

CSE 140L – Summer Session II – 2017

Instructor: Mohsen Imani

Homework #5

Checkoff Deadline: Wednesday, September 6

PROBLEM 1: GENERAL QUESTIONS

Answer the following questions in a concise way. You are encouraged to look for the answer online. These questions will not be graded for the homework and you are not required to provide a report for them or submit anything, but they might be part of the final exam.

- 1) Draw the State diagram for the following verilog code.

```
module fsm( clk, rst, inp, outp);
    input clk, rst, inp;
    output outp;
    reg [1:0] state;
    reg outp;
    always @( posedge clk, posedge rst )
    begin
        if( rst )
            state <= 2'b00;
        else
            begin
                case( state )
                    2'b00:
                        begin
                            if( inp ) state <= 2'b01;
                            else state <= 2'b10;
                        end
                    2'b01:
                        begin
                            if( inp ) state <= 2'b11;
                            else state <= 2'b10;
                        end
                    2'b10:
                        begin
                            if( inp ) state <= 2'b01;
                            else state <= 2'b11;
                        end
                    2'b11:
                        begin
                            if( inp ) state <= 2'b01;
                            else state <= 2'b10;
                        end
                endcase
            end
        end
    end

    always @(posedge clk, posedge rst)
    begin
        if( rst )
            outp <= 0;
    end
end
```

```

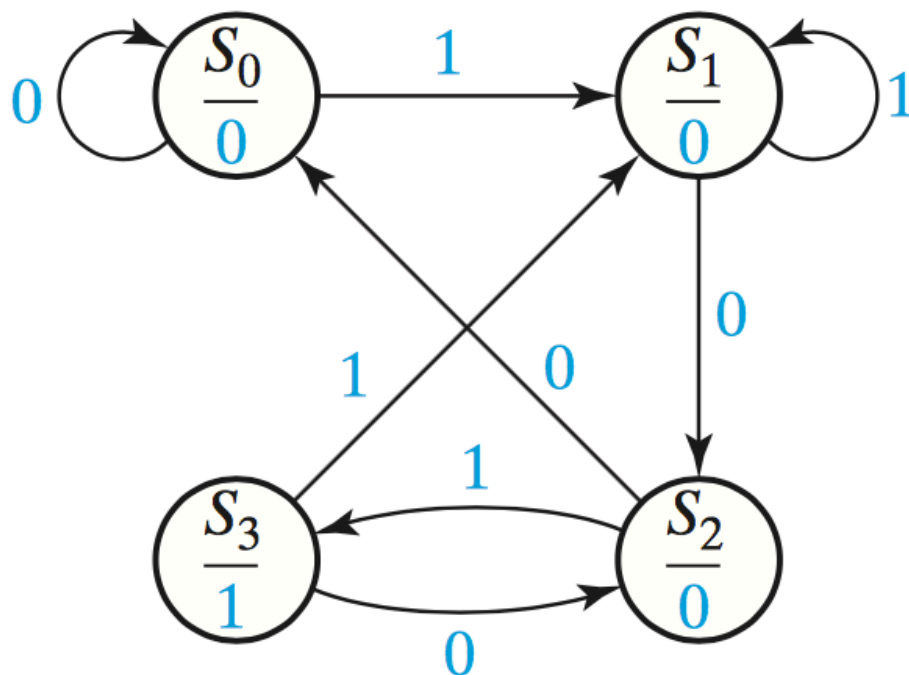
        else if( state == 2'b11 )
            outp <= 1;
        else outp <= 0;
    end
endmodule

```

- 2) Explain briefly what is transition and states in FSM and how would you model them in verilog.
- 3) Write the verilog code for a JK master-slave flip flop and use it to create a T flipflop
- 4) List few commonly used encoding techniques in digital design and their advantages
- 5) Briefly describe the sources of clock signal in the real world electronic devices, think what keeps your wrist watch or a wall clock ticking.
- 6) What is the metric to determine the number of flipflops we need for implementing a given FSM?
- 7) Does better verilog coding help contain the power requirements and area utilized by the hardware? Explain with an example.

PROBLEM 2: FINITE STATE MACHINE: FROM DIAGRAM TO VERILOG

An FSM has a 1-bit input, a 1-bit output and 4 states (S_0 is the starting state). The behavior is described by the state diagram below. Implement this FSM in Verilog.



PROBLEM 3: FINITE STATE MACHINE: FROM DESCRIPTION TO VERILOG

Implement in Verilog a finite state machine with a 1-bit input (X) and a 1-bit output (Z). The machine examines groups of four consecutive inputs and produces an output $Z = 1$ if the input sequence 0101 or 1001 occurs, and $Z = 0$ otherwise. The machine resets after every four inputs.

PROBLEM 4: TRAFFIC LIGHT CONTROLLER

Design an FSM for a traffic light controller with a 1-bit input and a 2-bit output. Pressing the button gives an input = 1, which turns the light from red (output = 10) to green (output = 11). The light has to be green for 5 clock cycles, then it turns to orange (output = 01) for 2 clock cycles before becoming red again.

PROBLEM 5: BINARY DIVISIBILITY BY 3

Construct an FSM that checks if a binary number is divisible by 3. Specifically, your FSM should take the bits sequentially (i.e. one by one) starting from the LSB and output REM = 1 if the number is not divisible and REM = 0 if it is divisible. Do not try *modulo* statement in verilog, as it is not synthesizable. Modulo only works for mod2 (This is synthesizable using shift operations).

Checkoff List (To be completed with a tutor, a TA or the instructor by Wednesday 09/06).

NOTE: Before starting the checkoff, you MUST submit your zipped project via TED. The submitted material MUST contain all the source code. The checkoff can only be attempted once, so make sure that your solution is working fine.

For two of the FSMs you implemented in Problems 2 to 5, selected by TA/tutor, you will be asked to:

- 1) Explain the functionality including state diagrams
- 2) Review the Verilog code
- 3) Compile the code
- 4) Review and run testbench to verify functionality

Design your testbench to make the checkoff procedure as easy as possible. You should be able to explain to your grader what is the FSM functionality and how you designed your testbench according to that.