Summary of Learning Objectives

1: Introduction to Digital Design with Verilog HDL

After this lecture you should be able to:

- define a module with single- and multi-bit `logic` inputs and outputs;
- write Verilog numeric literals in binary, decimal and hexadecimal bases;
- evaluate the value and length of expressions using `logic` signals, arrays, numeric literals and the operators described below;
- draw schematics for Verilog descriptions of multiplexers and registers; and
- use `assign` statements to design combinational logic and `always_ff` statements to design registers.

2: Flip-Flops and Registers

After this lecture you should be able to:

- predict the relationship between the input and output waveforms of D flip-flops, registers, counters and shift registers
- draw block diagrams and write Verilog descriptions for these

3: State Machines

After this lecture you should be able to: design a state machine based on an informal description of its operation, document it using state transition diagrams and tables, write a synthesizable Verilog description of it and convert between these three descriptions.

4: More Verilog

After this lecture you should be able to:

- declare modules with parameters and ports
- instantiate modules using positional, named and wildcard parameters and signals
- declare arrays of arrays and use initialized arrays as lookup tables
- convert between high/low logic levels and true/-false truth values for active-high and active-low interfaces

5: Interfaces

After this lecture you should be able to: classify an interface as serial or parallel, synchronous or asynchronous, uni- or bi-directional and explain the advantages of each; draw the schematic or write the Verilog for a synchronous serial transmitter or receiver; convert data transmitted over an asynchronous serial or SPI interface to the interface waveform(s); extract the data from these waveforms.

6: Timing Analysis

After this lecture you should be able to be able to: identify features and specifications on a timing diagram, identify a specification as a requirement or guaranteed response, apply the terms defined in this lecture, and do calculations involving clock rate, propagation delays and setup/hold time requirements.

7: Memory System Design

After this lecture you should be able to: answer short questions about different memory types and memory terminology; identify read/write timing specifications from a timing diagram (as for flip-flops); draw a schematic showing how memory ICs are connected to increase the data bus size and the total amount of memory; design (combinational) address decoding logic to place memory at specific address ranges.

8: Verification

After this lecture you should be able to select an appropriate verification strategy including: selecting simulation or hardware testing; stimulus-only or self-checking testbenches; selection of test inputs; use of "known-good" models; unit testing; regression testing; distinguish between functional (RTL) and gate-level (timing) simulations; use delays and event controls to generate waveforms in a System Verilog testbench.

9: Implementation of Digital Logic Circuits

After this lecture you should be able to: state which transistors are on and off in a CMOS totem-pole output; determine the direction of current flow between driver and receiver; determine from a data sheet: if an input or output voltage would be high, low or invalid and calculate noise margin; compute the effect of frequency and voltage changes on the power consumption of CMOS logic circuits; determine the RC time constant and current consumption of an open-collector output; describe the causes and consequences of ESD; design simple circuits to convert between logic levels; distinguish between DIP, TQFP, BGA and CSP packages.

10: Programmable Logic Applications and Architectures

After this lecture you should be able to: explain the growth of digital electronics; select software versus hardware and PLDs versus ASICs to solve a particular problem; explain the terms: Moore's Law, ASIC, CPLD, FPGA, feature size, VLSI, fabless, wafer, die, NRE, FPGA, LE and LUT.