Designing with Verilog

Course Description

This comprehensive course is a thorough introduction to the Verilog language. The emphasis is on writing Register Transfer Level (RTL) and behavioral source code. This class addresses targeting Xilinx devices specifically and FPGA devices in general. The information gained can be applied to any digital design by using a top-down synthesis design approach. This course combines insightful lectures with practical lab exercises to reinforce key concepts. You will also learn advanced coding techniques that will increase your overall Verilog proficiency and enhance your FPGA optimization. This course covers Verilog 1995 and 2001.
In this three-day course, you will gain valuable hands-on experience. Incoming students with little or no Verilog knowledge will finish this course empowered with the ability to write efficient hardware designs and perform high-level HDL simulations.

Level: FPGA 1

Training Duration: 3 days

Who Should Attend?

Engineers who want to use Verilog effectively for modeling, design, and synthesis of digital designs

Prerequisites:

- Basic digital design knowledge

Software Tools:

- Vivado® Design or System Edition 2014.1

Hardware:

- Architecture: N/A*
- Demo board: Kintex®-7 FPGA KC705 board*

Skills Gained: After completing this training, you will be able to:

- Write RTL Verilog code for synthesis
- Write Verilog test fixtures for simulation
- Create a Finite State Machine (FSM) by using Verilog
- Target and optimize Xilinx FPGAs by using Verilog
- Use enhanced Verilog file I/O capability
- Run a timing simulation by using Xilinx Simprim libraries
- Create and manage designs within the Vivado™ Design Suite environment
- Download to the evaluation demo board

Course Outline

1. Hardware Modeling Overview
2. Verilog Language Concepts
3. Modules and Ports
4. Demo: Multiplexer
5. Lab 1: Building Hierarchy
6. Introduction to Testbenches
7. Lab 2: Verilog Simulation and RTL Verification
8. Verilog Operators and Expressions
9. Continuous Assign Statements
10. Lab 3: Memory
11. Verilog Procedural Statements
12. Lab 4: Clock Divider and Address Counter
13. Controlled Operation Statements
14. Lab 5: n-bit Binary Counter and RTL Verification
15. Verilog Tasks and Functions
16. Advanced Language Concepts
17. Finite State Machines
18. Lab 6: Finite State Machines
19. Targeting Xilinx FPGAs
20. Lab 7: Implement and Download
21. Advanced Verilog Testbenches
22. Lab 8: Using Verilog File I/O

Lab Description

The labs for this course provide a practical foundation for creating synthesizable RTL code. All aspects of the design flow are covered in the labs. The labs are written, synthesized, behaviorally simulated, and implemented by the student. The focus of the labs is to write code that will optimally infer reliable and high-performance circuits. The labs culminate in a functional calculator that students verify in simulation.