**Subject**

DIGITAL LOGIC DESIGN

**Topic**

Traffic Signal Controller

**By**

21BEC006(Shabbir Aglodiya)

* **INTRODUCTION**

The design code for a traffic light controller utilising a Finite State Machine is shown below. The input signals are clk and rst, while the output signals are nl, sl, el, and wl. "001" stands for green light, "010" for yellow light, and "100" for red light in the output signal. When the reset signal is activated, the design will enter the north state and begin producing output when the reset signal is turned off. Green light will be on for eight clock cycles and Yellow light will be on for four clock cycles. The design will begin with the north, then move to the south, then east, and ultimately west, and so on.

* **BLOCK DIAGRAM**

ASM

COMBINATION LOGIC

MAIN RED

YELLOW

GREEN

STATE TRANSISTION

SIGNAL

v

TRANSTITION

INPUT

INPUT

SIDE RED

YELLOW

GREEN

CLOCK

SELECTED

TIMER OUTPUT

TIMERS CONTROLS

TIMING CIRCUITS

* **VERILOG CODE**

module tsc(nl,sl,el,wl,clk,rst);

output reg [2:0] nl,sl,el,wl;

input clk;

input rst;

reg [2:0] state;

parameter [2:0] north=3'b000;

parameter [2:0] northy=3'b001;

parameter [2:0] south=3'b010;

parameter [2:0] southy=3'b011;

parameter [2:0] east=3'b100;

parameter [2:0] easty=3'b101;

parameter [2:0] west=3'b110;

parameter [2:0] westy=3'b111;

reg [2:0] count;

always @(posedge clk, posedge rst)

begin

if (rst)

begin

state=north;

count =3'b000;

end

else

begin

case (state)

north :

begin

if (count==3'b111)

begin

count=3'b000;

state=northy;

end

else

begin

count=count+3'b001;

state=north;

end

end

northy :

begin

if (count==3'b011)

begin

count=3'b000;

state=south;

end

else

begin

count=count+3'b001;

state=northy;

end

end

south :

begin

if (count==3'b111)

begin

count=3'b0;

state=southy;

end

else

begin

count=count+3'b001;

state=south;

end

end

southy :

begin

if (count==3'b011)

begin

count=3'b0;

state=east;

end

else

begin

count=count+3'b001;

state=southy;

end

end

east :

begin

if (count==3'b111)

begin

count=3'b0;

state=easty;

end

else

begin

count=count+3'b001;

state=east;

end

end

easty :

begin

if (count==3'b011)

begin

count=3'b0;

state=west;

end

else

begin

count=count+3'b001;

state=easty;

end

end

west :

begin

if (count==3'b111)

begin

state=westy;

count=3'b0;

end

else

begin

count=count+3'b001;

state=west;

end

end

westy :

begin

if (count==3'b011)

begin

state=north;

count=3'b0;

end

else

begin

count=count+3'b001;

state=westy;

end

end

endcase // case (state)

end // always @ (state)

end

always @(state)

begin

case (state)

north :

begin

nl = 3'b001;

sl = 3'b100;

el = 3'b100;

wl = 3'b100;

end // case: north

northy :

begin

nl = 3'b010;

sl = 3'b100;

el = 3'b100;

wl = 3'b100;

end // case: north\_y

south :

begin

nl = 3'b100;

sl = 3'b001;

el = 3'b100;

wl= 3'b100;

end // case: south

southy :

begin

nl= 3'b100;

sl= 3'b010;

el = 3'b100;

wl = 3'b100;

end // case: south\_y

west :

begin

nl = 3'b100;

sl = 3'b100;

el = 3'b100;

wl = 3'b001;

end // case: west

westy :

begin

nl = 3'b100;

sl = 3'b100;

el = 3'b100;

wl= 3'b010;

end // case: west\_y

east :

begin

nl = 3'b100;

sl = 3'b100;

el = 3'b001;

wl =3'b100;

end // case: east

easty :

begin

nl = 3'b100;

sl = 3'b100;

el = 3'b010;

wl = 3'b100;

end // case: east\_y

endcase // case (state)

end // always @ (state)

endmodule

* **TESTBENCH CODE FOR VERILOG**

`timescale 1ns/1ps

module tsc\_tb;

wire [2:0] nl,sl,el,wl;

reg clk,rst;

tsc DUT (nl,sl,el,wl,clk,rst);

initial

begin

clk=1'b1;

forever #5 clk=~clk;

end

initial

begin

rst=1'b1;

#15;

rst=1'b0;

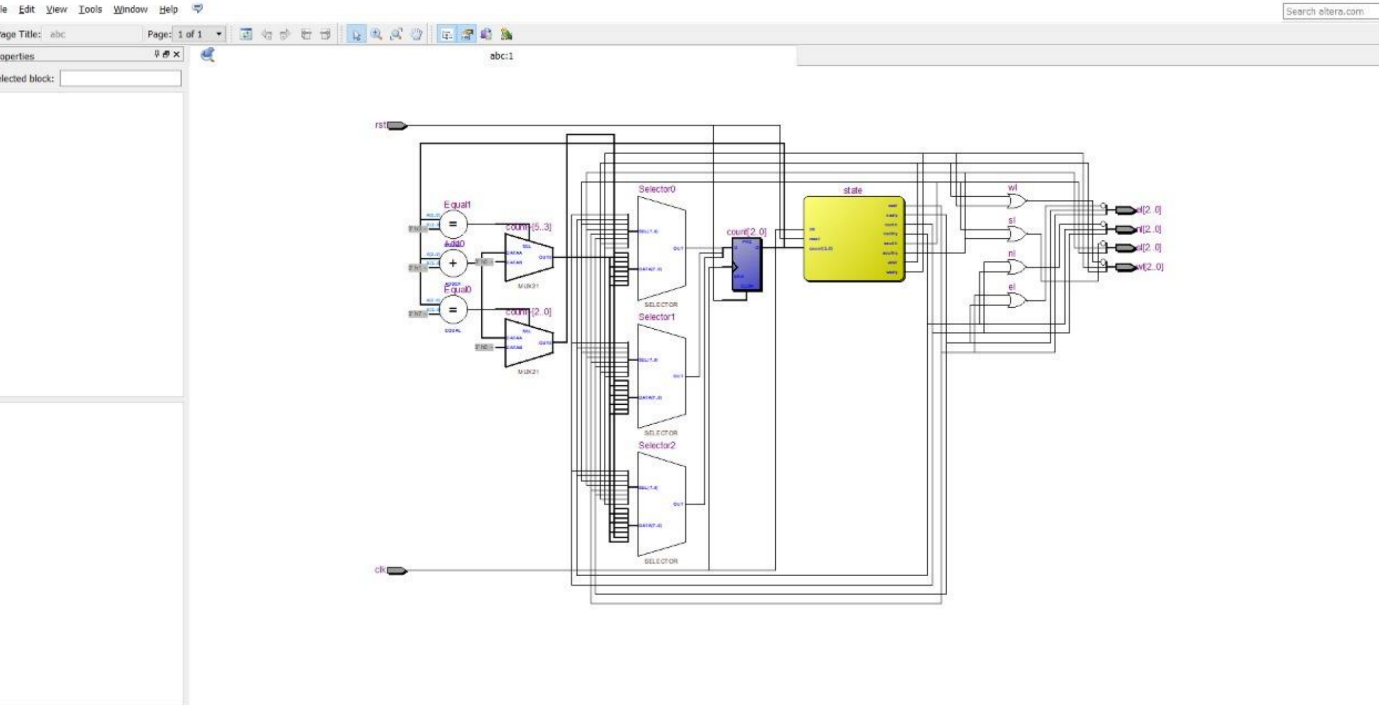
#1000;

$stop;

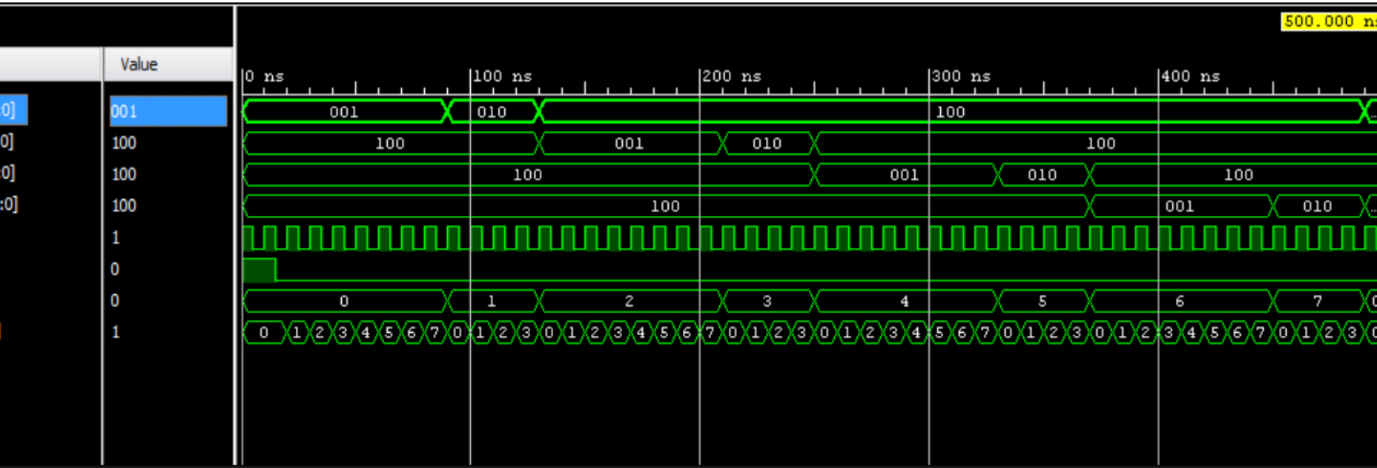
end

endmodule

* **GATE LEVEL**

****

* **RTL SIMULATION**

****

* **CASES**

**case 1:**

first the state is defined to north i.e, green when clock is zero then to change the

state from north to south by first shifting to north\_y i.e shifting to yellow then

the transitions takes place to south

**case 2:**

in this if the state is north\_y then convert it into south i.e, green and if it is not

in north\_y state then first convert it into north\_y i.e, yellow

**case 3:**

in this if the state is south then convert into south\_y i.e, yellow and if it is not in

state south then first it is converted into south state

**case4:**

in this if the state is south\_y then convert into east i.e, yellow and if it is not

south\_y then it is converted into south\_y

**case 5:**

in this if it is east then it is converted into east\_y i.e, yellow and if it is not in

state east then first it is converted into east state

**case 6:**

in this if it is east\_y then it is converted to west and then from west\_y i.e,

yellow it is again converted to north i.e, green

* **CONCLUSION**

The Traffic Light is an important agent in preventing road accident. Thus, Designing and Implementing traffic light controller is a task itself. With this Experiment a person can see that just using the given cases we can branch out implementing state diagram, state table and the Verilog code easily.