

## SystemVerilog for Design and Synthesis

### Overview

*SystemVerilog for Design and Synthesis* is a comprehensive workshop covering the complete Verilog Hardware Description Language and the synthesizable portions of the SystemVerilog extensions to Verilog. The synthesizable subset of SystemVerilog includes 2-state data types, user-defined types, enumerated types, structures, and self-verifying decision statements. Emphasis is placed on proper SystemVerilog coding styles for top-down design with synthesis and simulation. Special attention is given to subtleties, such as how blocking and non-blocking assignments affect simulation and synthesis. About forty percent of the course is devoted to hands-on experience in labs that reinforce the principles presented, including a 5-hour final project modeling a small DSP processor.

**Course length:** 4-days on-site. 5-days *eTutored™ live*. 5 to 60 days *eTutored™ self-paced*.

### 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 using Verilog and SystemVerilog for modeling, simulation and synthesis. Both new Verilog/SystemVerilog users, as well as those who are familiar with Verilog/SystemVerilog and desire a more in-depth knowledge of the language, will benefit from this course.

### 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.

### Included Materials

- Comprehensive binder with all PowerPoint slides (printed in color), lab instructions, and supplemental white papers. (*eTutored™ self-paced* courses use an eBook and other online materials instead of a training binder.)
- A handy “*Verilog HDL Quick Reference Guide*” (\$15 value).
- All lab files, including example solutions that illustrate proper and efficient coding styles.

### Software Tools Used

Engineers will use both Verilog simulation and synthesis tools during class labs. The tools used can be the Cadence *NC-Verilog™*, Synopsys *VCS™*, or Mentor Graphics *ModelSim™* simulators, and the Synopsys *DC™* or *SynplifyPro™*, or Mentor *Precision™* synthesis compilers.

### 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 or as an *eTutored™ live* online class. We also offer several public *eTutored™ live* workshops throughout the year. For more information, please visit [www.sutherland-hdl.com](http://www.sutherland-hdl.com), or call us at +1-503-692-0898.

(company names and product names are trademarks or registered trademarks of their respective companies)

## Syllabus — 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.