System Verilog Assertions
&
Functional Coverage

Language, Methodology and Applications

2 Day Training Class with 6 LABs

Ashok B. Mehta

DefineView Consulting

http://www.defineview.com

2 Day Training :: Abstract

- **What is SVA (SystemVerilog Assertions)?**
  - System Verilog Assertions (SVA) is a powerful subset of the IEEE 1800 System Verilog standard.
  - Its hardware oriented concurrent semantics allow for intuitive development of complex multi-clock domain checkers to catch those elusive bugs at the source.
  - SVA works both with VHDL and Verilog. Clean separation between RTL design and SVA assertions.
- **What is FC (Functional Coverage)?**
  - Functional Coverage (FC) is another subset of System Verilog that allows you to measure how much of design *intent* have you covered with your tests/regressions.
- **Course Highlights**
  - Each operator/feature is explained in detail using comprehensive examples, timing diagrams and simulation logs.
  - Real life applications are discussed to put it all in perspective.
  - A language reference grade handout book is provided to the class. It has comprehensive detail on each page that can serve as excellent reference material for future.
  - Labs are geared to solidify understanding of key concepts using application oriented designs.
  - The course includes IEEE 1800 – Features from LRM 2005, 2009 and 2012.

2 Day Training :: DAY 1 Agenda

- **Introduction to Assertions**
  - What is an assertion?
  - Advantages of Assertion Based Verification (ABV) .
  - Assertion Based Verification (ABV) Methodology components

- **Assertions :: Syntax and Semantics**
  - Immediate assertions
  - Concurrent assertions - Basics
  - clocking basics; threads
  - Sequence and Property
  - Binding properties.
  - Sampled value functions
  - $rose, $fell, $stable
  - Sequence Operators
  - ##m and ##[m:n] clock delay
  - [*] and [*m:n] – Consecutive repetition operator
  - [=] and [=m:n] – Non-consecutive repetition operator
  - [->] and [-> m:n] – Goto (non-consecutive) repetition operator
  - ‘throughout’, ‘within’, ‘intersect’, 'first_match'
  - 'and' and 'or' of sequences
  - ‘intersect’ vs. ‘and’
  - ‘not’ operator
  - If … else
  - ‘disable iff’
  - Recursive property
  - System functions -$onehot, $isunknown, $countones

- **DAY 1 LABs**
  - LAB 1: 'bind’; implication operators
  - LAB 2: pipelined threads
  - LAB 3: Synchronous FIFO

2 Day Training :: DAY 2 Agenda

- Multiple Clocks
- Local variables
- Detecting and using endpoint of a sequence
- The ‘expect’, 'assume' statement
- Calling subroutines
- Asynchronous Assertions
- ‘let’ declaration
- ‘checker’
- ‘strong’ and ‘weak’ properties, deferred immediate assertions
- $sampled, $changed, $inferred_clock, $inferred_disable
- past and future global clock sampling functions
- ‘followed by’ property operators: #-# and #=#
- ‘always’, ‘eventually’, ‘until’, ‘until_with’, ‘s_until’, ‘s_until_with’, ‘nexttime’, ‘case’, $inferred_clock and $inferred_disable, etc.

- **System Verilog Functional Coverage**
- **Code coverage vs. Functional coverage**
- **Coverage driven methodology**
- **Features**
  - ‘covergroup’
  - ‘coverpoint’
  - ‘bins’
  - ‘cross’ coverage
  - Transition coverage
  - Wildcard bins
  - ‘ignore_bins’ and ‘illegal_bins’
  - ‘binsof’ ‘intersect’
- **Coverage options**
  - Instance specific
  - ‘covergroup’ type
- **System tasks for coverage**
- **Coverage methods for use in procedural code**

DAY 2 LABs

- **LAB 4: COUNTER Assertions**
  - Code different properties to meet various Counter fail conditions.
  - Enforces the use of Local Variables, $past system task, etc.

- **LAB 5: DATA TRANSFER BUS PROTOCOL Assertions**
  - Code different properties to meet bus protocol fail conditions.
  - Exemplifies temporal domain assertions coding ($stable, $rose, throughout, etc.)
  - Shows two different ways to code the same property.

- **LAB 6: PCI PROTOCOL Assertions**
  - Create a test plan for a basic PCI Read Bus Transaction.
  - Write properties to catch key temporal domain protocol violations.

CUSTOMER TESTIMONIALS …

"Ashok is a very good instructor – very impressive."

John Reykjalin, President, Grizzly Peak Engineering, Inc.

"The class was excellent, well distributed between the fundamentals and the practical examples. With a little tweak, we can use those example assertions presented right now in our design development!
In addition I would like to thank you for seminar associated text material. The handout (book) is a few levels above anything I have previously seen. The rules and example descriptions cover the associated topic completely."

Thomas Slee, Sr. Electrical Engineer, ASIC Verification Lead, Space System Loral

"The seminar on SVA was very educative and informative.
The material was in-depth, was from a hardware design/verification person's perspective and it was vendor neutral. The information was very good and I hope to have a chance to use it in the future."

Shubha Umesh, Senior Logic Engineer, LeCroy Corporation

"Ashok, I take this opportunity to personally thank you for your dedication. You have very good knowledge on the subject. I intend to now use assertions heavily on my next verification project and will use your examples extensively. This class helped me strengthen my knowledge on SVA."

Mohammad Ashraf, Sr. Electrical Engineer, Space System LORAL

Ashok Mehta

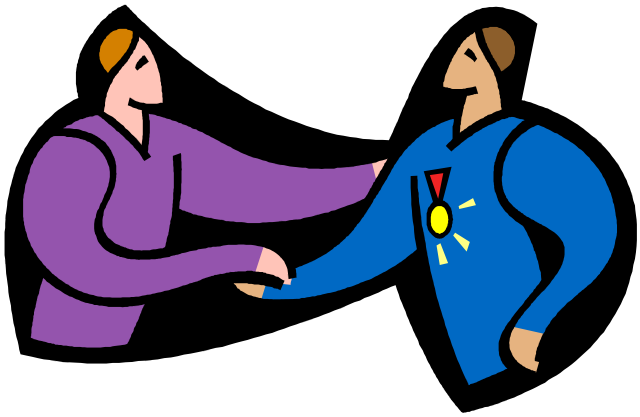
INTRODUCTION

- **30+ years of experience in SoC, CPU design and verification**
- **Author of two books**
  - SystemVerilog Assertions and Functional Coverage (2nd edition - 2016)
    - 400+ pages
  - ASIC/SoC Functional Design Verification (2018)
    - 320+ pages
- **17 US Patents on 3DIC and SoC verification**

CURRENT ACTIVITIES

- **Verification Methodology Adviser to FPGA, SoC and Systems companies.**
  - Methodology evaluation; creation; customization
- **SVA and Functional Coverage Training**
  - Customized to your requirements
- **SVA and FC Training class on UDEMY**
  - 33 lectures; 8.5 hours; life time access; in-depth from scratch

happy asserting…



Please visit our web site for detail on scheduling and pricing.

DefineView Consulting
(www.defineview.com)