ECE 274a Lab Exam 3 Fall 2023

Date: Sunday Nov 26, 2023 (8 am) – **4.50 pm Friday December 1, 2023 (Demo due date)**

Dec 4 - 6, 2023 is late demo (see item C below)

Submit your code and **work by hand on D2L by 11.59 pm Dec 6, 2023**

README:

a) **You have an option to work by yourself or form a group of 2 people** (you and another person who does not have to be your lab partner) to complete this lab exam. Both of you MUST show up together at the demo time – otherwise, you are not a group.

b) By submitting your code in the designated D2L Dropbox, **you certify that the work in this exam is entirely your own or your group**. You agree to give the instructor the right to deduct the scores as she sees fit if any part of your work is determined to be similar to the work of other groups. See https://deanofstudents.arizona.edu/policies/code-academic-integrity

c) **Demo** your lab exam during the **week of Nov 27 – Dec 1, 2023** in

any lab sessions: Mon and Wed 9 – 11.50 am, Tues, Thurs and Fri 2-4.50 pm.
my office hours (ECE 456a): Mon/Wed 1.30 – 3.30 pm, Tues 2.30 – 4.30 pm
TA office hours (lab room): Fri 9 – 10.50 am.

late demo:

- Demo during Mon (Dec 4, 2023) 9-11.50 am lab session, 15% deduction of lab exam 3 score.
- Demo during Tues (Dec 5, 2023) 2-4.50 pm lab session, 30% deduction of lab exam 3 score.
- Demo during Wed (Dec 6, 2023) 9-11.50 am lab session, 45% deduction of lab exam 3 score.

No demo to TA or instructor by 11.50 am on Dec 6, 2023, results in ZERO lab exam 3 score.
Submit the code on D2L with no demo to TA or instructor also results in ZERO lab exam 3 score.

d) Demo but fail to submit your lab exam code and work by hand on D2L by 11.59 pm Dec 6, 2023 - 5% deduction in your lab exam 3 score. If you work in a group, you are still required to submit the work in your own D2L Dropbox. **NO late submission.**

e) Bonus points:

- Successfully demo and get 100 points by 4.50 pm Tuesday Nov 28, 2023, 6 extra points
- Successfully demo and get 100 points by 11.50 am Wed Nov 29, 2023, 4 extra points
- Successfully demo and get 100 points by 4.50 pm Thurs Nov 30, 2023, 2 extra point

e) Grading rubric:

- Your system MUST use the given Register File (RF). **Only 1 RF should be used (to do the read and write operations) in your code.**
  - Instantiating/calling more than one Register file results in 25% deduction in your score.
  - Not using the register file in your code results in 25% deduction in your score.
- At the demo time, if you have
  - Only HLSM diagram, max of 10/100 points
  - Incomplete code or too many Syntax errors, max of 20/100 points
  - Complete code but nothing works – no correct output waveform, max of ***25***/100 points
  - ***Complete code*** and if each part is working correctly in each simulation,

	Behavioral simulation	**Post-synthesis** simulation
done	+3	+5
countodd	+8	+15
average (sum is correct)	+15	+25
Write operation in the register file	+15	+30 (require retaining signals)

Note: +xx is the scores added into ***25*** points given that that part is working correctly in either behavioral or posy-synthesis simulation. For example,

- if all outputs work only in *behavioral* sim., you will get ***25*** + 3 + 8+ 15 +15 = 66 points
- if only countodd works in post-synthesis but done, average and register file work only in behavioral sim, you will get ***25*** + 3 + 15 + 15 + 15 = 73 points
- if all outputs work in post-synthesis except the register file (only work in behavioral sim.), you will get ***25*** + 5 + 15+ 25 +15 = 85 points.

Exam Problem: Given a **C-like code**,

Inputs: byte A[16], go (1 bit),

Outputs: done (1 bit), countodd(?? bits), average(?? bits)

Local Storage: This part is your responsibility and part of this exam. You are also to **decide**

1) **minimum** number of bits used for sum such that an overflow will not occur.

2) minimum number of bits used for countodd

```
while(1){
        while(!go);
        done = 0; countodd = 0;
        sum = 0; average = 0;
        for(i = 0; i < 16; i++){
                temp = A[i];
                if ( (temp%2) == 1) {
                        countodd++;
                }
                sum = sum + temp;
                if( i > 0){
                        temp1 = A[i-1];
                        A[i-1] = (temp+temp1)/2;
                }
        }
        average = sum/16;
        done = 1;
}
```

1) **Draw its HLSM**

2) **Write Verilog code(s)** and **testbench** to perform both **behavioral** and **post-synthesis functional simulations** of your designed digital system. Use a clock period of 200ns in the testbench.
   Note: If you decide to use a structural way of Verilog code, you must show at the demo time and submit work by hand of the designed Datapath and Controller.

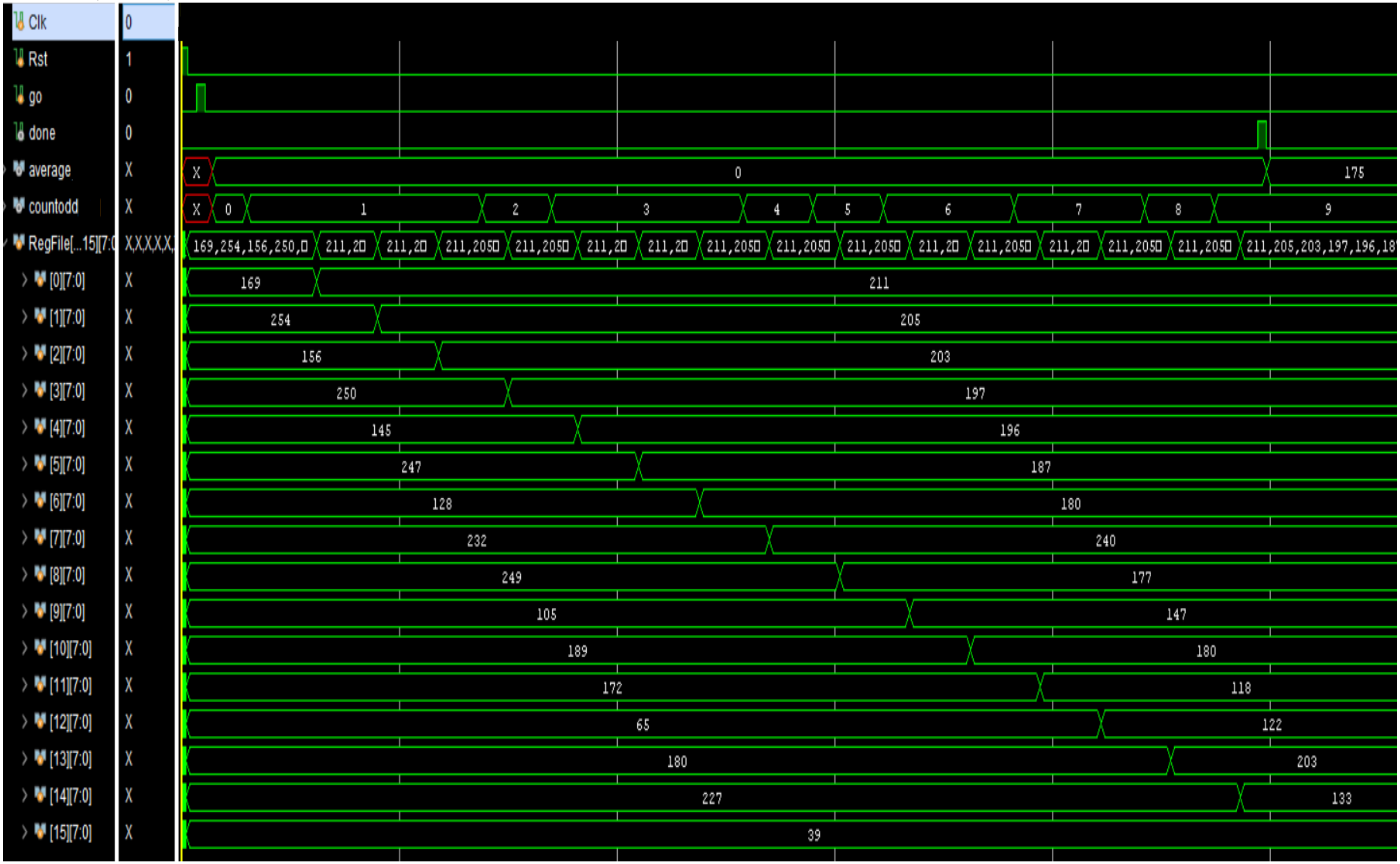
Verilog code for 16x8 Register file (RegFile16x8) is given on the next page to implement an array A in the pseudo code given above.

```
`timescale 1ns / 1ps
module Register16_8(R_Addr, W_Addr, R_en, W_en, R_Data, W_Data, Clk, Rst);
  input [3:0] R_Addr, W_Addr;
  input R_en, W_en;
  output reg [7:0] R_Data;
  input [7:0] W_Data;
  input Clk, Rst;

  reg [7:0] RegFile [0:15];

  always @(posedge Clk) begin
    if (Rst==1) begin
        RegFile[0] = 8'd169;
        RegFile[1] = 8'd254;
        RegFile[2] = 8'd156;
        RegFile[3] = 8'd250;
        RegFile[4] = 8'd145;
        RegFile[5] = 8'd247;
        RegFile[6] = 8'd128;
        RegFile[7] = 8'd232;
        RegFile[8] = 8'd249;
        RegFile[9] = 8'd105;
        RegFile[10] = 8'd189;
        RegFile[11] = 8'd172;
        RegFile[12] = 8'd65;
        RegFile[13] = 8'd180;
        RegFile[14] = 8'd227;
        RegFile[15] = 8'd39;
    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 <= 32'hZZZZZZZZ;
  end
endmodule
```

Behavioral simulation waveform. Note: Clk signal is used but NOT shown below. Number of bits for countodd, average are also NOT shown since it is part of your work.



Post-synthesis functional simulation waveform. Note: Clk signal is used but NOT shown below. Number of bits for countodd, average are also NOT shown since it is part of your work.