AXI to APB Bridge v3.0

LogiCORE IP Product Guide

Vivado Design Suite

PG073 November 18, 2015

XILINX
ALL PROGRAMMABLE™

Table of Contents

Introduction

The LogiCORE™ IP Advanced Microcontroller Bus Architecture (AMBA®) AXI to Advanced Peripheral Bus (APB) Bridge core translates AXI4-Lite transactions into APB transactions. It functions as a slave on the AXI4-Lite interface and as a master on the APB interface. The AXI to APB Bridge is used primarily to connect the APB slaves with AXI masters.

Features

The Xilinx AXI to APB Bridge core is a soft IP core with these features:

- AXI interface is based on the AXI4-Lite specification
- APB interface is based on the APB3 specification, supports optional APB4 selection
- Connects as a 32-bit slave on a 32-bit AXI4-Lite interface
- Connects as a 32-bit master on a 32-bit APB3/APB4 interface
- Supports optional data phase timeout

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ Families, Zynq®-7000, 7 Series, UltraScale™ Architecture
Supported User Interfaces	AXI4-Lite, APB3, APB4
Resources	See Table 2-1
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	N/A
Simulation Model	None
Supported S/W Driver	N/A
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite Vivado
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. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The main function of the LogiCORE™ IP AXI to Advanced Peripheral Bus (APB) Bridge core is to connect APB slaves to AXI masters. It translates AXI4-Lite transactions into APB transactions.

Feature Summary

The 32-bit AXI4-Lite interface on the AXI to APB Bridge core is based on the *AMBA® AXI and ACE Protocol Specification v2.0* [Ref 1]. The core functions as a 32-bit slave on this interface.

The 32-bit APB interface of the core is based on the AP3 interface as described in the *AMBA APB Protocol Specification v2.0* [Ref 1]. The core supports the optional APB4 interface as well. The core functions as a 32-bit master on the APB3/APB4 interface.

The AXI to APB Bridge core supports a 1:1 (AXI:APB) synchronous clock ratio as well as data phase timeout.

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 on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

The AXI to APB Bridge core translates AXI4-Lite transactions into APB transactions. The bridge functions as a slave on the AXI4-Lite interface and as a master on the APB interface.

The AXI to APB Bridge core block diagram, shown in Figure 2-1, is described in subsequent sections.

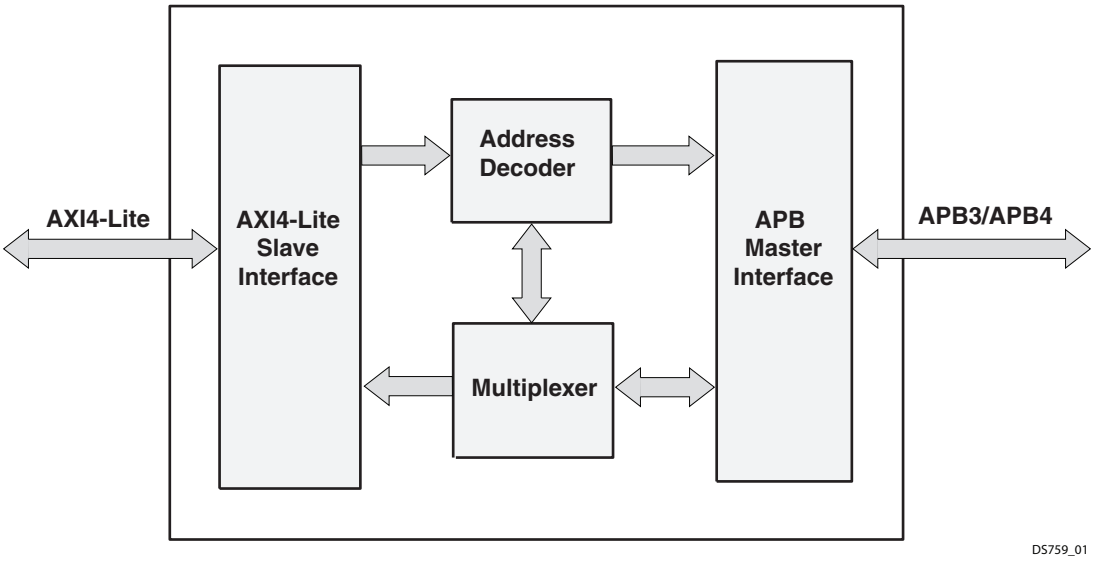

Figure 2-1: **AXI to APB Bridge Core Block Diagram**

AXI4-Lite Slave Interface

The AXI4-Lite Slave Interface module provides a bidirectional slave interface to the AXI interface. The AXI address and data bus widths are always fixed at 32 bits. When both write and read transfers are simultaneously requested on the AXI4-Lite interface, the read request has higher priority than the write request. This module also contains the data phase timeout logic for generating a SLVERR response on the AXI interface when an APB slave does not respond.

APB Master Interface

The APB Master module provides the APB master interface on the APB. This interface can be APB3 or APB4, which can be selected by setting the generic for APB protocol. When it is set to APB4, the `m_apb_pstrb` and `m_apb_pprot` signals are driven at the APB interface. The APB address and data bus widths are fixed at 32 bits.

Standards Compliance

The AXI to APB Bridge core is based on the AMBA® AXI4-Lite specification [Ref 1] and the APB3/APB4 specification [Ref 1].

Resource Utilization

Because the AXI to APB Bridge core is used with other designs in the FPGA, the resource utilization and timing numbers reported in this section are estimates only.

The AXI to APB Bridge core resource utilization benchmarks for several parameter combinations, measured on 7 series FPGAs, are shown in Table 2-1.

Table 2-1: **Performance and Resource Utilization Benchmarks 7 Series FPGAs**

Parameter Values (other parameters at default value)			Device Resources			Performance
Number of slaves	APB Protocol	Timeout value	Slices	Slice Flip-Flops	LUTs	F_{MAX} (MHz)
1	APB4	64	119	364	286	200
4	APB4	64	136	370	372	200
8	APB4	64	174	386	475	200
16	APB3	64	181	402	549	200

Note: Performance and utilization numbers for UltraScale architecture-based devices and Zynq®-7000 devices are expected to be similar to those for 7 series devices.

Port Descriptions

Table 2-2 shows the I/O signals for the AXI to APB Bridge core.

Table 2-2: **I/O Signal Description**

Signal Name	Interface	I/O	Initial State	Description
s_axi_aclk	Clock	I	-	AXI clock. This signal in available only in legacy and XIP modes.
s_axi_aresetn	Reset	I	-	AXI reset. This signal in available only in legacy and XIP modes.
AXI Interface Signals				
s_axi_*		I/O		See the *AXI Reference Guide* (UG761) [Ref 2]. This signal is available only in legacy and XIP modes.
APB Signals				
m_apb_paddr[31:0]	APB	O	0	Address. This is the APB address bus and is fixed to 32 bit.
m_apb_pprot[2:0](1)	APB	O	0	Protection type. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access.
m_apb_psel	APB	O	0	Select. The AXI to APB Bridge core generates this signal to each peripheral bus slave. It indicates that the slave device is selected and that a data transfer is required. There is a `m_apb_psel` signal for each slave. Port width depends on the number of slave connected to the bridge.
m_apb_penable	APB	O	0	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
m_apb_pwrite	APB	O	0	Direction. This signal indicates an APB write access when High and an APB read access when Low.
m_apb_pwdata[31:0]	APB	O	0	Write data. This bus is driven by the AXI to APB Bridge core during write cycles when `m_apb_pwrite` is High. This bus is fixed to 32-bits wide.
m_apb_pstrb[3:0](1)	APB	O	0	Write strobes. This signal indicates which byte lanes to update during a write transfer. Write strobes must not be active during a read transfer.

Table 2-2: **I/O Signal Description *(Cont'd)***

Signal Name	Interface	I/O	Initial State	Description
m_apb_pready	APB	I	-	Ready. The APB slave uses this signal to extend an APB transfer. The port width depends on the number of slave interfaces created.
m_apb_prdata[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed to 32-bits wide.
m_apb_prdata2[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata3[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata4[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata5[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata6[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata7[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata8[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata9[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata10[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata11[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata12[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata13[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.

Table 2-2: **I/O Signal Description *(Cont'd)***

Signal Name	Interface	I/O	Initial State	Description
m_apb_prdata14[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata15[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_prdata16[31:0]	APB	I	-	Read Data. The selected slave drives this bus during read cycles when `m_apb_pwrite` is Low. This bus is fixed at 32-bits wide.
m_apb_pslverr[c_apb_num_slaves-1:0]	APB	I	-	This signal indicates a transfer failure.

Notes:

1. This signal is only used when APB protocol is set to APB4.

Designing with the Core

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

General Design Guidelines

Memory Mapping

The AXI memory map and the APB memory map are one single complete 32-bit (4 GB) memory space. The AXI to APB Bridge core does not modify the address for APB; hence, the address that is presented on the APB is exactly as received on the AXI interface.

Read and Write Ordering

When a read request and a write request are issued simultaneously (`s_axi_awvalid`/ `s_axi_wvalid` and `s_axi_arvalid` are asserted High) from the AXI4-Lite interface, the AXI to APB Bridge core gives priority to the read request over the write request. When both write and read requests are valid, the write request is initiated on the APB after the read is requested on the APB.

AXI Response Signaling

The AXI Slave interface does not support exclusive read or write access. An `exokay` response is never seen from the AXI interface.

Endianness

Both AXI and APB are little-endian.

Address/Data Translation

No address/data translation/conversion from AXI4-Lite to APB takes place inside AXI to APB Bridge core. The write/read address from AXI4-Lite is passed to APB address. AXI4-Lite write data is passed on to APB and APB read data is passed on to AXI4-Lite read data.

APB4 Operation

When the APB protocol generic is set to apb4, the AXI to APB Bridge core drives `m_apb_pstrb` and `m_apb_pprot` signals. `s_axi_wstrb` is passed to `m_apb_pstrb` during write transfers. `s_axi_arprot` is assigned to `m_apb_pprot` during a read transfer, and `s_axi_awprot` is assigned to `m_apb_pprot` during a write transfer.

Bridge Error Conditions

`m_apb_pslverr` on the APB results in SLVERR on the AXI4-Lite interface. The AXI to APB Bridge core never generates DECERR. In case of timeout, where there is no response from the APB slave, the AXI-Lite interface generates SLVERR. For more information about timeouts, see Basic, page 15.

Bridge Timeout Condition

A data phase timeout is implemented in the AXI to APB Bridge core, when `c_dphase_timeout` is not equal to 0. When a request is issued from the AXI interface, the AXI to APB Bridge core translates this request into a corresponding APB transfer. If there is no response to the request by the APB slave (`m_apb_pready` is not asserted), the core waits for the number of clock cycles defined in the timeout generic and then responds to AXI with SLVERR response (and drives zeroes on `s_axi_rdata` during the read transfer). For more information about timeouts, see Basic, page 15.

Clocking

The AXI to APB Bridge core is a synchronous design and uses `s_axi_aclk` at both AXI and APB interfaces.

Resets

`s_axi_aresetn` is a synchronous, active-Low reset input that resets the AXI to APB Bridge core upon assertion. The `s_axi_aresetn` signal is also used to reset the APB interface.

Timing Diagram

The timing diagram shown in Table 3-1 illustrates the AXI to APB Bridge core operation for various read and write transfers. This diagram shows that when both write and read requests are active, the read request is given higher priority.

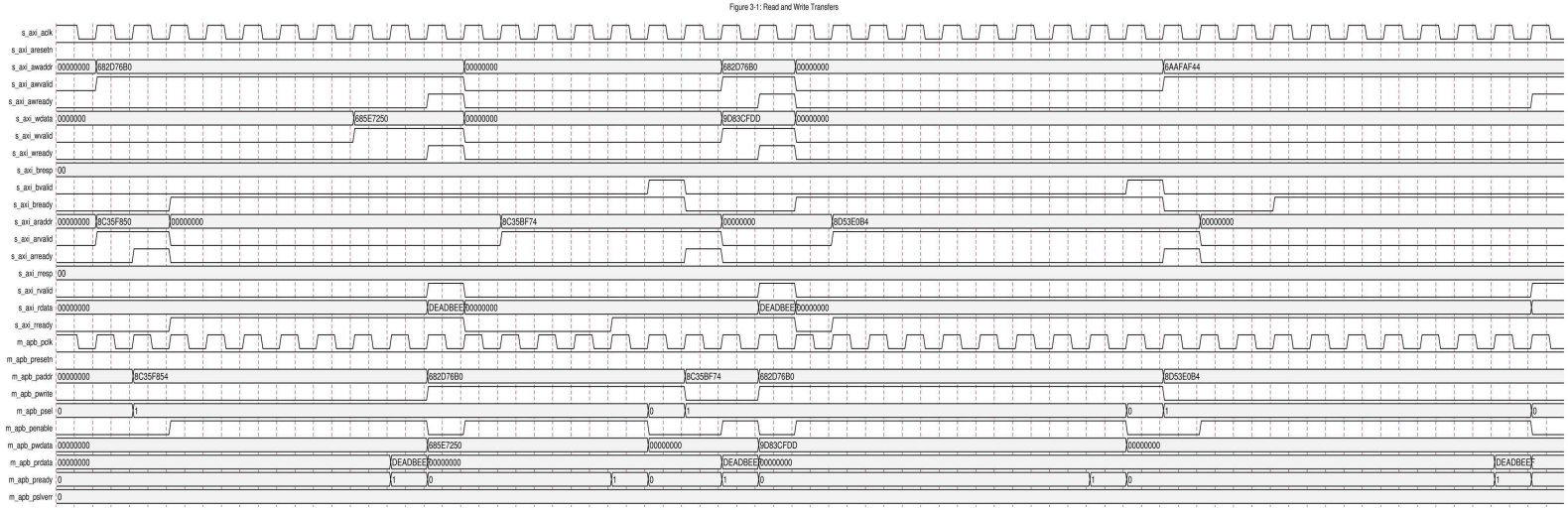

Figure 3-1: **AXI to APB Bridge Timing**

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 chapter includes information on using the Vivado Design Suite to customize and generate the core.

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 you can run the `validate_bd_design` command in the Tcl console.

Vivado Integrated Design Environment

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**.
2. In the Vivado IP catalog, expand **AXI_Infrastructure** in the View by Function pane.
3. Select **AXI APB Bridge** in the IP catalog.

4. Double-click the 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: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). This layout might vary from the current version.

The Customize IP dialog box for the AXI to APB Bridge (Figure 4-1) has two pages to configure the core.

Basic

The Basic page consists of configurations such as the number of APB slaves, APB protocol, and timeout value. See Figure 4-1.

In a timeout condition, a counter decrements during an active APB operation. A data acknowledge from the target address space forces the counter to reload. If the APB does not respond and generate `apb_ready` within the number of clock cycles, AXI generates a `ready` so that it does not hang.

IMPORTANT: *Each slave base address/ high address must be configured in the Slave Addresses page.*

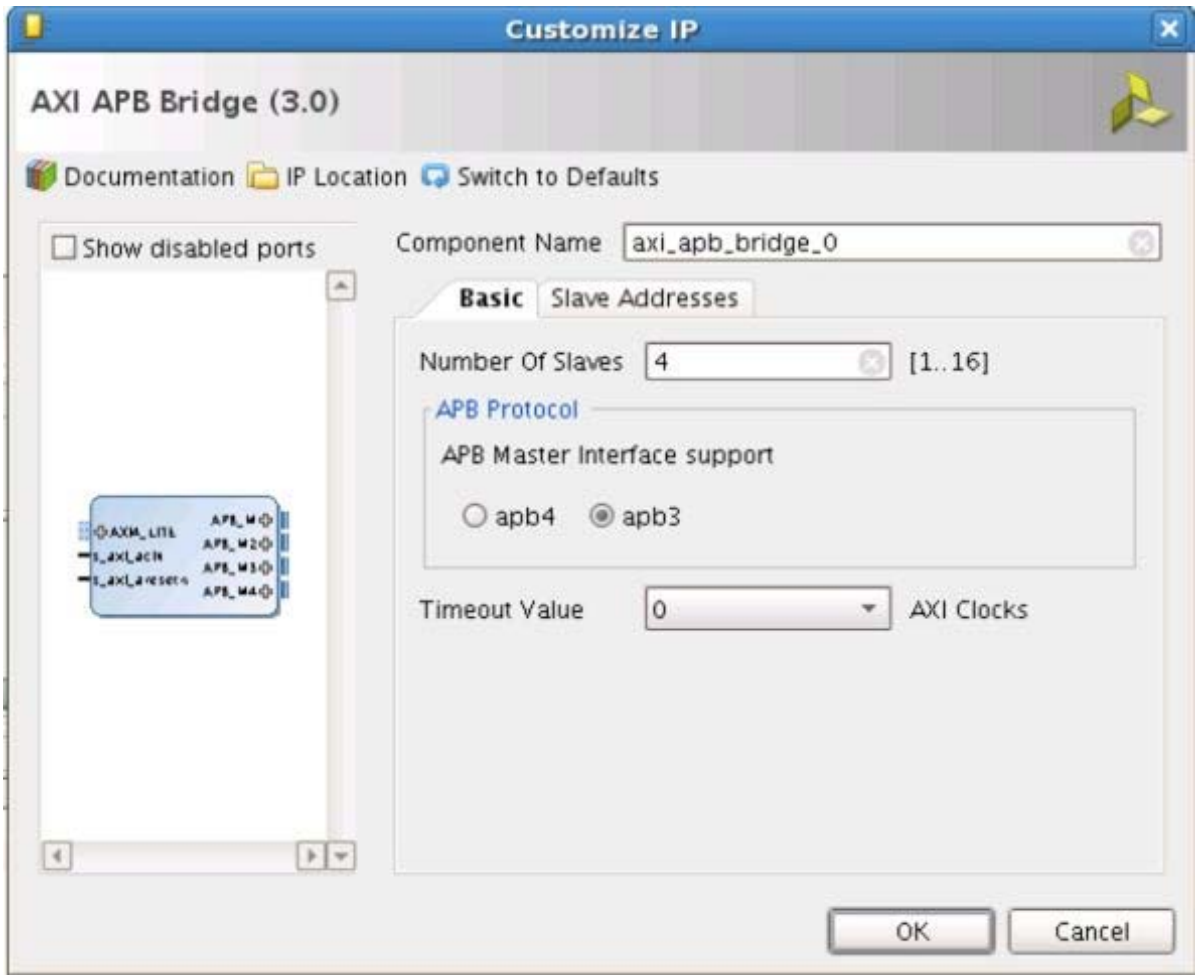

Figure 4-1: **AXI APB Bridge Core Basic Configuration**

Slave Addresses

The Slave Addresses tab contains the APB slave base addresses and high addresses.

Note: *In IP Integrator, all base and high addresses of all slaves are greyed out and auto-updated when a slave is connected.*

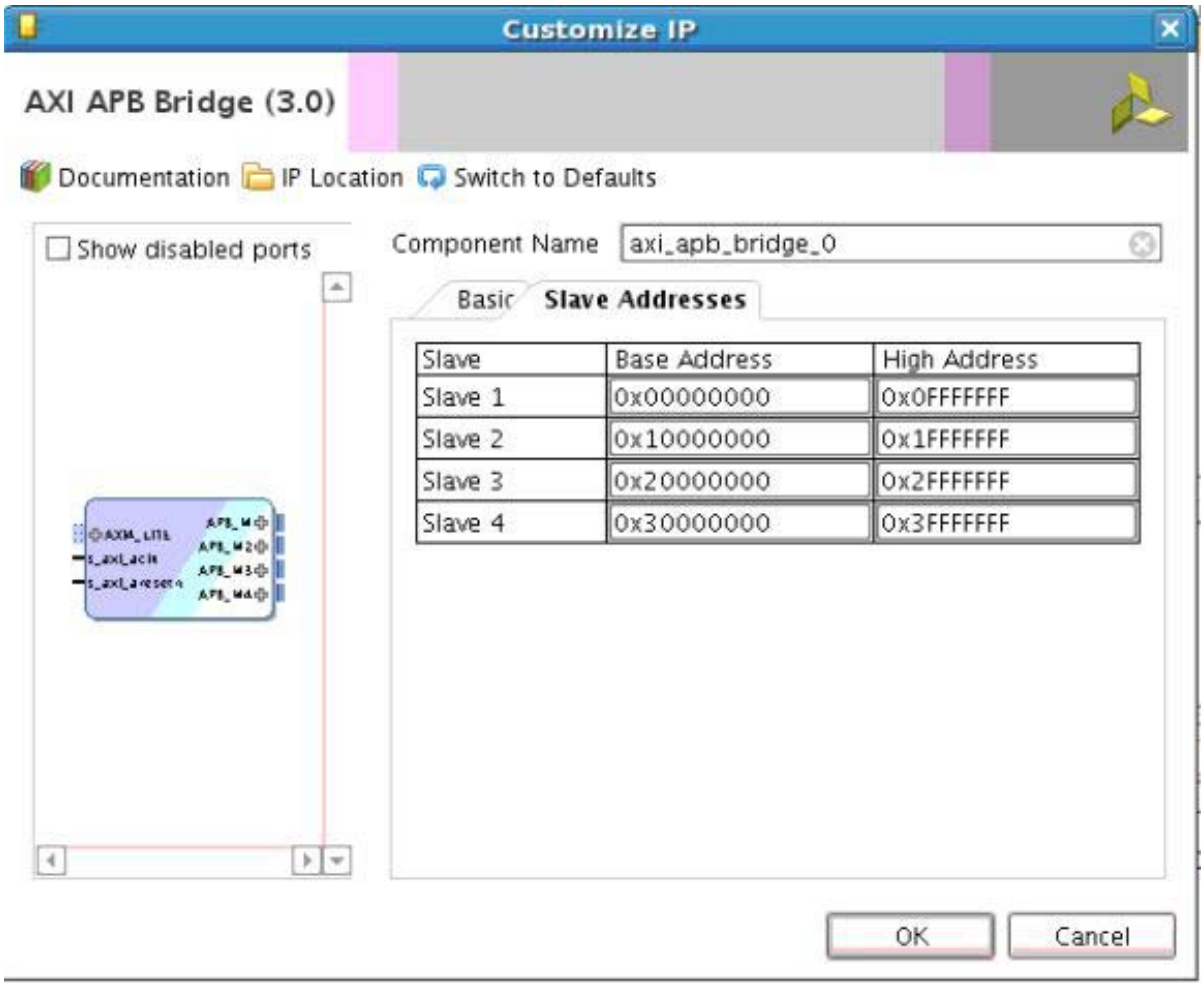

Figure 4-2: **AXI APB Bridge Core Slave Addresses Configuration**

Output Generation

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

Constraints

The necessary Xilinx design constraints (XDCs) are delivered when the core is generated.

Simulation

For comprehensive information about Vivado Design Suite simulation components, as well as information about using supported third party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 6].

For information about simulating the example design, see Simulating the Example Design.

Synthesis and Implementation

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

For information about synthesizing and implementing the example design, see Implementing the Example Design.

Example Design

This chapter contains information about the example design provided in the Vivado® Design Suite.

Overview

The top module instantiates all components of the core and example design that are needed to implement the design in hardware, as shown in Figure 5-1. This includes clock generator (MMCME2) and example design module with logic for AXI transaction generator and APB transaction checker.

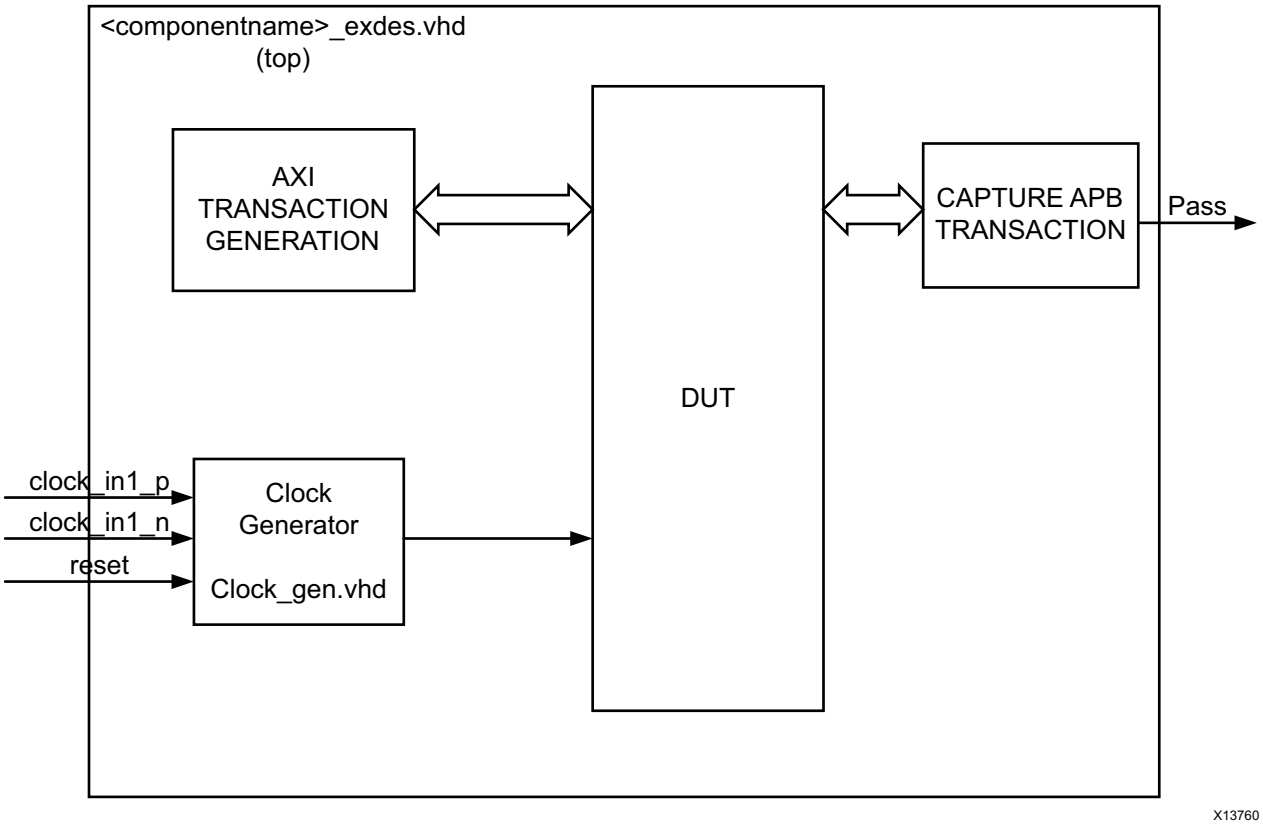

Figure 5-1: **Block Diagram of Example Design**

This example design demonstrates transactions on AXI interfaces of the DUT.

Clock generator: MMCME2 is used to generate the clock for the example design. It generates 100 MHz clock for `s_axi_aclk` of the DUT. The DUT is under reset until MMCME2 is locked.

AXI transaction generation: Handles write and read transactions on the AXI-Lite interface of the bridge.

Capture APB transaction: Serves as the APB slave to the bridge and handles write and read transactions from the AXI interface of the bridge.

Implementing the Example Design

After following the steps described in Chapter 4, Customizing and Generating the Core, implement the example design as follows:

1. Right-click the core in the Hierarchy window, and select **Open IP Example Design**.
2. A new window opens where you can specify a directory for the example design. Select a new directory, or keep the default directory.
3. A new project is automatically created in the selected directory and it is opened in a new window.
4. Uncomment the IO constraints settings in `<component_name>_exdes.xdc` specified in Table 5-3.
5. In the Flow Navigator (left side pane), click **Run Implementation** and follow the directions.
6. Note that GPIO_LED_7 on the KC705 board glows when the test case in the example design has passed. For more information, see the *KC705 Evaluation Board for the Kintex-7 FPGA User Guide* (UG810) [Ref 7].

Example Design Directory Structure

In the current project directory, a new project called `<component_name>_example` is created and the files are delivered in the `<component_name>_example/<component_name>_example.srcs/` directory. This directory and its subdirectories contain all the source files that are required to create the AXI to APB Bridge core example design.

Example Design Directory

Table 5-1 shows the files delivered in the directory.

Table 5-1: **Example Design Directory**

Name	Description
<component_name>_exdes.vhd	Top-level HDL file for the example design.
clock_gen.vhd	Clock generation module for example design.

Simulation Directory

Table 5-2 shows the files delivered in the directory.

Table 5-2: **Simulation Directory**

Name	Description
<component_name>_exdes_tb.vhd	Test Bench for Exdes.

Constraints Directory

Table 5-3 shows the files delivered in the directory.

Table 5-3: ***Constraints*** **Directory**

Name	Description
<component_name>_exdes.xdc	Top-level constraints file for the example design.

Simulating the Example Design

Using the example design delivered as part of the AXI to APB Bridge core, you can quickly simulate and observe the behavior of the core.

The AXI Transaction generation block generates a write and a read transaction to the DUT. The DUT then converts the transaction to APB. The APB Transaction is then captured through the write data and provides the data requested.

Setting up the Simulation

The Xilinx simulation libraries must be mapped into the simulator. To set up the Xilinx simulation models, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 5]. To switch simulators, click **Simulation Settings** in the Flow Navigator (left pane). In the Simulation options list, change **Target Simulator**.

The example design supports functional (behavioral) and post-synthesis simulations. For information about how to run simulation, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 5].

Simulation Results

The simulation script compiles the AXI to APB Bridge core example design, and supporting simulation files. It then runs the simulation and checks to ensure that it completed successfully.

If the test passes, the following message is displayed:

```
Test Completed Successfully
```

If the test fails, the following message is displayed.

```
Test Failed !! Test Timed Out.
```

Test Bench

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

Figure 6-1 shows the test bench for the AXI to APB Bridge core example design. The top level test bench generates a 200 MHz clock and drives the initial reset to the example design.

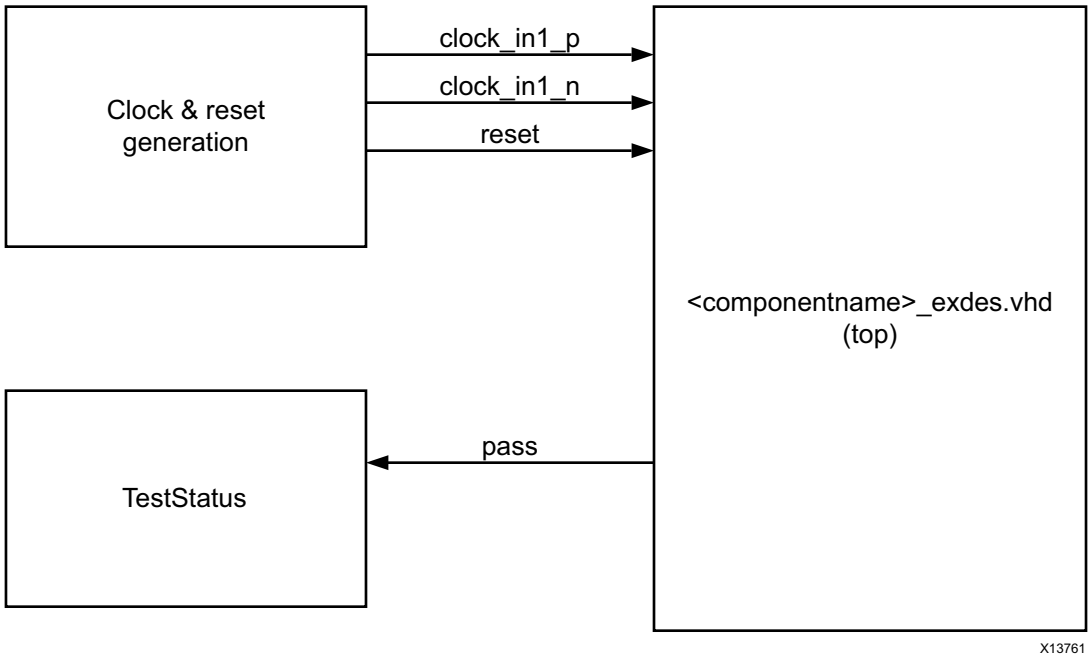

Figure 6-1: **AXI to APB Bridge Core Example Design Test Bench**

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® Design Suite to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impacts to user logic are included.

Migrating to the Vivado Design Suite

For information on migrating to the Vivado® Design Suite, see *Vivado Design Suite Migration Methodology Guide* (UG911) [Ref 8].

Upgrading in the Vivado Design Suite

There are no port or parameter changes.

Debugging

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

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI to ABP Bridge, the Xilinx Support web page (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 AXI to ABP 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 AXI to ABP Bridge

AR: 54439

Technical Support

Xilinx provides technical support in 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 AXI to ABP 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 rin 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 LogiCORE 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 9].

- 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 core clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation or a Vivado lab tools capture that the waveform is correct for accessing the AXI4-Lite interface.

Appendix C

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. ARM® AMBA® documentation (infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0051a/index.html):
   - *AMBA AXI and ACE Protocol Specification, AXI3, AXI4, and AXI4-Lite, v2.0*
   - *AMBA APB Protocol Specification v2.0*
2. *AXI Reference Guide* (UG761)
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. *KC705 Evaluation Board for the Kintex-7 FPGA User Guide* (UG810)
8. *ISE® to Vivado Design Suite Migration Methodology Guide* (UG911)
9. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
10. *7 Series FPGAs Overview* (DS180)
11. *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	3.0	Added support for UltraScale+ families.
10/28/2014	3.0	Removed internal writer note.
04/02/2014	3.0	Updated core to v3.0. Removed m_apb_pclk and m_apb_presetn.
12/18/2013	2.0	Added UltraScale architecture support information.
10/02/2013	2.0	• Updated core to v2.0. • Added Vivado IP integrator support. • Changed signal names to lowercase. • Removed design parameter descriptions. • Added example design and test bench details. • Added Debugging appendix.
07/25/2012	1.0	Initial Xilinx release. This release supports Vivado Design Suite 2012.2 and Xilinx Platform Studio. This document replaces DS788, *LogiCORE IP AXI to APB Bridge Data Sheet*.

Please Read: Important Legal Notices