University of Engineering and Technology, Peshawar

Department of Computer Systems Engineering.

Course Lab: CSE-308 Digital System Design

Zubair Khan **19** PWCSE **1797**

Section A

Batch 21 (Spring_2022)

Submitted to Engr. Madiha Sher

LAB # 9 TITLE

IMPLEMENT A TRAFFIC LIGHT CONTROLLER USING FSM.

INTRODUCTION:

The lab this week will continue the introduction to FSMs with a simple Traffic Light Controller design project. In the lab, you are required to create a state diagram of a Mealy machine, which implements a traffic light controller, based on provided guidelines. You will then use this state diagram to write the behavioral Verilog description of the traffic light controller. The testing of your traffic light controller will take place during the lab session using the ISE development tools and the Spartan 6 board. As with the previous lab, we will make use of the character LEDs to display the outputs of our digital circuit.

THE TRAFFIC LIGHT CONTROLLER:

This week you will design a traffic light controller for the highway and farm road intersection shown in Figure 1. For simplicity, we will assume that the traffic lights on opposite ends of the intersection are the same; in other words, if the light for the North bound traffic is green, then the light for the South bound traffic is also green. The traffic light controller outputs two 2-bit signals, highwaySignal and farmSignal, for the highway road and the farm road, respectively.

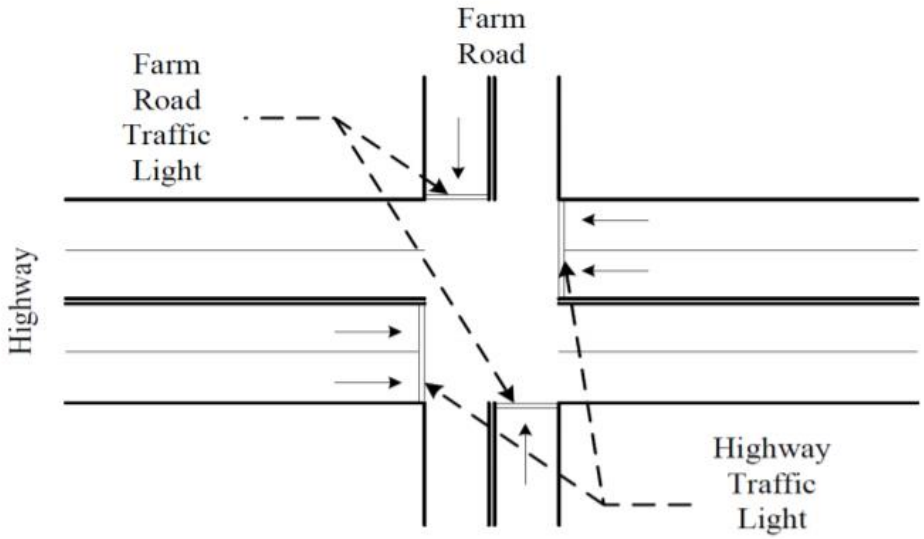


Figure 1: Example Illustration

The following encoding is used for both signals:

001 : Green

010 : Yellow

100 : Red

Obviously, to avoid accidents, the highway lights and farm road lights must not be green at the same time. Furthermore, the transitions from green on each road must be followed by 3 seconds of yellow. Because the traffic on the highway is much greater than that of the farm road, the traffic light must remain green on the highway longer than on the farm road. For this design, the highway light must remain green unless there is a vehicle on the farm road, while the farm road light must remain green for only 10 seconds.

The brain of the traffic light controller is the FSM. For the initial design, Vehicle, Clock and Rst constitute the only input to the FSM. The output of the FSM includes the highwaySignal and farmSignal. The operation of the FSM came be described by using a state diagram as shown in figure 2.

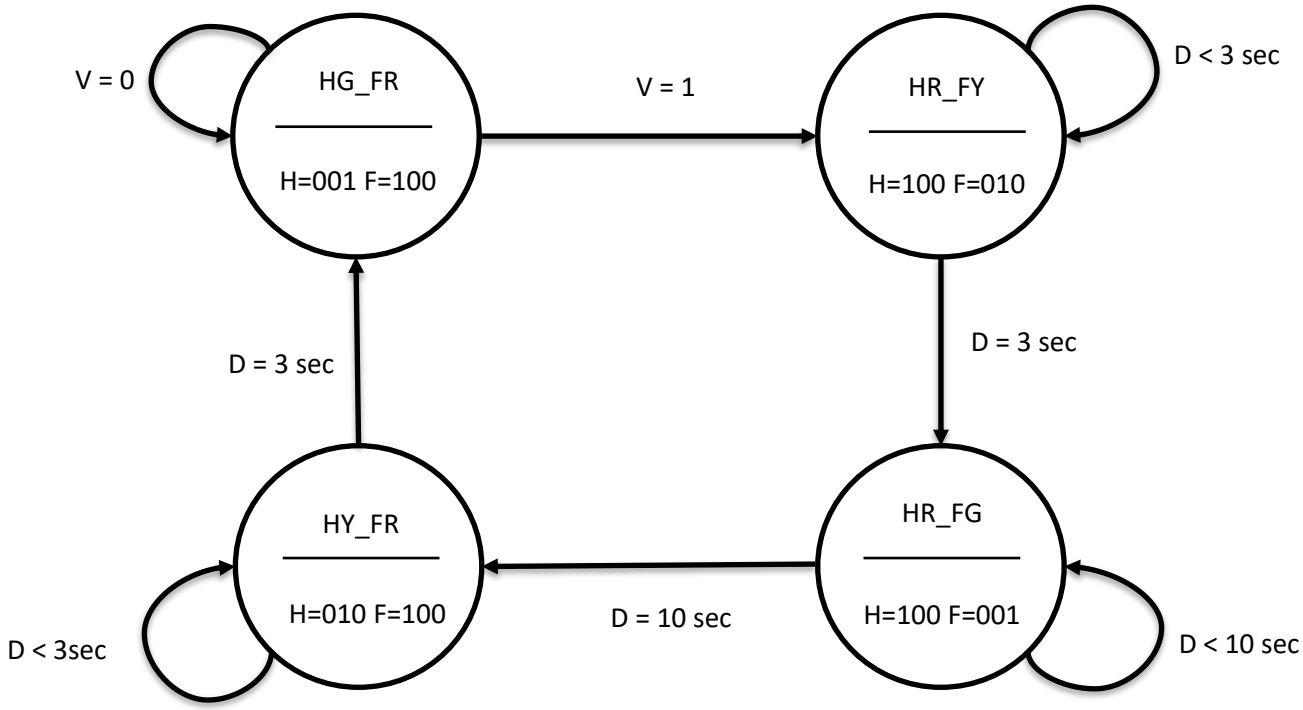


Figure 2: State Transition Diagram

CODE:

```
module D_FF (q, d, clock, reset);
```

```
        output reg q;
        input d, clock, reset;
        always @(negedge clock or posedge reset)
        begin
                if (reset)
                        q = 1'b0;
                else
                        q = d;
        end
endmodule


module syncronizer(
   input clk, rst, in,
   output out
);
   wire w1;
   D_FF DF01   (w1, in, clk, rst);
   D_FF DF02   (out, w1, clk, rst);
endmodule



module clk_divider(clk, clk1, rst);
        integer cnt;
        input clk, rst;
        output reg clk1;
        always @(posedge clk)
                begin
                        if(rst)
                                begin
                                        cnt = 0;
                                        clk1 = 0;
                                end
                        else
                                begin
                                        cnt = cnt + 1;
```

```
                                        if(cnt == 100000000)
                                                begin
                                                        clk1 = ~clk1;
                                                        cnt = 0;
                                                end
                                end
                        end
endmodule


module L2P(
   input clk1, rst, in,
   output reg out
);

   syncronizer sync (clk1, rst, in, synin);

   parameter s0 = 0;
   parameter s1 = 1;

   reg  state;
   reg next_state;

   always @(posedge clk1)
           if(rst)
                   begin
                           state = s0;
                   end
                   else
     state = next_state;

 always @(*) begin

    case (state)
    s0:
     begin
        if(synin == 0)
        begin
```

```
            out = 0;
            next_state = s0;
          end
          else if(synin== 1)
          begin
            out = 1;
            next_state = s1;
          end
       end
     s1:
     begin
        if(synin== 0)
        begin
           out = 0;
           next_state = s0;
        end
        else if(synin == 1)
        begin
           out = 0;
           next_state = s1;
        end

     end

     endcase
  end
endmodule

//module lock (on, off, reset,clk,onpulse,offpulse, out);


module traffic_controller(reset, clk, VEHICAL_IN, HighWayLight, FarmLight,state7seg,enable);

   input reset, VEHICAL_IN, clk;
   output reg [2:0] HighWayLight, FarmLight;
          output [6:0] state7seg;
          output enable;
```

```
reg [1:0] state, next_state;
        assign enable =0;
clk_divider clk_Div (clk, clk1, reset);
L2P l1(clk1, reset, VEHICAL_IN, VEHICAL);
        seven_seg seg1(state, state7seg);

parameter RED = 3'b100;
parameter GREEN = 3'b001;
parameter YELLOW = 3'b010;

parameter FR_HG=0;
parameter FY_HR=1;
parameter FG_HR=2;
parameter FR_HY=3;

always @(posedge clk1)
          if(reset)
                    begin
                              state = FR_HG;
                    end
                   else
      state = next_state;

always @(*) begin
   case(state)
      FR_HG:
         begin
            if(VEHICAL == 1)
            begin
               next_state = FY_HR;
               FarmLight = YELLOW;
               HighWayLight = RED;
            end
            else
            begin
               next_state = FR_HG;
               HighWayLight = GREEN;
               FarmLight = RED;
```

```
            end
          end

        FY_HR:
          begin
            FarmLight = YELLOW;
            HighWayLight = RED;
            next_state = FG_HR;
          end

        FG_HR:
          begin
            FarmLight = GREEN;
            HighWayLight = RED;
            next_state = FR_HY;
          end

        FR_HY:
          begin
            FarmLight = RED;
            HighWayLight = YELLOW;
            next_state = FR_HG;
          end
      endcase
  end
endmodule



module seven_seg(in, out);
  input [1:0] in;
  output [6:0] out;
       assign out = (in == 0) ? 7'b1000000:
                                (in == 1) ? 7'b1111001:
                                (in == 2) ? 7'b0100100:
                                (in == 3) ? 7'b0110000:  7'b1111111;
endmodule
```

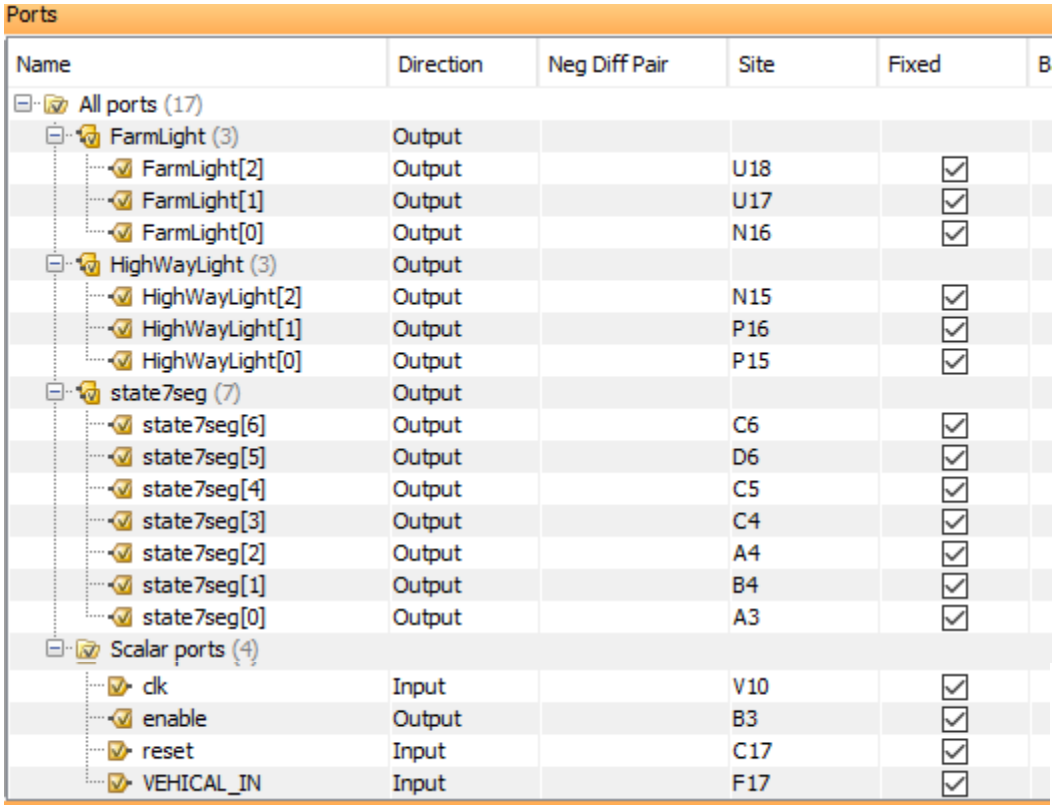
```
// module seven_seg(in, out);
//     input [1:0] in;
//     output [6:0] out;
//         assign out = (in == 0) ? 7'd0:
//                                  (in == 1) ? 7'b1111001:
//                                  (in == 2) ? 7'b0100100:
//                                  (in == 3) ? 7'b0110000:
//                                  (in == 4) ? 7'b0011001:
//                                  (in == 5) ? 7'b0010010:
//                                  (in == 6) ? 7'b0000010:
//                                  (in == 7) ? 7'b1111000:
//                                  (in == 8) ? 7'b0000000:
//                                  (in == 9) ? 7'b0011000: 7'b1111111;
// endmodule
```

UCF FILE:

```
//reset, clk, VEHICAL_IN, HighWayLight, FarmLight,state7seg

NET "reset" LOC = C17 | PULLUP;
NET "clk" LOC = V10 | PULLUP;
NET "VEHICAL_IN" LOC = F17 | PULLUP;

NET "HighWayLight[0]" LOC = P15;
NET "HighWayLight[1]" LOC =  P16;
NET "HighWayLight[2]" LOC =  N15;

NET "FarmLight[0]" LOC =  N16;
NET "FarmLight[1]" LOC =  U17;
NET "FarmLight[2]" LOC =  U18;

NET "state7seg[0]" LOC = A3;
NET "state7seg[1]" LOC = B4;
NET "state7seg[2]" LOC = A4;
NET "state7seg[3]" LOC = C4;
NET "state7seg[4]" LOC = C5;
```

```
NET "state7seg[5]" LOC = D6;
NET "state7seg[6]" LOC = C6;

NET "enable" LOC = B3;
```

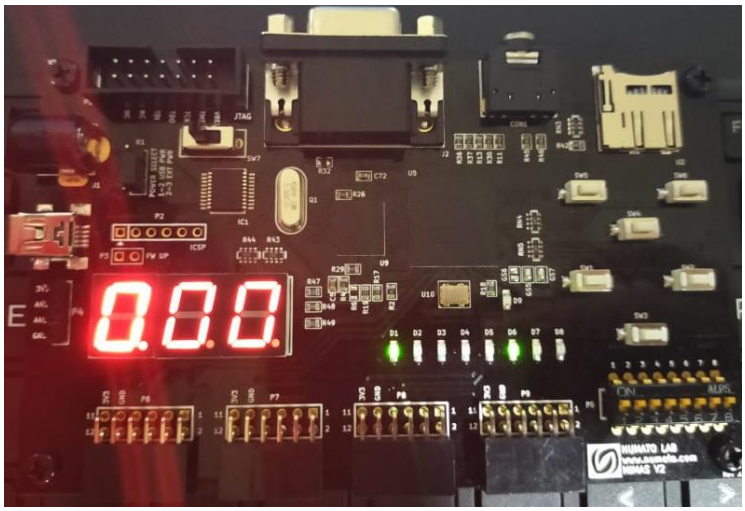
I/O PIN PLANNING:

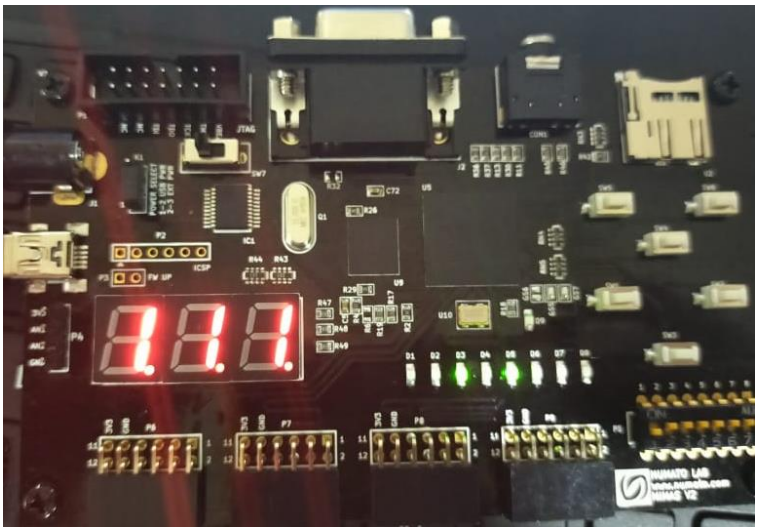


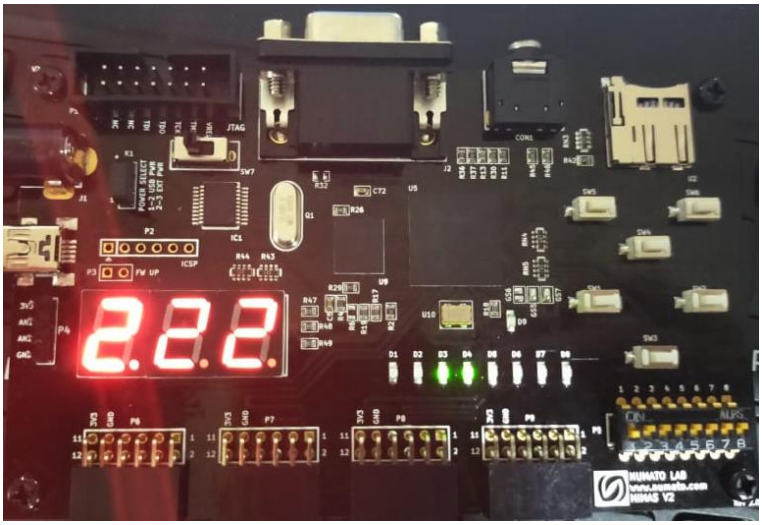
Ports

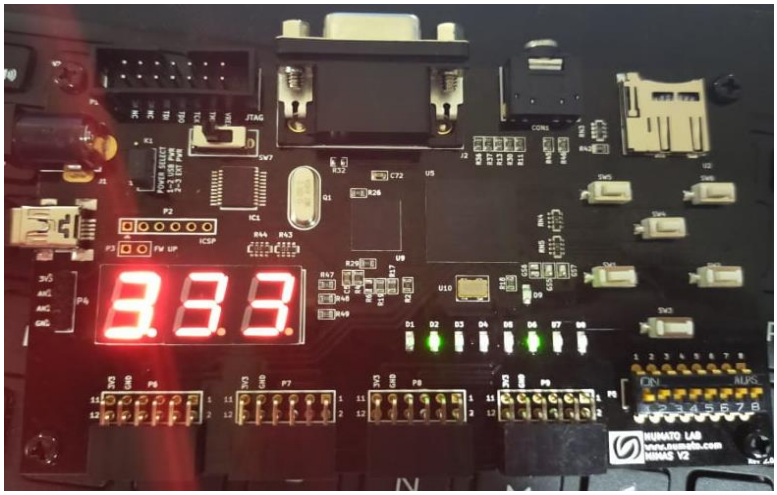
Name	Direction	Neg Diff Pair	Site	Fixed	B
All ports (17)					
FarmLight (3)	Output				
FarmLight[2]	Output		U18	☑	
FarmLight[1]	Output		U17	☑	
FarmLight[0]	Output		N16	☑	
HighWayLight (3)	Output				
HighWayLight[2]	Output		N15	☑	
HighWayLight[1]	Output		P16	☑	
HighWayLight[0]	Output		P15	☑	
state7seg (7)	Output				
state7seg[6]	Output		C6	☑	
state7seg[5]	Output		D6	☑	
state7seg[4]	Output		C5	☑	
state7seg[3]	Output		C4	☑	
state7seg[2]	Output		A4	☑	
state7seg[1]	Output		B4	☑	
state7seg[0]	Output		A3	☑	
Scalar ports (4)					
clk	Input		V10	☑	
enable	Output		B3	☑	
reset	Input		C17	☑	
VEHICAL_IN	Input		F17	☑	

OUTPUT:


0.0.0.


1.1.1.


2.2.2.


JTAG
ICSP
FW UP
3V3
GND
P6
P7
P8
P9
ON
AUS
HUMATO LAB
NIMAS V2