UE21EC343AB5 VERIFICATION OF DIGITAL SYSTEMS (4-0-0-0-4)

Course Description: Verification of Digital Systems will impart training in understanding the complexities of digital system verification and the importance of hardware verification languages (HVLs). The course describes the lexical elements of the system Verilog and equips the students to create the layered test benches for modern-day complex digital designs in SystemVerilog. Digital Design & Computer Organization is required for this course as a prerequisite.

Course Objectives:

- Understanding the complexity of Digital verification
- Understanding the OOPS in System Verilog
- Understand and use the System Verilog RTL design and synthesis features, including new data types, literals, procedural blocks, statements, and operators, relaxation of Verilog language rules, fixes for synthesis issues, enhancements to tasks and functions, new hierarchy and connectivity features, and interfaces.
- Appreciate and apply the System Verilog verification features, including classes, constrained random stimulus, coverage, strings, queues, and dynamic arrays, and learn how to utilize these features for more effective and efficient verification.
- Understanding of complete layered testbench of System Verilog

Course Outcomes:

Students completing the course should be able to

- Understand the necessity of Verification
- Understand the complexity of Verification: directed testbench and layered testbench.
- Students will be able to write system Verilog code for both design and verification.
- Understand the constraints for verification and other constructs like interface, clocking etc.
- Students using System Verilog constructs will be able to achieve better functional coverage.

Course Content:

Unit 1: Introduction & Connecting The Testbench: Verification Guidelines: The Verification Process, Basic Test bench Functionality, Directed Testing, Methodology Basics, Constrained-Random Stimulus Data Types: Built-In Data Types, Fixed-Size Array, Dynamic Arrays, Associative Arrays, Array Methods, Choosing a Storage Type, Creating New Types with typedef, Creating User-Defined Structures, Type conversion, Constants, Strings Procedural Statements And Routines: Procedural Statements, Tasks, Functions, Void Functions, Task and Function Overview, Routine Arguments, Returning from a Routine, Local Data Storage, Time Values.

Separating the Testbench and Design: The Interface Construct, Stimulus Timing, Interface Driving, and Sampling, Connecting It All Together, Top-Level Scope, Program - Module Interactions, System Verilog Assertions, The Four-Port ATM Router, and The ref Port Direction. **15 Hours**

Unit 2: Basic OOP & Randomization:

Basic OOP: Where to Define a Class, OOP Terminology, Creating New Objects, Object Deallocation, Using Objects, Static Variables vs. Global Variables, Class Methods, Detaining Methods Outside of the Class, Scoping Rules, Using One Class Inside another, Understanding Dynamic Objects, Copying Objects, Public vs. Local, Straying Off Course, Building a Testbench.

Randomization: Introduction, What to Randomize, Randomization in System Verilog, Constraint Details, Solution Probabilities, Controlling Multiple Constraint Blocks, Valid Constraints, In-line Constraints, The pre_randomize and post_randomize Functions, Random Number Functions, Constraints Tips and Techniques, Common Randomization Problems, Iterative, and Array Constraints, Atomic Stimulus Generation vs. Scenario Generation, Random Control, Random Number Generators, Random Device Configuration, Conclusion Threads, and Interprocess Communication: Working with Threads, Interprocess Communication, Events, Semaphores, Mailboxes, Building a Testbench with Threads and IPC.

15 Hours

Unit 3: Functional Coverage: Coverage Types, Functional Coverage Strategies, Simple Functional Coverage Example, Anatomy of a Cover Group, Triggering a Cover Group, Data Sampling, Cross Coverage, Generic Cover Groups, Coverage Options, Analysing Coverage Data, Measuring Coverage Statistics During Simulation, Conclusion. **12 Hours**

Unit 4: Advanced Interfaces: Virtual Interfaces with the ATM Router, Connecting to Multiple Design Configurations, Procedural Code in an Interface.

A Complete System Verilog Testbench: Design Blocks, Testbench Blocks, Alternate Tests.

Interfacing with C: Passing Simple Values, Connecting to a Simple C Routine, Connecting to C++, Simple Array Sharing, Open arrays, Sharing Composite Types, Pure and Context Imported Methods, Communicating from C to System Verilog. **14 Hours**

Text Books:

1. "SystemVerilog for Verification: A Guide to Learning the Testbench Language Features", Chris Spear, Greg Tumbush, Springer Publication, ISBN-13: 9781489995001, 2014

Reference Books:

1. "System Verilog For Design Second Edition: A Guide To Using SystemVerilog For Hardware Design And Modeling", Peter Flake, Simon Davidmann, Stuart Sutherland, Springer, 2010, ISBN: 9781441941251.