ECE 274a Lab/Practical Exam 3 and 4 Fall 2018

(You can complete this lab exam in a group of 2 students)

Exam score: 100 points (about 7-8% in your final grade)

Date: Monday November 26, 2018 at 8 am – **Friday November 30, 2018 by 4.45 PM (Demo)**

Submit your code in the designated Dropbox on D2L by 11.59 PM on Friday November 30, 2018

If no HLSM and code submission on D2L, 10 points will be deducted from your score.

If you use structural way in Verilog code, also submit your work for Datapath and Controller.

Late policy: if you demo late, submit all your files in the late Dropbox

50% deduction if demo by 11.45 am Monday December 3, 2018

75% deduction if demo by 4.45 pm Tuesday December 4, 2018

0 point after 4.45 pm 12/14/2018.

When finished,

- **Demo your working prototype on FPGA board and your post-synthesis functional waveform simulation** to the instructor or TAs during lab sessions or office hours.
  10 points deduction if you/your team do not demo this lab exam in person.
  At the demo time, both teammates must be present. The absent one will get 10 points deduction.
- **Each person *(even though you work in a group of two)***, submit
  ALL your design, testbench, .xdc files in the designated Dropbox on D2L.
  your HLSM (and Datapath and controller (if you use structural way in your Verilog code)) in the D2L Dropbox - take a picture (make sure that the submitted pictures are readable)

How we will grade:

- If no HLSM (and Datapath and controller (if you use structural way in your Verilog code)) is shown during the demo time, 10 points will be deducted.
- If works in a team, the person who is absent during the demo time, 10 points will be deducted for that person.

During demo time, first -> demo with the given register file. Then the TA/instructor will ask you to change the content of the register file and demo it again.

a) a correct HLSM but INcomplete code	maximum of 20 out of 100 points
b) a correct HLSM, a complete code but the behavioral simulation is NOT working	maximum of 40 out of 100 points
c) a correct HLSM, a complete code, working behavioral simulation but the post-synthesis simulation is NOT working.	maximum of 50 out of 100 points
d) a correct HLSM, a complete code, both behavioral and post-synthesis simulations are working BUT no code to implement on FPGA board	maximum of 60 out of 100 points
e) a correct HLSM, both simulations work, a code and .xdc to attempt to download to the board but NOT working.	maximum of 70/100 points
f) a correct HLSM, both simulations work, only 'count' is displayed on the board when done (LED) is 1 (LED is lit up)	maximum of 85/100 points
g) all complete and both A[i] for all i and count are displayed correctly on the FPGA board for only the given register file	95/100 points
h) all complete and both A[i] for all i and count are displayed correctly on the FPGA board for both the given register file and the one given by TA/GLA/ULA	100/100 points

Exam Problem: given the C code (pseudo-code) below

1) Convert the C code provided to **HLSM diagram**
2) **Write Verilog code (behavioral or structural way – your choice)**
3) **Perform simulations (both behavioral and post-synthesis) for your HLSM diagram.** Use Clk of 200 ns period to perform both behavioral and post-synthesis functional simulations. See waveform on page 4 when it is working.

   Retaining signals for Register file (all locations) should be used for post-synthesis simulation.
4) **Write the top level code** (using Clk div, two-digit display, any modules/code that you want to use to make it display as required, …**) and .xdc file to implement your designed circuit on the FPGA board** (See details below)

Note: Register file is a component used in this circuit (even though, it shows as Inputs in the given C code below). Verilog code for 16x8 Register file (RegFile16x8) is given on the next page (page 3).
Your system MUST use the given register file.

On FPGA board, use **2-Hz clock frequency** for clock of your circuit.

1) During the time that done = 0, make it display value of ***count***.
2) When done = 1 (LED for done is lit), display
   a) Each ***value of A[i]*** (one at a time starting from i = 0 to i = 15)
   b) ***count*** should be displayed as the last number on the board.

See the video on D2L for working prototype.

At the demo time, you will first demo what you have and then you will be asked to change the content in the register file and then demo it again. Your code should work for any given register file that we use to test.

Inputs: byte A[16], go (1 bit),
Outputs: count (7 bits), done (1 bit)

```
while(!go);
done = 0;
count = 0;
i = 0;
while(i < 16){
        temp = A[i];
         if((temp > 47) && (temp <58))
         {
                count++;
                A[i] = temp - 48;
         }
         i = i + 1;
}
done = 1;
while(1);
```

This means that when your HLSM diagram reaches the state that done = 1, make it stay at that state.

```
`timescale 1ns / 1ps

module RegFile16x8(R_Addr, W_Addr, R_en, W_en, R_Data, W_Data, Clk, Rst);
  input [3:0] R_Addr, W_Addr;
  input Clk, Rst, R_en, W_en;
  output reg [7:0] R_Data;
  input [7:0] W_Data;

  //simple memory declaration
  reg [7:0] RegFile [0:15];

  // Write procedure
  always @(posedge Clk) begin
    if (Rst == 1) begin
      RegFile[0] <= 8'd48;
      RegFile[1] <= 8'd53;
      RegFile[2] <= 8'd68;
      RegFile[3] <= 8'd57;
      RegFile[4] <= 8'd55;
      RegFile[5] <= 8'd59;
      RegFile[6] <= 8'd40;
      RegFile[7] <= 8'd49;
      RegFile[8] <= 8'd31;
      RegFile[9] <= 8'd38;
      RegFile[10] <= 8'd54;
      RegFile[11] <= 8'd50;
      RegFile[12] <= 8'd63;
      RegFile[13] <= 8'd58;
      RegFile[14] <= 8'd70;
      RegFile[15] <= 8'd51;
    end
    else if (W_en==1) begin
        RegFile[W_Addr] <= W_Data;
    end
  end

  // Read procedure
  always @(*) begin
    if (R_en==1)
      R_Data <= RegFile[R_Addr];
    else
      R_Data <= 8'bZZZZZZZZ;
  end

endmodule
```

The waveform below does not show the clock signal to prevent you from trying to reverse engineering to get the HLSM diagram.

Below is what the register file and count should be for the given register file (on page 3) when your circuit is working (done = 1).

When done = 1, count is 8 since there are 8 values that are > 47 and < 58.
The register file contents are changed when its original values are > 47 and < 58.