TRAFFIC LIGHT CONTROLLER

The purpose of this project is to design a methodology using Verilog to control the traffic with specified time delays for a Plus (+) -Shaped Road.

INTRODUCTION:

Managing traffic control is a significant challenge in many urban areas due to the sheer number of vehicles and the highly dynamic nature of traffic systems. Inefficient traffic systems often lead to accidents and significant time losses. This approach aims to reduce vehicle waiting times at traffic signals. The hardware design for this system has been developed using Verilog Hardware Description Language (HDL) programming.

Verilog is a hardware descriptive language that focuses on hardware design and simulation. Traditional methods of mounting various electronic components on breadboards or PCB circuits can be cumbersome, time-consuming, and prone to errors due to improper connections. Verilog offers a solution to these challenges by allowing designers to code the process and upload it directly to the circuit. This approach ensures that the circuit functions as intended according to the written code.

HDL is commonly used for designing sequential circuits, such as shift registers, and combinational logic circuits, like adders and subtractors. It describes digital systems, such as microprocessors and memory. Designs described in HDL are independent, possess unique states of operation, and are easy to simulate, design, and debug. This makes HDL more advantageous than schematics, especially for large circuits. Consequently, the use of Verilog design is increasingly prevalent in addressing the complexities of modern circuit design.

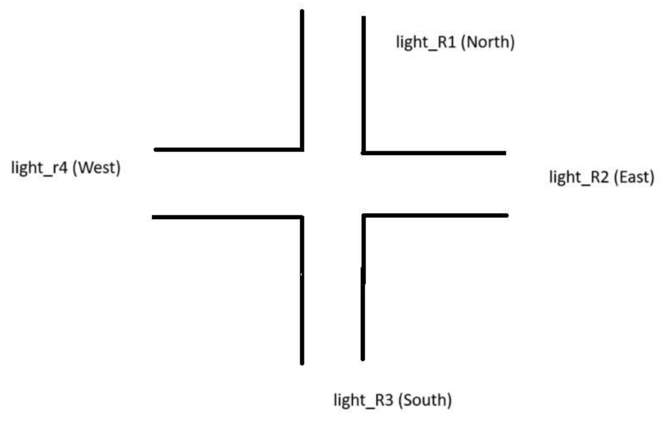
This project involves designing a basic traffic light control system for a (+)-shaped intersection. The system's output has been tested using VIVADO 2016.2.

A traffic light, also known as a traffic signal or stop light, is a signaling device installed at road intersections, pedestrian crossings, and other locations to indicate when it is safe to drive, ride, or walk using a universal color code.

In modern traffic systems:

- A red light means all traffic must stop.
- A yellow light indicates that cross-town traffic should slow down.
- A green light signals that it is safe to go or proceed.

LOGIC DIAGRAM:



In this diagram:

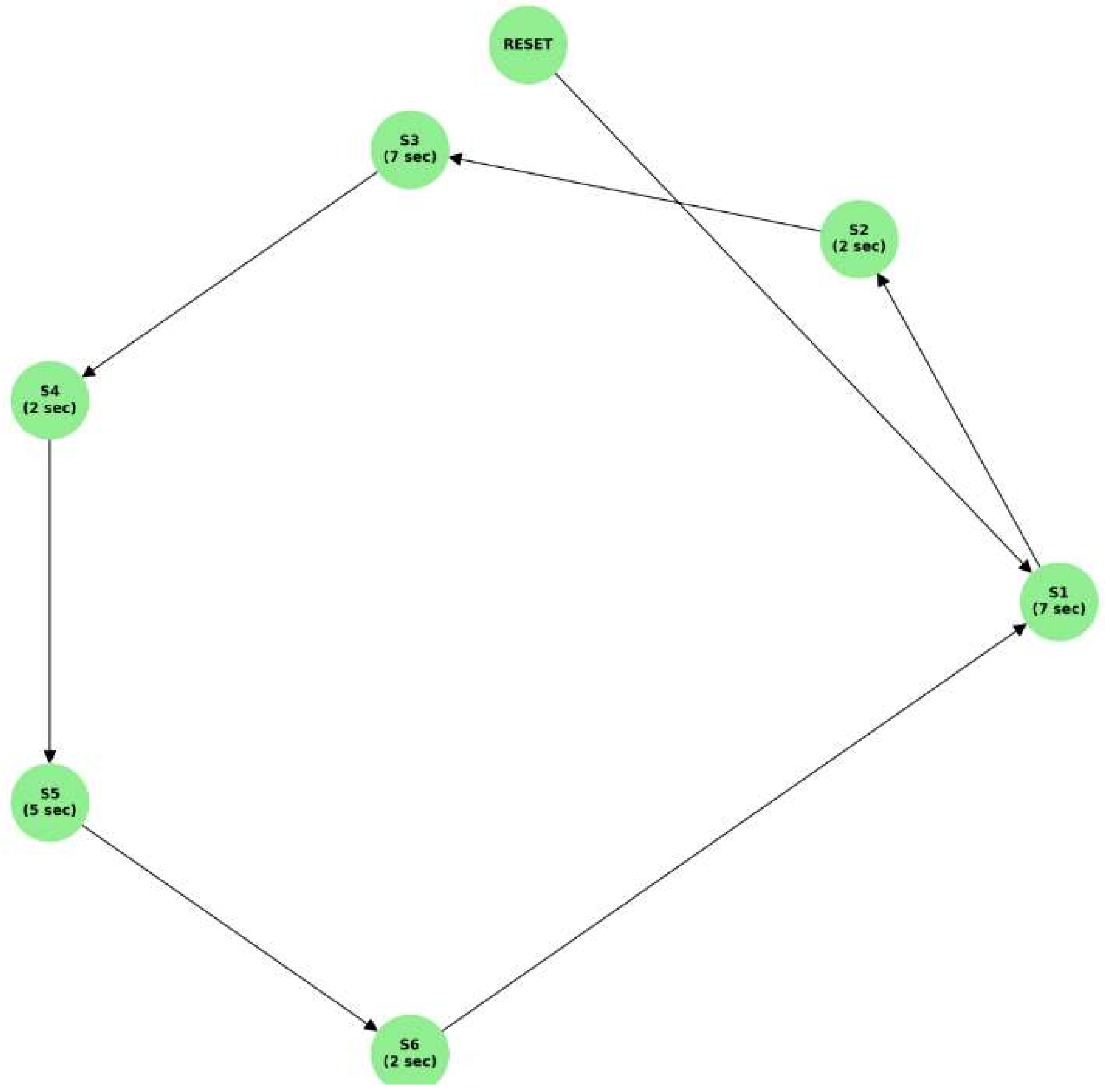
R1 represents North.

R2 represents East.

R3 represents South.

R4 represents West.

STATE DIAGRAM:



Timing Definitions:sec7:

7 seconds (time duration for green light)

sec5: 5 seconds (time duration for yellow light)

sec2: 2 seconds (time duration for yellow light and transition)

sec3: 3 seconds (time duration for yellow light and transition)

VIVADO CODE:

```
`timescale 1ns / 1ps

module Traffic_Light_Controller_Cross_Shaped (
    input clk,
    input rst,
    output reg [2:0] light_R1, // Road 1 lights (North)
    output reg [2:0] light_R2, // Road 2 lights (East)
    output reg [2:0] light_R3, // Road 3 lights (South)
    output reg [2:0] light_R4  // Road 4 lights (West)
);

    // State encoding
    parameter S1 = 0, S2 = 1, S3 = 2, S4 = 3, S5 = 4, S6 = 5;
    reg [2:0] ps; // present state
    reg [3:0] count;
    parameter sec7 = 7, sec5 = 5, sec2 = 2, sec3 = 3;

    // State transition and output logic
    always @(posedge clk or posedge rst) begin
        if (rst) begin
            ps <= S1;
            count <= 0;
        end else begin
```

```
case (ps)
   S1: if (count < sec7) begin
         ps <= S1;
         count <= count + 1;
      end else begin
         ps <= S2;
         count <= 0;
      end
   S2: if (count < sec2) begin
         ps <= S2;
         count <= count + 1;
      end else begin
         ps <= S3;
         count <= 0;
      end
   S3: if (count < sec7) begin
         ps <= S3;
         count <= count + 1;
      end else begin
         ps <= S4;
         count <= 0;
      end
   S4: if (count < sec2) begin
```

```
            ps <= S4;
            count <= count + 1;
          end else begin
            ps <= S5;
            count <= 0;
          end
        S5: if (count < sec5) begin
            ps <= S5;
            count <= count + 1;
          end else begin
            ps <= S6;
            count <= 0;
          end
        S6: if (count < sec2) begin
            ps <= S6;
            count <= count + 1;
          end else begin
            ps <= S1;
            count <= 0;
          end
        default: ps <= S1;
      endcase
    end
```

```
end

// Output logic based on current state (Mealy outputs)
always @* begin
    case (ps)
        S1: begin
            light_R1 = 3'b001; // Green for R1
            light_R2 = 3'b100; // Red for R2
            light_R3 = 3'b001; // Green for R3
            light_R4 = 3'b100; // Red for R4
        end
        S2: begin
            light_R1 = 3'b010; // Yellow for R1
            light_R2 = 3'b100; // Red for R2
            light_R3 = 3'b010; // Yellow for R3
            light_R4 = 3'b100; // Red for R4
        end
        S3: begin
            light_R1 = 3'b100; // Red for R1
            light_R2 = 3'b001; // Green for R2
            light_R3 = 3'b100; // Red for R3
            light_R4 = 3'b001; // Green for R4
        end
```

```
S4: begin
   light_R1 = 3'b100; // Red for R1
   light_R2 = 3'b010; // Yellow for R2
   light_R3 = 3'b100; // Red for R3
   light_R4 = 3'b010; // Yellow for R4
end
S5: begin
   light_R1 = 3'b001; // Green for R1
   light_R2 = 3'b100; // Red for R2
   light_R3 = 3'b001; // Green for R3
   light_R4 = 3'b100; // Red for R4
end
S6: begin
   light_R1 = 3'b010; // Yellow for R1
   light_R2 = 3'b100; // Red for R2
   light_R3 = 3'b010; // Yellow for R3
   light_R4 = 3'b100; // Red for R4
end
default: begin
   light_R1 = 3'b000; // Off
   light_R2 = 3'b000; // Off
   light_R3 = 3'b000; // Off
   light_R4 = 3'b000; // Off
```

```
        end
      endcase
    end

Endmodule
```

TESTBENCH:

```
`timescale 1ns / 1ps


module Traffic_Light_Controller_Cross_Shaped_TB;
    reg clk;
    reg rst;
    wire [2:0] light_R1; // Road 1 lights (North)
    wire [2:0] light_R2; // Road 2 lights (East)
    wire [2:0] light_R3; // Road 3 lights (South)
    wire [2:0] light_R4; // Road 4 lights (West)

    Traffic_Light_Controller_Cross_Shaped uut (
        .clk(clk),
        .rst(rst),
        .light_R1(light_R1),
        .light_R2(light_R2),
        .light_R3(light_R3),
        .light_R4(light_R4)
    );


    initial begin
        clk = 0;
        forever #5 clk = ~clk; // 10ns clock period
    end
    initial begin
        rst = 1;
        #20 rst = 0; // Release reset after 20ns
        #2000 $finish; // End the simulation after 2000ns
    end
endmodule
```

THEORY:

The code provided controls the traffic lights for a cross-shaped intersection with four directions: North (R1), East (R2), South (R3), and West (R4).

State Definitions:

S1: North-South (R1 and R3) green, East-West (R2 and R4) red.

S2: North-South yellow, East-West red.

S3: East-West green, North-South red.

S4: East-West yellow, North-South red.

S5: North-South turning green, East-West red.

S6: North-South turning yellow, East-West red.

Explanation of States:

State S1 (North-South Green):

light_R1 = Green (North-South traffic can move)

light_R2 = Red

light_R3 = Green

light_R4 = Red

State S2 (North-South Yellow):

light_R1 = Yellow (North-South traffic prepare to stop)

light_R2 = Red

light_R3 = Yellow

light_R4 = Red

State S3 (East-West Green):

light_R1 = Red

light_R2 = Green (East-West traffic can move)

light_R3 = Red

light_R4 = Green

State S4 (East-West Yellow):

light_R1 = Red

light_R2 = Yellow (East-West traffic prepare to stop)

light_R3 = Red

light_R4 = Yellow

State S5 (North-South Turning Green):

light_R1 = Red

light_R2 = Red

light_R3 = Green (North-South turning traffic can move)

light_R4 = Red

State S6 (North-South Turning Yellow):

light_R1 = Red

light_R2 = Red

light_R3 = Yellow (North-South turning traffic prepare to stop)

light_R4 = Red

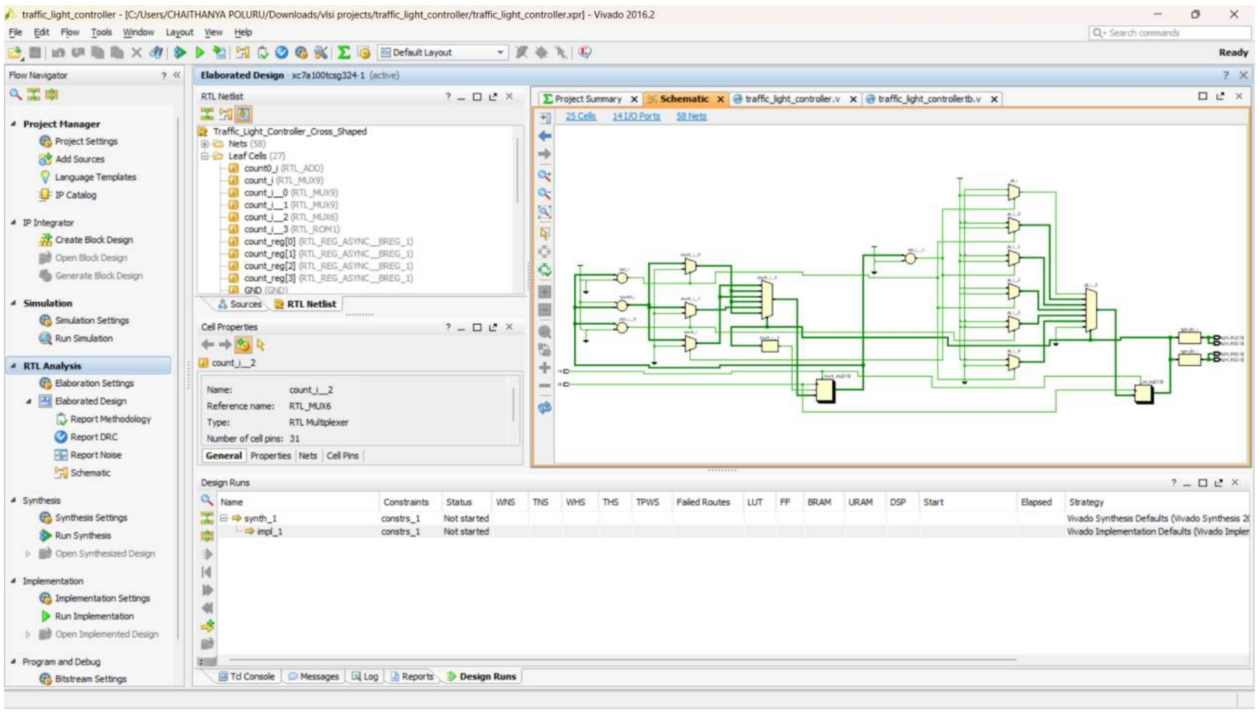
Default State:

All lights are off, indicating a reset or malfunction state.

STATE TABLE:

State	**light_R1**	**light_R2**	**light_r3**	**light_r4**	**Description**
S1	Green	Red	Green	Red	North-South green, East-West red
S2	Yellow	Red	Yellow	Red	North-South yellow, East-West red
S3	Red	Green	Red	Green	East-West green, North-South red
S4	Red	Yellow	Red	Yellow	East-West yellow, North-South red
S5	Red	Red	Green	Red	North-South turning green, East-West red
S6	Red	Red	Yellow	Red	North-South turning yellow, East-West red
default	off	off	off	off	All lights off (reset)

SCHEMATIC DIAGRAM:


traffic_light_controller - [C:/Users/CHAITHANYA POLURU/Downloads/vlsi projects/traffic_light_controller/traffic_light_controller.xpr] - Vivado 2016.2

Elaborated Design - xc7a100tcsg324-1 (active)

RTL Netlist

```
Traffic_Light_Controller_Cross_Shaped
  Nets (58)
  Leaf Cells (27)
    count0_i (RTL_ADD)
    count_i (RTL_MUX9)
    count_i__0 (RTL_MUX9)
    count_i__1 (RTL_MUX9)
    count_i__2 (RTL_MUX6)
    count_i__3 (RTL_ROM1)
    count_reg[0] (RTL_REG_ASYNC__BREG_1)
    count_reg[1] (RTL_REG_ASYNC__BREG_1)
    count_reg[2] (RTL_REG_ASYNC__BREG_1)
    count_reg[3] (RTL_REG_ASYNC__BREG_1)
    GND (GND)
```

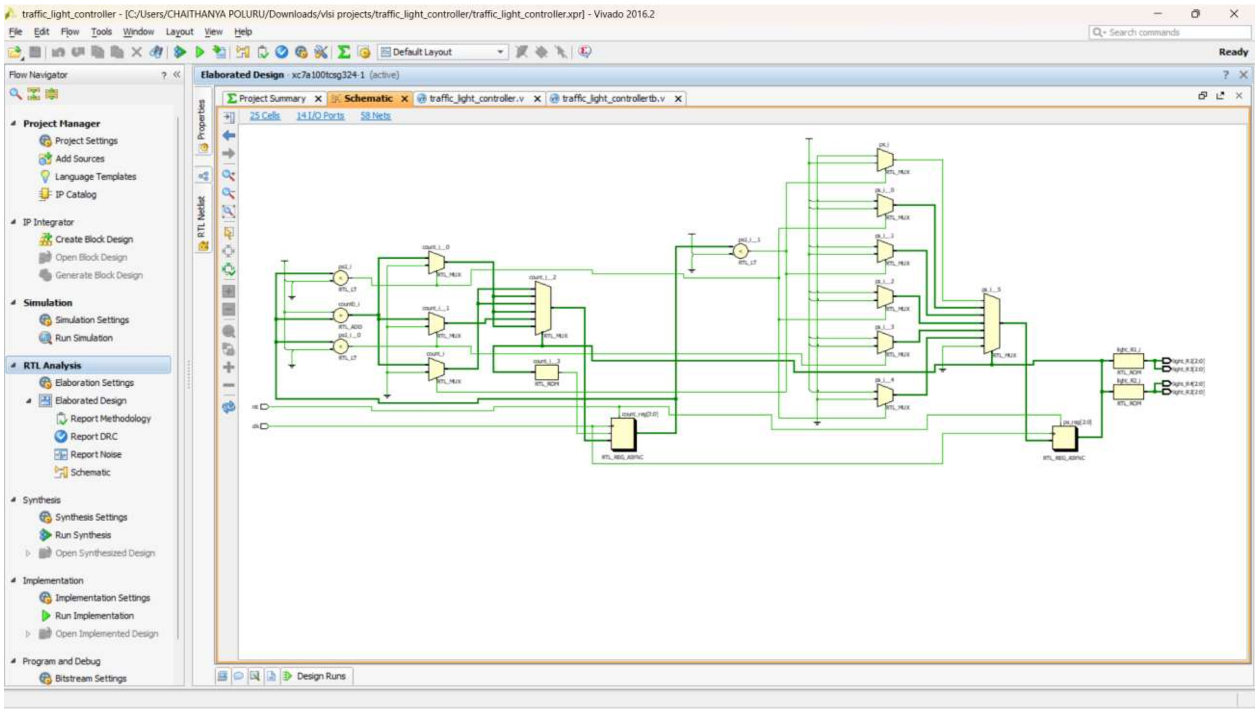
Cell Properties

count_i__2

```
Name:                 count_i__2
Reference name:       RTL_MUX6
Type:                 RTL Multiplexer
Number of cell pins:  31
```

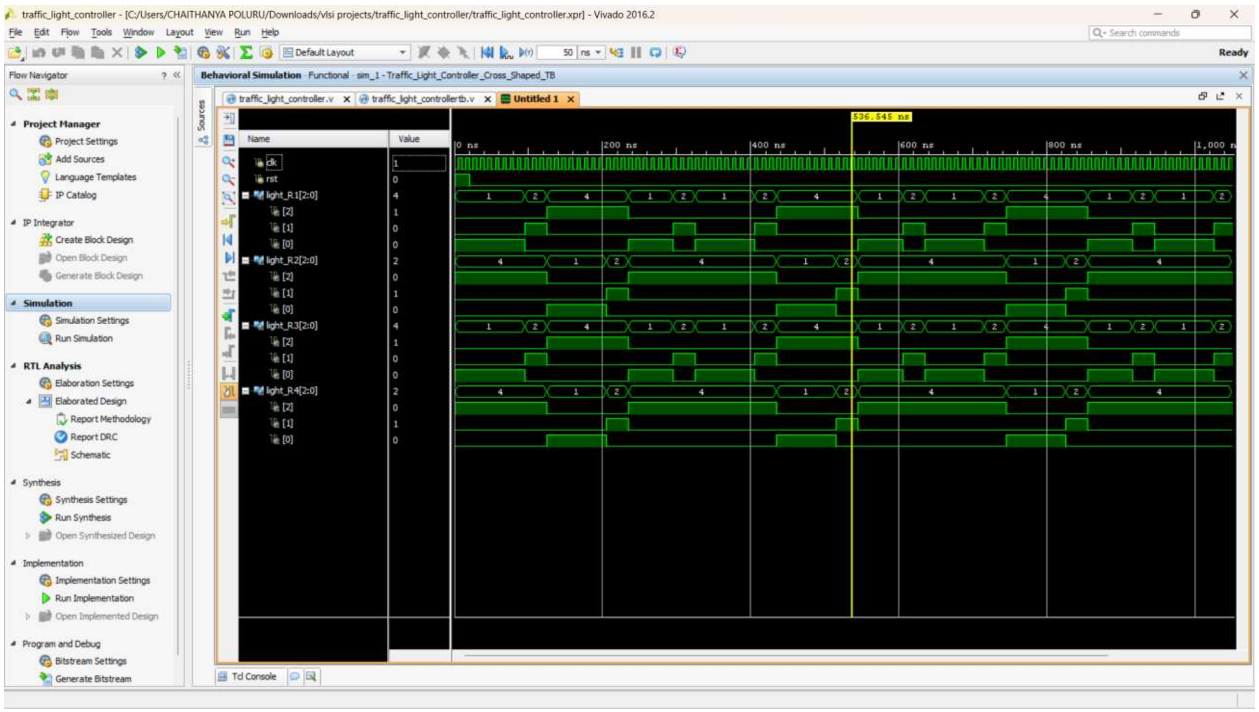
Design Runs

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Strategy
synth_1	constrs_1	Not started														Vivado Synthesis Defaults (Vivado Synthesis 2(
impl_1	constrs_1	Not started														Vivado Implementation Defaults (Vivado Impler


traffic_light_controller - [C:/Users/CHAITHANYA POLURU/Downloads/vlsi projects/traffic_light_controller/traffic_light_controller.xpr] - Vivado 2016.2

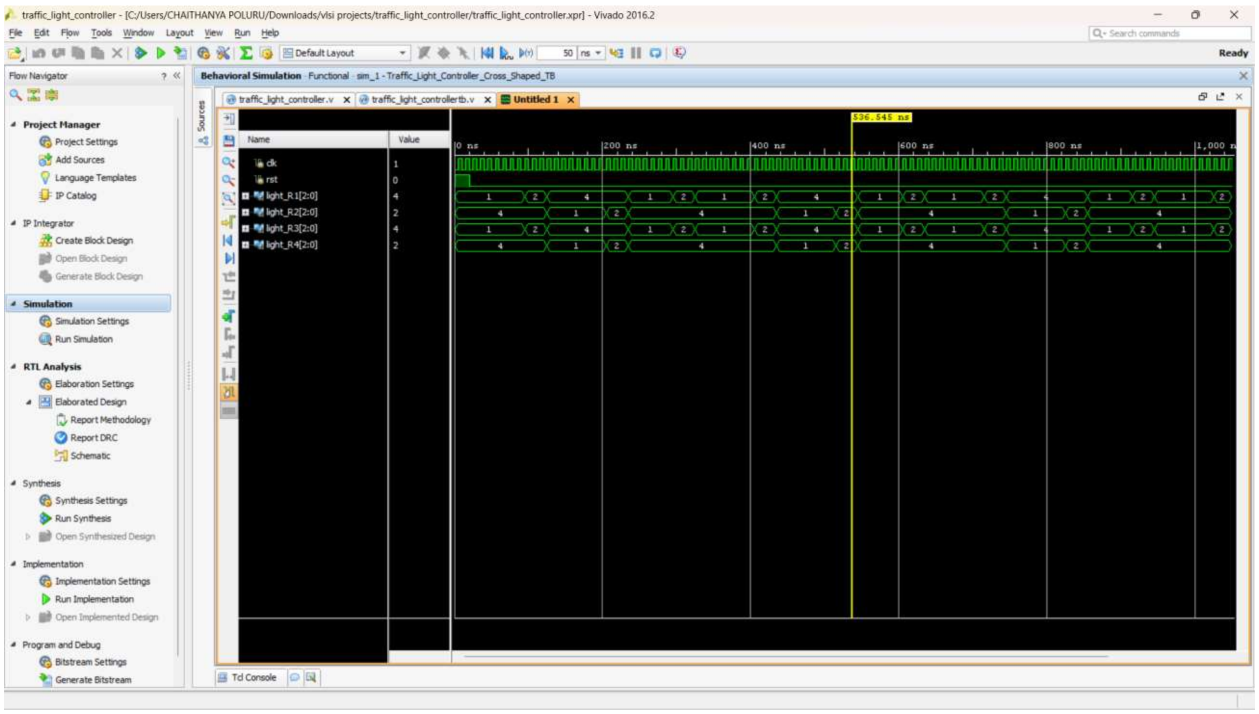
Elaborated Design - xc7a100tcsg324-1 (active)

OUTPUTS:


traffic_light_controller - [C:/Users/CHAITHANYA POLURU/Downloads/vlsi projects/traffic_light_controller/traffic_light_controller.xpr] - Vivado 2016.2

Behavioral Simulation - Functional - sim_1 - Traffic_Light_Controller_Cross_Shaped_TB

Name	Value
clk	1
rst	0
light_R1[2:0]	4
[2]	1
[1]	0
[0]	0
light_R2[2:0]	2
[2]	0
[1]	1
[0]	0
light_R3[2:0]	4
[2]	1
[1]	0
[0]	0
light_R4[2:0]	2
[2]	0
[1]	1
[0]	0


traffic_light_controller - [C:/Users/CHAITHANYA POLURU/Downloads/vlsi projects/traffic_light_controller/traffic_light_controller.xpr] - Vivado 2016.2

Behavioral Simulation - Functional - sim_1 - Traffic_Light_Controller_Cross_Shaped_TB

Name	Value
clk	1
rst	0
light_R1[2:0]	4
light_R2[2:0]	2
light_R3[2:0]	4
light_R4[2:0]	2

CONCLUSION:

This traffic light control system operates on the concept of fixed time allocation for each side of the junction, which cannot be adjusted according to varying traffic densities. The timings allotted at every junction are predetermined. Occasionally, higher traffic density on one side of the junction requires a longer green signal duration than the standard allotted time.

Thus, the traffic light control system facilitates the orderly flow of vehicles. However, there are numerous issues such as obstacles and high levels of accidents that occur daily. The traffic signal controller helps prevent such occurrences. Nonetheless, many areas or small towns still lack traffic light control facilities, leading to frequent accidents in those regions. Therefore, it is crucial to implement such facilities to control and maintain traffic flow effectively.