University of Tehran

Electrical and Computer Engineering Department

ECE (8101) 342

Object Oriented Modeling of Electronic Circuits – Spring 1402-03

Homework 2-3

Homework 2: C++ Gate Modeling
Due Date: Farvardin 2024

RTL description
Yosys synthesis
C++ gate-level netlist
C++ procedural gate-level simulation.

In this homework you are to design a 4-bit sequential multiplier at the RT level, synthesize it with Yosys, generate a Verilog netlist, translate it to a C++ netlist, and then convert it to C++ expressions and flip-flops. You are to perform simulations in each step.

1. Design a 4-bit sequential add-and-shift multiplier. The multiplier has a start signal and a ready signal. When *start* is issued the multiplication begins and after the completion of the task, the ready is issued and remain asserted until the next start is issued. Show the Verilog description. Perform Verilog simulations and show that your design works correctly.
2. Synthesize the circuit of Part 1 with Yosys. Use a library of gates as those you used in Assignment 1. If you run into a problem using the gates of Assignment 1, just use NOR gates with 2, 3, and 4 inputs as your library elements. The Verilog output of this part consists of concurrent instantiations of Yosys library elements.
3. Perform post-synthesis Verilog simulation to make sure the synthesis is done correctly.
4. Write a C++ program to order gates in the Yosys generated netlist to instantiate gates that are dependent on other gates are instantiated after instantiation of gated that they depend on. You do not need to handle the flip-flops and the ordered list applies to the combinational part of the circuit only.
5. Manually or using a C++ program convert the ordered Verilog netlist (Part 4) to a netlist of gates according to the gates we discussed in class.
6. Develop a C++ main() for testing your netlist. Read datafiles from external files.
7. Using your program from Assignment 1, convert the Yosys Verilog output (Part 2) to Verilog assign expressions. If Yosys cannot produce a netlist of AND, OR and NOT gates, redo Assignment 1 using NOR gates..
8. Convert the Verilog assign expressions of Part 7 to C++ expressions, add flip-flop models, include the circuit in a C++ main() and run tests to verify functionality of your circuit.