

Comprehensive Verilog

Standard Level - 4 days

Comprehensive Verilog is a 4-day training course teaching the application of the Verilog[®] Hardware Description Language for programmable logic and ASIC design.

The syllabus covers the Verilog language, coding for register transfer level (RTL) synthesis, developing test fixtures, and using Verilog tools. The course also provides an overview of SystemVerilog.

Because Doulos is independent, delegates can usually use their choice of design tools during the workshops. Workshops are based around carefully designed exercises to reinforce and challenge the extent of learning, and comprise approximately 50% of class time.

Who should attend?

- Engineers about to embark on their first Verilog design project
- Engineers who have already acquired some practical experience in the use of Verilog, but wish to consolidate and extend their knowledge within a formal training environment using the tools of their choice.
- Engineers who are proficient in VHDL but need to become competent in the application of, and interaction with, Verilog HDL as well

What will you learn?

- How Verilog fits into the FPGA/ASIC design flow
- How to use the Verilog language for hardware design and logic synthesis
- How to write thorough Verilog test fixtures to verify your designs
- How to avoid common mistakes when coding Verilog for synthesis

Pre-requisites

Delegates must have attended Essential Digital Design Techniques (or equivalent), or have a good working knowledge of digital hardware design. No previous knowledge of VHDL or software language experience is required.

Course materials

Doulos course materials are renowned as 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. Course fees include:

- Fully indexed course notes creating a complete reference manual
- Workbook full of practical examples to help you apply your knowledge
- Doulos Golden Reference Guide for Verilog language, syntax, semantics and tips
- Tool tour guides (to support the tools and technologies of your choice).

Continued...

For further information contact your local Doulos [Sales Office](#).



Comprehensive Verilog

Standard Level - 4 days

Structure and content

What is Verilog? ♦ Scope of Verilog ♦ Design flow for ASICs, CPLDs and FPGAs ♦ Introduction to synthesis ♦ Synchronous design ♦ Timing constraints ♦ Verilog books and internet resources

Modules

Modules & ports ♦ Continuous assignments ♦ Wire assignments ♦ Comments ♦ Names ♦ Nets and strengths ♦ Design hierarchy ♦ Module instances ♦ Primitive instances ♦ Text fixtures ♦ \$monitor ♦ Initial blocks ♦ Logic values ♦ Vectors ♦ Registers

Numbers, Wires and Regs

Numbers ♦ Output formatting ♦ Timescales ♦ Always blocks ♦ \$stop and \$finish ♦ Using wires and registers correctly

Always Blocks

Event control ♦ If statements ♦ Begin-end ♦ Incomplete assignment and latches ♦ FPGAs and latches ♦ Unknown and don't care ♦ Conditional operator ♦ Tristates

Procedural Statements

Case, casez and casex statements ♦ full_case and parellel_case directives ♦ For, repeat, while and forever loops ♦ integers ♦ Self-disabling blocks ♦ Combinational logic synthesis

Clocks and Flipflops

Synthesising flip-flops & latches ♦ Avoiding simulation race hazards ♦ Nonblocking assignments ♦ Asynchronous & synchronous resets ♦ Clock enables ♦ Synthesizable always templates ♦ RTL synthesis technology ♦ Inferring flip-flops ♦ Making best use of RTL synthesis

Operators, Parameters, Hierarchy

Bitwise, reduction, logical and equality operators ♦ Part selects ♦ Concatenation & replication ♦ Shift registers ♦ Conditional compilation ♦ Parameters ♦ Hierarchical names

Finite State Machines

Designing state machines ♦ State machine architectures ♦ Verilog code-based FSM strategy ♦ State encoding ♦ Unreachable states & safe design practices ♦ One-hot machines

Synthesis of Arithmetic and Counters

Arithmetic operators and their synthesis ♦ Signed and unsigned values ♦ Adder architectures ♦ WYSIWYG arithmetic synthesis ♦ Resource sharing ♦ Integer and vector arithmetic

Tasks, Functions and Memories

Understanding tasks ♦ Task arguments ♦ Task synchronization ♦ Tasks and synthesis ♦ Functions ♦ Memory arrays ♦ RAM modelling and synthesis ♦ \$readmemb and \$readmemh

Test Fixtures

Designing test fixtures ♦ Writing to files ♦ File access using MCDs ♦ Reading from files ♦ The Verilog Programming Language Interface (PLI) ♦ Automated design verification using Verilog ♦ Force and release ♦ Gate-level simulation ♦ Back annotation using SDF. ♦ PLD and ASIC design flow ♦ Verilog libraries ♦ Command-line options ♦ Behavioural modelling

Continued...

For further information contact your local Doulos [Sales Office](#).



Comprehensive Verilog

Standard Level - 4 days

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 ♦ Understanding unsynthesizable Verilog constructs

Project Management & Good Practice

Writing Verilog for simulation and synthesis ♦ Coding standards for synthesis ♦ File organisation ♦ Design data control ♦ Functional validation ♦ Methodical testing ♦ Hierarchical design ♦ Gated clocks ♦ Asynchronous design

Supplementary Subjects

The PLI

What is the PLI? ♦ What is the PLI for? ♦ How to use the PLI ♦ TF, ACC and VPI routines ♦ creating tests in C

Gate Level Verilog

Structural Verilog ♦ Using built-in primitives ♦ Net types & drive strengths ♦ UDPs ♦ Gate, net & path delays ♦ Specify blocks ♦ Smart paths ♦ Pulse rejection ♦ Cell library modelling

Verilog-2001

What is Verilog-2001? ♦ “Cosmetic” changes ♦ General enhancements ♦ Parameterisation and generate ♦ Configuration and libraries ♦ File I/O

SystemVerilog

Background ♦ Who is SystemVerilog for? ♦ Current status of SystemVerilog ♦ RTL enhancements ♦ Interfaces ♦ Assertions ♦ Testbenches ♦ C interface

Verilog is a registered trademark of Cadence Design Systems Inc

Pre-cursor courses

- Essential Digital Design Techniques (2 days)
Book Essential Digital Design Techniques at the same time as Comprehensive Verilog and **save 10%** with our Fast-track Designer Induction pricing. Ask the sales team for details.

Follow-on courses

- Expert Verilog (incorporating Design and Verification modules)
- Comprehensive SystemVerilog
- Modular SystemVerilog (including tool specific variants)

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.

For further information contact your local Doulos [Sales Office](#).

