

## SystemVerilog Testbench for Verilog Verification Engineers

### Overview

*SystemVerilog Testbench for Verilog Verification Engineers* is a 3-day fast-paced workshop covering the testbench constructs in the IEEE 1800 SystemVerilog standard. SystemVerilog unifies the strengths of Verilog, VHDL, C, VERA and PSL in one language, making it a true “Hardware Description and Verification Language” (HDVL). SystemVerilog enables the verification of very large, complex designs using a single language for all aspects of the design cycle, including simulation, synthesis and formal verification.

The focus of this workshop is on the verification aspects of SystemVerilog. Concepts presented include new special testbench modeling constructs, defining clocking domains, Object Oriented verification, constrained random verification, coverage, and assertions. Just a few of the SystemVerilog verification constructs that are presented include dynamic arrays, dynamic processes, object oriented inheritance and polymorphism, mailboxes, semaphores, specifying randomization constraints, specifying functional coverage, and an overview of SystemVerilog assertions. Several labs reinforce the principles presented, with forty percent of the class time devoted to hands-on experience.

Course materials include a *comprehensive student guide*, and a “*Verilog HDL Quick Reference Guide*” (\$15 value).

### Course Objectives

At the conclusion of this workshop, engineers will understand how to take full advantage of the rich set of advanced verification extensions to the Verilog language to develop object-oriented testbenches that utilize constrained-random tests, functional coverage, mailboxes and scoreboarding.

### Intended Audience

This workshop is for experienced Verilog engineers. The course presupposes a working knowledge of Verilog, and only covers the SystemVerilog testbench enhancements to Verilog. Verification engineers who are not familiar with Verilog should instead take Sutherland HDL’s “*SystemVerilog Testbench for non-Verilog Verification Engineers*” workshop.

### Prerequisites (essential)

*Knowledge of the Verilog HDL is mandatory.* Without this background, students cannot fully benefit from this course. This course presents the substantial extensions to the Verilog HDL, and assumes a working knowledge of Verilog.

### Software Tools Used

The Cadence *Incisive*<sup>™</sup>, Synopsys *VCST*<sup>™</sup> or the Mentor Graphics *Questa*<sup>™</sup> simulator will be used for labs.

### Quotes From Students

*“The instructor was clearly an expert and extremely good at presenting the material.”*

*“Best workshop I’ve attended. Very informative and insightful into what SystemVerilog offers.”*

*“The labs were particularly effective — long enough to learn the topic and to make you think.”*

### Workshop Locations

This workshop can be presented on-site, at your facilities. We also offer several open-enrollment workshops throughout the year. For more information, please refer to our web page, [www.sutherland-hdl.com](http://www.sutherland-hdl.com), or call us at +1-503-692-0898.

## Syllabus — SystemVerilog Testbench for Verilog Verification Engineers

### Day One

---

#### Introduction to SystemVerilog

- The purpose of SystemVerilog
- Backward compatibility with Verilog
- Software tools supporting SystemVerilog

#### Strategies for High-level Verification Programs

- Black-box versus white-box testing
- Assertion based design methods
- Constrained random testing
- Test synchronization
- Object-oriented testing

#### Verilog and SystemVerilog Data Types

- Verilog's 4-state data types and rules
- SystemVerilog's 2-state data types and rules
- Casting
- User-defined types
- Enumerated types
- 2-state modeling and verification "gotchas"
- Lab: Verification with 2-state and enumerated data types

#### SystemVerilog Testbench Programming Constructs

- Time units and precision
- Packages
- Design hierarchy
- Interfaces
- Procedural blocks
- Enhanced tasks and functions
- Lab: Working with complex hierarchical designs

#### Program Blocks and Clocking Domains

- Program blocks
- Clocking domains
- Final blocks
- Lab: Developing a test program

### Day Two

---

#### Object-Oriented Programming, Part One

- SystemVerilog's class data type
- Defining class objects
- Class methods
- Class inheritance
- Lab: Creating a simple OO testbench

#### Object-Oriented Programming, Part Two

- Extending class definitions (inheritance)
- Virtual methods
- Virtual classes
- Public and private classes
- Lab: Creating an advanced OO testbench

#### Dynamic Arrays and Scoreboards

- Dynamic arrays
- Associative arrays
- Queues
- Strings
- Fork—join dynamic processes
- Lab: Create a scoreboard using dynamic arrays

#### Process Synchronization

- Built-in mailbox classes
- Built-in semaphore classes
- Enhanced event data types
- Synchronized verification methodologies
- Lab: Using mailboxes for verification

### Day Three

---

#### Constrained Random Value Generation

- Built-in SystemVerilog random classes
- Defining constrained random values
- Constrained random verification methodologies
- Lab: Using constrained random test values

#### Functional Coverage

- Defining cover groups
- Defining cover points
- Defining cover bins
- Cross coverage
- Lab: Using coverage with constrained random tests

#### Assertions for Verification Engineers

- Assertion concepts
- Concurrent assertions and sequences
- Basic sequence definitions
- Disabling assertions during reset
- Controlling assertion messages
- Binding assertions to design blocks
- Lab: Using assertions at the testbench level