



## Modular SystemVerilog

SystemVerilog is a significant new language based on the widely used and industry-standard Verilog® hardware description language. The SystemVerilog extensions enhance Verilog in a number of areas, providing productivity improvements for RTL designers, verification engineers and for those involved in system design and architecture. SystemVerilog is expected to become IEEE standard 1800 in 2005.



Modular SystemVerilog consists of four modules, which can be combined and customised into a single program to fulfil team-based training requirements. Presented from a vendor independent perspective, the workshops can be provided in the context of the customer's own choice of tools, including Mentor Graphics QuestaTMSim.

The Modular SystemVerilog package is only available for in-house or customer-dedicated training.

Modular SystemVerilog comprises:

- **Verilog Primer for SystemVerilog (1-3 days)**
- **Fundamentals of SystemVerilog (2 days)**
- **SystemVerilog Assertions (1 day)**
- **SystemVerilog Testbench Automation (2 days)**

Based on these modules, team-based and in-house training can be customised to fit the content, scope and duration needed to best-fit a specific customer requirement. In particular, the following SystemVerilog topics can be emphasised:

- Using SystemVerilog as a "better Verilog", with an emphasis on synthesis and bus modelling.
- Using SystemVerilog Assertions for design and verification
- Increasing verification productivity by utilising testbench automation techniques.
- Applying the SystemVerilog language to the task of verifying existing Verilog or VHDL designs.

### Who should attend?

- Hardware design engineers who wish to become skilled in the practical use of SystemVerilog for digital hardware design and verification
- Verification engineers who wish to become skilled in the practical use of SystemVerilog for creating testbenches



## What will you learn?

- The SystemVerilog core language, data types and interfaces
- How to synthesize hardware from a SystemVerilog description
- How to call functions written in C from SystemVerilog and how to call SystemVerilog tasks and functions from a C program
- How to write and apply assertions and use an assertion-based verification methodology
- How to model busses, and apply efficient synthesis and verification techniques
- How to create structured testbenches and use cycle-based timing
- How to use object-oriented programming as implemented by SystemVerilog classes and how to apply this to constrained-random test stimulus generation
- How to create and use functional coverage information

## Pre-requisites

- Verilog Primer for SystemVerilog. A good working knowledge of VHDL is essential. This is not suitable as a first course in a hardware description language. If you have not used either VHDL or Verilog, you should attend the Doulos Comprehensive Verilog course instead.
- Fundamentals of SystemVerilog. A working knowledge of Verilog or attendance of the Verilog Primer for SystemVerilog is essential.
- SystemVerilog Assertions. The ability to read and understand simple examples of VHDL or Verilog code is essential, and experience running HDL simulations is recommended. The ability to write original VHDL, Verilog or SystemVerilog code is not required. Attendance of Fundamentals of SystemVerilog, whilst not absolutely essential, is strongly recommended.
- SystemVerilog Testbench Automation. Attendance of Fundamentals of SystemVerilog is essential. Attendance of Assertion-Based Verification with SVA is strongly recommended.

## Structure and Content

- Verilog Primer for SystemVerilog (2 or 3 days)
  - Introduction
  - Verilog Basics
  - Combinational Logic
  - Sequential Logic
  - Miscellaneous Topics
  - Test Fixtures
  - Supplementary Subjects
  - Behavioural Verilog
  - Gate Level Verilog
- SystemVerilog Fundamentals (2 days)
  - Introduction
  - Verilog-2001 and Verilog-2005



- Improving RTL Productivity
- Improving RTL Productivity...continued
- Interfaces
- Transaction-Level Modelling
- Using C functions in SystemVerilog Simulations
- SystemVerilog Assertions (1 day)
- Assertion-Based Verification
- The SystemVerilog Assertions Language (SVA)
- Properties, Assertions and Sequences
- More on Properties and Sequences
- Introduction
- Testbench Structure and Timing
- Modelling for Verification
- Verification Methodology
- Classes – the Building-Blocks of Automated Testbenches
- Constraining and Directing the Stimulus Generation
- Checking the simulation results
- Functional Coverage