ECE413s

Smart Traffic Light Controller

Supervisor: Dr. DiaaEl-Din S Khalil

Fall 2024

Contents

Members

Name	ID	Load
Mohamed Atta El-Sayed Atta	2101521	Design
Mohamed Sameh Abdelrahman Ahmed	2100401	Design
Mina Medhat Abdelmalak	2001486	Testbench
Akram Emad El-Din Ahmed	2101512	Testbench + Report

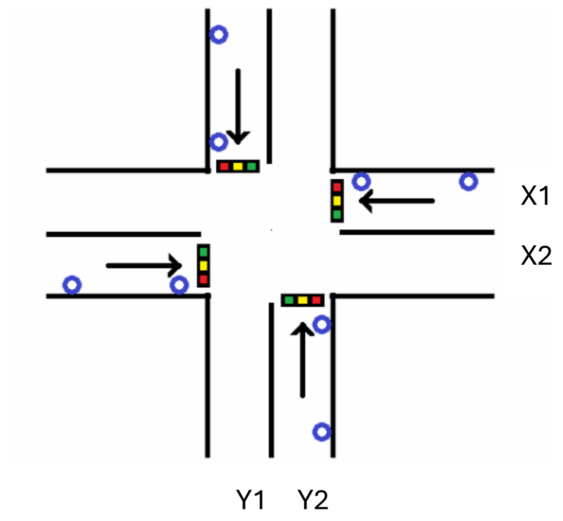
Abstract

The Aim of this report is to present an idea of implementation of a Smart Traffic light System using Verilog and provide the proper testbench.

Introduction

Smart Traffic Light System aims to control the time of waiting depending on the traffic flow of our streets. By sensing each road we can determine whether the road is Empty, Jammed and non-jammed. That's the data needed to control our traffic light.

Algorithm Explanation

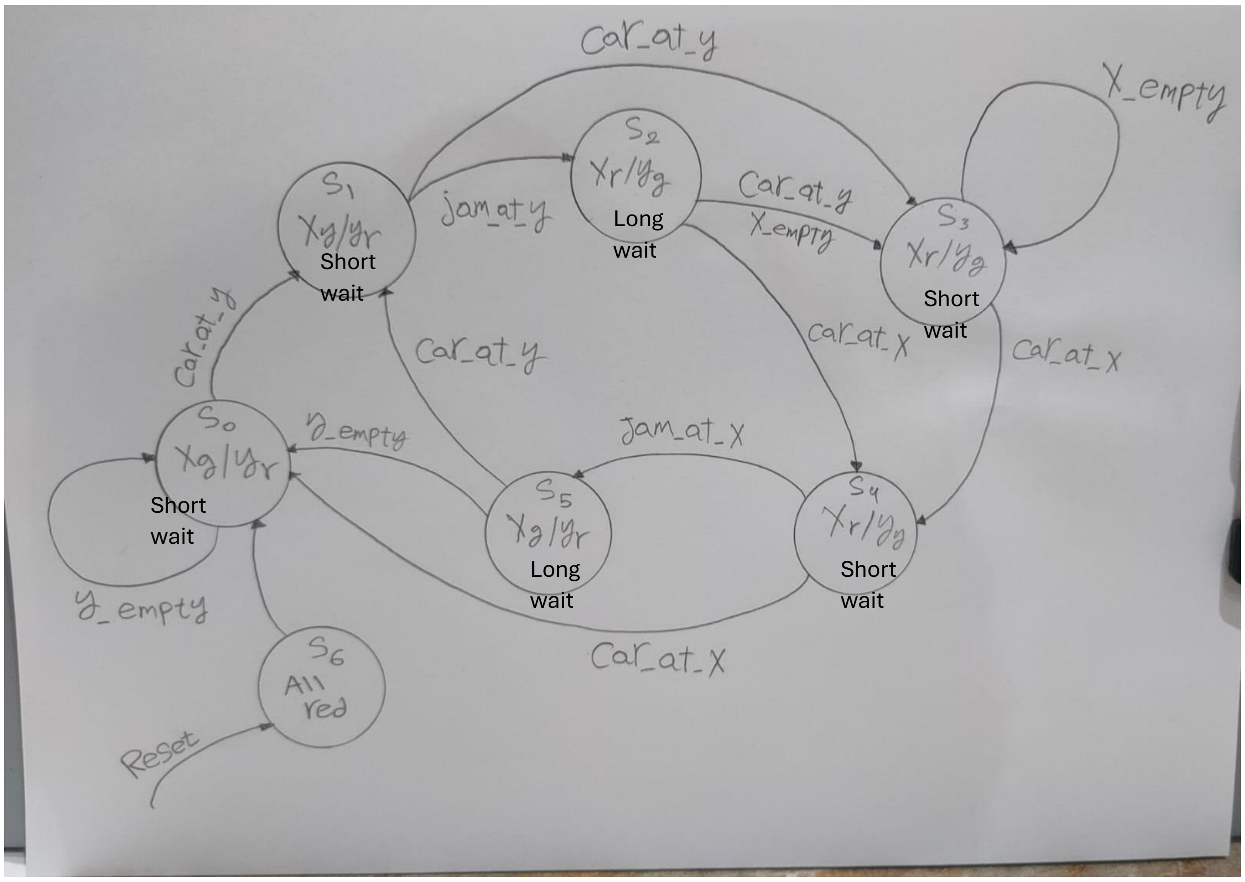


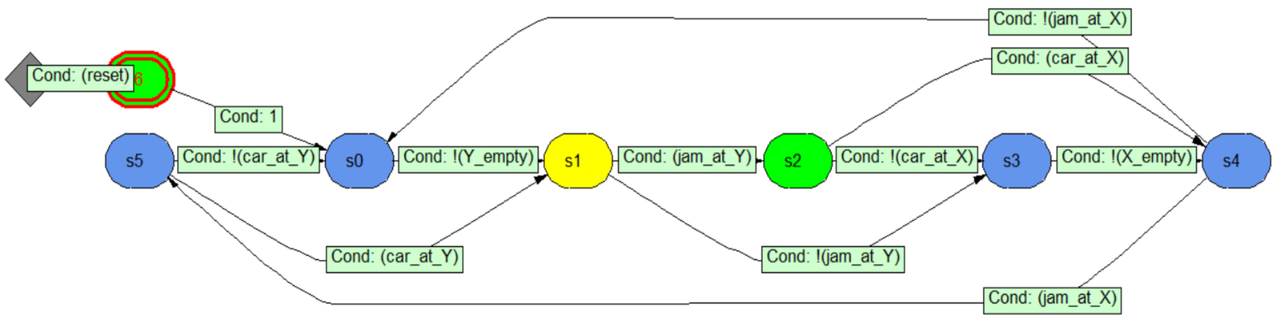
To implement this algorithm, we applied the concept of FSM (Finate State Machine) to identify each state and its changes.

- For each road we have 2 sensors, one at the beginning and the other at some distance from the first sensor, where we can detect:
    - The road is empty.
    - The road is Jammed.
    - The road is neither empty nor jammed.
- At the beginning of the system, Road X1and X2 will be allowed to move for **a short time** then changes to yellow then red
- The system checks the sensor inputs in the yellow state and determine whether the controller lid the red led for **a short time or a long time**
- We open both X1 and X2 at the same time or Y1 and Y2 at the same time
- Although our effort to reduce car accidents, there's no guarantee that car accidents will not happen, that's why we added a reset signal which turn all the red LEDs to block the streets.
- Y1,Y2 is now opened and X1.X2 are closed
- At the yellow state of Y1,Y2 The same sensor check happens to determine the waiting time at the red state (Long wait or short wait).

FSM

The following FSM illustrates our algorithm that will be codded into verilog



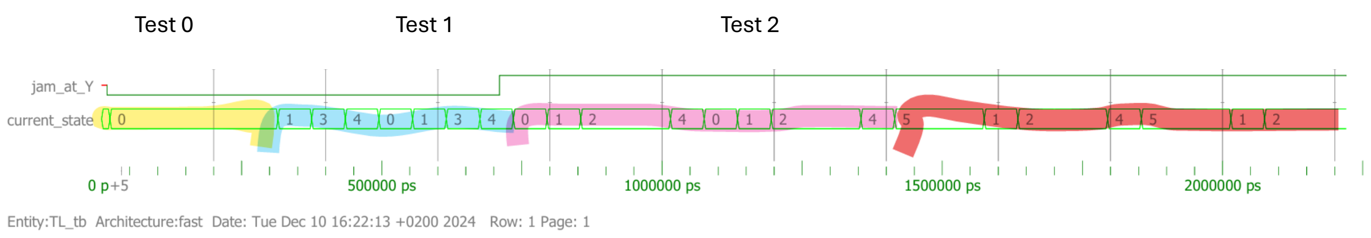


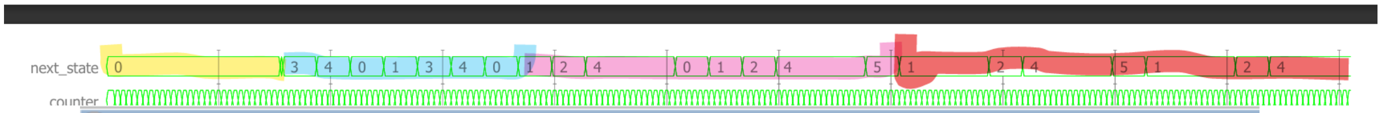
Test Plan and Simulation

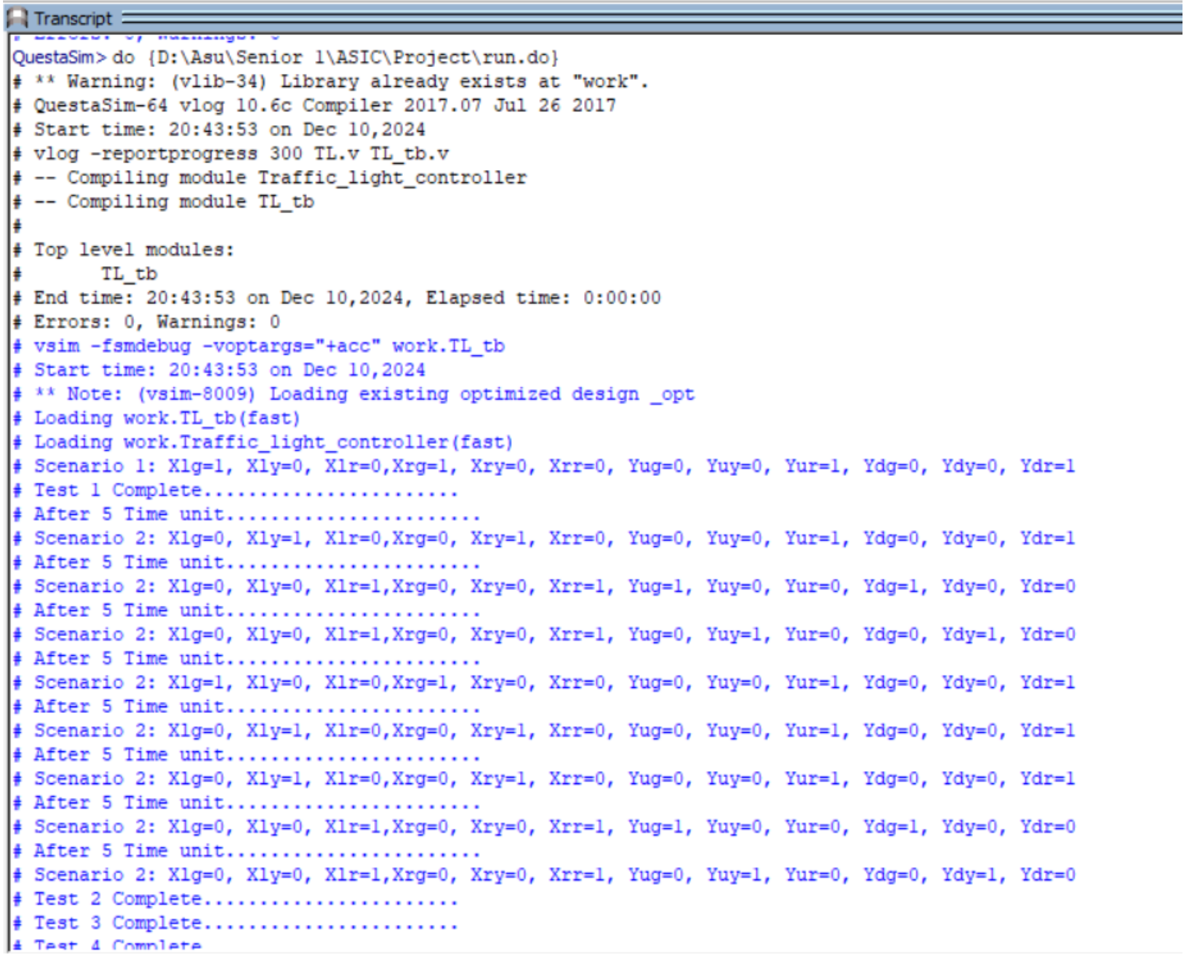
- Test 0: Reset=1, Testing the initial state and car accident state
    - Expected Outputs: All Traffic lights Are red
    - Expected State: State S6
- Test 1: Reset= 0, All roads are empty
    - Expected Outputs: X will be opened and won't be closed as long as there's no appearance of car at Y1 or Y2
    - Expected State: State S0
- Test 2: car at x and y (no jam)
    - Expected Outputs: X1,X2 will be opened then turns to yellow. Y1,Y2 will be opened after X1,X2 Closes then turns to yellow then red and it loops as long as that is the state **"Short Waiting Time"**
    - Expected State: State S0>S1>S3>S4>S0 and it loops
- Test 3: car at x and y is jammed
    - Expected Outputs: X1,X2 (Short green time). Y1,Y2 (Long green time)
    - Expected State: State S0>S1>S2>S4>S0 and it loops
- Test 4: Car jam at x and y
    - Expected Outputs: X1,X2 will be opened then turns to yellow. Y1,Y2 will be opened after X1,X2 Closes then turns to yellow then red and it loops as long as that is the state **"Short Waiting Time"**
    - Expected State: State S0>S1>S3>S4>S0 and it loops

Test Simulation

Test 3







```
QuestaSim> do {D:\Asu\Senior 1\ASIC\Project\run.do}
# ** Warning: (vlib-34) Library already exists at "work".
# QuestaSim-64 vlog 10.6c Compiler 2017.07 Jul 26 2017
# Start time: 20:43:53 on Dec 10,2024
# vlog -reportprogress 300 TL.v TL_tb.v
# -- Compiling module Traffic_light_controller
# -- Compiling module TL_tb
#
# Top level modules:
#       TL_tb
# End time: 20:43:53 on Dec 10,2024, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
# vsim -fsmdebug -voptargs="+acc" work.TL_tb
# Start time: 20:43:53 on Dec 10,2024
# ** Note: (vsim-8009) Loading existing optimized design _opt
# Loading work.TL_tb(fast)
# Loading work.Traffic_light_controller(fast)
# Scenario 1: Xlg=1, Xly=0, Xlr=0,Xrg=1, Xry=0, Xrr=0, Yug=0, Yuy=0, Yur=1, Ydg=0, Ydy=0, Ydr=1
# Test 1 Complete.......................
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=1, Xlr=0,Xrg=0, Xry=1, Xrr=0, Yug=0, Yuy=0, Yur=1, Ydg=0, Ydy=0, Ydr=1
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=0, Xlr=1,Xrg=0, Xry=0, Xrr=1, Yug=1, Yuy=0, Yur=0, Ydg=1, Ydy=0, Ydr=0
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=0, Xlr=1,Xrg=0, Xry=0, Xrr=1, Yug=0, Yuy=1, Yur=0, Ydg=0, Ydy=1, Ydr=0
# After 5 Time unit.......................
# Scenario 2: Xlg=1, Xly=0, Xlr=0,Xrg=1, Xry=0, Xrr=0, Yug=0, Yuy=0, Yur=1, Ydg=0, Ydy=0, Ydr=1
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=1, Xlr=0,Xrg=0, Xry=1, Xrr=0, Yug=0, Yuy=0, Yur=1, Ydg=0, Ydy=0, Ydr=1
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=1, Xlr=0,Xrg=0, Xry=1, Xrr=0, Yug=0, Yuy=1, Yur=1, Ydg=0, Ydy=0, Ydr=1
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=0, Xlr=1,Xrg=0, Xry=0, Xrr=1, Yug=1, Yuy=0, Yur=0, Ydg=1, Ydy=0, Ydr=0
# After 5 Time unit.......................
# Scenario 2: Xlg=0, Xly=0, Xlr=1,Xrg=0, Xry=0, Xrr=1, Yug=0, Yuy=1, Yur=0, Ydg=0, Ydy=1, Ydr=0
# Test 2 Complete.......................
# Test 3 Complete.......................
# Test 4 Complete
```

Test Bench Code

```
`timescale 1ns / 1ps

module TL_tb();


reg s1_xL, s2_xL, s1_xR, s2_xR;
reg s1_yU, s2_yU, s1_yD, s2_yD;
reg clk, reset;
wire Xlg,Xrg,Xly,Xry, Xlr,Xrr, Yug,Ydg, Yuy,Ydy, Yur,Ydr;
integer x=0;
```

```
// Instantiate the DUT (Device Under Test)
Traffic_light_controller dut (.*);

// Clock generation
initial begin
    clk = 0;
    forever #5 clk = ~clk; // 10 ns clock period (100 MHz clock frequency)
end

initial begin
    // Initialize inputs
    //Test 0 Reset >>> Expected All Leds Are red, State S6
    reset = 1;
    @(negedge clk);
    //Test 1 Senario 1 Empty roads @ x,y >>> Expected State S0, Expected
Output x is green, Y is red
    reset = 0;
    s1_xL = 0; s2_xL = 0; s1_xR = 0; s2_xR = 0;
    s1_yU = 0; s2_yU = 0; s1_yD = 0; s2_yD = 0;
    repeat (30) @(negedge clk);
    $display("Scenario 1: Xlg=%b, Xly=%b, Xlr=%b,Xrg=%b, Xry=%b, Xrr=%b,
Yug=%b, Yuy=%b, Yur=%b, Ydg=%b, Ydy=%b, Ydr=%b", Xlg, Xly, Xlr, Xrg, Xry,
Xrr,Yug,Yuy,Yur,Ydg,Ydy,Ydr);
    $display("Test 1 Complete.......................");
    //Test 2 car at x,y (no jam) >>> Expected X yellow then red, y green then
yellow "Short wait" , Expected states: S0>S1>S3>S4>S0
    s1_xL = 1; s2_xL = 0; s1_xR = 1; s2_xR = 0;
    s1_yU = 1; s2_yU = 0; s1_yD = 1; s2_yD = 0;
    for(x=0;x<8;x=x+1)begin
    repeat (5) @(negedge clk);
    $display("After 5 Time unit.......................");
    $display("Scenario 2: Xlg=%b, Xly=%b, Xlr=%b,Xrg=%b, Xry=%b, Xrr=%b,
Yug=%b, Yuy=%b, Yur=%b, Ydg=%b, Ydy=%b, Ydr=%b", Xlg, Xly, Xlr, Xrg, Xry,
Xrr,Yug,Yuy,Yur,Ydg,Ydy,Ydr);
    end
    x=0;
    $display("Test 2 Complete.......................");
    //Test 3 No jam X, Jam Y >>Expected X Yellow then red "Long wait", y will
turn green"Long wait" then yellow, Expected states: S0>S1>S2>S4>S0
    s1_xL = 1; s2_xL = 0; s1_xR = 1; s2_xR = 0;
    s1_yU = 1; s2_yU = 1; s1_yD = 1; s2_yD = 1;
        repeat (60) @(negedge clk);
    $display("Test 3 Complete.......................");
    s1_xL = 1; s2_xL = 0; s1_xR = 1; s2_xR = 1;
    s1_yU = 1; s2_yU = 0; s1_yD = 1; s2_yD = 1;
    $display("Test 4 Complete.......................");

   repeat (90) @(negedge clk);


    @(negedge clk);
    $stop;
end
endmodule
```

Design Code

```
module Traffic_light_controller(
input s1_xL, s2_xL, s1_xR, s2_xR, s1_yU, s2_yU, s1_yD, s2_yD, clk, reset,
output reg Xlg,Xrg, Xly,Xry, Xlr,Xrr, Yug,Ydg, Yuy,Ydy, Yur,Ydr
    );
localparam s0=0,s1=1,s2=2,
            s3=3,s4=4,s5=5,s6=6,
            short_wait_time = 5, // 5 time units
            long_wait_time = 15; // 15 time units
        wire    X_empty  = ~s1_xL | ~s1_xR;
        wire    Y_empty  =~s1_yU | ~s1_yD;
        wire    car_at_X = s1_xL | s1_xR;
        wire    car_at_Y = s1_yU | s1_yD;
        wire    jam_at_X = s2_xL | s2_xR;
        wire    jam_at_Y =s2_yU | s2_yD;
reg [2:0] current_state, next_state;
reg [3:0] counter;
always@(posedge clk, posedge reset)begin
    if(reset)begin
        current_state <= s6; //all red if reset is pressed
        counter <= 0;
        end
    else if(counter == 0)begin
        current_state <= next_state;
        case (next_state)
            s0:counter <= short_wait_time;
            s1:counter <= short_wait_time;
            s2:counter <= long_wait_time;
            s3:counter <= short_wait_time;
            s4:counter <= short_wait_time;
            s5:counter <= long_wait_time;
            s6:counter <= short_wait_time;
        endcase
    end
    else begin
        counter <= counter - 1;
    end
end

always@(*)begin
 next_state = current_state;
    case(current_state)
        s0: if(Y_empty)
            next_state = s0;
        else
            next_state = s1; // car at y
        s1: if(jam_at_Y)
            next_state = s2; //longer time case
        else
            next_state= s3; //normal time case
        s2:if(car_at_X)
            next_state = s4;
        else
            next_state = s3;
        s3: if(X_empty)
            next_state = s3;
        else
            next_state = s4;
        s4:if(jam_at_X)
            next_state = s5;
        else next_state = s0;
        s5: if(car_at_Y)
            next_state = s1;
        else
            next_state = s0;
        default: next_state = s0; // X is main direction
    endcase
end

always@(*)begin
    Xlg = 0;
    Xrg = 0;
    Xly = 0;
    Xry = 0;
    Xlr = 0;
    Xrr = 0;
    Yug = 0;
    Ydg = 0;
    Yuy = 0;
    Ydy = 0;
    Yur = 0;
    Ydr = 0;
    case(current_state)
        s0,s5: begin
            Xlg = 1;
            Yur = 1;
            Xrg = 1;
            Ydr = 1;
        end
        s1:begin
            Xly = 1;
            Yur = 1;
            Xry = 1;
            Ydr = 1;
        end
        s2,s3:begin
        Yug = 1;
        Xlr = 1;
        Ydg = 1;
        Xrr = 1;
        end
    s4:begin
        Yuy = 1;
        Xlr = 1;
        Ydy = 1;
        Xrr = 1;
        end
    s6: begin
        Xlr = 1;
        Yur = 1;
        Xrr = 1;
        Ydr = 1;
        end
    default: begin
    Xlg = 0;
    Xrg = 0;
    Xly = 0;
    Xry = 0;
    Xlr = 0;
    Xrr = 0;
    Yug = 0;
    Ydg = 0;
    Yuy = 0;
    Ydy = 0;
    Yur = 0;
    Ydr = 0;
    end
    endcase
end

endmodule
```