

System Verilog Assertions

LAB Material

Ashok B. Mehta

<http://www.defineview.com>

© 2006-2013

Copyright Notice

Copyright Notice

© 2006-2013

The material in this training guide is copyrighted by DefineView Consulting/Ashok B. Mehta of Los Gatos, California. All rights reserved. No material from this guide may be duplicated or transmitted by any means or in any form without the express written permission of Ashok B. Mehta

DefineView Consulting

<http://www.defineview.com>

Ashok B. Mehta

501 Pine Wood Lane

Los Gatos, CA 95032

(408) 309-1556

Email: ashok@defineview.com

Verilog is a registered trademark of Cadence Design Systems, San Jose, California.

Lab 1 ...

how to 'bind' a design module
with it's properties ...

LAB 1 : Basic Property

LAB Overview

This LAB is to help you understand how to 'bind' a design module to a property module (which contains assertions for the design module.)

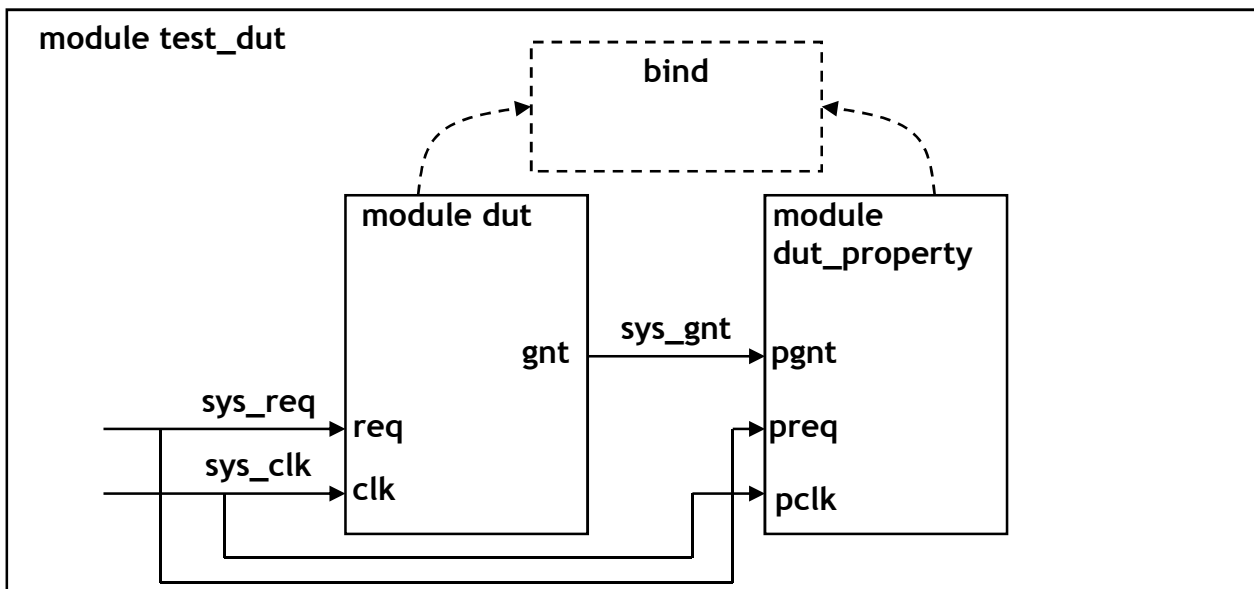
It also highlights how property/sequence behaves with and without an implication and understand the vacuous pass phenomenon ;-)

LAB Objectives

1. 'bind' a design module to it's property module.
2. You will also see how the property can be 'cover'ed and how it can avoid a vacuous pass.

LAB Design

Here's the block diagram showing connection between 'dut' and 'dut_property' modules. You need to bind these two modules.



LAB 1 : 'bind'ing design/property module

LAB: Database

FILES:

1. *dut.v :: Verilog module that drives a simple req/gnt protocol.*
2. *dut_property.sv :: File that contains 'dut' properties/assertions.*
3. *test_dut.sv :: Testbench for the 'dut'. This is the file in which you'll 'bind' the 'dut' with 'dut_property'*
4. *LAB_QUESTIONS :: This file has questions that you will answer.*

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

1. `% cd <myDir>/SVA_LAB/LAB1`

2. `% vi test_dut.sv`
Search for "LAB EXERCISE - START"

*Add code after the comments to
'bind' the design 'dut' with it's property module 'dut_property'*

Save the file.

Then follow the compile/simulate steps described below.

3. `% run_no_implication`

*This will create test_dut_no_implication.log
Compare ('diff') your log with the one stored in
.solution/test_dut_no_implication.log*

*If you did a correct bind of 'dut' and 'dut_property', you will see a few PASS/FAIL
in the log. If not, you won't see any pass or fail*

-) *Study test_dut_no_implication.log*
-) *Answer the questions embedded in the file ./LAB_QUESTIONS
for +define+no_implication*

LAB 1 : 'bind'ing design/property module

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

4. `% run_implication`

*This will create test_dut_implication.log
Compare ('diff') your log with the one stored in
.solution/test_dut_implication.log*

*If you did a correct bind of 'dut' and 'dut_property', you will see a few PASS/FAIL
in the log. If not, you won't see any pass or fail*

-) Study test_dut_implication.log
-) Answer the questions embedded in the file ./LAB_QUESTIONS
for +define+implication

5. `% run_implication_novac`

*This will create test_dut_implication_novac.log
Compare ('diff') your log with the one stored in
.solution/test_dut_implication_novac.log*

*If you did a correct bind of 'dut' and 'dut_property', you will see a few PASS/FAIL
in the log. If not, you won't see any pass or fail*

-) Study test_dut_implication_novac.log
-) Answer the questions embedded in the file ./LAB_QUESTIONS
for +define+implication_novac