Introduction to Verilog and ModelSim

(Part 1 - Background)

Overall Content

- Part 1 – Background
- Part 2 – Verilog Basics
- Part 3 – Running ModelSim
- Part 4 – Combinational Logic
- Part 5 – Sequential Logic

Scope and Objective

- Verilog language standards
  - Mainly Verilog95 (some Verilog2001)
  - Focusing on synthesizable constructs
- ModelSim as simulation tool
  - Other tools can be used
  - Focus on getting familiar with Verilog
- Introductory level design
  - Basic design flow (i.e. entry, checking, simulate)
  - Basic implementation

Computer Aided Design

- CAD tools (software) used in various fields
- Common CAD-based design flow:
  - Design entry (input)
  - Design check (optional)
  - Design simulation/synthesis (process)
- End result (output) can be either used for analysis or further processing (e.g. fabrication)
- May consist multiple stages of similar structure

Design Entry (Electronics)

- Representation of a design
- Classic design entry methods:
  - Graphical - Schematic Capture
  - Text – Netlist, Hardware Description Language (HDL)
- Not-so-classic method:
  - abstract representation (e.g. state diagrams)
- Many digital systems design work involves using HDL (schematics still used for simple systems)

Hardware Description Language

- HDL coding… not programming!
  - Encoding a system in text form
  - May describe structure, function or behavior
- Mostly used HDL:
  - VHDL (VHSIC HDL)
  - Verilog HDL
- Most industry-standard tools supports both VHDL and Verilog

VHDL and/or Verilog

- VHDL
  - strongly typed, verbose
  - efficient at representing a design
- Verilog
  - C-like standard, relatively simple
  - easier for entry-level engineers
- Both has its strength/weaknesses
  - Some uses both! We choose Verilog (… go figure!)

case in-sensitive

Good for system level integration

case sensitive

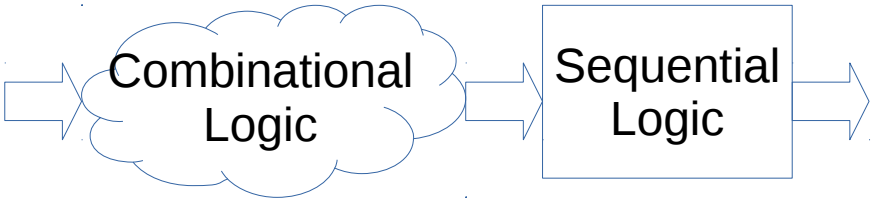
Good for simulation work

HDL Approaches

- Mainly, 3 styles of HDL coding:
  - Structural – netlist format (specify exact electrical lines) → Absolutely synthesizable
  - RTL – functionality of the system (usually in digital logic form) → Should be synthesizable but may not be optimized
  - Behavioral – usually abstract behavior of the system (e.g. black box) → Most of the time NOT synthesizable (i.e. for simulation only)

Register Transfer Level (RTL)

- Any digital systems can be built from logic gates (combinational) and flip-flops (sequential)



- Some tools can generate RTL codes from other advanced forms of graphical entry (e.g. flow charts, truth table, state diagrams)
- For most people, RTL codes are synthesizable codes

Design Simulation

- Many simulation tools for Verilog-based design
  - Icarus Verilog (http://iverilog.icarus.com)
- A commonly used simulation tool is ModelSim
  - Complete design environment by Mentor Graphics
  - Code Editor, Simulator, Waveform viewer