

## Comprehensive Verilog and SystemVerilog for Design and Synthesis

### Overview

*Comprehensive Verilog and SystemVerilog for Design and Synthesis* is a packed 4-day workshop covering the complete IEEE Verilog Hardware Description Language standard, including the synthesizable portions of the SystemVerilog-2009 extensions to Verilog. All aspects of Verilog are presented, with a focus on using Verilog for top-down design with synthesis and simulation. The synthesizable subset of the SystemVerilog extensions are integrated into the course topics. These powerful extensions include 2-state data types, user-defined types, enumerated types, structures, and self-verifying decision statements. The importance of proper coding styles is emphasized throughout the course. Special attention is given to subtleties, such as how blocking and non-blocking assignments affect simulation and synthesis. Several labs reinforce the principles presented. Forty percent of the class time is devoted to hands-on experience, including a 5-hour final project modeling a complete DSP processor. A handy “*Verilog HDL Quick Reference Guide*” (\$15 value) is included with the course materials.

### Intended Audience and Objectives

This workshop is for digital engineers who will be designing with Verilog and SystemVerilog. This workshop will enable students to immediately be productive with Verilog and SystemVerilog languages, and with simulation and synthesis software tools. Both new Verilog users, as well as those who are familiar with Verilog and desire a more in-depth knowledge of the language, will benefit from this course.

### Software Tools Used

Students will use both Verilog simulation and synthesis tools during class labs. The tools used can be the Cadence *NC-Verilog*<sup>™</sup>, Synopsys *VCS*<sup>™</sup>, or Mentor Graphics *ModelSim*<sup>™</sup> simulators, and the Synopsys *DCT*<sup>™</sup>, Synplicity *Synplify*<sup>™</sup>, or Mentor *Precision*<sup>™</sup> synthesis compilers.

### Prerequisites (essential)

*Knowledge of digital design engineering is required.* Without this background, students cannot fully benefit from this course. Labs include writing models of digital circuits such as shift registers, arithmetic logic units, FIFOs and a DSP.

### Comments From Students

*“An excellent comprehensive study of Verilog with perspectives on Verilog as a simulation, modeling and synthesis language; backed with valuable labs.”*

*“Excellent balance between lecture and lab. Very good structure. For sure the best course I've ever had!”*

*“Excellent, wonderful class. Definitely worth the 4 days. The instructor was extremely well prepared. He kept my interest the entire time.”*

### 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—Verilog and SystemVerilog for Design and Synthesis

### Day One

#### Introduction to Verilog and SystemVerilog

- Concepts of top-down design
- Overview of RTL models
- Overview of gate/switch models

#### Design Verification Using Simulation

- Writing verification testbenches in Verilog
- Running your preferred Verilog simulator
- Debugging designs with simulation
- Lab: running your simulator and debug tools

#### Verilog HDL Syntax and Semantics

- Identifier names
- Logic values and numbers
- Data types
- SystemVerilog extensions to Verilog data types
- Exercise: selecting the correct data types

#### Procedures, Programming Statements and Operators

- Procedural blocks
- SystemVerilog enhanced procedural blocks
- Tasks and functions
- Continuous assignments
- Programming statements and operators
- Exercise: programming statement subtleties (“gotchas”)
- Lab: modeling a simple ALU and testbench

### Day Two

#### Synthesizing RTL Models

- General synthesis guidelines
- Running your preferred synthesis compiler
- Lab: synthesize a shift/storage register

#### RTL Models of Combinational Logic

- Always procedures and sensitivity lists
- Continuous assignments
- Synthesis full case and parallel case statements
- SystemVerilog unique and priority decision statements
- Lab: model, verify and synthesize an ALU

#### RTL Models of Sequential Logic

- Flip-flops and latches
- Synchronous and asynchronous inputs
- Lab: model, verify and synthesize a Johnson counter

#### Modeling State Machines

- Modeling Mealy and Moore state machines
- Modeling state encoding sequences
- SystemVerilog enumerated types
- Lab: model, verify and synthesize a UART

### Day Three

#### Modeling Structural Netlists—After Synthesis

- Design hierarchy
- Module instantiation
- Generating arrays of instances
- Parameterized models and redefining parameters
- Verilog constructs used in ASIC/FPGA libraries
- Delay calculation and backannotation
- SDF files
- Lab: model and verify a hierarchical design, simulate with SDF delay backannotation

#### Modeling RAMs and ROMs

- Modeling memories
- Modeling bi-directional ports
- Testing bi-directional ports
- Timing constraints
- Lab: model and verify a dual-port RAM

### Day Four

#### Design Verification with Verilog

- Design documentation
- Using Verilog as a verification language
- Structured tests
- Configurable test benches
- Writing output files
- Reading test vector files
- Adding built-in error checking
- Lab: verify a design using test vectors

#### Verilog Wizardry (Lab Intensive Project)

- Using all aspects of Verilog in a design project
- Simulating and verifying larger designs
- Lab: model and verify an embedded DSP processor

*“The IEEE 1800 SystemVerilog standard enables modeling larger, more complex designs, and, at the same time, makes it easier to write Verilog models that synthesize correctly and optimally. Every design engineer learning Verilog should also learn the synthesizable portions of SystemVerilog, and know how to take full advantage of these powerful extensions to Verilog.”*

Stuart Sutherland, President of Sutherland HDL, Inc.