



AHB-PCI Bridge IP Core

Testsuite Reference Manual

Version 3.8.2 February 2008
Copyright © PLDA 1996-2008

AHB-PCI Bridge IP Core Testsuite

Reference Manual

Documentation Change History

Date	Version Number	Change
February 2008	3.8.2	No change
August 2007	3.8.1	New format, no content changes.
January 2005	3.3.100	Revision of content and new format.

Proprietary Notice

Words and logos marked with ® or ™ are registered trademarks or trademarks owned by PLDA SA. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by PLDA in good faith. This document is provided “as is” with no warranties whatsoever, including any warranty of merchantability, non infringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample.

This document is intended only to assist the reader in the use of the product. PLDA shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product. Nor shall PLDA be liable for infringement of proprietary rights relating to use of information in this document. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Product Status

The information in this document is final content pertaining to the PLDA AHB-PCI Bridge IP Core.

Web Address

<http://www.plda.com>

Table of Contents

Preface	4
	About this Document	4
	Additional Reading	4
	Feedback and Contact Information	5
Chapter 1	Introduction	6
1.1	Overview	6
1.2	Requirements	6
1.3	Directory Structure	7
Chapter 2	Simulating with the Testsuite	8
2.1	Simulation under Modelsim, NCSim, VCS	8
2.1.1	Windows	8
2.1.2	Unix	8
Chapter 3	Understanding the Testsuite Scripts	9
3.1	Simplebridge Scripts	9
3.2	Simplebridge Cardbus Scripts	14
3.3	Hostbridge Scripts	14
3.4	Hostbridge Cardbus Scripts	18

Preface

About this Document

Intended Audience

This document has been written for design managers, system engineers, and designers of ASICs and FPGAs who are evaluating or using the AHB-PCI Bridge Core.

Scope

This document describes the testsuite used to verify PLDA's AHB-PCI Bridge Core.

Typographical Conventions

<i>italic</i>	Highlights important notes or publications
bold	Highlights interface elements.
COURIER NEW	DENOTES TEXT USED IN A CODE EXAMPLE OR A SIGNAL.

Additional Reading

This section lists additional resources from PLDA and third-parties.

PLDA periodically updates its documentation. Please contact PLDA at support@plda.com or check the Web site at <http://www.plda.com> for current versions.

PLDA Publications

Please refer to the following documents for further information:

- *AHB-PCI Bridge IP Core Reference Manual*: The *AHB-PCI Bridge Reference Manual* provides the complete functional description of the PLDA AHB-PCI Bridge Core.
- *AHB-PCI Bridge IP Core: Getting Started*: This document provides information to enable designers to integrate the Core into their design flow as quickly as possible (installing, customizing, integrating, and simulating the Core).
- *AHB Testbench Reference Manual*: Describes the functionality of the AHB Testbench.
- *Build History*: The *Build History* lists changes made in each version and build of the Core.

Other Publications

Please refer to the following documents for information on specification standards:

- *AHB Specification*, revision 2.0 - ARM, May 1999
- *PCI Local Bus Specification*, revision 2.3 - PCI Special Interest Group, April 2002
- *PCI Compliance Checklist*, revision 2.3 - PCI Special Interest Group, April 2002
- *PCI-PCI Bridge Architecture Specification*, revision 1.1 - PCI Special Interest Group, December 1998

Feedback and Contact Information

Feedback about this Document

PLDA welcomes comments and suggestions pertaining to this documentation. Please contact PLDA at support@plda.com and provide the following information:

- the title of the document
- the page number to which your comments refer
- a description of your comments

Contact information

Corporate Headquarters

PLDA
Parc club du golf - Bât. 11a
Rue Guilibert
13856 Aix-en-Provence Cedex 3 - France

Tel: USA +1 408 273 4528 - International +33 442 393 600

Fax: +33 442 394 902

Sales

For sales questions, please contact sales@plda.com.

Technical Support

For technical support questions, please contact support@plda.com.

Chapter 1 Introduction

1.1 Overview

The AHB-PCI Bridge testsuite simulates and tests the essential features of the AHB-PCI Bridge Core (vhdl or verilog) by calling a set of PCI and AHB scripts that are executed successively on the Testsuite design. The Testsuite is defined as a main tcl/tk script and must be used in conjunction with a simulation tool, such as Modelsim or NCSim.

1.2 Requirements

- **Modelsim:** version 6.1 or later
- **NCSim:** 5.5 p004 or later.

1.3 Directory Structure

- **ahb_testbench**
 - **sim_lib**
 - **modelsim_vhdl**: Modelsim vhdl simulation library
 - **modelsim_vlog**: Modelsim vlog simulation library
 - **sources**
 - **vhdl**: vhdl source code
 - **vlog**: vlog source code
- **Core**
 - **sim_lib**
 - **modelsim_vhdl**: Modelsim vhdl simulation library
 - **modelsim_vlog**: Modelsim vlog simulation library
 - **syn_lib**
 - **sources**
 - **vhdl**: vhdl source code
 - **vlog**: vlog source code
 - **synthesis**
 - **dc**
- **pcixpci_testbench**
 - **sim_lib**
 - **modelsim_vhdl**: Modelsim vhdl simulation library
 - **modelsim_vlog**: Modelsim vlog simulation library
 - **sources**
 - **vhdl**: vhdl source code
 - **vlog**: vlog source code
- **ref_designs**
 - **testsuite**
 - **script**: Testsuite script set
 - **sources**
 - **vhdl**: top-level for testsuite design
 - **vlog**: top-level for testsuite design
 - **modelsim**
 - **vhdl**: Modelsim vhdl Testsuite design
 - **vlog**: Modelsim vlog Testsuite design
 - **nctsim**
 - **vhdl**: NCSim vhdl Testsuite design
 - **vlog**: NCSim vlog Testsuite design
 - **vcs**
 - **vlog**: VCS vlog Testsuite design

Chapter 2 Simulating with the Testsuite

The testsuite is a set of scripts designed to verify the functionality of the Core. Each script is designed to automatically report errors in log files during execution. Each log file is saved in the log directory and is verified by comparing the generated log record with a keyword list.

All Testsuite scripts are found in `./reference_designs/testsuite/script`.

`scriptlog.htm` reports the status of executed scripts and is found in `./reference_designs/testsuite/(simu)/(vlog/vhdl)/testsuite_log`. All detailed AHB and PCI script reports are also saved in this directory.

Note that:

- some test scripts are designed to insert expected transaction errors (through the PCI Testbench).
- If you simulate a script through a graphical waveform, an "unknown" value might appear on the PCI_AD Bus due to uninitialized internal memory. This is normal behavior in the test environment and does not correspond to a valid data phase.

2.1 Simulation under Modelsim, NCSim, VCS

Your chosen simulator initiates the Testsuite in batch mode without displaying a waveform.

Before launching a simulation you must set the following variable in the header of the testsuite .tcl file (`testsuite_batch.tcl`):

- **set compile_pcixpci_testbench_sources**: 1 to compile selected source code, 0 if using selected ModelSim library
- **set compile_ahb_testbench_sources**: 1 to compile selected source code, 0 if using selected ModelSim library
- **set compile_ahbpci_bridge_v3x_sources**: 1 to compile selected source code, 0 if using selected ModelSim library

2.1.1 Windows

For Windows, a .tcl macro is provided that automatically configures the preceding variables and launches the testsuite (`testsuite_batch.tcl`). Note that launching the Testsuite requires use of tcl interpreter software.

1. Launch your tcl interpreter and browse to:
`./reference_designs/testsuite/<"modelsim" or "NCSim" or "VCS">/(vlog/vhdl/`
2. Execute the following command:
`sh testsuite_batch.tcl`

2.1.2 Unix

For Unix, a .tcl macro is provided that automatically configures the preceding variables and launches the testsuite (`testsuite_batch.tcl`). To launch the testsuite, do the following:

1. Open a terminal console and change directory to:
`./reference_designs/testsuite/<"modelsim" or "NCSim" or "VCS">/(vlog/vhdl/`
2. Execute the following command:
`sh testsuite_batch.tcl`

Chapter 3 Understanding the Testsuite Scripts

A set of PCI and AHB scripts execute the tests described in this section.

3.1 Simplebridge Scripts

simplebridge_reg_ahb_00

This script verifies that all AHB registers are implemented.

- Write data on the AHB-PCI AHB registers and reads it back from AHB bus
- Write FFFFFFFFh on mask registers and verifies that the corresponding bits in simple bridge mode are correctly implemented

simplebridge_reg_pci_00

This script verifies that all PCI registers are implemented.

- Configure the AHB-PCI PCI registers and read from PCI bus
- writes FFFFFFFFh on these registers and verifies that these registers in simple bridge mode are correctly implemented.

simplebridge_pciahb_00

This script verifies that the PCI-AHB windows operate correctly and that the PCI-AHB registers are implemented.

- Transfer a burst of data in PCI slave mode from PCI to AHB. AHB verifies that the data are correctly received.

simplebridge_pciahb_01

This script verifies that the PCI-AHB windows operate correctly and that the PCI-AHB registers are implemented.

- Transfer a burst of data in PCI slave mode from PCI to AHB using different cbe commands (the cbe command permits you to specify byte enable of data).
- AHB verifies that data are correctly received.

simplebridge_pciahb_02

This script verifies that the PCI-AHB windows operates correctly and that the PCI-AHB registers are implemented.

- Transfers two data with different cbe commands from PCI to AHB (the cbe command permits you to specify byte enable of data) in PCI slave mode (by using PCI wrap-around access).
- AHB verifies that data are correctly received. Note that initial waitstates are added before data is retrieved.

simplebridge_dma_00

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Small DMA-length bursts (DMA READ) are initiated from PCI to AHB
- Small DMA-length bursts (DMA WRITE) are initiated from AHB to PCI
- AHB verifies that data are correctly received

simplebridge_dma_01

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Small and medium-length bursts (DMA READ) from PCI to AHB
- Small and medium-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

simplebridge_dma_02

This script verifies that the DMA transfer operates and AHB DMA registers are implemented.

- Medium DMA-length bursts (DMA READ) from PCI to AHB
- Medium and large DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received.

simplebridge_dma_03

This script verifies that the DMA transfer operates and AHB DMA registers are implemented.

- Large DMA-length bursts (DMA READ) from PCI to AHB
- Large DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

simplebridge_dma_stop_00

This script verifies that the DMA transfer can be stopped.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB and stop transfer after several cycles
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI and stop transfer after several cycles
- AHB verifies that data are correctly received

simplebridge_dma_shift_00

This script verifies that the DMA transfer can be aligned on byte.

- Small and medium DMA-length bursts (DMA READ) by testing all realignment capabilities from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) by testing all realignment capabilities from AHB to PCI
- AHB verifies that necessary data are correctly re-aligned and received

simplebridge_dma_int_00

This script verifies post DMA transfer that the interrupt bit-through status register is set.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that status bit and data are correctly received.

simplebridge_dma_int_01

This script verifies post DMA transfer that the interrupt bit-through status register is set and that the bridge_irq interrupt is enabled.

- Small and Medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and Medium burst (DMA WRITE) from AHB to PCI from AHB to PCI
- AHB verifies that status bit, interrupt, and data are correctly received

simplebridge_multiple_transfer_00

This script verifies that mixed DMA transfers and PCI-AHB transfers are correctly managed by the Core.

- Small and Medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and Medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers small DMA-length (read and write) bursts to AHB during the DMA bursts.
- AHB and PCI verifies that data are correctly received.

simplebridge_multiple_transfer_01

This script verifies that mixed DMA transfers and PCI-AHB transfers are correctly managed by the Core.

- Small and Medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and Medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers small DMA-length (read and write) bursts to AHB during the DMA bursts.
- AHB and PCI verify that data are correctly received

simplebridge_multiple_transfer_02

This script verifies that mixed DMA transfers and PCI-AHB transfers are correctly managed by the Core.

- Small and Medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and Medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers small DMA-length (read and write) bursts to AHB during the DMA bursts.
- AHB and PCI verify that data are correctly received

simplebridge_multiple_transfer_03

This script verifies that the DMA transfer and AHB-PCI transfer are correctly managed by the Core.

- Small and Medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and Medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers small DMA-length (read and write) bursts to AHB during the DMA bursts.
- AHB verifies that data are correctly received

simplebridge_doorbell_pci_to_cpu_00

This script verifies that the doorbell feature from PCI to CPU operates correctly.

- PCI successively fires the four CPU doorbell interrupts
- AHB verifies that the interrupt on `bridge_irq` is fired, verifies the status, and clears the interrupt

simplebridge_doorbell_cpu_to_pci_00

This script verifies that the doorbell feature from CPU to PCI operates correctly.

- AHB successively fires the four CPU doorbell interrupts
- PCI verifies that interrupt `#INTA` is fired, verifies the status, and clears the interrupt

simplebridge_pciahb_timer_00

This script verifies that the PCI-AHB timer feature operates correctly.

- AHB configures AHB-TIMER and de-masks PCI-AHB timer interrupts
- PCI writes a burst of data and reads only one data on the requested burst
- PCI-AHB overflow and a `bridge_irq` interrupt is fired
- AHB verifies the bit and clears this interrupt

simplebridge_pciahb_f_error_00

This script verifies that the PCI-AHB fetch error feature operates correctly.

- AHB enables PCI-AHB fetch error interrupt
- PCI writes a data burst and tries to read data
- During the data reading, an OS error (PCI Testbench) is asserted
- PCI-AHB fetch error appears and a `bridge_irq` interrupt is fired
- AHB verifies the bit and clears the interrupt

simplebridge_pciahb_p_error_00

This script verifies that the PCI-AHB post error feature operates correctly.

- AHB enables PCI-AHB post error interrupt
- PCI tries to write a data burst
- During the data writing, an OS error (PCI Testbench) is asserted
- PCI-AHB post error appears and a `bridge_irq` interrupt is fired
- AHB verifies the bit and clears the interrupt

simplebridge_pciahb_ordering_00

This script verifies that the PCI-AHB ordering feature operates correctly.

- AHB testbench inserts read latency
- AHB initiates a DMA Read and DMA Write
- PCI writes data in PCI slave mode from PCI to AHB
- PCI read data in PCI slave mode from AHB to PCI
- AHB verifies data received from DMA
- PCI verifies data received

simplebridge_dma_end_00

This script verifies that the DMA end status bit operates correctly.

- AHB enable DMA end interrupt.
- AHB starts a DMA read to receive data from PCI
- At the end of the DMA read, a bridge_irq interrupt is fired
- AHB verifies the bit status and clears the interrupt
- The same process is done with DMA Write when transferring data to PCI
- AHB verifies data received

simplebridge_dma_abort_00

This script verifies that the DMA abort status bit operates correctly.

- AHB enable DMA abort interrupt
- AHB starts a DMA read to receive data from PCI
- During the DMA process, the PCI agent (PCI Testbench) aborts transfer
- An Interrupt bridge_irq is fired
- AHB verifies the bit status and clears the interrupt
- The same process is done with a DMA Write when transferring data to PCI
- AHB verifies data received

simplebridge_dma_ahb_tb_error_00

This script verifies that the AHB error status bit operates correctly.

- AHB enable AHB error interrupt
- AHB starts a DMA read to receive data from PCI
- During the DMA process, AHB (AHB Testbench) asserts an OS error
- An Interrupt bridge_irq is fired
- AHB verifies bit status and clears the interrupt
- The same process is done with DMA Write when transferring data to PCI
- AHB verifies data received.

simplebridge_ahbpci_00

This script verifies that the AHB-PCI windows operate correctly.

- AHB transfers a burst of data from AHB to PCI
- AHB transfers a burst of data from PCI to AHB
- AHB verifies that data are correctly received

simplebridge_ahbpci_01

This script verifies that the AHB-PCI windows operate correctly.

- AHB transfers a burst of data (using byte and Word data) from AHB to PCI
- AHB transfers a burst of data (using byte and word data) from PCI to AHB
- AHB verifies that data are correctly received

simplebridge_ahbpci_p_error_00

This script verifies that the AHB-PCI post error feature operates correctly.

- AHB enables AHB-PCI post error interrupt
- AHB tries to write a data burst to PCI
- During the data write, an abort is asserted by PCI (PCI Testbench)
- AHB-PCI post error appears and a bridge_irq interrupt is fired
- AHB verifies bit and clear this interrupt

simplebridge_ahbpci_f_error_00

This script verifies that the AHB-PCI fetch error feature operates correctly.

- AHB enables AHB-PCI fetch error interrupt
- AHB writes a data burst to PCI and tries to read data from PCI
- During the data read, an Abort is asserted by PCI (PCI Testbench)
- AHB-PCI fetch error appears and a bridge_irq interrupt is fired
- AHB verifies bit and clear this interrupt

simplebridge_pme_int_00

This script verifies that the PME interrupt operates correctly.

- AHB asserts PME interrupt
- PCI enables PME interrupt, detects PME interrupt, verifies bit status and clears this interrupt

simplebridge_pme_int_01

This script verifies that the PME feature operates correctly.

- PCI switches PM mode to D3 then switches to D0
- AHB reads and verifies the PME status register
- When switching to D0, PCI verifies that the AHB-PCI bridge BARs are reset

simplebridge_clkrun_00

This script verifies that the clock-run feature operates correctly.

- PCI agent (PCI testbench) clocks down and clocks up the clockrun of the AHB-PCI simplebridge
- AHB-PCI simplebridge maintains the clockrun signal

3.2 Simplebridge Cardbus Scripts

simplebridge_cardbus_pme_int_00

This script verifies that the PME interrupt operates correctly.

- AHB asserts the PME interrupt
- PCI enables the PME interrupt, detects the PME interrupt, verifies bit status, and clears the interrupt

simplebridge_cardbus_pme_int_01

This script verifies that the PME feature operates correctly.

- PCI switches PM mode to D3 then switches to D0
- AHB reads and verifies the PME status register
- When switching to D0, PCI verifies that the AHB-PCI bridge BARs are reset.

simplebridge_cardbus_reg_00

This script verifies that the cardbus CIS pointer, Cint, and PME interrupt operate correctly.

- AHB configures and reads the cardbus CIS pointer
- PCI reads the cardbus CIS register
- PCI enables Cint interrupt and fires a Cint interrupt
- PCI verifies the Cint status register and clears the interrupt
- PCI enables the PME interrupt and fires a PME interrupt
- PCI verifies the PME status register and clears the interrupt

3.3 Hostbridge Scripts

hostbridge_reg_ahb_00

This script verifies that all AHB registers are implemented.

- Write data to the AHB register and reads it back from the AHB bus
- Writes FFFFFFFh on mask registers and verifies that the corresponding bits in host bridge mode are correctly implemented

hostbridge_pciahb_00

This script verifies that the PCI-AHB windows operate correctly and that the PCI-AHB registers are implemented.

- Transfers a burst of data from PCI to AHB in PCI slave mode
- AHB verifies that data are correctly received

hostbridge_pciahb_01

This script verifies that the PCI-AHB windows operate correctly and that the PCI-AHB registers are implemented.

- Transfer a burst of data in PCI slave mode from PCI to AHB using different cbe commands (the cbe command permits you to specify byte enable of data).
- AHB verifies that data are correctly received

hostbridge_pciahb_02

This script verifies that the PCI-AHB windows operate correctly and that the PCI-AHB registers are implemented.

- Transfers two data with different cbe commands from PCI to AHB (the cbe command permits you to specify byte enable of data) in PCI slave mode (by using PCI wrap-around access).
- AHB verifies that data are correctly received. Note that initial waitstates are added before retrieving data.

hostbridge_dma_00

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Small DMA-length bursts (DMA READ) from PCI to AHB
- Small DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

hostbridge_dma_01

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Medium DMA-length bursts (DMA READ) from PCI to AHB
- Medium DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

hostbridge_dma_02

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Medium DMA-length bursts (DMA READ) from PCI to AHB
- Medium and Large DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

hostbridge_dma_03

This script verifies that the DMA transfer operates and that the AHB DMA registers are implemented.

- Large DMA-length bursts (DMA READ) from PCI to AHB
- Large DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB verifies that data are correctly received

hostbridge_dma_stop_00

This script verifies that the DMA transfer can be stopped.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB and stops transfer after several cycles
- Large DMA-length bursts (DMA WRITE) from AHB to PCI and stops transfer after several cycles
- AHB verifies that necessary data are correctly received

hostbridge_dma_shift_00

This script verifies that the DMA transfer can be aligned on byte.

- Small and medium DMA-length bursts (DMA READ) by testing all realignment capabilities from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) by testing all realignment capabilities from AHB to PCI
- AHB verifies that necessary data are correctly re-aligned and received

hostbridge_multiple_transfer_00

This script verifies that the DMA transfer and PCI-AHB transfer can be managed correctly by the Core.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers small DMA-length (read and write) bursts to AHB during the DMA bursts
- AHB and PCI verify that data are correctly received

hostbridge_multiple_transfer_01

This script verifies that the DMA transfer and PCI-AHB transfer can be managed correctly by the Core.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers medium DMA-length burst data at the same time to AHB and PCI reads it after several cycles
- AHB and PCI verifies that data are correctly received

hostbridge_multiple_transfer_02

This script verifies that the DMA transfer and PCI-AHB transfer can be managed correctly by the Core.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI
- PCI transfers medium DMA-length (read and write) bursts to AHB during the DMA bursts
- AHB and PCI verifies that data are correctly received

hostbridge_multiple_transfer_03

This script verifies that the DMA transfer and AHB-PCI transfer are managed correctly by the Core.

- Small and medium DMA-length bursts (DMA READ) from PCI to AHB
- Small and medium DMA-length bursts (DMA WRITE) from AHB to PCI
- AHB transfers medium burst data at the same time to PCI and AHB reads data after several cycles
- AHB and PCI verify that data are correctly received

hostbridge_pciahb_timer_00

This script verifies that the PCI-AHB operates correctly.

- AHB configures AHB-TIMER and de-masks PCI-AHB timer interrupt
- PCI writes a burst of data and reads only one data on the requested burst
- PCI-AHB overflow and a bridge_irq interrupt is fired
- AHB verifies the bit and clears the interrupt

hostbridge_pciahb_f_error_00

This script verifies that the PCI-AHB fetch error function operates correctly.

- AHB enables PCI-AHB fetch error interrupt
- PCI writes a data burst and tries to read data
- During the data reading, an OS error (PCI Testbench) is asserted
- PCI-AHB fetch error appears and a bridge_irq interrupt is fired
- AHB verifies bit status and clears the interrupt

hostbridge_pciahb_p_error_00

This script verifies that the PCI-AHB post error function operates correctly.

- AHB PCI-AHB post error interrupt
- PCI tries to write a data burst
- During data writing, an OS error (PCI Testbench) is asserted
- PCI-AHB posts an error and a bridge_irq interrupt is fired
- AHB verify bit status and clear the interrupt

hostbridge_pciahb_ordering_00

This script verifies that the PCI-AHB ordering feature operates correctly.

- AHB testbench inserts read latency
- AHB initiates a DMA Read and DMA Write
- PCI writes data in PCI slave mode from PCI to AHB
- PCI read data in PCI slave mode from AHB to PCI
- AHB verifies data received from DMA
- PCI verifies data received

hostbridge_dma_end_00

This script verifies that the DMA end status bit operates correctly.

- AHB enable DMA end interrupt
- AHB start a DMA read to receive data from PCI
- At the DMA end, a bridge_irq interrupt is fired
- AHB verifies bit status and clears the interrupt
- The same process is done with a DMA Write to transfer data to PCI
- AHB verifies data received

hostbridge_dma_abort_00

This script verifies that the DMA abort status bit operates correctly.

- AHB enable DMA abort interrupt.
- AHB start a DMA read to receive data from PCI
- During the DMA process, PCI (PCI Testbench) aborts transfer
- An Interrupt bridge_irq is fired
- AHB verifies bit status and clears the interrupt
- The same process is done with DMA Write when transferring data to PCI
- AHB verifies data received

hostbridge_dma_ahb_tb_error_00

This script verifies that the AHB error status bit operates correctly.

- AHB enables AHB error interrupt
- AHB starts a DMA read to receive data from PCI
- During the DMA process, AHB (AHB Testbench) asserts an OS error
- An Interrupt bridge_irq is fired
- AHB verifies bit status and clears the interrupt
- The same process is done with DMA Write when transferring data to PCI
- AHB verifies data received

hostbridge_ahbpci_00

This script verifies that the AHB-PCI windows operate correctly.

- Shows how AHB transfers a burst of data from AHB to PCI
- Shows how AHB transfer a burst of data from PCI to AHB
- AHB verifies that data are correctly received

hostbridge_ahbpci_01

This script verifies that the AHB-PCI windows operate correctly.

- Shows how AHB transfers a burst of data (using byte and word data) from AHB to PCI
- Shows how AHB transfer a burst of data (using byte and word data) from PCI to AHB
- AHB verifies that data are correctly received

hostbridge_ahbpci_p_error_00

This script verifies that the AHB-PCI post error function operates correctly.

- AHB enables AHB-PCI post error interrupt.
- AHB tries to write a data burst to PCI.
- During the data writing, an Abort is asserted by PCI (PCI Testbench). AHB-PCI post error appears and a bridge_irq interrupt is fired. AHB verifies bit and clear this interrupt.

hostbridge_ahbpci_f_error_00

This script verifies that the AHB-PCI fetch error function operates correctly.

- AHB enables AHB-PCI fetch error interrupt
- AHB writes a data burst to PCI and tries to read data from PCI
- During the data reading, an Abort is asserted by PCI (PCI Testbench)
- AHB-PCI fetch error appears and a bridge_irq interrupt is fired
- AHB verifies bit status and clear the interrupt

hostbridge_conf_type_00

This script verifies that the hostbridge configuration Type0 operates correctly.

- AHB uses DMA to configure (using the configuration command) the connected PCI agent (PCI Testbench). Configuration commands used are type 0.
- When the PCI agent is configured, AHB starts a DMA read and a DMA write in order to ensure that the PCI agent is correctly configured
- AHB verifies data received

hostbridge_conf_type_01

This script verifies that the hostbridge configuration Type1 operates correctly.

- AHB uses DMA in order to configure (using the configuration command) the connected PCI agent (PCI Testbench). Configuration commands used are type 1.
- When the PCI agent is configured, AHB starts a DMA read and a DMA write in order to ensure that the PCI agent is correctly configured
- AHB verifies data received

hostbridge_pme_int_00

This script verifies that the PME interrupt operates correctly.

- PCI asserts and deasserts PME interrupt
- AHB enables PME interrupt
- AHB detects PME interrupt by verifying that bridge_irq AHB is set
- AHB verifies bit status and clears the bridge_irq interrupt

hostbridge_clkrun_00

This script verifies that the clock-run feature operates correctly.

- AHB clocks down and clocks up the clockrun of the PCI agent (PCI testbench)
- PCI agent maintains the clockrun signal

3.4 Hostbridge Cardbus Scripts

hostbridge_cardbus_pme_int_00

This script verifies that the PME interrupt operates correctly.

- PCI assert and de-assert PME interrupt
- AHB enables PME interrupt
- AHB detects PME interrupt by verifying that bridge_irq AHB is set
- AHB verifies bit status and clears the bridge_irq interrupt