UltraScale Devices Debug Bridge v1.0

LogiCORE IP Product Guide

Vivado Design Suite

PG245 (v1.0) April 6, 2016

XILINX
ALL PROGRAMMABLE™

Table of Contents

Introduction

The Xilinx® LogiCORE™ IP Debug Bridge core is a controller which connects the JTAG interface to debug cores in the design. To debug Tandem with Field Updates designs, the Debug Bridge IP is required to 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.

Note: The current version is for the Tandem with Field Updates solution. General Partial Reconfiguration support is planned for a future release. For more information, see *UltraScale Architecture Gen3 Integrated Block for PCI Express* (PG156) [Ref 1].

Features

- User selectable mode for adding a Debug Bridge instance in each Reconfigurable Module (RM) region which would connect to debug cores like ILA, VIO, Memory IP, and JTAG2AXI.
- User selectable mode for adding a Debug Bridge that contains a BSCAN primitive to the Static region of a design.

LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family(1)	UltraScale™ Devices
Supported User Interfaces	IEEE Standard 1149.1 – JTAG
Resources	See Table 2-1.
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 PR boundary. Another instance of Debug Bridge needs to be instantiated in the Static region with a BSCAN primitive and the output of the latter needs to be connected to BSCAN pins at the PR boundary. These two connected modules, one in the RM and one in Static, enable the Tandem with Field Updates designs to be debugged using the Xilinx debug cores (ILA, VIO, Memory IP, and JTAG2AXI).

Figure 1-1 and Figure 1-2 show the Debug Bridge IP block configured in two modes.

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

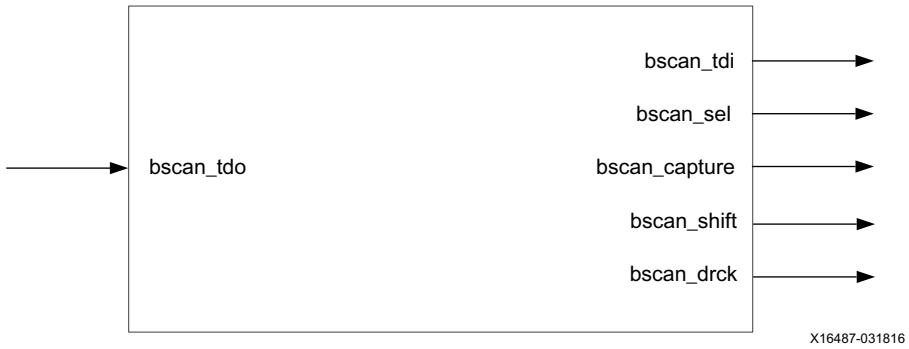

Figure 1-2: **Debug Bridge Configured with BSCAN_primitive Mode**

Figure 1-3 shows the Debug Bridge connections in Static and Reconfigurable regions.

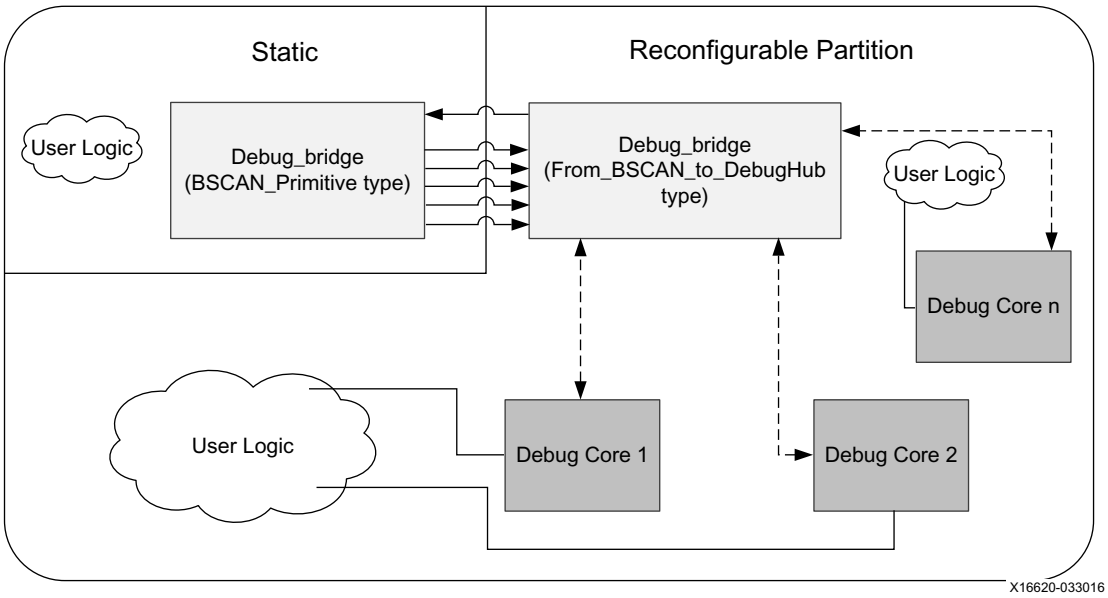

Figure 1-3: **Debug Bridge Connections in Static and Reconfigurable Partition Regions**

Licensing and Ordering Information

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 this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information about 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.

Resource Utilization

UltraScale Devices

Table 2-1 provides approximate resource counts for the various core options on UltraScale devices.

Table 2-1: **Device Utilization – XCZU9EG-FFVB1156**

Configuration	Slices	LUTs	Flip-Flops	Block RAMs	BSCAN
From_BSCAN_to_DebugHub	0	0	5	0	0
BSCAN_Primitive	0	0	0	0	1

The resource usage results do not include the characterization registers and represent the true logic used by the core. LUT counts include SRL16s or SRL32s.

Resources required for the Debug Bridge core have been estimated for the UltraScale devices (Table 2-1). These values were generated using the Vivado® IP catalog. They are derived from post-synthesis reports, and might change during implementation.

Start by choosing the device, maximum frame size, and minimum block size of the core. If using the Streaming Video or pCore interfaces, add the corresponding resources.

Port Descriptions

Table 2-2 and Table 2-3 show the signals for From BSCAN to DebugHub and BSCAN Primitive modes.

Table 2-2: **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.
drck	I	This port needs to connect to the DRCK pin of BSCAN primitive.
shift	I	This port needs to connect to the SHIFT pin of BSCAN primitive.
tdi	I	This port needs to connect to the TDI pin of BSCAN primitive.
update	I	This port needs to connect to the UPDATE pin of BSCAN primitive.
tdo	O	This port needs to connect to the TDO pin of BSCAN primitive.
sel	I	This port needs to connect to the SEL pin of BSCAN primitive.

Table 2-3: **BSCAN Primitive Mode**

Signal	I/O	Description
bscan_tdo	I	This port needs to connect to the tdo pin of debug_bridge in BSCAN to DebugHub Mode.
bscan_tdi	O	This port needs to connect to the tdi pin of debug_bridge in BSCAN to DebugHub Mode.
bscan_sel	O	This port needs to connect to the sel pin of debug_bridge in BSCAN to DebugHub Mode.
bscan_shift	O	This port needs to connect to the shift pin of debug_bridge in BSCAN to DebugHub Mode.
bscan_update	O	This port needs to connect to the update pin of debug_bridge in BSCAN to DebugHub Mode.
bscan_drck	O	This port needs to connect to the drck pin of debug_bridge in BSCAN to DebugHub Mode.

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 2]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 3]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 4]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 5]

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 2] 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 3] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 4].

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.

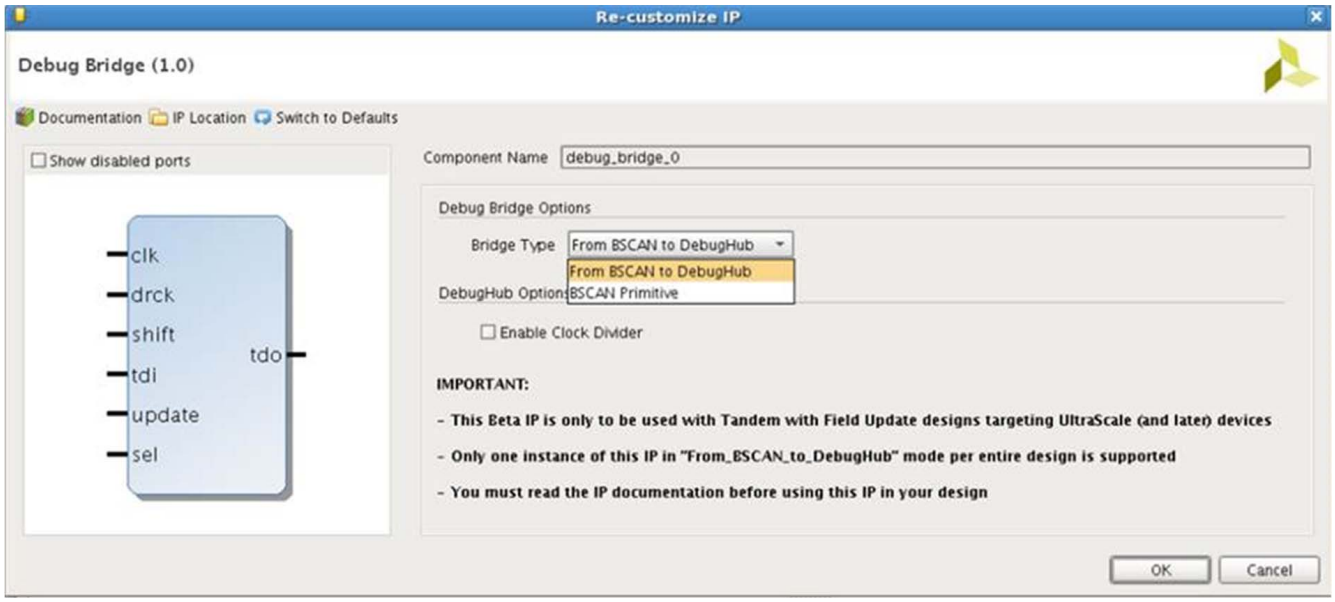

Figure 4-1: **Debug Bridge Customize IP**

- **Component Name** – Use this text field to provide a unique module name for the Debug Bridge core.
- **Bridge Type** – This option is used to switch between two modes, **From BSCAN to DebugHub** and **BSCAN Primitive**.
- **Enable Clock Divider** – This option is used to divide down the clock frequency on `clk` port to 100 MHz. This option is only enabled for **From BSCAN to DebugHub** mode.
- **User Scan Chain** – This option is used to select JTAG_CHAIN on BSCAN primitive. This option is only enabled for **BSCAN Primitive** 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
User Scan Chain	C_USER_SCAN_CHAIN	1

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 two different modes. One is **From_BSCAN_to_DebugHub** and the other is **BSCAN_Primitive**.
- **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:
  - `tdo` – Output
  - `tdi` – Input
  - `sel` – Input
  - `shift` – Input
  - `capture` – Input
  - `drck` – Input

  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.

  - **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 would divide 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.
- **BSCAN_Primitive** – This mode is only used while instantiating the `debug_bridge` IP in the Static region of the design. The block pins need to be connected to the BSCAN pins defined on the PR boundary.
  - **User Scan Chain** – This option sets the JTAG_CHAIN for BSCAN Primitive.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 3].

Constraining the Core

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

Required Constraints

The Debug Bridge core includes an XDC file if you select **BSCAN Primitive** option for the **Bridge Type** parameter. It contains appropriate clock constraints for the BSCAN primitive. It is also expected that the clock signal connected to the `clk` input port of the Debug Bridge is properly constrained in your design.

Device, Package, and Speed Grade Selections

This IP is only supported for UltraScale™ and UltraScale+™ devices and packages. Currently, only UltraScale devices are supported for the Tandem with Field Updates solution.

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 3].

Test Bench

This chapter contains information about the test bench provided in the Vivado® Design Suite.

There is no test bench for this IP core release.

Appendix A

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.

Migrating and Upgrading

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

Upgrading in the Vivado Design Suite

This section is not applicable for the first release of the core.

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 6].

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

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/06/2016	1.0	Initial Xilinx release.

Please Read: Important Legal Notices