN to DebugHub Mode**

- **Component Name** – Use this text field to provide a unique module name for the Debug Bridge core.
- **Debug Bridge Options** – This option switches between six different modes:
  - **From_BSCAN_to_DebugHub**
  - **From_AXI_to_BSCAN**
  - **From_AXI_to_JTAG**

- **From_JTAG_to_BSCAN**
- **From_PCIE_to_BSCAN**
- **From_PCIE_to_JTAG**

- **DebugHub Options** – There are two options:
  - The **Enable Clock Divider** divides down the clock frequency on the `clk` port to 100 MHz. This option enables the **From_BSCAN_to_DebugHub** mode.
  - The **Enable BSCANID Vector** enables the BSCANID vector signal for backwards compatibility with the previous versions of this mode.

Figure 4-2 shows the Debug Bridge in **From_AXI_to_BSCAN** mode.



Component Name: debug_bridge_0

Debug Bridge Options

Bridge Type: From AXI to BSCAN

AXI JTAG Options

Axi Addr Width: 5

Axi Data Width: 32

Soft BSCAN Options

User Scan Chain: 1

JTAG Options

TCK Clock Ratio: 8

From AXI to BSCAN:

This bridge type is intended for designs that use Xilinx Virtual Cable (XVC) to remotely debug on FPGA or SoC device through Ethernet or other interfaces without the need for JTAG cable. In this mode, the Debug Bridge receives XVC Commands via AXI4-Lite interface For more information refer to PG245

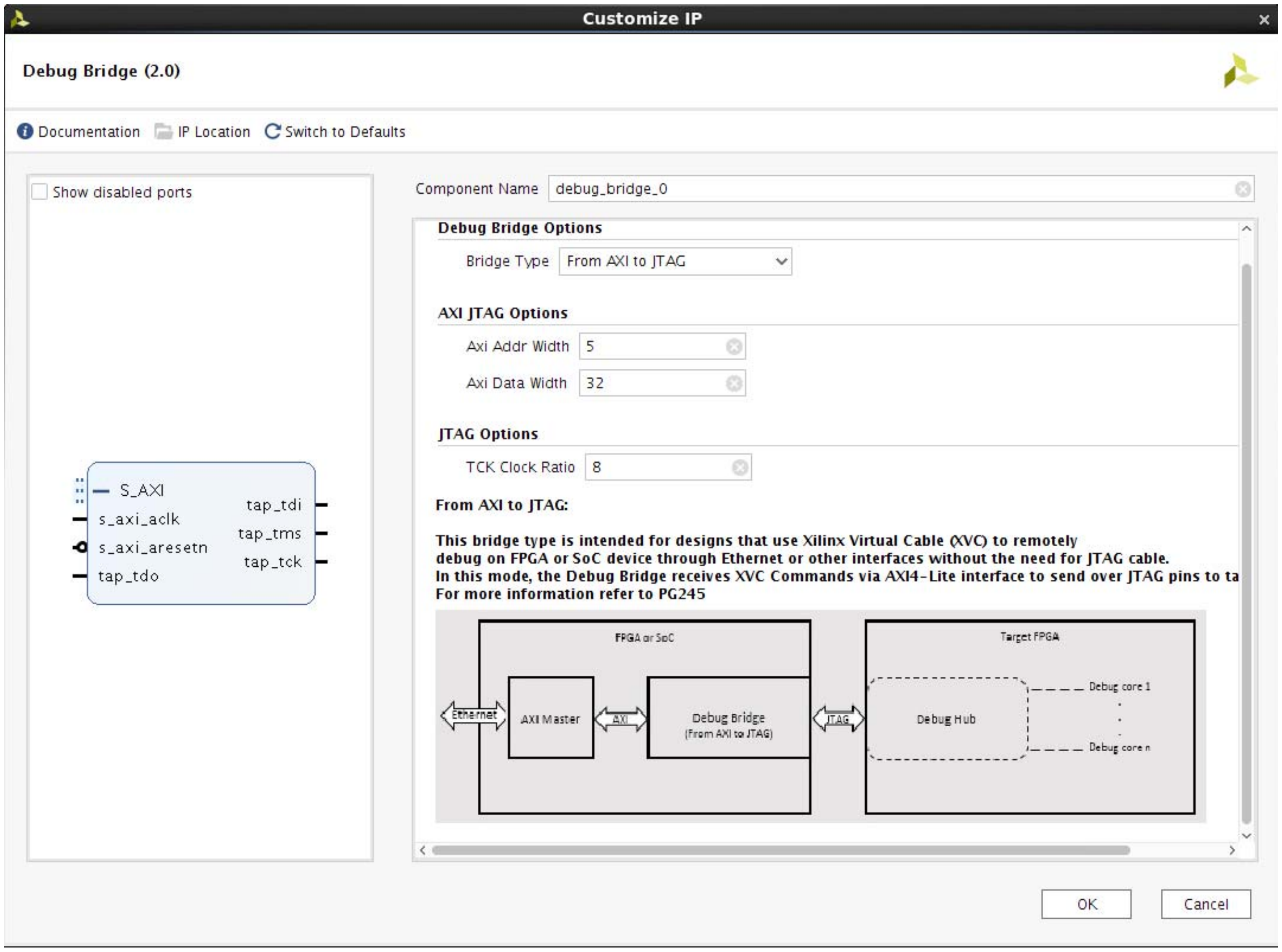
Figure 4-2: **Debug Bridge Customize IP – From AXI4 to BSCAN Mode**

- **AXI JTAG Options** – These options show the address width, data width, and clock ratio settings.
  - **AXI Addr Width** – This parameter sets the address width of the AXI4 interface.

- **AXI Data Width** – This parameter sets the data width of the AXI4 interface.

- **Soft BSCAN Options** – This option sets the User Scan Chain.
  - **User Scan Chain** – This parameter sets the JTAG_CHAIN for soft BSCAN.
- **JTAG Options** – These options are used for generating Soft TAP clock.
  - **TCK Clock Ratio** – This parameter specifies the divider value to generate TCK clock for Serial (BSCAN) communication.

Figure 4-3 shows the Debug Bridge in **From_AXI_to_JTAG** mode.



Component Name debug_bridge_0

Debug Bridge Options

Bridge Type From AXI to JTAG

AXI JTAG Options

Axi Addr Width 5

Axi Data Width 32

JTAG Options

TCK Clock Ratio 8

From AXI to JTAG:

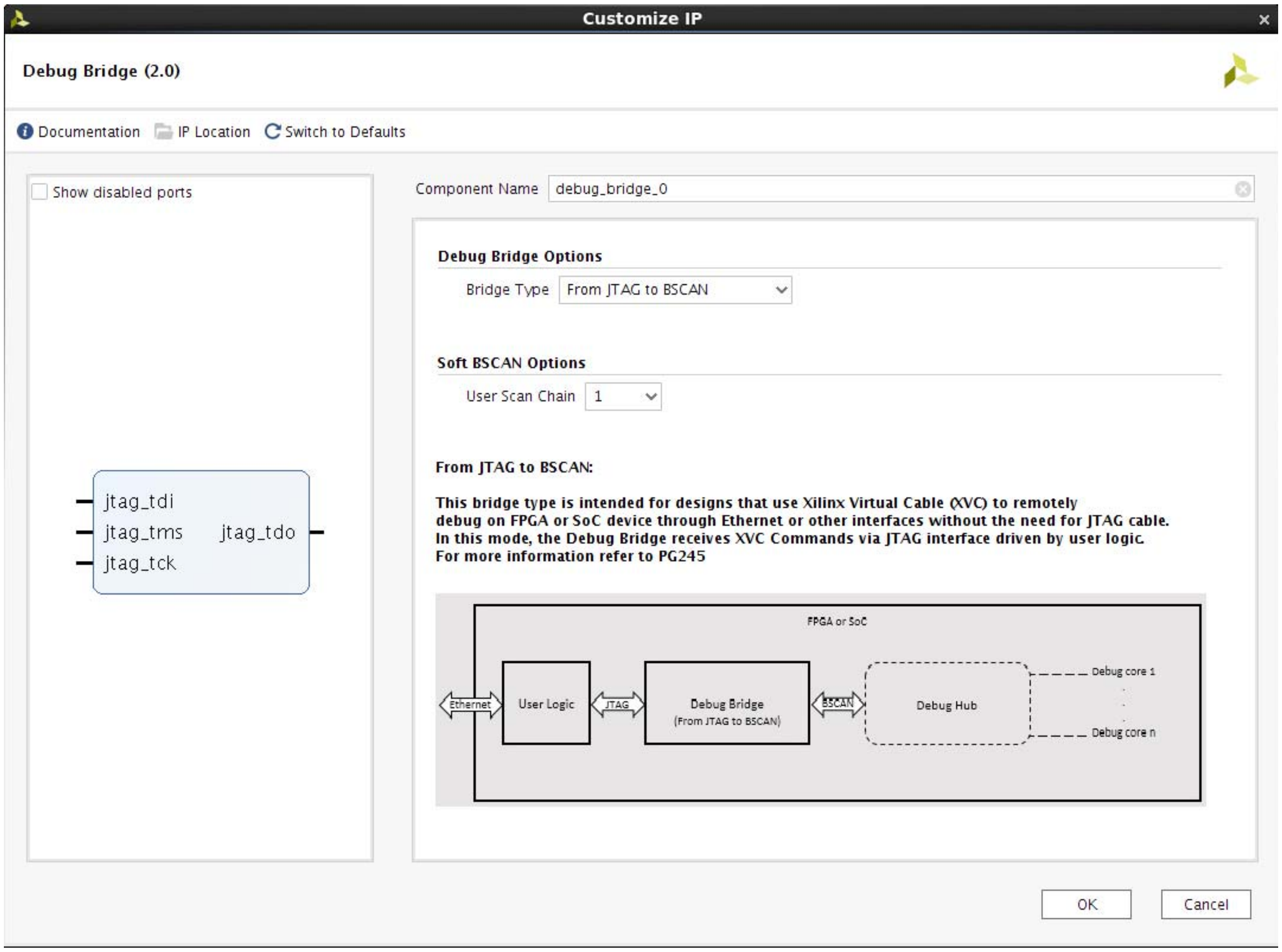
This bridge type is intended for designs that use Xilinx Virtual Cable (XVC) to remotely
debug on FPGA or SoC device through Ethernet or other interfaces without the need for JTAG cable.
In this mode, the Debug Bridge receives XVC Commands via AXI4-Lite interface to send over JTAG pins to ta
For more information refer to PG245

Figure 4-3: **Debug Bridge Customize IP – From AXI4 to JTAG Mode**

- **AXI JTAG Options** – These options show the address width, data width, and clock ratio settings.
  - **AXI Addr Width** – This parameter sets the address width of the AXI4 interface.
  - **AXI Data Width** – This parameter sets the data width of the AXI4 interface.
- **JTAG Options** – These options are used for generating Soft TAP clock.

  - **TCK Clock Ratio** – This parameter specifies the divider value to generate TCK clock for Serial (BSCAN) communication.

Figure 4-4 shows the Debug Bridge in **From_JTAG_to_BSCAN** mode.



Component Name: debug_bridge_0

Debug Bridge Options

Bridge Type: From JTAG to BSCAN

Soft BSCAN Options

User Scan Chain: 1

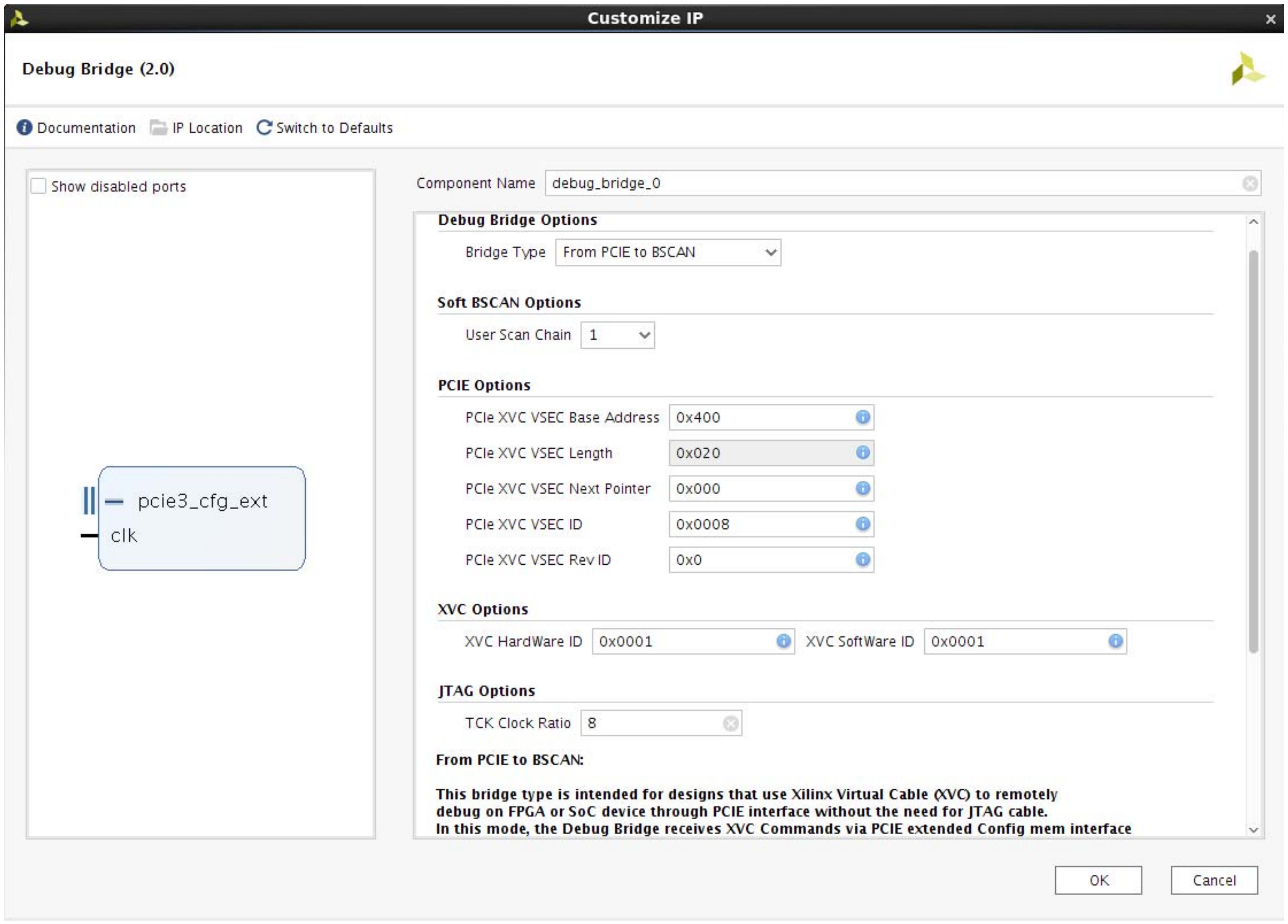
From JTAG to BSCAN:

This bridge type is intended for designs that use Xilinx Virtual Cable (XVC) to remotely debug on FPGA or SoC device through Ethernet or other interfaces without the need for JTAG cable. In this mode, the Debug Bridge receives XVC Commands via JTAG interface driven by user logic. For more information refer to PG245

Figure 4-4: **Debug Bridge Customize IP – From JTAG to BSCAN Mode**

- **Soft BSCAN Options** – This option sets the User Scan Chain.
  - **User Scan Chain** – This parameter sets the JTAG_CHAIN for soft BSCAN.

Figure 4-5 shows the Debug Bridge in **From_PCIE_to_BSCAN** mode.



Component Name: debug_bridge_0

Debug Bridge Options

Bridge Type: From PCIE to BSCAN

Soft BSCAN Options

User Scan Chain: 1

PCIE Options

PCIe XVC VSEC Base Address: 0x400

PCIe XVC VSEC Length: 0x020

PCIe XVC VSEC Next Pointer: 0x000

PCIe XVC VSEC ID: 0x0008

PCIe XVC VSEC Rev ID: 0x0

XVC Options

XVC HardWare ID: 0x0001 XVC SoftWare ID: 0x0001

JTAG Options

TCK Clock Ratio: 8

From PCIE to BSCAN:

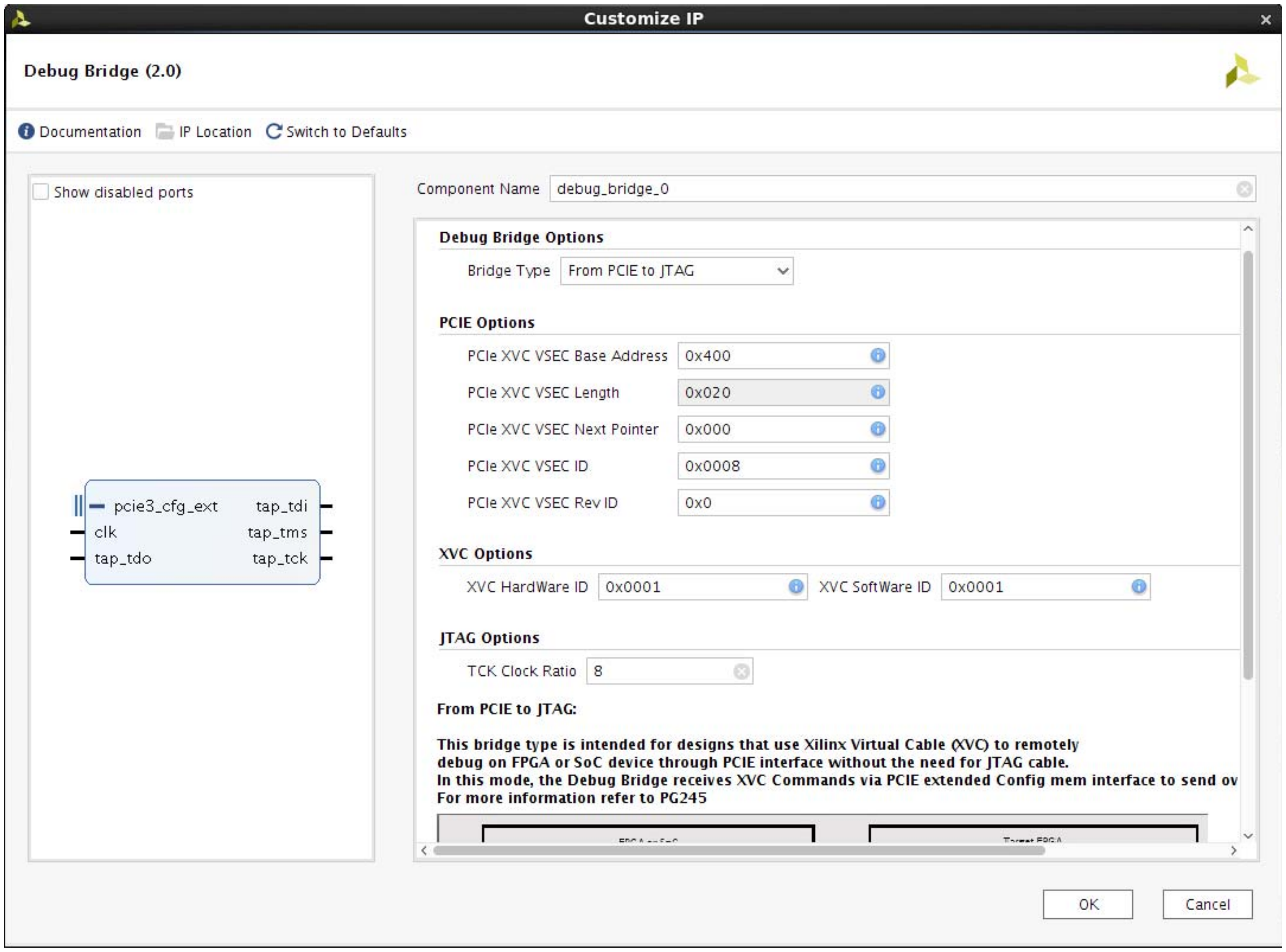
**This bridge type is intended for designs that use Xilinx Virtual Cable (XVC) to remotely debug on FPGA or SoC device through PCIE interface without the need for JTAG cable.
In this mode, the Debug Bridge receives XVC Commands via PCIE extended Config mem interface**

Figure 4-5: **Debug Bridge Customize IP – From PCIE to BSCAN Mode**

- **Soft BSCAN Options** – This option sets the User Scan Chain.
    - **User Scan Chain** – This parameter sets the JTAG_CHAIN for soft BSCAN.
- **PCIE Options** – These options show the PCIe VSEC parameters required to set PCIe master to communicate with the Debug Bridge over the Extended config space.
    - **PCIe XVC VSEC Base Address** – XVC VSEC Extended config address range, usually ranging from 0x3FF to 0x400.
    - **PCIe XVC VSEC Length** – XVC VSEC length in bytes, this is a fixed length for XVC VSEC.
    - **PCIe XVC VSEC Next Pointer** – XVC VSEC next capability offset pointer. This is set to the address offset of the next PCIe extended capability or set to 0x000 to terminate the extended capability chain.

  - **PCIe XVC VSEC ID** – XVC VSEC ID identifies the XVC extended capability. This value should be validated by the Host software to identify the XVC extended capability. This value should not be modified when using the Xilinx Vendor ID of 0x10EE for the corresponding PCIe IP configuration.
  - **PCIe XVC VSEC Rev ID** – XVC VSEC Rev ID identifies the revision of the XVC extended capability. This value should be validated by the Host software to identify the XVC extended capability revision. This value should not be modified when using the Xilinx Vendor ID of 0x10EE for the corresponding PCIe IP configuration.
- **XVC Options** – These options are for the PCIe driver and runtime tool to identify the XVC versions.
  - **XVC HardWare ID** – Hardware ID value.
  - **XVC SoftWare ID** – Software ID value.
- **JTAG Options** – These options are used for controlling Soft TAP clock.
  - **TCK Clock Ratio** – This parameter specifies the divider value to generate TCK clock for Serial (BSCAN) communication.

Figure 4-6 shows the Debug Bridge in **From_PCIE_to_JTAG** mode.



Component Name debug_bridge_0

Debug Bridge Options

Bridge Type: From PCIE to JTAG

PCIE Options

PCIe XVC VSEC Base Address	0x400
PCIe XVC VSEC Length	0x020
PCIe XVC VSEC Next Pointer	0x000
PCIe XVC VSEC ID	0x0008
PCIe XVC VSEC Rev ID	0x0

XVC Options

XVC HardWare ID 0x0001 XVC SoftWare ID 0x0001

JTAG Options

TCK Clock Ratio 8

From PCIE to JTAG:

This bridge type is intended for designs that use Xilinx Virtual Cable (XVC) to remotely debug on FPGA or SoC device through PCIE interface without the need for JTAG cable. In this mode, the Debug Bridge receives XVC Commands via PCIE extended Config mem interface to send ov For more information refer to PG245

Figure 4-6: **Debug Bridge Customize IP – From PCIE to JTAG Mode**

For PCIE Options, XVC Options, and JTAG Options, the descriptions for the **From_PCIE_to_JTAG** mode is the same as **From_PCIE_to_BSCAN** mode.

User Parameters

Table 4-1 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/Value[1]	User Parameter/Value[1]	Default Value
Bridge Type	C_DEBUG_MODE	From_BSCAN_to_DebugHub
Enable Clock Divider	C_ENABLE_CLK_DIVIDER	FALSE
Clock Frequency (In Hertz)	C_CLK_INPUT_FREQ_HZ	300000000
AXI Addr Width	C_S_AXI_ADDR_WIDTH	5

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship *(Cont'd)***

Vivado IDE Parameter/Value[1]	User Parameter/Value[1]	Default Value
AXI Data Width	C_S_AXI_DATA_WIDTH	32
Tck Clock Ratio	C_TCK_CLOCK_RATIO	8
User Scan Chain	C_USER_SCAN_CHAIN	1
Enable BSCANID Vector	C_EN_BSCANID_VEC	FALSE
PCIe XVC VSEC Base Address	C_PCIE_EXT_CFG_BASE_ADDR	0x400
PCIe XVC VSEC Length	C_PCIE_EXT_CFG_VSEC_LENGTH	0x020
PCIe XVC VSEC Next Pointer	C_PCIE_EXT_CFG_NEXT_PTR	0x000
PCIe XVC VSEC ID	C_PCIE_EXT_CFG_VSEC_ID	0x0008
PCIe XVC VSEC Rev ID	C_PCIE_EXT_CFG_VSEC_REV_ID	0x0
XVC Hardware ID	C_XVC_HW_ID	0x0001
XVC Software ID	C_XVC_SW_ID	0x0001

Notes:

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

- **Bridge Type** – This parameter configures `debug_bridge` in six different modes: **From_BSCAN_to_DebugHub**, **From_AXI_to_BSCAN**, **From_AXI_to_JTAG**, **From_JTAG_to_BSCAN**, **From_PCIE_to_BSCAN**, and **From_PCIE_to_JTAG**.
  - **From_BSCAN_to_DebugHub** – This mode is only used while instantiating the `debug_bridge` IP in the PR region. While adding this block to the PR region design part, the following pins need to be added as part of the PR boundary and also need to be connected to the `debug_bridge` instance:

Signal	I/O
S_BSCAN	Slave BSCAN Interface
S_BSCAN_VEC	Slave BSCAN Interface
clk	I

    In this mode, there is another `clk` port on `debug_bridge` which needs to be connected to a stable free running clock or one of the clock connected to a debug core in the PR region. There are two sub-parameters in this mode which qualify the `clk` port.
  - **From_AXI_to_BSCAN** – This mode is only used while debugging a design on the same FPGA over the Xilinx Virtual Cable (XVC). This block is connected as an AXI4 slave to an AXI4 master.
  - **From_AXI_to_JTAG** – This mode is only used while debugging a design on a different FPGA over the XVC. This block is connected as an AXI4 slave to an AXI4

master. The following four JTAG ports are connected to I/Os so that they can be connected to JTAG pins of another board.

Signal	I/O
tap_tdo	I
tap_tdi	O
tap_tck	O
tap_tms	O

- **From_JTAG_to_BSCAN** – This mode is only used while debugging over soft BSCAN where the JTAG connection is through regular I/Os. The following pins are connected to regular I/Os to connect to JTAG connector.

Signal	I/O
jtag_tdo	O
jtag_tdi	I
jtag_tck	I
jtag_tms	I

- **From_PCIE_to_BSCAN** – This mode is used to add a Debug Bridge instance in the design with a PCIe master. This mode is a slave connected on the Extended Config interface on the PCIe master to debug cores like ILA, VIO, Memory IP, and JTAG2AXI in the same chip.

  For more information on how to use the PCIe with the XVC, see the *UltraScale Devices Gen3 Integrated Block for PCI Express LogiCORE IP Product Guide* (PG156) [Ref 2].

- **From_PCIE_to_JTAG** – This mode is used to add a Debug Bridge instance in the design with a PCIe master. This mode is a slave connected on the Extended Config interface on the PCIe master while bringing out the JTAG pins out of the FPGA through I/O pins. This mode is mainly used to debug design on another board over XVC.

Signal	I/O
tap_tdo	I
tap_tdi	O
tap_tck	O
tap_tms	O

- **PCIe XVC VSEC Base Address** – XVC VSEC Extended config address range, usually ranging from 0x3FF to 0x400.
- **PCIe XVC VSEC Length** – XVC VSEC length in bytes, this is a fixed length for XVC VSEC.

- **PCIe XVC VSEC Next Pointer** – XVC VSEC next capability offset pointer. This is set to the address offset of the next PCIe extended capability or set to 0x000 to terminate the extended capability chain.
- **PCIe XVC VSEC ID** – XVC VSEC ID identifies the XVC extended capability. This value should be validated by the Host software to identify the XVC extended capability. This value should not be modified when using the Xilinx Vendor ID of 0x10EE for the corresponding PCIe IP configuration.
- **PCIe XVC VSEC Rev ID** – XVC VSEC Rev ID identifies the revision of the XVC extended capability. This value should be validated by the Host software to identify the XVC extended capability revision. This value should not be modified when using the Xilinx Vendor ID of 0x10EE for the corresponding PCIe IP configuration.
- **XVC HardWare ID** – Hardware ID value.
- **XVC SoftWare ID** – Software ID value.
- **Enable BSCANID Vector** – This parameter uses the BSCANID 32-bit signal for backwards compatibility with the previous versions.
- **Enable Clock Divider** – This parameter is checked only when the clock frequency of the signal connected to the `clk` port is >100 MHz. This option divides down the clock frequency of clock at the `clk` port of `debug_bridge` to 100 MHz.
- **Clock Frequency (in Hz)** – This parameter is visible only when **Enable Clock Divider** options is enabled. The value of this parameter should be the frequency of the clock signal connected to the `clk` port of Debug Bridge instance in Hz.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 4].

Constraining the Core

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

Required Constraints

This section is not applicable for this IP core.

Device, Package, and Speed Grade Selections

This IP supports Tandem with Field Updates and Xilinx Virtual Cable in UltraScale+™, UltraScale™, and 7 series devices and packages. Currently, only PCIe is supported in UltraScale+ and UltraScale devices.

Clock Frequencies

This section is not applicable for this IP core.

Clock Management

This section is not applicable for this IP core.

Clock Placement

This section is not applicable for this IP core.

Banking

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

This section is not applicable for this IP core.

Simulation

This IP core does not support simulation.

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 4].

Test Bench

There is no test bench for this IP core release.

Verification, Compliance, and Interoperability

This appendix provides details about how this IP core was tested for compliance with the protocol to which it was designed.

Xilinx® has verified the Debug Bridge core in a proprietary test environment, using an internally developed bus functional model.

Upgrading

This appendix contains information about upgrading to a more recent version of the IP core.

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations between core versions.

Changes from v1.1 to v2.0

Added **From_PCIE_to_BSCAN** and **From_PCIE_to_JTAG** modes with new PCIE Options and XVC Options.

Changes from v1.0 to v1.1

While upgrading the designs with `debug_bridge_v1_0` to `debug_bridge_v1_1`, the following steps need to happen:

1. The Debug Bridge instance in **BSCAN_Primitive** mode in the static region of Tandem Field Updates design needs to be selected.
2. The Debug Bridge instance in **From_BSCAN_to_DebugHub** mode needs to be regenerated and the BSCAN pins crossing the PR boundary has to be expanded per latest version. These BSCAN pins on the Debug Bridge boundary of RM blocks need to ground for inputs and left unconnected for outputs.

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 Debug Bridge, 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 Debug Bridge. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

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

Answer Records

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

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

AR: 66939

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

There are many tools available to address Debug Bridge 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 7].

Additional Resources and Legal Notices

Xilinx Resources

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

Documentation Navigator and Design Hubs

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

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

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

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

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

References

These documents provide supplemental material useful with this product guide:

1. *Vivado Design Suite User Guide: Partial Reconfiguration* (UG909)
2. *UltraScale Devices Gen3 Integrated Block for PCI Express LogiCORE IP Product Guide* (PG156)
3. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
4. *Vivado Design Suite User Guide: Designing with IP* (UG896)
5. *Vivado Design Suite User Guide: Getting Started* (UG910)
6. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
7. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
8. *Vivado Design Suite User Guide: Implementation* (UG904)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/28/2017	2.0	• Updated Resources in IP Facts section. • Updated Resource Utilization section.
04/05/2017	2.0	• Updated IP Facts section. • Updated Overview chapter. • Updated Device Utilization table. • Updated Port Descriptions section. • Updated description and figures in General Options Panel section. • Updated User Parameter section.
10/05/2016	1.1	• Updated IP Facts section. • Updated Overview chapter. • Updated Device Utilization table. • Updated Port Descriptions section. • Updated description and figures in General Options Panel section. • Updated User Parameter section. • Updated Upgrading appendix.
04/06/2016	1.0	Initial Xilinx release.

Please Read: Important Legal Notices