CECS 225 – Spring 2020

Lab Programming Assignment #4

Points Possible: 20

Due Date: Friday, May 1 by 11:59 pm

Instructions:

Submit two files (do not zip them):
Programming Assignment 4-<Name of Submitter>-Design.txt
Programming Assignment 4-<Name of Submitter>-Testbench.txt.

Only submission via BeachBoard will be accepted.

In addition, as a comment at the top within your code for each file, provide the EDA Playground URL for your saved code, so that I can directly access and execute it.

Be sure to also type in your name within each file, at the top as a comment.

Failure to do any of the above will result in point deductions. No exceptions.

Requirements:

A priority encoder has 2^N inputs. It produces an N-bit binary output indicating the most significant bit of the input that is 1 (TRUE).

Using the EDA Playground implement an 8:3 priority encoder. The 8-bit input will have at most one bit as a 1 value. If no bit is 1, then your design should output don't cares for all three bits (xxx).

Example console output of what I expect to see is the following:

```
[2020-03-28 00:48:27 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp
a.out
VCD info: dumpfile encoder.vcd opened for output.
 in=00000000, code=xxx
 in=00000001, code=000
 in=00000010, code=001
 in=00000100, code=010
 in=00001000, code=011
 in=00010000, code=100
Done
```

Note, regarding the example output above:

- It only shows six test cases, whereas I expect to see all nine test cases (scenario with no 1's + the eight scenarios with a 1).
- "in" indicates the input, "code" indicates the encoded output.

You can use use the below design and test bench as a template. The highlighted asterisks indicate the parts of the code that you'd need to replace with your own code.

Instructor: Ali Sharifian

Design:

```
module encoder (in, code);
  input [7:0] in;
  output reg [2:0] code;

  always @ (in)
    begin
      if (in[0])      code = *******;
      else if (in[1]) code = *******;
      else if (in[2]) code = *******;
      else if (in[3]) code = *******;
      else if (in[4]) code = *******;
      else if (in[5]) code = *******;
      else if (in[6]) code = *******;
      else if (in[7]) code = *******;
      else            code = *******;
    end
endmodule
```

Testbench:

```
module testbench;
  reg [7:0] in;
  wire [2:0] code;

  encoder enc (in, code);

  initial
    begin
                    *******
    end
endmodule
```