SystemVerilog (IEEE 1800™) 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.

Modular SystemVerilog consists of several modules that can be combined and customised into an integrated program to fulfil team-based training requirements. Workshops comprise approximately 50% of class time, and are based around carefully designed exercises to reinforce and challenge the extent of learning.

Doulos is uniquely qualified to give you the complete view of SystemVerilog’s capabilities in any tool context. Active in SystemVerilog training development and project support since 2003, our comprehensive expertise combined with corporate independence, teamed with co-operative vendor relationships, enables Doulos delegates to benefit from objective tuition whilst learning in the context of their chosen tool and methodology. Leading tools supported for this class include:

- **Simulation**: Cadence Incisive®, Mentor Graphics Questa™Sim, Synopsys VCS®
- **Synthesis**: Mentor Graphics Precision™, Synopsys Design Compiler®, Synplicity Synplify®

Other tools may be available on request. Please contact Doulos if your preferred tools are not listed here.

**Who should attend?**

- Engineers and managers who wish to use or evaluate SystemVerilog for ASIC or FPGA design or verification
- EDA support engineers who wish to understand how their customers use SystemVerilog

**What class modules are available?**

- **Fast-track Verilog for VHDL Users** (1-3 days) provides a secure grounding in the essentials of the classic Verilog language, enabling engineers whose HDL experience has been predominantly in VHDL to gain maximum benefit from SystemVerilog training modules.
- **Fundamentals of SystemVerilog for Design** (1½ days) trains engineers in the practical use of SystemVerilog for synthesisable RTL design.
- **Fundamentals of SystemVerilog for Verification** (½ day) provides verification engineers with the necessary background in SystemVerilog to embark on the remaining verification modules. (Offered for verification teams in place of Fundamentals of SystemVerilog for Design)
- **SystemVerilog Assertions** (½ day) teaches the principles of assertion-based verification and design, key features of the SystemVerilog assertion language for creating your own custom assertions, and how to package and deploy libraries of assertion checkers.
- **Module-based SystemVerilog Verification** (1 day) shows how to use SystemVerilog to build effective block-level testbenches, building on existing best-practice testbench architecture based on Verilog modules.

For further information contact your local Doulos Sales Office.
Modular SystemVerilog

In-house Training Module Options

- **Class-based SystemVerilog Verification** describes how to write sophisticated object-oriented testbenches using SystemVerilog’s testbench automation capabilities, which support a constrained-random, coverage-driven verification methodology. These features enable you to write testbenches at higher levels of abstraction and be more productive than is possible with standard hardware description languages. The material leverages Doulos's years of experience in teaching object-oriented verification concepts, making these challenging topics accessible to engineers with a wide variety of backgrounds and providing ideal preparation for your subsequent adoption of a sophisticated verification methodology.

- **Adopter Classes** (2-3 days) provide vital additional information, code examples and know-how to enable you to deploy vendor-advocated verification methodologies such as, **OVM** (Cadence & Mentor) **AVM** (Mentor) **URM** (Cadence) and **VMM** (Synopsys).

Prerequisites

A working knowledge of Verilog is essential. Engineers already proficient in VHDL can follow the **Fast-track Verilog for VHDL Users** module to meet this prerequisite. For engineers with other backgrounds Doulos can offer tailored Verilog training, either as a separate class or integrated into a team-based training package. Contact Doulos to discuss the options that best suit your needs.

Which modules should you choose?

The modular packaging allows Doulos to work flexibly with you to prepare a customized training class that matches your specific needs. We have carefully planned two standard tracks to meet the majority of training requirements, and you are invited to consider one of these as a starting point for any discussion about customization (see diagram on the next page):

**Design Group Track (3 - 5 days)**

This track is designed to meet the needs of RTL design teams who wish to use SystemVerilog to enhance their designs, and who expect to create design-specific module-level testbenches using SystemVerilog to improve verification productivity and quality.

- Fast-track Verilog for VHDL Users *(if required)*
- Fundamentals of SystemVerilog for Design
- SystemVerilog Assertions
- Module-based SystemVerilog Verification

**Verification Specialist Track (4 - 7 days)**

This track is focused on the needs of verification teams who wish to use SystemVerilog either to move their verification activity forward to use modern testbench automation techniques, or who aim to use SystemVerilog to replace other testbench solutions. It assumes delegates are already proficient in VHDL or Verilog – contact Doulos to discuss options for those who don’t meet this prerequisite.

- Fast-track Verilog for VHDL Users *(if required)*
- Fundamentals of SystemVerilog for Verification
- SystemVerilog Assertions
- Module-based SystemVerilog Verification

For further information contact your local Doulos [Sales Office](mailto:sales@doulos.com).
Modular SystemVerilog
In-house Training Module Options

- Class-based SystemVerilog Verification
- Adopter Class (optional)

Training materials
Doulos training materials are renowned for being the most comprehensive and user friendly available. Their style, content and coverage is unique in the HDL training world, and has made them sought after resources in their own right. The materials include:

- Fully indexed class notes creating a complete reference manual
- Workbook full of practical examples and solutions to help you apply your knowledge

Structure and content of each module

Verilog for VHDL Users (1 to 3 days)

Introduction
What is Verilog? • brief history and current status • the PLI • scope of Verilog • design flow • Verilog-2001 • SystemVerilog • Verilog books and Internet resources

For further information contact your local Doulos Sales Office.
Modular SystemVerilog

In-house Training Module Options

Differences between VHDL and Verilog

“Philosophy” • red tape • strong typing • determinism • data abstraction • structure vs behaviour – nets vs registers • language structure – architecture, packages, configurations, files • identifiers • output ports • implicit wires • arrays • aggregates • signedness • operators • signal vs variables/nets • process vs initial/always • if, case, loop differences • file i/o • hierarchical names

Verilog basics

Modules & ports • continuous assignments • comments • names • nets and strengths • design hierarchy • module instances • primitive instances • text fixtures • $monitor • initial blocks • logic values • vectors • registers • numbers • output formatting • timescales • always blocks • $stop and $finish • using nets and variables correctly

Combinational logic

Event control • if statements • begin-end • incomplete assignment and latches • unknown and don’t care • conditional operator • tristates • case, casez and casex statements • full_case and parallel_case directives • for, repeat, while and forever loops • integers • self-disabling blocks • combinational logic synthesis

Sequential logic

Synthesising flip-flops & latches • avoiding simulation race hazards • nonblocking assignments • asynchronous & synchronous resets • clock enables • synthesizable always templates • designing state machines • state machine architectures • Verilog code-based FSM strategy • state encoding • unreachable states & safe design practices • one-hot machines

Other features of Verilog

Verilog operators • part selects • concatenation & replication • shift registers • conditional compilation • parameterisation and generate • hierarchical names • arithmetic operators and their synthesis • signed and unsigned values • memory arrays • RAM modelling and synthesis • $readmemb and $readmemh

Tasks and functions

Understanding tasks • task arguments • task synchronization • tasks and synthesis • functions

Test fixtures

File I/O – writing to files; file access using MCDs; reading from files • automated design verification using Verilog • force and release • gate-level simulation • back annotation using SDF • “traditional” Verilog libraries • configuration and libraries • command-line options • behavioural modelling

Behavioural Verilog

Algorithmic coding • synchronization using waits & event control • concurrent-disabling of always blocks • named events • fork & join • high-level modelling using tasks, implicit FSMs and concurrent-disabling • understanding intra-assignment controls • overcoming clock skew • blocking and nonblocking assignments • continuous procedural assignment

For further information contact your local Doulos Sales Office.
Modular SystemVerilog

In-house Training Module Options

Fundamentals of SystemVerilog for Design (1½ days)

The SystemVerilog data type system
enum • typedef • struct • union • packed/unpacked • packages and $unit • using arrays in SystemVerilog • array and structure literals, assignment patterns

Nets and variables
Key changes in Verilog-2005 and SystemVerilog • continuous assignment to variables • modified driver and connection rules • data types on ports and nets

Modules and processes
Port connection shorthand • type parameters • synthesis idioms for processes • miscellaneous improvements to the language

Design applications of interfaces
The interface construct • interfaces to encapsulate communication • modports • synthesis of interfaces and modports • imported functions for design

Fundamentals of SystemVerilog for Verification (½ day)

Key features of the SystemVerilog data type system
enum • typedef • struct • union • packed/unpacked • packages

Nets and variables
Key changes in Verilog-2005 and SystemVerilog

Testbench applications of interfaces
The interface construct • modports

SystemVerilog Assertions (½ day)

Introduction to assertions
Assertions, properties, sequences • clocking and sampling • property implication • uses of assertions • simulation of assertions • formal tools

Assertion methodology
Methodology consequences of assertion-based design and verification • assertion and assumption • benefits of assertions to the designer • protocol checkers

A brief introduction to SVA syntax
Writing simple assertions of your own • sequences and the ## operator • repetition and time ranges • sequence fusion • overview of temporal operators • local variables and actions in assertions

Packaging assertions
Assertions in interfaces and modules • the bind construct • deploying verification IP, particularly assertion-based IP

For further information contact your local Doulos Sales Office.
Modular SystemVerilog

In-house Training Module Options

Module-based SystemVerilog Verification (1 day)

Verification for design groups
Bus functional models • testbench architecture in classic Verilog • stimulus and response timing

Using SystemVerilog to construct module-level testbenches
Clocking blocks to manage timing • testbench applications of interfaces • task and function enhancements in SystemVerilog • decoupling test cases from the testbench

Dynamic data types
strings • queues • dynamic arrays • associative arrays • queue and array methods • foreach loop

Testbench automation
Introduction to testbench automation concepts • randomisation, checking and coverage • the need for constraints • randomisation of stimulus data using std::randomize and traditional Verilog distribution functions • procedural randomisation: randcase, randsequence • collecting functional coverage data

Class-based SystemVerilog Verification (2 days)

Introducing classes
SystemVerilog’s class syntax • describing stimulus data and a stimulus generator • randomization of class members (without constraints) • objects and references • constructors and new • shallow copy using new • writing a custom copy method

Hooking classes to the DUT
Dynamically-constructed test environment vs. statically-elaborated DUT and test harness • using virtual interface and class-based BFMs • the role of clocking and program blocks • appropriate structure for DUT, clock generators and other structural elements • constructing and launching the test environment using program+initial • simple class-based testbench architecture

Varying the Stimulus
Generator template objects • introduction to constraints • implication constraints • derived classes • upcasting and the is-a relationship • virtual methods

Components and Channels
FIFO channels to decouple components • base class for transaction data • downcasting and $cast • parameterized classes and macros for specialization • running self-contained components with fork…join

Reusable Testbench Components
Maintaining a component instance hierarchy • virtual base class for components • launching a task with fork…join_none • testbench component architecture • preview of standard methodologies (OVM, VMM)

Monitor and Check Components
Passive monitors and unbounded FIFOs • checker components and scoreboards • stopping the test cleanly • semaphore for mutual exclusion

For further information contact your local Doulos Sales Office.
Modular SystemVerilog
In-house Training Module Options

Coverage in Classes
Coverage-driven TBA methodology • coverage planning as the first step in a verification process • analysing and interpreting coverage data • SystemVerilog coverage constructs in detail • covergroup sampling • per-instance coverage in testbench components • covergroup options • transition and cross coverage • controlling bins structure • coverage reports

Adopter Classes
Each of these classes provides a quick-start, practical introduction to one of the vendor-advocated verification methodology approaches that are rapidly gaining popularity in the SystemVerilog verification community. They can be presented as a 2-3 day class giving hands-on experience of the chosen methodology, or ½ to 1-day overview to introduce the key ideas and benefits. They are designed for use within the context of Doulos team-based SystemVerilog verification training, but can also be used stand-alone.
- OVM Adopter Class
- AVM Adopter Class
- URM Adopter Class
- VMM Adopter Class

Project services
Doulos Project Services enable clients to access our world-leading technical know-how and apply it directly to projects. Expert-on-call, Expert-design and Expert-support options can be flexibly packaged and delivered to provide valuable expertise and additional resource just when it is needed.

How to book a class
To make a provisional booking, or to obtain pricing information, please contact your local Doulos sales team. You will find contact details on our website.

Doulos acknowledges all trademarks and registered trademarks as the property of their respective owners.