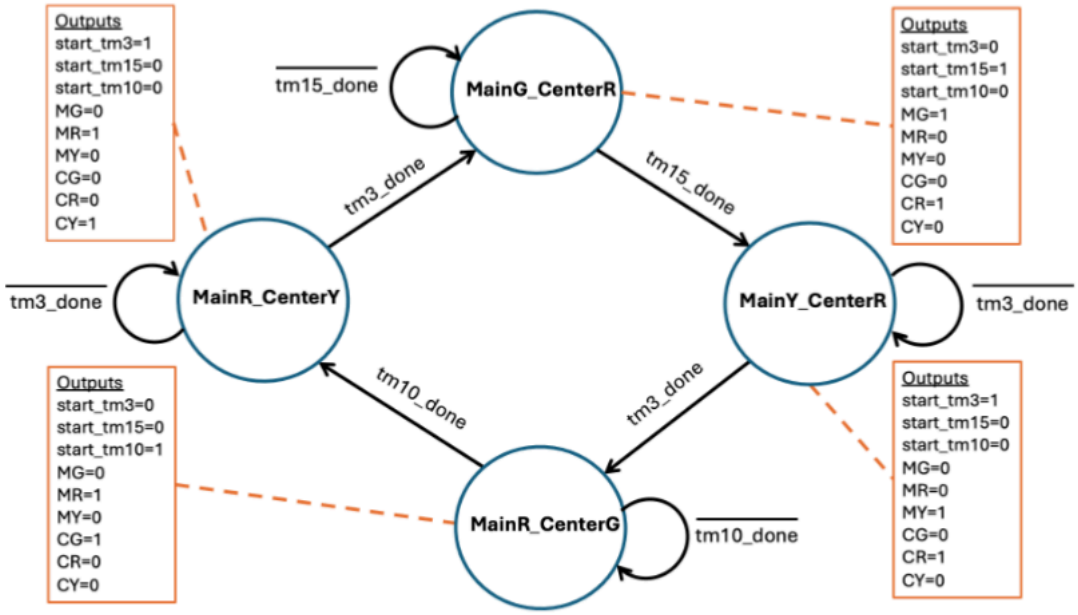
Include final State Diagram showing all input and output signals (5 points)



- LED 0: Main Street Red Light
- LED 1: Main Street Yellow Light
- LED 2: Main Street Green Light
- LED 4: Center Street Red Light
- LED 5: Center Street Yellow Light
- LED 6: Center Street Green Light

Include the code for the overall Traffic Light Controller (5 points)

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 11/10/2025 10:08:52 AM
// Design Name:
// Module Name: traffic_light_controller_top
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
```

```
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module traffic_light_controller_top(
    input clk,
    input btnC,
    output [6:0] led
    );

    reg [1:0] current_state, next_state;
    reg start_tm3, start_tm10, start_tm15;
    reg MG, MR, MY, CG, CR, CY;
    reg start_tm3_dly, start_tm10_dly, start_tm15_dly;

    wire clk_1Hz, start_tm3_pulse, start_tm10_pulse, start_tm15_pulse;
    wire tm3_done, tm10_done, tm15_done;

    parameter [1:0] MainG_CenterR = 2'b00, MainY_CenterR = 2'b01, MainR_CenterG =2'b10,
MainR_CenterY = 2'b11;


    clk_gen U1(
        .clk(clk),
        .rst(btnC),
        .clk_1Hz(clk_1Hz)
    );


    always @(posedge clk) begin
        start_tm3_dly <= start_tm3;
        start_tm10_dly <= start_tm10;
        start_tm15_dly <= start_tm15;
    end

    assign start_tm3_pulse = start_tm3 && !start_tm3_dly;
    assign start_tm10_pulse = start_tm10 && !start_tm10_dly;
    assign start_tm15_pulse = start_tm15 && !start_tm15_dly;

    timer #(
        .MAX_COUNT(3)
```

```
) tm3_instance (
    .clk_1Hz(clk_1Hz),
    .rst(btnC),
    .start_pulse(start_tm3_pulse),
    .tm_done(tm3_done)
);

timer #(
    .MAX_COUNT(10)
) tm10_instance (
    .clk_1Hz(clk_1Hz),
    .rst(btnC),
    .start_pulse(start_tm10_pulse),
    .tm_done(tm10_done)
);

timer #(
    .MAX_COUNT(15)
) tm15_instance (
    .clk_1Hz(clk_1Hz),
    .rst(btnC),
    .start_pulse(start_tm15_pulse),
    .tm_done(tm15_done)
);

always @(current_state) begin
    MG = 0;
    MY = 0;
    MR = 0;
    CG = 0;
    CY = 0;
    CR = 0;
    start_tm3 = 0;
    start_tm10 = 0;
    start_tm15 = 0;
    case(current_state)
        MainG_CenterR: begin
            MG = 1;
            CR = 1;
            start_tm15 = 1;
        end
        MainY_CenterR: begin
            MY = 1;
            CR = 1;
```

```
            start_tm3 = 1;
        end
        MainR_CenterG: begin
            MR = 1;
            CG = 1;
            start_tm10 = 1;
        end
        MainR_CenterY: begin
            MR = 1;
            CY = 1;
            start_tm3 = 1;
        end

    endcase
end

always @(*) begin
    next_state = current_state;

    case(current_state)
        MainG_CenterR: begin
            if(tm15_done) next_state = MainY_CenterR;
        end
        MainY_CenterR: begin
            if(tm3_done) next_state = MainR_CenterG;
        end
        MainR_CenterG: begin
            if(tm10_done) next_state = MainR_CenterY;
        end
        MainR_CenterY: begin
            if(tm3_done) next_state = MainG_CenterR;
        end
    endcase
end

always @(posedge clk_1Hz, posedge btnC) begin
    if(btnC) current_state <= MainG_CenterR;
    else current_state <= next_state;
end

assign led[0] = MR;
assign led[1] = MY;
assign led[2] = MG;
assign led[4] = CR;
```

```
    assign led[5] = CY;
    assign led[6] = CG;

endmodule
```

Include a test bench with waveforms that demonstrate correct operation (5 points)

I had to speed things up because it was taking forever to get to the required seconds.

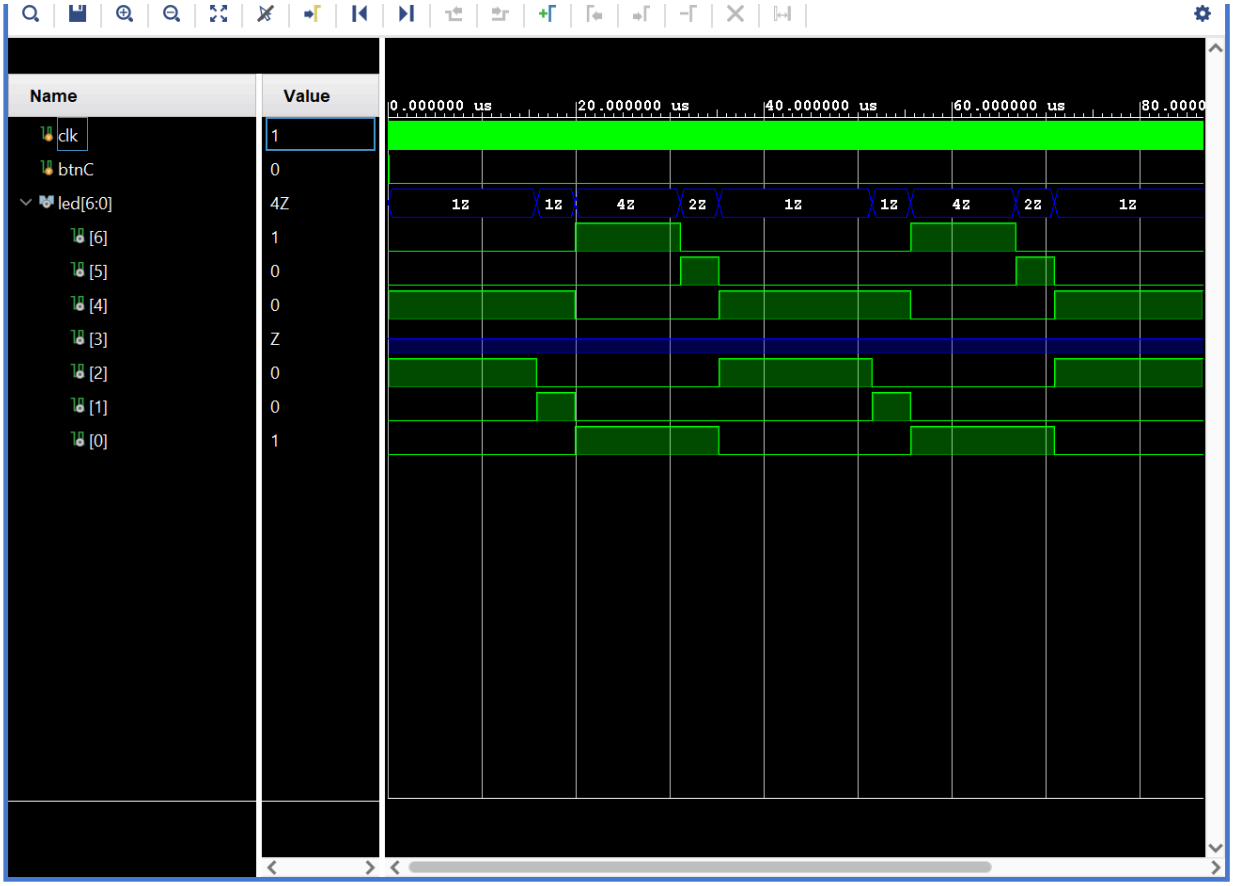


Figure 1: testbench simulation

Include a Conclusion statement describing your fulfillment of the objectives along with your observations (5 points)

I was able to achieve 100% for the lab by fulfilling all requirements. I did not do any extra credit. I credit a main file incharge of initiating my other files. I had a clk_gen file to create a slower clock signal however in the simulation I sped it up because getting to the required amount of seconds to see anything happen was taking far too long. I created a timer method that could

take any value of seconds and create a timer for it. Lastly I created a test bench to simulate that it indeed worked.

In your conclusion statement, indicate the level of functionality (in percent) of your lab (5 points)