SystemVerilog for Design and Verification

Course Title

SystemVerilog for Design and Verification

Course Description

This course presents SystemVerilog for design and verification. SystemVerilog is an extension of Verilog that combines both hardware description and verification into one language.

SystemVerilog introduces some new constructs that allow more robust RTL coding and better compile-time checks compared to Verilog.

SystemVerilog provides a number of new features, such as object-oriented modeling, constrained randomization, functional coverage, assertions and others that enable powerful verification methods for increasingly complex chips.

SystemVerilog is also used to develop UVM to provide standard verification methodology. Verification can be performed with SystemVerilog alone, or in combination with UVM.

Target Audience

This course is intended for those who are involved in digital design and verification. They include chip design engineers, verification engineers and technical managers.

Prerequisite

Verilog language

Duration

- 4 days (9 a.m. – 5 p.m.)

Course Presentation

- Each participant will receive training materials and source code files for lab exercises.
- Lecture is 70%, lab exercises is 30%
- The lab exercises are mostly coded and ready to run. Participants will study and understand the code, run them, and review the outputs and waveforms. In some lab exercises, there will be minor changes to run different scenarios
- Each participant will receive a Certificate of Attendance after completing at least 75% of the course

Course Instructor

Chang Chee Keng, Sys-ASIC Designs Pte Ltd

C.K. Chang, B.S.E.E., M.S.E.E., has 30 years of experience in chip design and verification, and has participated in a number of chip development projects. He was a technical staff of AMCC/JNI Corporation (Singapore) and participated in design, verification and architecture of 10Gbit Fibre Channel host adapter ASICs. Prior to that, he founded Digital Media Interconnect Pte Ltd to provide consulting and training services to companies in Singapore and the region. Previous to that, he was a Senior Staff Engineer at Silicon Systems (Singapore) Pte Ltd, a subsidiary of Texas Instruments Inc, and was involved in the design of Firewire storage integrated circuits. Prior to returning to Asia, he worked in Silicon Valley, California, USA, at ESS Technology, 3DO Company and Sun Microsystems in design, verification and architecture of various ASIC development projects.

Contact

Sys-ASIC Designs Pte Ltd
21 Bukit Batok Crescent
#09-79, WCEGA Tower
Singapore 658065
Tel: (65) 6022 1812
Email: ckchang@sys-asic.com
Website: www.sys-asic.com

Course Contents

SystemVerilog for Design and Verification

Chapter 5: Tasks and Functions (Pages 100-111)

Comparing Tasks and Functions

ANSI and non-ANSI styles

Formal Argument Directions and Data Types

input, output, inout, ref

Executing Tasks

Default Argument and Name Binding

Static Tasks

Automatic Tasks

Static and Automatic Variables

Argument Pass By Value

Argument Pass By Reference

SystemVerilog Functions

Automatic and Constant Functions

Chapter 6: Processes and Control (Pages 112-130)

Procedural Statements

Types of always Procedures

always_comb, always_latch, always_ff

always_comb vs always @*

always_comb for RTL coding

always vs always_ff

always_ff for RTL coding

Simulation and Synthesis Behavior

Initial and Final Procedures

Parallel Block Statements

fork-join, join_any, join_none

Procedural Timing Control

Delay Control

Intra-Assignment Control

@ Operator Event Control

posedge, negedge, edge

or Event Control

Level-Sensitive and Edge Control

Conditional Qualifier iff

Process Control wait fork, disable fork

Event Synchronization and Trigger

Chapter 7: Arrays and Structures (Pages 131-147)

Array Definition and History

Packed and Unpacked Arrays

Accessing Array Elements

Array Size and Dimensions

Dynamic Arrays

Associative Arrays

Queues and Queue Methods

General Array Methods

Structure Definition

Packed and Unpacked Structures

Structure Assignments

Chapter 8: Modules and Hierarchy (Pages 148-170)

Module Header Definition

ANSI and non-ANSI Styles

Connecting Module Instances

Driving Module Outputs

Compilation Unit

Compilation-Unit Scope

Top-Level Modules

$unit and $root

Packages Overview

Package Reference with Full Scope

Explicit import pkg::

Wildcard import pkg::*

Package in Module Headers

Package Importing other Packages

Package Chaining with export

Package Use Cases

Package vs Include Usages

Chapter 9: Interfaces (Pages 171-182)

Interfaces Overview

Interface with Functionalities

Interface in Verification Testbench

Interface connecting to RTL code

Interface with modport

Interface with ports

Multiple Interface Instances

Virtual Interface

Virtual Interface use in UVM

Chapter 10: System Tasks and Scheduling Semantics (Pages 183-193)

Utility System Tasks and Functions

$countbits, $countones, $isunknown

Sampled Value Functions
$rose, $fell, $changed, $sampled, etc
$past in Assertion
Sequence .triggered Method
Bit Vector Functions
Local Variables and Formal Arguments (Pages 112-117)
SVA Local Variables
Property Evaluation Threads
SVA Formal Arguments
Generating Multiple Assertions
SVA Binding (Pages 118-124)
Assertion Binding and Advantages
Bind Statement
Global and Instance Binding
SVA Directives (Pages 125-139)
SVA Directives – assert, assume, cover, restrict
Simulation and Formal Usages
Cover Property and Cover Sequence
assume and restrict for Formal
Strong and Weak Properties
SVA Event Scheduling (Pages 140-149)
Assertion Event Scheduling
Immediate Assertion Scheduling
Concurrent Assertion Regions
Use of $sample in Assertions

Course Outline

Day 1:

- Chapter 1: Literals and Basic Data Types
- Chapter 2: Strings and Enumerated Types
- Chapter 3: Operators
- Chapter 4: Assignments and Procedural Statements
- Chapter 5: Tasks and Functions
- Chapter 6: Processes and Control
- Lab Exercises: Sections 1-5

Day 2:

- Chapter 7: Arrays and Structures
- Chapter 8: Modules and Hierarchy
- Chapter 9: Interfaces
- Chapter 10: System Tasks and Scheduling Semantics
- Chapter 11: System Verilog Classes
- Lab Exercises: Sections 6-9

Day 3:

- Chapter 12: Constrained Random Generation
- Chapter 13: Functional Coverage
- Chapter 14: Direct Programming Interface
- Chapter 15: Introduction to UVM
- Lab Exercises: Sections 10-11, Testbenches 1-3

Day 4:

- Lab Exercises: Testbenches 4-5
- Verification Methodologies Discussions
- Chapter 16: SystemVerilog Assertions
- SVA Lab Exercises: 1-14