UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department
Digital Logic Design, ECE 367, Fall 1398
Computer Assignment 4
Registers, State machines, and Combinational packages
Week 10

Name:
Date:

1. Implement the serial multi-broadcasting system (SMBS) of Computer Assignment 2. For this purpose, write the behavioral description of this circuit in SystemVerilog (using **always** statements is recommended). After testing it in ModelSim bring it into Quartus and use it as a component.
2. Use cross-coupled NOR gates and any extra logic as required to build a flip-flop with a D input, a falling edge clock, an active-high asynchronous reset, and an enable (EN) input for clock enabling. Base your delay values on those given in Assignment 1. Use this flip-flop to build a 6-bit shift-register. Test your flip-flops by using the shift-register in a testbench to shift its serial input one place to the right with every clock when its enable input is active.
3. Using the proper delay values write a 6-bit shift-register using **always** statements in SystemVerilog. After testing this shift-register in ModelSim bring it into Quartus and use it as a component, SReg6.
4. Using the SMBS of Problem 1, and the shift-registers of Problem 3. Design and implement a multi-broadcast system that simultaneously broadcasts its input over several outputs. The incoming data is serial on *serIn* that is synchronous with the input clock, *clk*. When there is no incoming data, *serIn* stays at 1. After this, the first 1 to 0 transition indicates serial data is starting. In this case the first 0 is just used as the start bit. Following the start bit (0), the next six bits indicate the destination port(s) (i.e., first four bits) and the line number to broadcast data on (i.e., the next two bits). In both cases, the least-significant bits come first in time.

   Immediately after the destination port number, serial data in chunks of 5 bits appear on *serIn*. A data chunk of size 5, is a group of 5 consecutive synchronous data bits that are followed by a spacer. The spacer is one clock duration 0 on *serIn* that is synchronized with the system clock.

   The data received as such will be sent to the specified port, and a *valid* output will accompany the valid data. The spacers are not valid data bits. This continues for as long as a 0 spacer is received after 5 bits of actual data. If a 1 is received instead of a 0 spacer, this indicates a stop-bit and transmission to the specified port ends. For as long as *serIn* remains at 1, no data is received. One again, the first 1 to 0 transition is indication of a 0 start-bit, after which bits for the next destination port start arriving.