CS223 Laboratory Assignment 3

Modeling Decoders and MUXs in System Verilog

Lab dates and times:

Section 1:	11.03.2021 Thursday	13:30-17:20
Section 2:	09.03.2021 Tuesday	08:30-12:20
Section 3:	10.03.2021 Wednesday	08:30-12:20
Section 4:	11.03.2021 Thursday	08:30-12:20

Location: EA Z04 (in the EA building, straight ahead past the elevators)
Groups: Each student will do the lab individually. Group size = 1

Preliminary Report (30points)

Today's lab needs considerable advance preparation. These advance designs and System Verilog models should be prepared in advance, and assembled neatly into a Preliminary Report with a printed cover page and printed pages for the schematics and System Verilog codes. Each page should have a proper heading. The contents of the report should be as follows:

a) A cover page which includes the following: course name and code number, the number of the lab, your name and student ID, date, number of your trainer pack (remember lab policies. You must always use same pack number).
b) Behavioral SystemVerilog module for 2-to-4 decoder and a testbench for it.
c) Behavioral SystemVerilog module for 4-to-1 multiplexer.
d) Schematic (block diagram) and structural System Verilog module of 8-to-1 MUX by using two 4-to-1 MUX modules, two AND gates, an INVERTER, and an OR gate. Prepare a test bench for it.
e) Schematic (block diagram) and SystemVerilog module for F(A,B,C,D)=∑(1,2,5,7,8,9,10,12,15) function, using one8-to-1 multiplexer and an inverter.

You can refer to the slides of chapter 4 of your textbook in Moodle while preparing your modules and testbenches. Don’t forget that you have to submit your Preliminary Report before your sections lab start time.

Part A: Decoders (25 points)

Decoders are widely used in digital design, as a building block. Although they themselves can be built with logic gates, their function is often described (and modeled in System Verilog) rather than their structure. As you will see, decoders can be composed into larger decoders.

A 2-to-4 decoder decodes a 2-bit input binary number by setting exactly one of the decoder's 4 outputs to 1. Unless it has an enable signal, one and only one output of a decoder will ever be 1 at the same time, corresponding to the current value of the inputs. With an enable signal, it is possible to make all the outputs be 0, when the decoder is disabled. When enabled, it behaves as described above. Decoder outputs are mutually exclusive, and in fact are the minterms of the inputs.

a) Write code: Give the SystemVerilog code which models a 2-to-4 decoder in behavioral style.
b) Simulate it: Using the System Verilog testbench code, verify in simulation that your 2-to-4 decoder with enable is working correctly. (Be sure to compare the order of the ports in your module with the order of the ports in the instantiation of your decoder in the testbench, to make sure they match 1-to-1.)
c) Make FPGA project: Now, follow the Xilinx design flow to synthesize, create programming file, and download your 2-to-4 decoder to your BASYS-3 FPGA board.
d) Test it: Using the switches and LEDs on BASYS-3 that you have assigned now, test your decoder. When you are convinced that it works correctly, show the physical implementation results to the TA. Be prepared to answer questions that you may be asked.

Part B: Multiplexers and Boolean function implementation (45points)

A multiplexer ("MUX" for short) is another higher-level building block, used widely in digital design. A M-to-1 multiplexer has M data inputs and 1 data output, and allows only one input to pass through to the output. A set of select inputs determines which input to pass through. MUXs can be composed into larger MUXs, as you will see in this part of the lab.

a) Write code: Give dataflow System Verilog module for a 4-to-1 multiplexer.
b) Simulate it: Simulate your 4-to-1 multiplexer and show it to your TA.
c) Write code: Write structural System Verilog code for an 8-to-1 multiplexer using two 4-to-1 multiplexers, two AND gates, an OR gate and an inverter.
d) Simulate it: Simulate 8-to-1 multiplexer and show it to your TA.
e) Test it: Set up the circuit you designed for $F(A,B,C,D)=\sum(1,2,5,7,8,9,10,12,15)$ in preliminary work in a new module, using a single 8-to-1 multiplexer. Using inverses of signals is allowed. Test your circuit using switches as input and a LED as output. Show your circuit to TA. Be prepared to answer questions that you may be asked.

Submit your code for MOSS similarity testing

Finally, when you are done and before leaving the lab, you need to combine all the design and simulation sources you wrote in a single file named *StudentID_SectionNumber.txt*. This will include all the Systemverilog codes you wrote. You don't need to add the constraints file(*.xdc file). If you have multiple files, just copy and paste them in order, one after another inside text file. Even if you didn't finish, or didn't get the Systemverilog part working, you

must submit your code to the Moodle Assignment for similarity checking. Your codes will be compared against all the other codes in all sections of the class, by the MOSS program, to determine how similar it is (as an indication of plagiarism). So be sure that the code you submit is code that you actually wrote yourself! All students must upload their code to the 'Moodle>Assignment' specific for your section. Check submission time and don't miss it before leaving the lab. After taking a backup of your work, don't forget to delete it from computer. Because students of other sections will work with your system too.

Clean Up

1) Clean up your lab station, and return all the parts, wires, the Beti trainer board, etc. Leave your lab workstation for others the way you would like to find it.

2) CONGRATULATIONS! You are finished with Lab #3 and are one step closer to becoming a computer engineer.

NOTES

--Advance work on this lab, and all labs, is strongly suggested.

--Be sure to read and follow the Policies for CS223 labs, posted in Moodle.