World Class Verilog & SystemVerilog Training

Sunburst Design - Expert Verilog-2001 for Synthesis & Verification

by Recognized Verilog & SystemVerilog Guru, Cliff Cummings of Sunburst Design, Inc.

Cliff Cummings is the only Verilog & SystemVerilog Trainer who helped develop every IEEE & Accellera Verilog, Verilog Synthesis and SystemVerilog Standard.

4 Days
50% Lecture, 50% Lab
Advanced Level

Course Objective

Simply stated, to give engineers *world class* Verilog, synthesis & verification training using award winning materials developed by renowned Verilog & SystemVerilog Guru, Cliff Cummings

Upon completion of this course, students will:

- Write efficient synthesizable Verilog-2001 RTL models
  - includes six different FSM coding styles
  - includes multi-clock and FIFO design techniques
  - includes experimentation with different synthesis coding styles
  - includes generating post-synthesis gate-level netlists for gate-level simulation
  - includes generating SDF files for gate-level timing simulations
- Write complex self-checking testbenches
  - includes building Verilog simulation-configuration files
  - includes working with actual Verilog project directory structures in all labs
  - includes using proven techniques for generating self-checking tests

Course Overview

Sunburst Design - Expert Verilog-2001 for Synthesis & Verification is a 4-day fast-paced intensive course on design and RTL coding styles for synthesis and verification. This is a design course, not a language syntax course.

The 1,000+ page binder for this 4-day course covers all of the important Verilog-2001 RTL coding styles for synthesis. More than a dozen problematic coding styles that cause mismatches between pre-synthesis and post-synthesis simulations are explained, and award-winning materials describing the problems associated with the synthesis directives "full_case" and "parallel_case" are both detailed in lecture and observed in lab exercises. Numerous proven RTL coding guidelines are taught and explained.

Complete coverage of blocking and nonblocking assignments is also presented using materials from two of Cliff's award-winning presentations on nonblocking assignments. 70 slides and a copy of an award-winning paper on nonblocking assignments with delays support the coding guidelines presented. Additional materials show transport delay-mode modeling for mixed RTL and gate-level simulation. 100+ pages will help you to master one of the most difficult and powerful topics in the Verilog language. No other course comes close to this coverage.

Another 100+ slides and two more award winning papers help to demonstrate multiple Finite State Machine (FSM) expert techniques including detailed techniques using four efficient FSM coding styles, and warnings about two common and inefficient coding styles. Labs include PCI bus arbiters and partial PCI target FSM designs (no silly traffic-light controllers, black jack games or soda pop change machines in this course!)

Included in the class are techniques for synchronous vs. asynchronous reset design and doing multi-asynchronous clock design. These techniques are not only advanced Verilog techniques, they are also advanced digital design techniques not covered by college courses. With more than 100 slides dedicated to this section and four award-winning and highly acclaimed papers for added reference material, this section alone is worth the price of admission.

After three days of coding, simulating and synthesizing actual designs, engineers will be shown the tricks and techniques used to build all of the self-checking testbenches used to test the RTL models on days 1-3. In-depth lecture on day four of the course is dedicated to teaching techniques for creating self-checking testbenches, including correct application of stimulus vectors and proper timing of output capture for verification while making use of the "SIMUTIL" self-checking testbench tasks and functions that were used to write all of the testbenches for the synthesis labs. All students will be given a copy of the SIMUTIL testbench utilities to take back and use at their own companies on their own designs.

A 1,000+-page student guide and 49-page Verilog-2001 HDL Quick Reference Guide supplement the lecture and provide excellent reference material for after the class. Numerous exercises and labs reinforce the principles presented. After you take this course, you will understand why Cliff Cummings has won 13 "Best Paper" awards for presentations made over the past 13 years.

By design, this course has more labs than can probably be completed in four days. Students will code each lab experimenting with multiple coding styles to observe first-hand the impact that coding styles have on synthesis results. You will learn the best coding styles through experimentation and comprehensive lab reviews. All of the synthesis labs include pre-coded Verilog headers and are accompanied by self-checking testbenches to maximize experimentation and minimize useless typing. Gate-level Verilog models for an ASIC library are included to permit experimentation with post-synthesis, gate-level simulation with SDF backannotation. All labs solutions are given to each course participant who completes the course.

FPGAs or ASICs? - There are small differences in the course materials based on the target audience. FPGA combinational coding and synthesis issues differ slightly from ASIC issues, so companies are encouraged to indicate their synthesis preference. For open enrollment classes and upon request, both ASIC and FPGA topics and lab reviews are included to accommodate both synthesis styles.

Target Audience

Sunburst Design - Expert Verilog-2001 for Synthesis & Verification

Target Audience

Expert Verilog-2001 for Synthesis & Verification is intended for experienced ASIC and FPGA design and verification engineers that require in-depth instruction on Verilog-2001 RTL synthesis coding styles, advanced state machine design techniques, advanced multi-clock design techniques and Verilog verification skills. Engineers with less Verilog and synthesis experience will complete fewer labs. Engineers with more Verilog and synthesis experience will complete most of the labs.

Prerequisites (mandatory)

This is a design course, not a language syntax course. This course assumes that students have a practical working knowledge of Verilog HDL or have completed Verilog HDL training. Engineers with VHDL synthesis experience and some Verilog exposure will do well in this class. Engineers with no prior HDL training or experience will struggle in this class. *This is an advanced class, not just an advanced beginner class!*

The Sunburst Design - Advantage

Who is teaching your "expert" and "advanced" classes? Most companies will not tell you because their instructors might not have much design experience or may never have participated on any of the Verilog Standards groups or presented at industry recognized conferences. Go to our web site and read about the Sunburst Design - Instructors - they are simply the best at what they do and they have the experience and qualifications to offer best-in-class training.

Sunburst Design Courses:

- *Sunburst Design - Advanced SystemVerilog for Design & Verification* - 4 days
    - *Sunburst Design - Advanced SystemVerilog for Design* - 3 days
    - *Sunburst Design - Advanced SystemVerilog for Verification* - 3 days
- *Sunburst Design - Expert Verilog-2001 for Synthesis & Verification* - 4 days
    - *Sunburst Design - Expert Verilog-2001 & Coding for RTL Design & Synthesis* - 2 days
    - *Sunburst Design - Expert Verilog-2001 Design, RTL Synthesis & Verification Techniques* - 2 days
- *Sunburst Design - Comprehensive Verilog-2001 Design & Best Coding Practices* - 4 days
    - *Sunburst Design - Introduction to Verilog-2001 & Best Coding Practices* - 2 days
    - *Sunburst Design - Advanced Verilog-2001 Knowledge & Design Practices* - 2 days
    - *Sunburst Design - Accelerated Introduction to Verilog-2001 & Best Known Coding Practices* - 1 day
- *Advanced Verilog PLI Courses* - (taught by a Sunburst Design training partner)

Course Customization? - Sunburst Design courses can be customized to include ***your*** company's coding guidelines or to modify the course for a different audience. Sections can be added or deleted from a course to meet you company's needs.

Course Syllabus

Day One

Introduction & Overview of Verilog Synthesis Resources

Configurations
- Important Verilog basics that are not generally understood by most Verilog users. Verilog-1995 and Verilog-2001 configurations, command lines switches, project directory structures, and commands for conditionally compiled Verilog designs and testbenches.

- Essential Verilog basics
- Working with project directory structures
- Verilog command switches
- Conditional compilation for design and verification
- Verilog-2001 Configuration files

Latches & Priority Encoders
- Introduction to Verilog synthesis design flows. Detailed description of two synthesis problem areas: latches and priority encoders. Detailed description of the synthesis directives "full_case" and "parallel_case", and why they should generally be avoided.

- always blocks & sensitivity lists
- Generating latches
- Generating priority encoders
- "full_case parallel_case," the "evil twins!"
- V2K1 & Verilog RTL Synthesis-2002 attributes
- Latch & priority encoder guidelines

Combinational Logic I
- RTL coding styles for combinational logic, including problems and inefficiencies that arise from poor coding styles. Includes Verilog-2001 (V2K1) combinational logic enhancements. Numerous combinational labs demonstrate many potential problem areas related to common combinational coding styles.

- Introduction to synthesis design flows
- Continuous assignments
- Always blocks
- V2K1 @* and comma-separated sensitivity lists
- Instantiated library elements
- Instantiated primitives
- Labs: Combinational labs I

SDF Backannotation
- This section details SDF (Standard Delay Format) file generation, writing Verilog gate-level netlists, gate-level simulations and gate-level simulations using SDF timing (may be important for simulation of full-board and system designs where Static Timing Analysis is not generally practical)

- SDF file basics
- Gate-level netlists
- SDF back annotation of gate-level designs
- Directory structures and command files for gate-level simulation
- Simulating with SDF files
- Conditional compilation of SDF files
- Lab: Post-synthesis gate-level netlist generation and simulation with SDF timing

Day Two

Combinational Labs Review
- An in-depth review of all the coding styles used in the combinational labs and the synthesized results. Conclusions are drawn about which coding styles infer the most efficient logic implementations.

- Review of Day-One labs

Combinational Logic II
- Additional RTL coding styles for combinational logic, including more problems and inefficiencies that occur from poor coding styles. Verilog-2001 (V2K1) enhancements are discussed including reasons to avoid over-usage generate statements. Example of poor usage includes I/O pad instantiation (use the more concise and better supported Array of Instance). More combinational labs demonstrate many potential problem areas related to common combinational coding styles.

- Synthesizable and non-synthesizable Verilog constructs
- Bitwise -vs- logical operators
- Tasks & functions
- Tri-state drivers
- Bi-directional busses
- Instantiating ASIC/FPGA library primitives
- (Synopsys) instantiating DesignWare components
- Guidelines
- Labs: Combinational labs II

Sequential Logic
- This section covers coding styles for sequential logic. Inferring efficient designs using adders and other large resources is also detailed. Also discusses and includes advantages and disadvantages of instantiation.

- Edge-sensitive sensitivity list
- Basic asynchronous & synchronous resets
- Additional flip-flop coding styles
- Simulation/synthesis differences
- Simulation efficiency
- Register banks
- Memories
- Instantiating Blocks
- Resource sharing
- Labs: Sequential labs

Synchronous & Asynchronous Reset Design
- Detailed material for selection and usage of synchronous and asynchronous reset design taken from actual design experiences.

- Synchronous vs. asynchronous resets
- Reset removal metastability
- Asynchronous reset synchronizer circuitry
- Reset distribution trees and techniques
- Multiple clock domains reset synchronization

Day Three

Nonblocking Assignments
- Detailed instruction on how Verilog blocking and nonblocking assignments work. Verilog scheduling algorithms are discussed and blocking and nonblocking assignment synthesizable model, usage-guidelines are presented. Includes important information that must be considered when doing mixed RTL and gate-level simulations.

- Simple flip-flops
- Blocking assignments
- Nonblocking assignments
- Inertial and transport delays
- Delay lines
- Mixed RTL & gate-level simulations
- Guidelines

IEEE Verilog 2001 Enhancements
- A concise summary of the important IEEE Verilog-2001 enhancements for both design and verification.

- The V2K1 top-5 enhancement requests

- V2K1 Multi-dimensional arrays
- Array of Instances (AOI) & V2K1 generate statements
- V2K1 Enhanced File I/O
- V2K1 reentrant tasks and functions
- V2K1 configurations
- New V2K1 port and parameter styles
- V2K1 sensitivity lists
- V2K1 RTL enhancements
- V2K1 signed arithmetic
- V2K1 IP development enhancements
- V2K1 miscellaneous enhancements
- Guidelines
- Labs: Combinational labs II

State Machines
- Fundamental and advanced coding styles for state machines. Includes important considerations for coding designs for easy debug and optimal synthesis. Why parameter state definitions are used instead of `define. Binary encoded, and efficient onehot coding styles are presented. FSMs with combinational outputs and sequential outputs are also presented.

- State machines
- Moore & Mealy styles
- Two always blocks implementation - combinational outputs
- Output assignments using always blocks and continuous assignments
- One always block implementation - Inefficient - avoid this style
- Three always block coding style - registered outputs
- Indexed one-hot implementation - registered outputs
- Encoded one-hot implementation - Inefficient - avoid this style
- Output encoded style - registered outputs
- Modified Mealy FSM to register outputs
- Efficiencies
- An early look at SystemVerilog 3.0 enumerated types
- Quick introduction to multi-clock design issues
- Labs: State machine labs experimenting with different coding styles and the "full_case parallel_case" synthesis directives. Simple multi-clock FIFO lab (behavioral model)

Day Four

Multi-Clock Design
- Detailed material for efficient coding, synthesizing, and verifying of multi-clock designs, taken from actual design experiences.

- Multi-clock metastability and synchronization
- Multi-bit synchronization
- Multi-clock design techniques
- Multi-clock partitioning and naming conventions
- Synthesis scripting considerations

- Static timing analysis considerations
- Gate-level simulation considerations

Multi-Clock FIFO Design
- Detailed material for efficient implementation of multi-clock FIFO designs.

- Multi-clock FIFO simulation inadequacies
- Gray codes
- RTL Gray code counters (ASIC vs. FPGA tradeoffs)
- Synchronized RTL FIFO coding style
- Asynchronous full-empty FIFO coding style technique

Testing & Testbenches
- This section covers various testbench stimulus and verification techniques, including use of Verilog tasks to generate bus functional models. Includes a detailed discussion of hierarchical referencing to probe designs and generate self-checking testbenches. Also details the contents of SIMUTIL, a set of simulation tasks for rapid development of self-checking testbenches.

- V2K1 named parameter passing and defparam avoidance
- V2K1 file I/O enhancements
- An early look at .name and .* SystemVerilog 3.0 implicit port connections
- Testing approaches
- Advanced loops, tasks, and functions & V2K1 automatic tasks and functions
- Bus functional model testing
- Global variables
- Hierarchical referencing
- ASCII "string" display
- Another early look at SystemVerilog 3.0 enumerated types
- Dumpfiles and waveform viewer strategies
- Self-checking tests
- Correct timing and techniques for applying stimulus and verifying results
- SIMUTIL self-checking testbench tasks
- Regression test generation
- Self-adapting pipeline testing

Classroom Details
Training is generally conducted at your facilities. For maximum effectiveness, we recommend having one workstation or PC for every two students, with licenses for your preferred Verilog simulator (we often can help provide the simulator and temporary training licenses).

For more information, contact:

Cliff Cummings - cliffc@sunburst-design.com - Sunburst Design, Inc. - 503-641-8446