ECE 464 / ECE 520
Homework 4

Hand in your solution, using the template at the end of this file as the first page of your solution. You can edit this page but it must be turned in as a single cover sheet.

Question 1

The purpose of this question is to give you practice in reading and interpreting Verilog. [20 points]

Sketch the logic being described in the following Verilog fragments (c is best captured as a Truth Table):

a.
```
wire [31:0] A, Abar, out;
assign Abar = ~A;
assign out = {A[31], A[31:1]};
```

b.
```
wire [7:0] all;
wire [3:0] select;
tri  shared;

assign shared = select[3] ? 1'bz : all[select[2:0]];
```

c.
```
always@(A)
  begin
    for (N = 0; N <=7; N = N+1)
      if (A==N) Y[N] = 1;
      else Y[N] = 0;
  end
```

d. Replace this if-then statement with a casex statement, and also sketch the logic. Hint: Use {} in the case select line.

```
reg A, B, C, D, E, F, G, H;
always@(*)
  if (A) H = F & G;
  else if (B) H = F | G;
  else if (C) H = F ^ G;
  else H = D & E;
```

Question 2

The following code obtains the even parity for the input A. (i.e. parity = 1 if A contains 0 1's or an even number of 1's)

```
reg [7:0] A;
wire parity;

assign parity = ~^A;
```

Show how you would use a for loop in a procedural block to obtain the same function. Hand in your Verilog code. [10 points]

Question 3

Hand-synthesize the following code fragment using a multiplexor.
ie. Draw a schematic showing that this would synthesize too if a 8:1 MUX were used. (Note: It is very unlikely that Synopsys would use an 8:1 mux for this design.)[15 points]

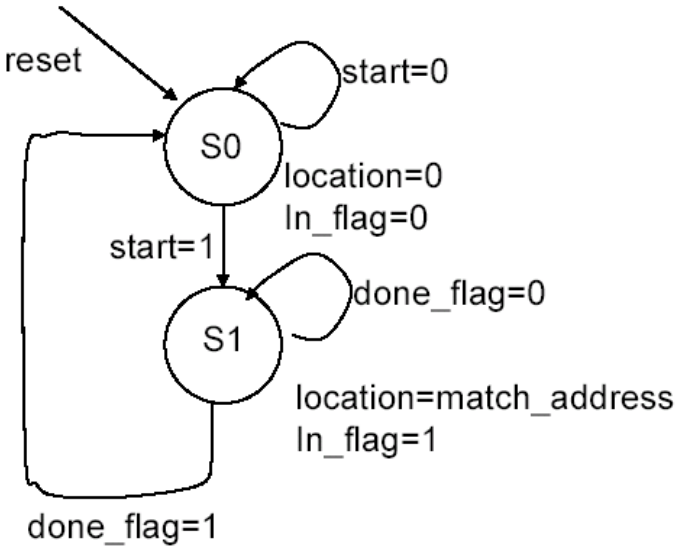
```
input [2:0] X;
input       Y, W;

reg       E, Ereg;

always@(posedge clock)
   Ereg <= E;

always@(X or Y or W)
  begin
    E = 1'b0;  \\ default prevents latches
    casex(X)
     3'b10x : E = Y;
     3'b111 : E = Y & W;
    endcase
  end
```

Question 4
This is a debugging question. Look at the Verilog file badFSM.v in the class locker. Simulate it with badFSM_test.v. Now read it into Synopsys. Identify and fix the problems Synopsys detects during the execution of the 'read' command. Also identify and repair any other potential concerns That you have about the file . The intended Moore FSM is below. Make sure it is consistent with that behavior.



Hand in your final Verilog file. Also hand in a `cut and paste' of the problems identified by Synopsys. [40 points]

Question 5.
Do tutorial 2 for ECE 520. It is available on the EDA Wiki. It describes how to get accurate timing and power numbers for a design. THE TUTORIAL IS CONSIDERED TO BE EXAMINABLE. Take the design and test fixture you generated for HW 3 and run it through this tutorial. Report the following numbers for your design:

a) total chip area after place and route,
b) the power consumed in your hardware: pre and post annotation of parasitics,
c) available timing slack in the critical path, pre and post annotation of parasitics(relax the clock to 10 ns to get a slack).

(Note, in general the total chip area will be about 10% more than the cell area number achieved during synthesis. This is explained in the Tutorial 2) [15 points]

Name: **Student ID:** **Signature:**

Q1.a.

b.

c.

Q2. (code)

Q3. (sketch)

Q4. (Put your Verilog Code on the NEXT page)

Problem (refer to line number or variable name)	**Fix**

Q5. Total Power (pre): **(post):** **Unmet slack (pre):** **(post):**
Cell Area: