System Verilog Assertions

LAB Material

Ashok B. Mehta

http://www.defineview.com

Copyright Notice

DefineView Consulting

http://www.defineview.com

Ashok B. Mehta
501 Pine Wood Lane
Los Gatos, CA 95032
(408) 309-1556
Email: ashok@defineview.com

Lab 6 …

PCI Read protocol …

LAB 6: PCI Read Protocol

LAB Overview

A simple system with a PCI Master and PCI Target modules designed to do a simple basic PCI Read operation.

The LAB shows how to derive and write simple but effective assertions for a PCI type bus.

LAB Objectives

1) Learn how to model temporal domain assertions for bus interface type logic.

2) Reinforce understanding of Edge sensitive sampled value functions, consecutive repetition, boolean expressions, etc.

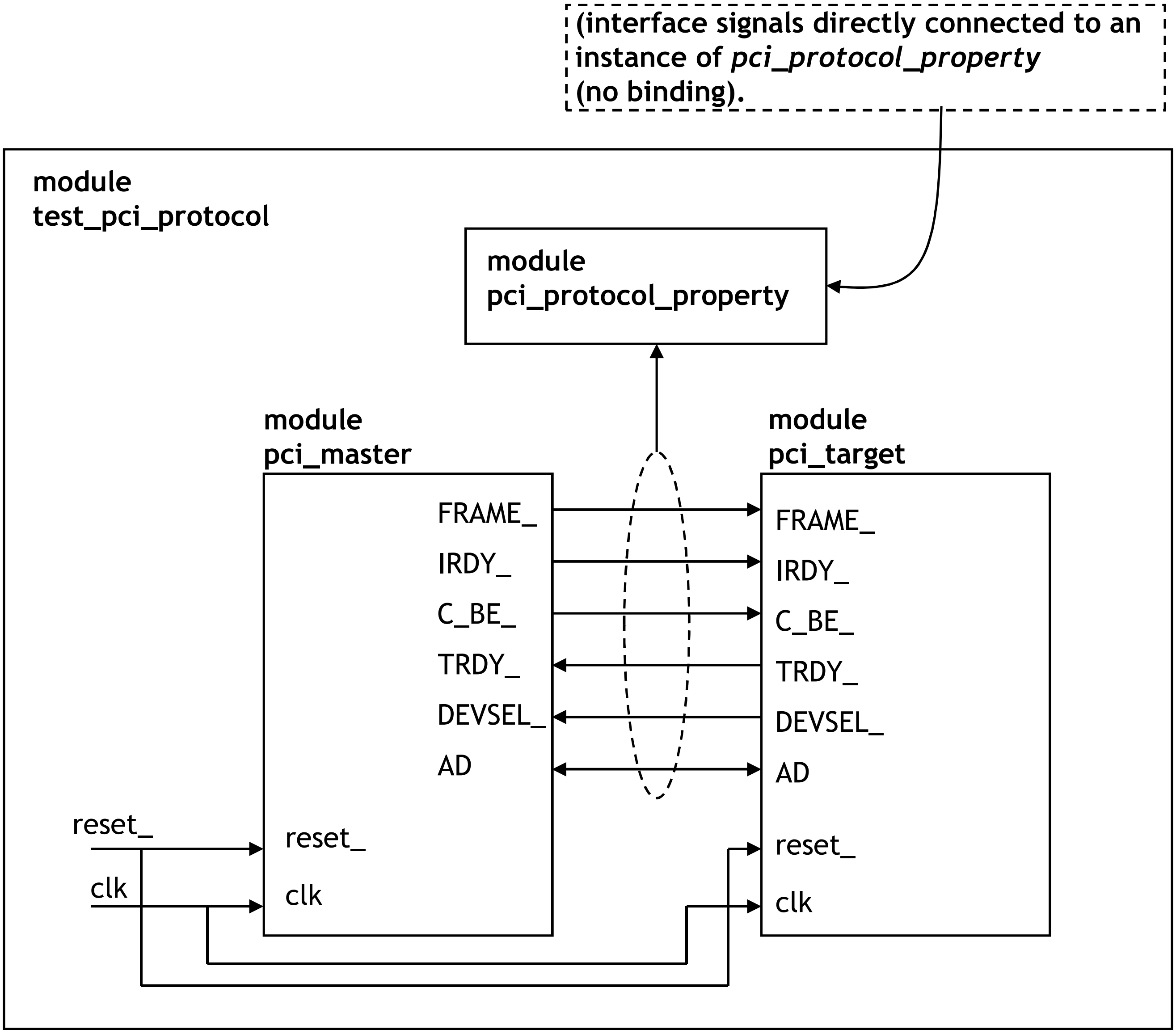
LAB: Database

FILES:

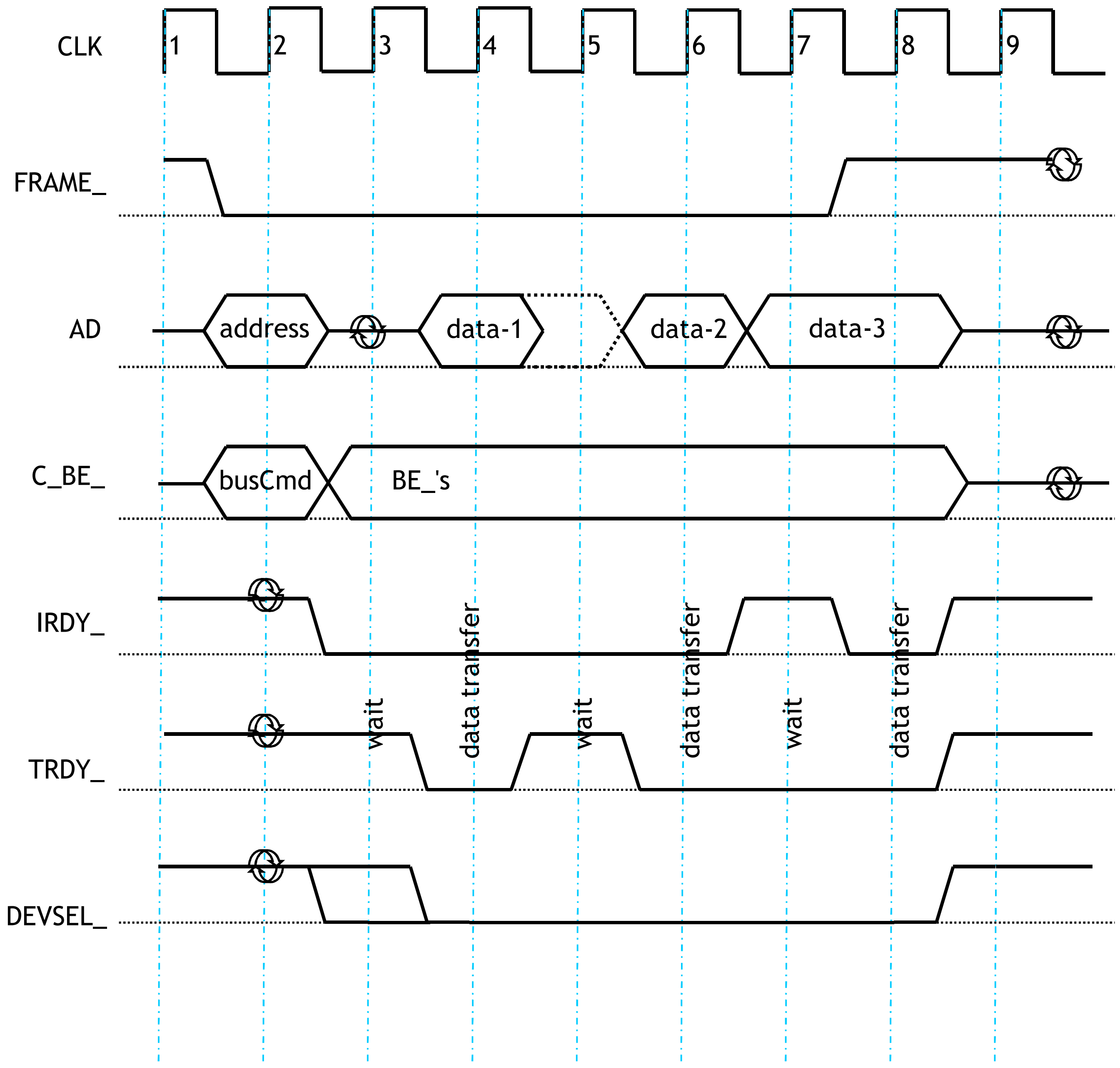
pci_master.v :: A (very) simple PCI Master module driving only a simple Read cycle.
pci_target.v :: A (very) simple PCI Target module responding to a simple Read Cycle.
pci_protocol_property.v :: SVA file for PCI Read cycle assertions.
Note that this file is only an empty module shell.
You will add properties that meet the specification described below.

test_pci_protocol.sv :: Testbench for the pci_protocol module.

LAB 6: PCI System



LAB6: PCI Read Protocol



PCI: Basic Read Protocol Checkers

LAB: Assertions to Code

Property Name	Description
checkPCI_AD_CBE (check1)	On falling edge of FRAME_, AD or C_BE_ bus cannot be unknown
checkPCI_DataPhase (check2)	When both IRDY_ and TRDY_ are asserted, AD or C_BE_ bus cannot be unknown
checkPCI_Frame_Irdy (check3)	FRAME can be de-asserted only if IRDY_ is asserted
checkPCI_trdyDevsel (check4)	TRDY_ can be asserted only if DEVSEL_ is asserted
checkPCI_CBE_during_trx (check5)	Once the cycle starts (i.e. at FRAME_ assertion) C_BE_ cannot float until FRAME_ is de-asserted.

LAB 6 : PCI Read Protocol

LAB: How to compile/simulate - step by step instructions

Follow the steps below to add your assertion for each check.

Then compile/simulate with each of your assertions and see that your results match with those stored in the ./.solution directory

Here's step by step instructions...

1. % cd <myDir>/SVA_LAB/LAB6
   % vi pci_protocol_property.sv

   Edit this file to add your properties.

   Note that DUMMY properties are coded in pci_protocol_property.sv to simply allow the module to compile.

   You must remove the DUMMY properties and code correct properties as required above.

2. % vi pci_protocol_property.sv
   - Look for `ifdef check1
   - Remove the 'DUMMY' property and code your property as specified above for CHECK #1
   - Save the file and run the following simulation.

   % run_check1
   - If you have coded the property correct, you should see a failure for the CHECK #1 specified above.
   - Simulation will create test_pci_protocol_check1.log
   - Compare test_pci_protocol_check1.log with .solution/test_pci_protocol_check1.log and see if your results match with the log in the .solution directory.
   - If your results don't match, revisit your property and repeat this step.

CONTINUED ➔

LAB 6 : PCI Read Protocol

LAB: How to compile/simulate - step by step instructions

3. % vi pci_protocol_property.sv
   - Look for `ifdef check2
   - Remove the 'DUMMY' property and code your property as specified above for CHECK #2
   - Save the file and run the following simulation.

   % run_check2
   - If you have coded the property correct, you should see a failure for the CHECK #2 specified above.
   - Simulation will create test_pci_protocol_check2.log
   - Compare test_pci_protocol_check2.log with .solution/test_pci_protocol_check2.log and see if your results match with the log in the .solution directory.
   - If your results don't match, revisit your property and repeat this step.

4. % vi pci_protocol_property.sv
   - Look for `ifdef check3
   - Remove the 'DUMMY' property and code your property as specified above for CHECK #3
   - Save the file and run the following simulation.

   % run_check3
   - If you have coded the property correct, you should see a failure for the CHECK #3 specified above.
   - Simulation will create test_pci_protocol_check3.log
   - Compare test_pci_protocol_check3.log with .solution/test_pci_protocol_check3.log and see if your results match with the log in the .solution directory.
   - If your results don't match, revisit your property and repeat this step.

CONTINUED ➔

LAB 6 : PCI Read Protocol

LAB: How to compile/simulate - step by step instructions

5. % vi pci_protocol_property.sv
- Look for `ifdef check4
- Remove the 'DUMMY' property and code your property as specified above for CHECK #4
- Save the file and run the following simulation.

% run_check4
- If you have coded the property correct, you should see a failure for the CHECK #4 specified above.
- Simulation will create test_pci_protocol_check4.log
- Compare test_pci_protocol_check4.log with .solution/test_pci_protocol_check4.log and see if your results match with the log in the .solution directory.
- If your results don't match, revisit your property and repeat this step.

6. % vi pci_protocol_property.sv
- Look for `ifdef check5
- Remove the 'DUMMY' property and code your property as specified above for CHECK #5
- Save the file and run the following simulation.

% run_check5
- If you have coded the property correct, you should see a failure for the CHECK #5 specified above.
- Simulation will create test_pci_protocol_check5.log
- Compare test_pci_protocol_check5.log with .solution/test_pci_protocol_check5.log and see if your results match with the log in the .solution directory.
- If your results don't match, revisit your property and repeat this step.

DONE… CONGRATULATIONS