PROJECT REPORT

TRAFFIC LIGHT CONTROLLER USING VERILOG HDL

Submitted by:
SHEFIN MUHAMMED M S

Under the Guidance of:
MR.MONISH & MS.VIMALA

SION SEMICONDUCTOR

October 2025

Contents

1. Aim

To design and implement a Traffic Light Controller using Verilog HDL that simulates the operation of a real-world traffic signal system at a four-way intersection by sequencing red, yellow, and green lights based on time delays and state transitions.

2. Objectives

The main objective of this project is to design and implement a Traffic Light Controller using Verilog HDL that simulates the operation of a real-world traffic signal system. The system controls the sequencing of red, yellow, and green lights at a four-way intersection based on predefined time delays and state transitions, ensuring smooth and efficient traffic flow management.

3. Tools Used

- **Software Tools:**
  - ModelSim – for simulation and waveform verification.
  - Xilinx Vivado – for synthesis and RTL implementation.

Methodology

- Study the basic working principles of a traffic light control system and identify its main components such as controller, sensors, and signal lights.
- Design the traffic light control logic using a Finite State Machine (FSM) with defined states for Red, Yellow, and Green lights.
- Develop the Verilog HDL code to implement the FSM for controlling light transitions based on timing sequences.
- Simulate the Verilog design using software tools like ModelSim or Vivado to verify timing, state transitions, and functional correctness.
- Test the implemented design and verify its operation to ensure proper sequencing and timing behavior.

4. Verilog Code and TestBench

4.1. Verilog Code

```verilog

module TLC(input clk,rst,output reg [2:0]NS,EW);
//FSM STATE
parameter s0=3'b000;
parameter s1=3'b001;
parameter s2=3'b010;
parameter s3=3'b011;
parameter s4=3'b100;
parameter s5=3'b101;

reg[3:0]state,next_state;
reg [3:0]count;
//COUNTER
always@(posedge clk)begin
if(rst)
count<=4'b0000;
else if (state==next_state) begin
case(state)
s0,s3:if(count<14) count<=count+1;
s1,s2,s4,s5: if(count<2) count<=count+1;
default: count <= 4'b0000;
endcase
end
else begin
count<=4'b0000;
end
end

always@(posedge clk)begin
if(rst)begin
state<=s0;
end
else
state<=next_state;
end
```

```
//STATE ADN DELAY
always@(*)begin
next_state=state;
case(state)
    s0:if (count==14) next_state=s1;
    //else next_state=s0;
    s1:if (count==2) next_state=s2;
    //else next_state=s1;
    s2:if (count==2) next_state=s3;
    //else next_state=s2;
    s3:if (count==14) next_state=s4;
    //else next_state=s3;
    s4:if (count==2) next_state=s5;
    //else next_state=s4;
    s5:if (count==2) next_state=s0;
    //else next_state=s5;
    default next_state=s0;
endcase
end
//FOR OUTPUT
//GREEN=100,//YELLOW=010,RED=001
always@(*)begin
    case(state)
    s0:begin
    NS=3'b100;//GREEN=100
    EW=3'b001;//RED=001
    end

    s1:begin
    NS=3'b010;//YELLOW=010
    EW=3'b001;//RED=001
    end

    s2:begin
    NS=3'b001;//RED=001
    EW=3'b001;//RED=001
    end

    s3:begin
    NS=3'b001;//RED=001
    EW=3'b100;//GREEN=100
    end

    s4:begin
    NS=3'b001;//GREEN=100
    EW=3'b010;//YELLOW=010
    end

    s5:begin
    NS=3'b001;//RED=001
    EW=3'b001;//RED=001
    end

    default:begin
    NS=3'b001;//RED=001
    EW=3'b001;//RED=001

    end
    endcase
end
endmodule
```

4.2. Testbench code

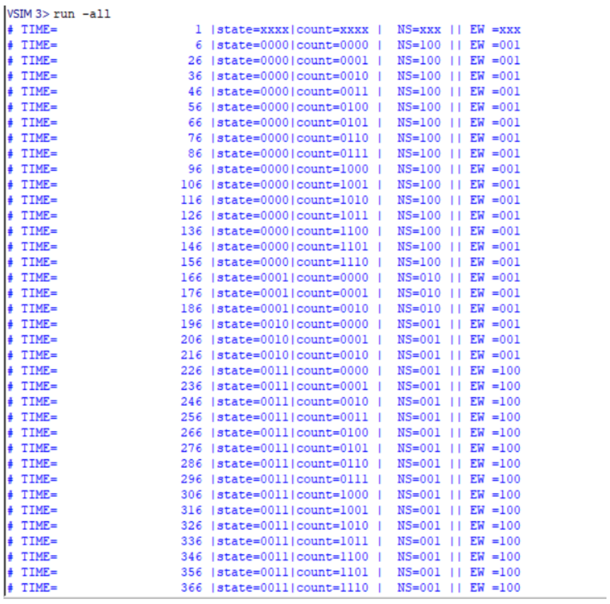
```
//TESTBENCH CODE FOR TRAFFIC LIGHT
module TLC_new_tb;
reg clk,rst;
wire [2:0]NS,EW;

TLC dut(.clk(clk),.rst(rst),.NS(NS),.EW(EW));

initial begin
 clk=0;#1;
 forever #5 clk = ~clk;
end

initial begin
```

```
#1;$monitor("TIME=%t |state=%b|count=%b |  NS=%b || EW =%b",$time
   ,dut.state,dut.count,NS,EW);
    rst=1;

#20 rst=0;

#500;

$finish;
$display("Simulation finished at time %0t", $time);
end
endmodule
```

5. Results

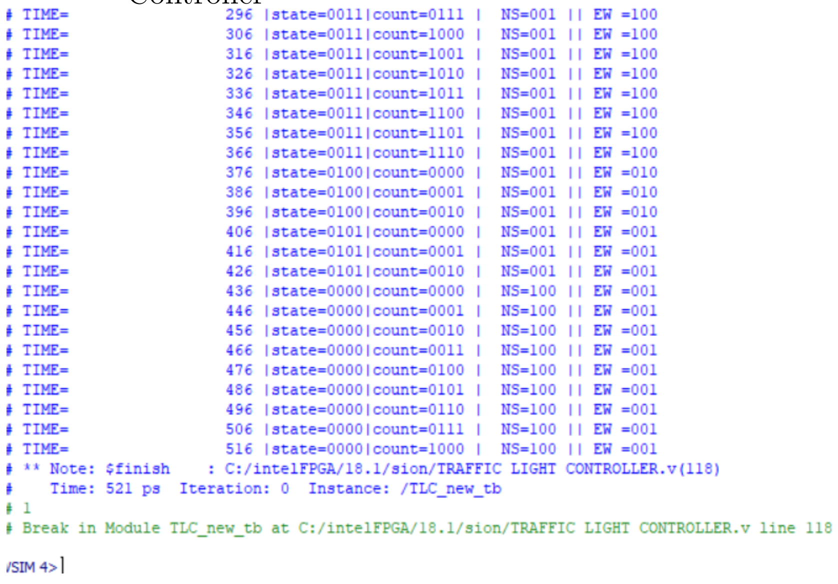
The Traffic Light Controller was successfully designed and implemented using Verilog HDL. The simulation results verified the correct operation of the finite state machine, ensuring proper sequencing of Red, Yellow, and Green lights according to the specified time delays and properly verified the output using the waveform.

5.1. Simulated Result



```
VSIM 3> run -all
# TIME=                  1 |state=xxxx|count=xxxx |  NS=xxx || EW =xxx
# TIME=                  6 |state=0000|count=0000 |  NS=100 || EW =001
# TIME=                 26 |state=0000|count=0001 |  NS=100 || EW =001
# TIME=                 36 |state=0000|count=0010 |  NS=100 || EW =001
# TIME=                 46 |state=0000|count=0011 |  NS=100 || EW =001
# TIME=                 56 |state=0000|count=0100 |  NS=100 || EW =001
# TIME=                 66 |state=0000|count=0101 |  NS=100 || EW =001
# TIME=                 76 |state=0000|count=0110 |  NS=100 || EW =001
# TIME=                 86 |state=0000|count=0111 |  NS=100 || EW =001
# TIME=                 96 |state=0000|count=1000 |  NS=100 || EW =001
# TIME=                106 |state=0000|count=1001 |  NS=100 || EW =001
# TIME=                116 |state=0000|count=1010 |  NS=100 || EW =001
# TIME=                126 |state=0000|count=1011 |  NS=100 || EW =001
# TIME=                136 |state=0000|count=1100 |  NS=100 || EW =001
# TIME=                146 |state=0000|count=1101 |  NS=100 || EW =001
# TIME=                156 |state=0000|count=1110 |  NS=100 || EW =001
# TIME=                166 |state=0001|count=0000 |  NS=010 || EW =001
# TIME=                176 |state=0001|count=0001 |  NS=010 || EW =001
# TIME=                186 |state=0001|count=0010 |  NS=010 || EW =001
# TIME=                196 |state=0010|count=0000 |  NS=001 || EW =001
# TIME=                206 |state=0010|count=0001 |  NS=001 || EW =001
# TIME=                216 |state=0010|count=0010 |  NS=001 || EW =001
# TIME=                226 |state=0011|count=0000 |  NS=001 || EW =100
# TIME=                236 |state=0011|count=0001 |  NS=001 || EW =100
# TIME=                246 |state=0011|count=0010 |  NS=001 || EW =100
# TIME=                256 |state=0011|count=0011 |  NS=001 || EW =100
# TIME=                266 |state=0011|count=0100 |  NS=001 || EW =100
# TIME=                276 |state=0011|count=0101 |  NS=001 || EW =100
# TIME=                286 |state=0011|count=0110 |  NS=001 || EW =100
# TIME=                296 |state=0011|count=0111 |  NS=001 || EW =100
# TIME=                306 |state=0011|count=1000 |  NS=001 || EW =100
# TIME=                316 |state=0011|count=1001 |  NS=001 || EW =100
# TIME=                326 |state=0011|count=1010 |  NS=001 || EW =100
# TIME=                336 |state=0011|count=1011 |  NS=001 || EW =100
# TIME=                346 |state=0011|count=1100 |  NS=001 || EW =100
# TIME=                356 |state=0011|count=1101 |  NS=001 || EW =100
# TIME=                366 |state=0011|count=1110 |  NS=001 || EW =100
```

Figure 1: Simulation Result of Traffic Light Controller



```
# TIME=                296 |state=0011|count=0111 |  NS=001 || EW =100
# TIME=                306 |state=0011|count=1000 |  NS=001 || EW =100
# TIME=                316 |state=0011|count=1001 |  NS=001 || EW =100
# TIME=                326 |state=0011|count=1010 |  NS=001 || EW =100
# TIME=                336 |state=0011|count=1011 |  NS=001 || EW =100
# TIME=                346 |state=0011|count=1100 |  NS=001 || EW =100
# TIME=                356 |state=0011|count=1101 |  NS=001 || EW =100
# TIME=                366 |state=0011|count=1110 |  NS=001 || EW =100
# TIME=                376 |state=0100|count=0000 |  NS=001 || EW =010
# TIME=                386 |state=0100|count=0001 |  NS=001 || EW =010
# TIME=                396 |state=0100|count=0010 |  NS=001 || EW =010
# TIME=                406 |state=0101|count=0000 |  NS=001 || EW =001
# TIME=                416 |state=0101|count=0001 |  NS=001 || EW =001
# TIME=                426 |state=0101|count=0010 |  NS=001 || EW =001
# TIME=                436 |state=0000|count=0000 |  NS=100 || EW =001
# TIME=                446 |state=0000|count=0001 |  NS=100 || EW =001
# TIME=                456 |state=0000|count=0010 |  NS=100 || EW =001
# TIME=                466 |state=0000|count=0011 |  NS=100 || EW =001
# TIME=                476 |state=0000|count=0100 |  NS=100 || EW =001
# TIME=                486 |state=0000|count=0101 |  NS=100 || EW =001
# TIME=                496 |state=0000|count=0110 |  NS=100 || EW =001
# TIME=                506 |state=0000|count=0111 |  NS=100 || EW =001
# TIME=                516 |state=0000|count=1000 |  NS=100 || EW =001
# ** Note: $finish    : C:/intelFPGA/18.1/sion/TRAFFIC LIGHT CONTROLLER.v(118)
#    Time: 521 ps  Iteration: 0  Instance: /TLC_new_tb
# 1
# Break in Module TLC_new_tb at C:/intelFPGA/18.1/sion/TRAFFIC LIGHT CONTROLLER.v line 118

VSIM 4>
```

Figure 2: Simulation Result of Traffic Light Controller

5.2. Waveform

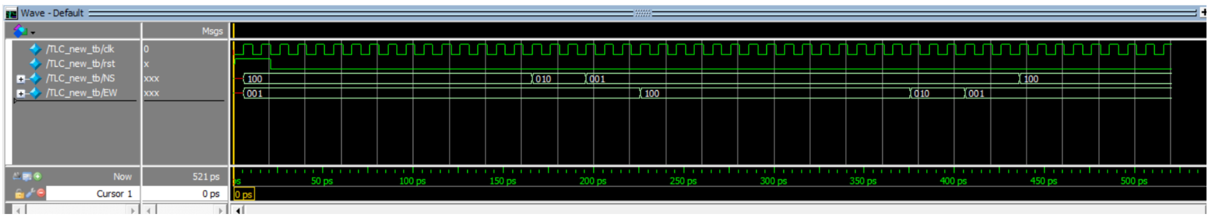


Figure 3: Waveform

5.3. Elaborated Design

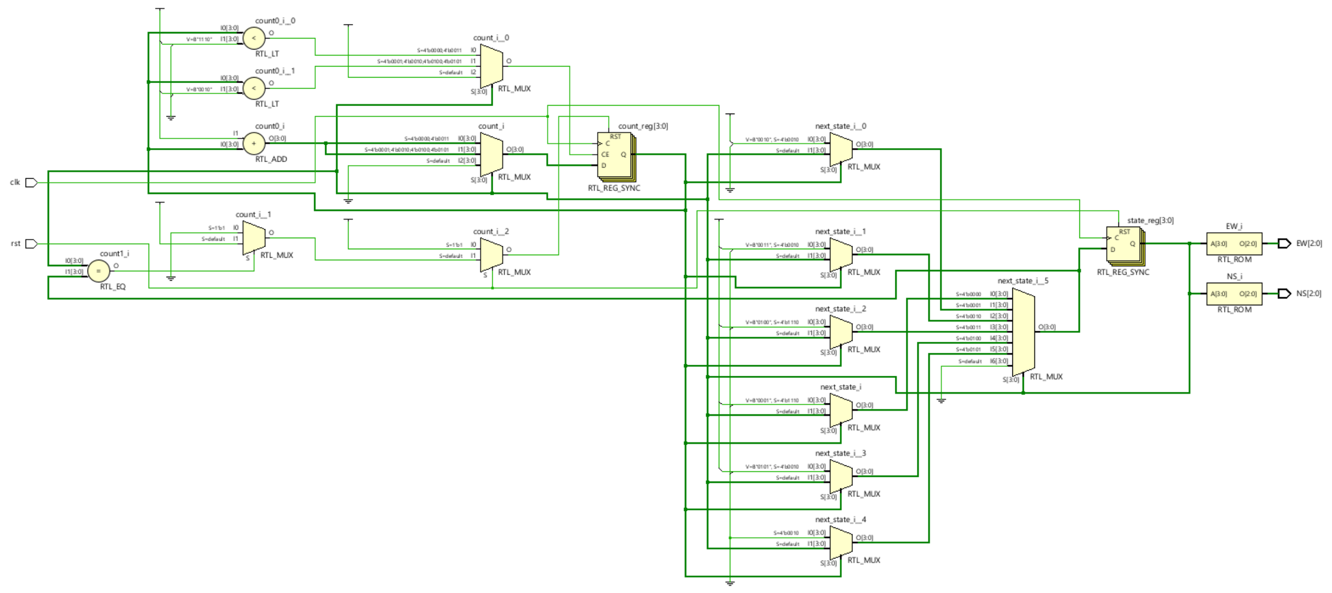


Figure 4: Elaborated Design using Vivado

6. Conclusion

The Traffic Light Controller was successfully designed, simulated, and implemented using Verilog HDL. The project demonstrated the practical application of digital design concepts such as finite state machines, timing control. The system effectively managed traffic light sequencing for a four-way intersection, ensuring smooth traffic flow. This project can be further enhanced by integrating sensors for vehicle detection to create an intelligent and adaptive traffic control system.