UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department
Digital Logic Design, ECE 367 / Digital Systems I, ECE 894, Fall 1400
Computer Assignment 2
Basic Gates and Simple Functions, Timing Simulation and Synthesis
Week 6-7

Name:
Date:

1. Using 2-state and 3-state NOT gates from Computer Assignment 1, show the design of a 2-to-1 multiplexer with tri-state control. Use NOT gates from Part 1 and 2 of the previous assignment.
   a. Show the schematic diagram of the circuit.
   b. Using worst-case delay values from Computer Assignment 1, manually hand simulate the multiplexer and, in a hand-drawn timing-diagram, show the three delay values To1, To0, and ToZ for the multiplexer.
   c. Write a testbench for the above circuit and apply inputs to cause the circuits' worst-case delays. Make sure the changes on inputs are far enough apart to put the circuits into a steady-state before the next input change.
   d. Compare your hand-simulation and the SystemVerilog simulation and explain the differences, if any.

2. Using the 2-to-1 multiplexer of Part 1, build a 4-to-1 multiplexer. Use inverters as needed. This 4-to-1 multiplexer does not have a tri-state control and its output can only take 1 and 0 values.
   a. Show the schematic diagram of the circuit.
   b. Using worst-case delay values from Problem 1 and Computer Assignment 1, manually hand simulate the 4-to-1 multiplexer and, in a hand-drawn timing-diagram, show the two delay values To1, and To0 for the multiplexer.
   c. Write a testbench for the above circuit and apply inputs to cause the circuits' worst-case delays. Make sure the changes on inputs are far enough apart to put the circuits into a steady-state before the next input change.
   d. Compare your hand-simulation and the SystemVerilog simulation and explain the differences, if any.

3. Using the 2-to-1 multiplexer of Part 1, build an octal 2-to-1 multiplexer. You need to instantiate the 2-to-1 multiplexer eight times.
   a. Show the schematic diagram of the circuit.
   b. The worst-case delay values for this multiplexer are no different than those of Problem 1. Explain the reason for this.
   a. Write a testbench for the above circuit and apply inputs to cause the circuits' worst-case delays.
   b. Use as **assign** statement to describe the octal 2-to-1 multiplexer circuit. Use SystemVerilog's condition operator (?:). Simulate the module with this **assign** statement alongside with the circuit of Part a and report the differences.

4. In this part you are to build an 8-bit two's complementor circuit. You will do so by first building a one-bit two's complementor and then cascade eight of these circuits to build the 8-bit one.
Read this italic part to learn how to build an 8-bit complementor circuit from 1-bit slices. Manually, to convert a number to its two's complement, you start from the right side of the number and you start complementing the bits. This means that each slice of the circuit has a data bit and a convert flag input. Initially the convert flag is 0. The complement slice takes the input and puts it on the output when the convert flag is 0 and complements it when the convert flag is 1. Each slice also has a convert next output. This output becomes 1 when either the convert input flag is 1 or a data bit is 1. The convert next output connects to the convert flag input of the next higher slice. When the first data bit of 1 from the right-hand side is detected, all bits to its left must be complemented.

a. Show the schematic diagram of the circuit.
b. Using gates based on transistors of Computer Assignment 1, hand-calculate the worst-case delay of a slice of a two's complementor circuit.
c. Write a testbench for an 8-bit two's complementor circuit. This circuit is built by eight slices each of which use SystemVerilog gate primitives with appropriate delay values. Verify your hand calculated delay values using this testbench.
d. Use as **assign** statement to describe the 8-bit two's complementor circuit. Note that for two's complementing you need to complement the number and add a 1 to it. Use the add operator. Use delays from Part c to annotate this **assign** statement. Simulate this **assign** statement alongside with the circuit of Part a and report the differences.

5. Using the multiplexer of Problem 3 and the complementor of Problem 4, build an 8-bit absolute value circuit.

a. Show the schematic diagram of the circuit.
b. Find the worst-case delay of the circuit using delays from Problem 3 and Problem 4.
c. In a module write a single **assign** statement to describe the 8-bit absolute value circuit.
d. Synthesize the description of Part c using Yosys.
e. Compare number of gates used in Part a and Part d.
f. Write a testbench for the circuits of Part a and Part c. Annotate the description of Part c to make its delays as closely as possible to those of the circuit of Part a.
g. Be able to evaluate the circuits in terms of gate count and timing delays.

Deliverables:

Generate a report that includes all the items below:

A. For all the above problems hand-drawn schematic diagrams and timing diagrams are required. Your SystemVerilog descriptions must correspond to the circuit diagrams. Your simulation run and the project built for this purpose must be demonstrated to the TA.
B. For Problem 5, your report should include images of the Yosys synthesis report. Simulation project files must be shown in the report. Be prepared to answer questions asked about the timing and gate counts of the varios circuits in this problem.

Make a PDF file of your report and name it with the format shown below:

FirstinitialLastnameStudentnumber-CAnn-ECEmmm

Where *nn* is a two-digit number for the Computer Assignment, *mmm* is the three-digit course number under which you are registered, and hopefully you know the rest. For the *Firstinitial* use only one character. For *Lastname* and for the multi-part last names use the part you are most identified with. Use the last five digits of your student id (exclude 8101) for the *Studentnumber* field of the report file name.