DIGITAL DESIGN COURSE PROJECT

TRAFFIC LIGHT CONTROLLER

Bhavya Manish Sharma (B20EE013)
Ghelani Shubham Bhaveshbhai (B20EE019)
Gohel Mruganshi Jatinbhai(B20CS014)

Introduction

- A design of a modern Traffic Light Controller System to manage the road traffic.
- The main objective was to design a hardware that directs the unit to glow correct traffic light in all the four directions of the road at appropriate times for the specified time intervals. Finite state machine model was used to implement the same.

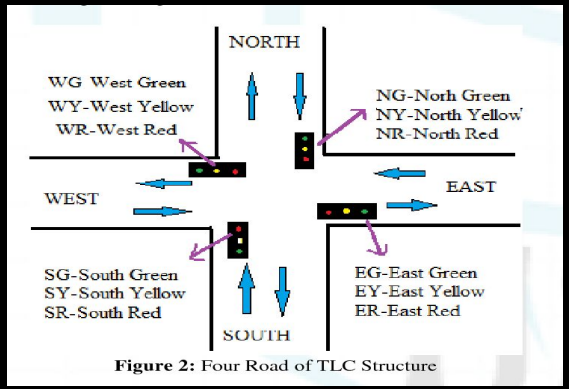
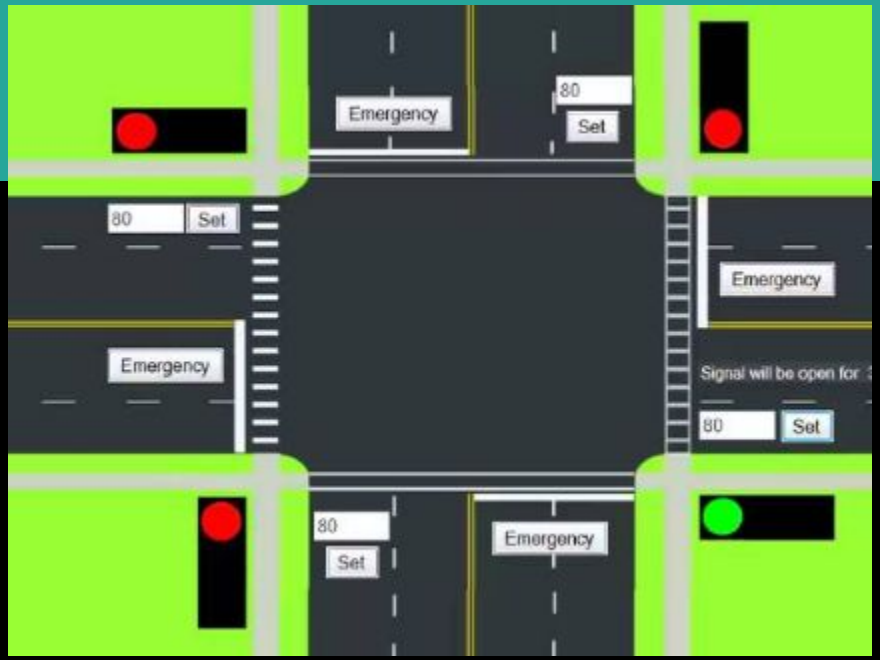
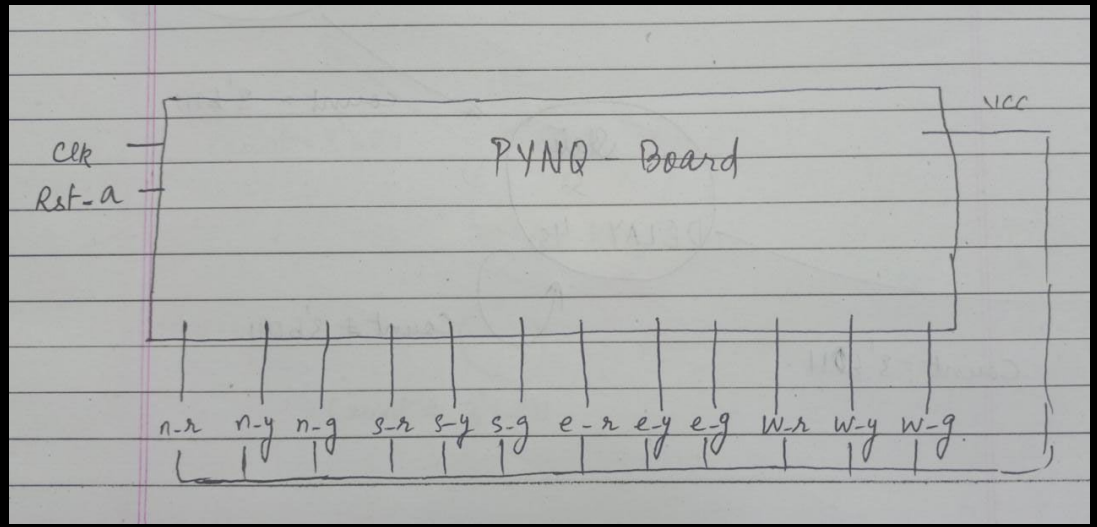
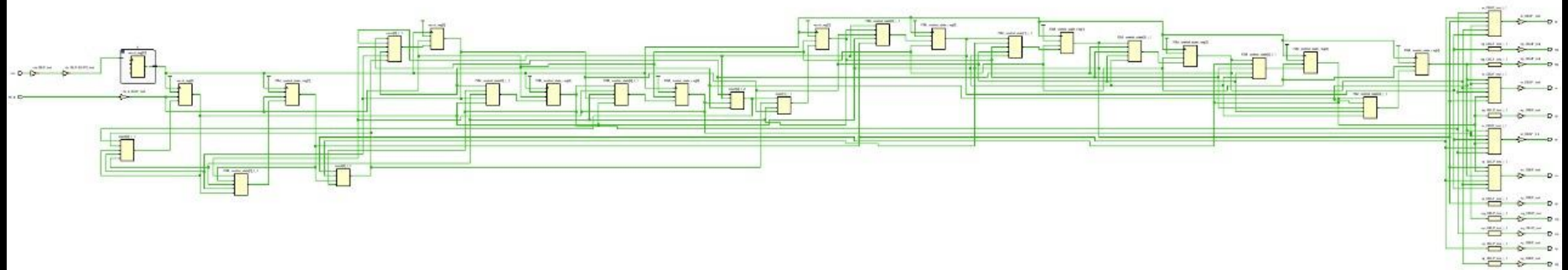


Figure 2: Four Road of TLC Structure


Emergency
80
Set
80
Set
Emergency
Emergency
Signal will be open for
80
Set
80
Set
Emergency

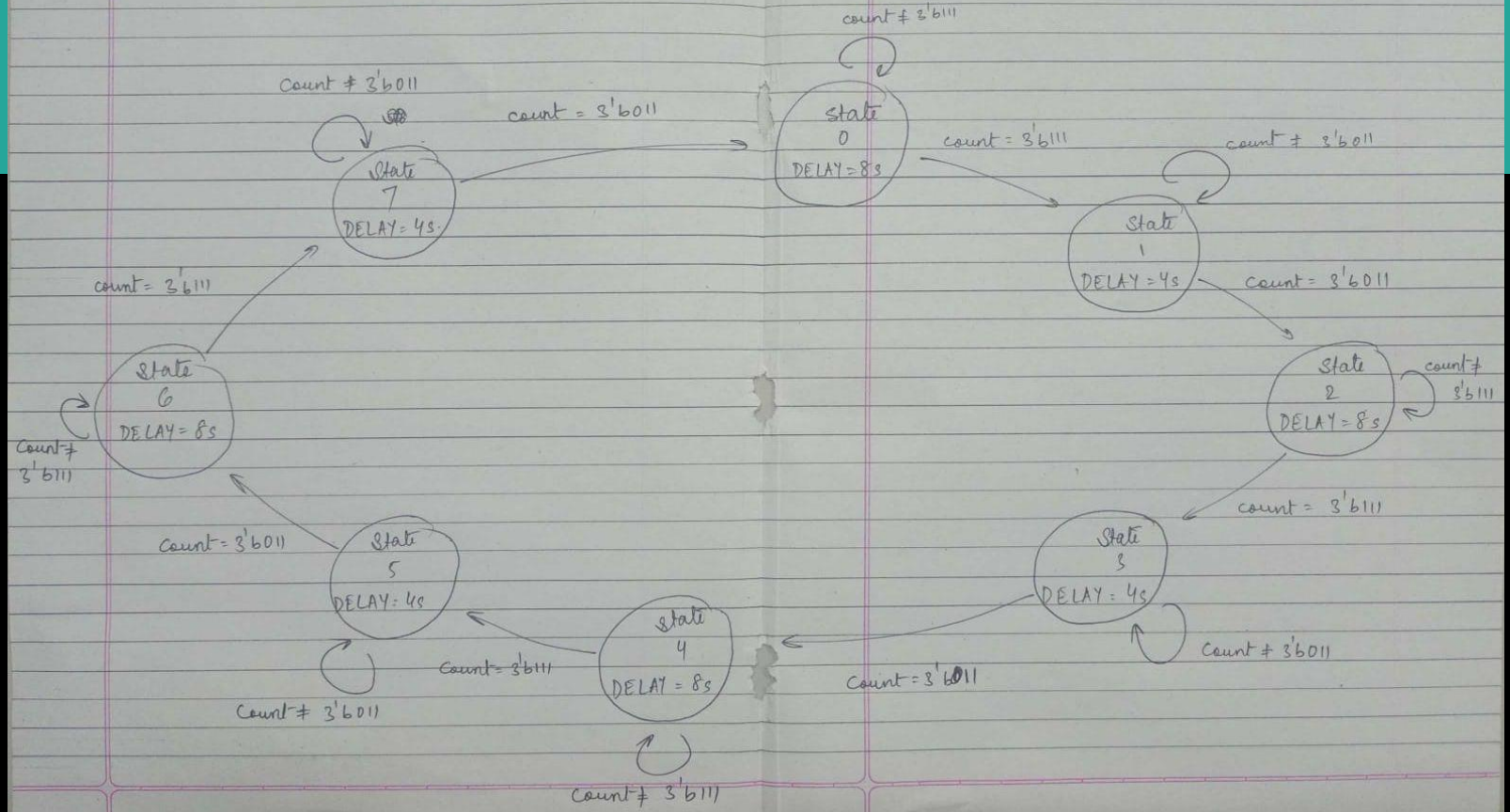
BLOCK DIAGRAM





STATE DIAGRAM

STATE DIAGRAM



STATE TABLE

STATE TABLE

States	N-lights			S-lights			E-lights			W-lights		
	n-r	n-y	n-g	s-r	s-y	s-g	e-r	e-y	e-g	w-r	w-y	w-g
STATE 0 (North)	0	0	1	1	0	0	1	0	0	1	0	0
STATE 1 (South-y)	1	0	0	0	1	0	1	0	0	1	0	0
STATE 2 (South)	1	0	0	0	0	1	1	0	0	1	0	0
STATE 3 (East-y)	1	0	0	1	0	0	0	1	0	1	0	0
STATE 4 (East)	1	0	0	1	0	0	0	0	1	1	0	0
STATE 5 (West-y)	1	0	0	1	0	0	1	0	0	0	1	0
STATE 6 (West)	1	0	0	1	0	0	1	0	0	0	0	1
STATE 7 (North-y)	0	1	0	1	0	0	1	0	0	1	0	0

General description of code

- We have implemented our design code for traffic light controller using Finite State Machine with 8 states.
- In this clk and rst_a are two input signal and n_lights, s_lights, e_lights and w_lights are 3 bit registers used to store value for the 12 output signals(n_r,n_y,n_g,s_r,s_y,s_g,e_r,e_y,e_g,w_r,w_y,w_g).
- The first bit of register gives the value for red light, second bit of register gives the value of yellow light and third bit of register gives the value of green light.
- On the reset signal, design will enter into north state and start giving output after reset will go low. Design will turn on green light for eight clock cycles and yellow light for four clock cycles.
- Design will start with north, then goes into south , then east and finally into west and by this it will keep going.

Detailed Logic of code

- We have 12 different outputs (here, LED's) and they are divided into 4 sets with 3 in each of them i.e. Red, Yellow and Green and the 4 sets are representing the 4 directions i.e. North, South, East and West.
- We have a clock and a reset button. Pressing the reset button will bring back the circuit to default state i.e. North state. After 8 counts, the yellow light of the next state will turn on and red will turn off. In this case south's yellow light will turn on. After next 4 counts, the yellow light will turn off and green light will turn on. Also the north will now turn red. Same will happen for other states.
- We have 3 bit binary numbers which will represent the state of every direction. After that using these binary numbers we will assign the single digit binary number to our 12 final variables.

```
module clk_divider(cin,clk);
input cin;
output clk;

reg [27:0] count=0;
always@(posedge cin)
    count = count + 1;

assign clk = count[27];

endmodule
```

```
always @(posedge clk, posedge rst_a)
  begin
      if (rst_a)
          begin
              state=north;
              count =3'b000;
          end
      else
          begin
              case (state)
              north :
                  begin
                      if (count==3'b111)
                          begin
                          count=3'b000;
                          state=south_y;
                          end
                      else
                          begin
                          count=count+3'b001;
                          state=north;
                          end
                  end
```

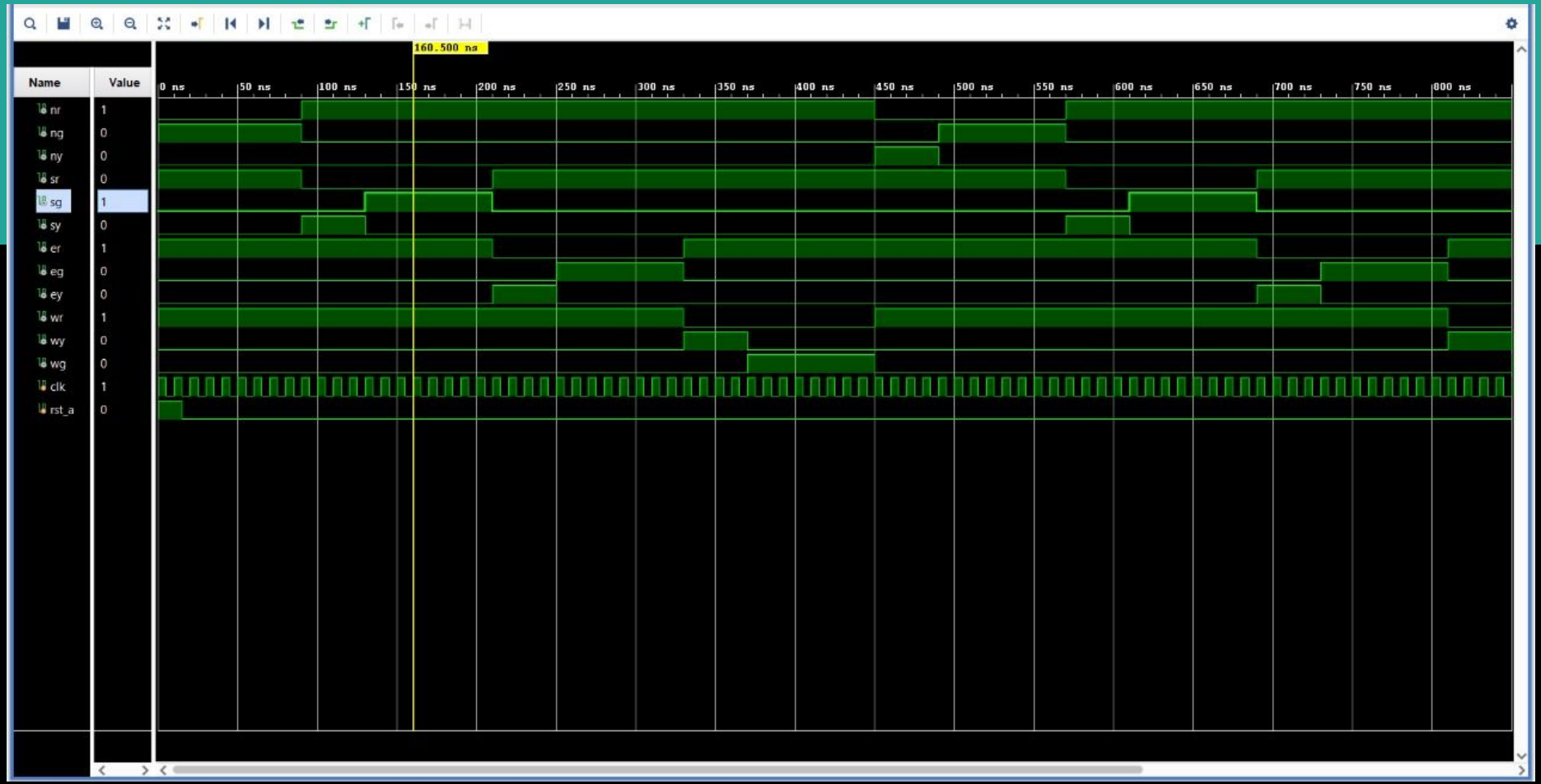
```
always @(state)
     begin
         case (state)
            north :
                begin
                    n_lights = 3'b001;
                    s_lights = 3'b100;
                    e_lights = 3'b100;
                    w_lights = 3'b100;
                end // case: north
```

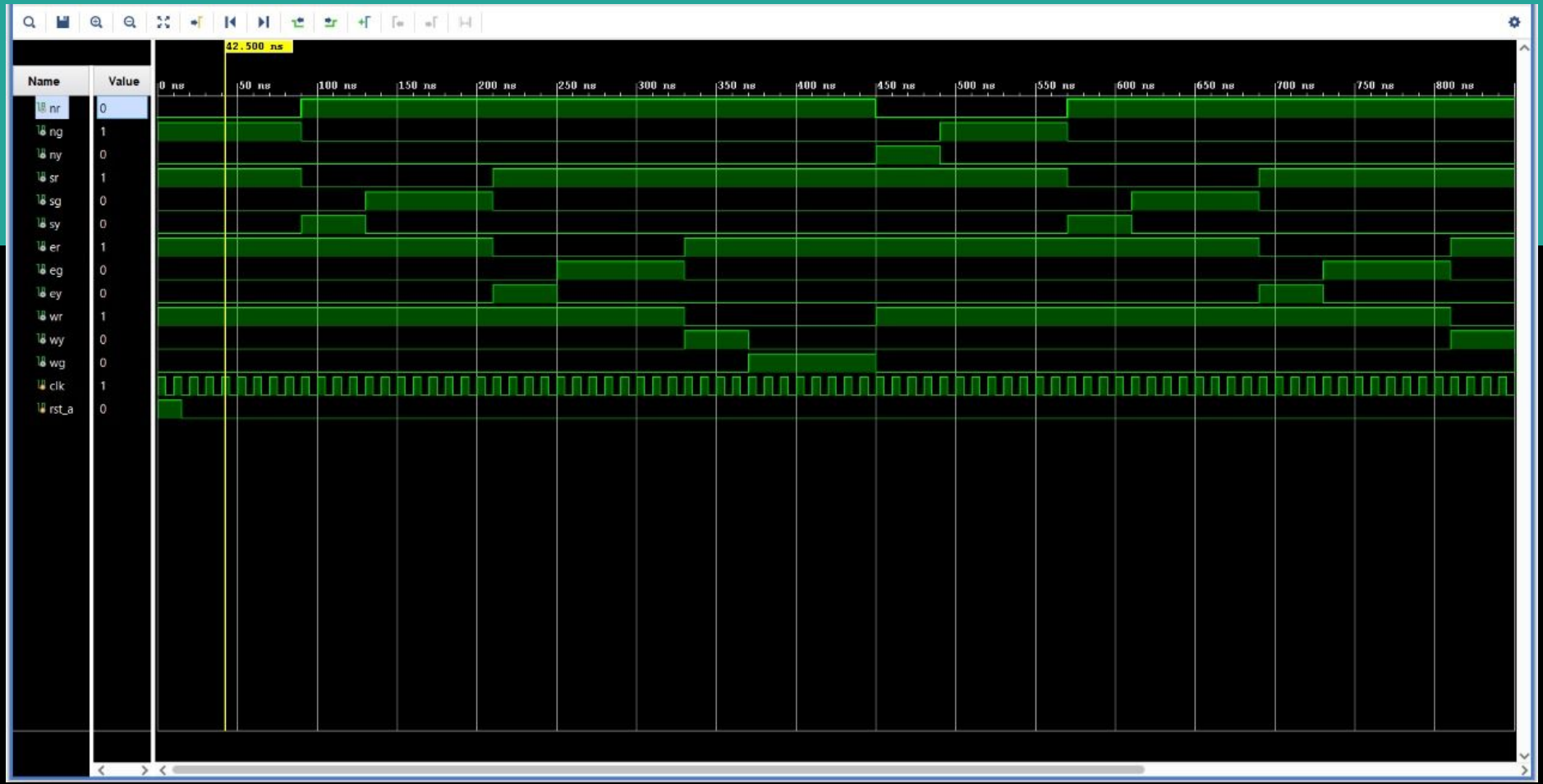
```
reg [2:0] count;
```

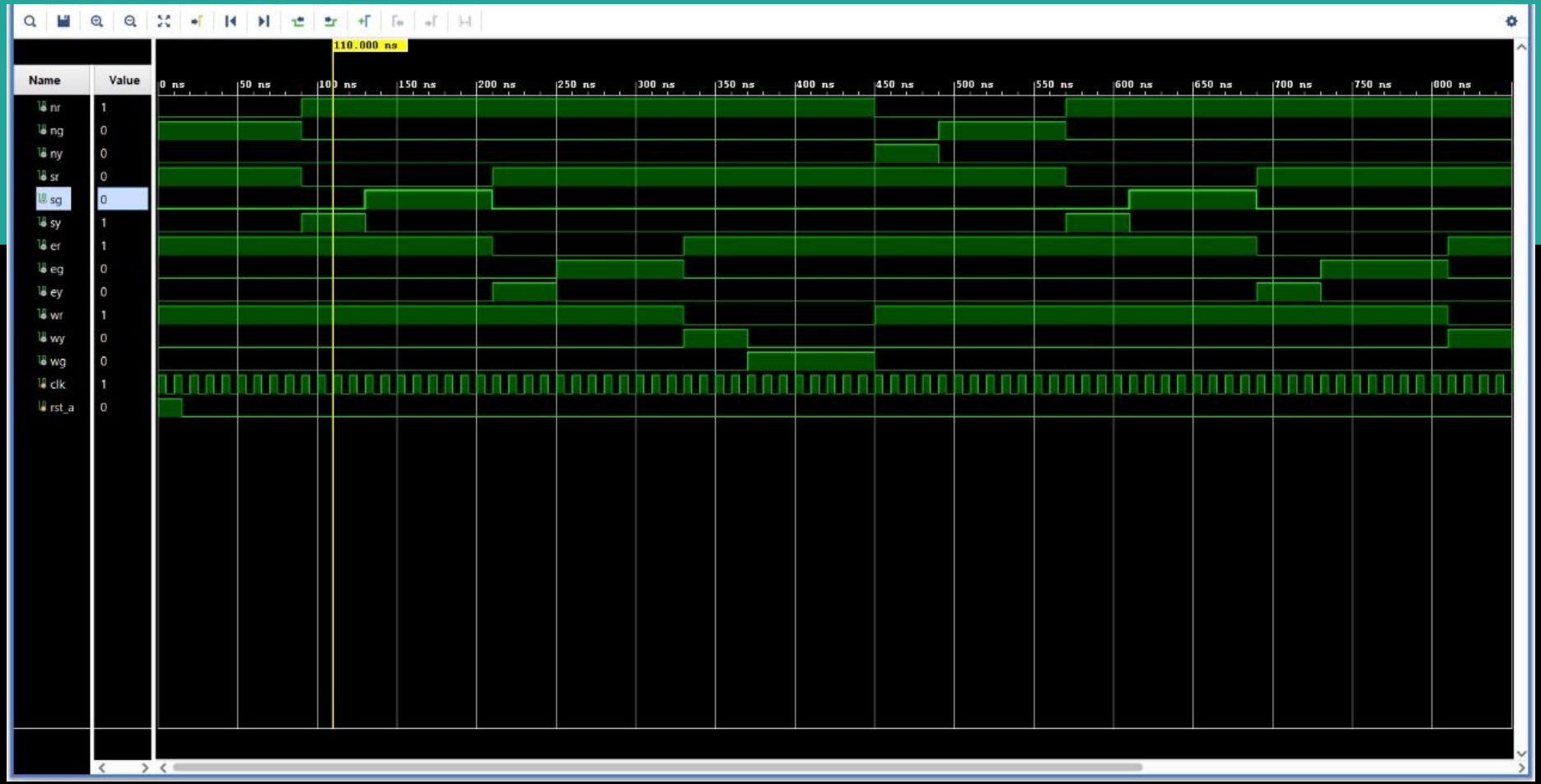
PINS AND VARIABLES

No.	Name of the pin	Direction	Width	Description
1	N_r, n_y, n_g	Output	1 each(total 3)	North lights
2	S_r, s_y, s_g	Output	1 each(total 3)	South lights
3	E_r, e_y, e_g	Output	1 each(total 3)	East lights
4	W_r, w_y, w_g	Output	1 each(total 3)	West lights
5	clk	Input	1	Clock signal
6	Rst_a	Input	1	Reset signal

SIMULATION


160.500 ns
Name
Value
0 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns
700 ns
750 ns
800 ns
nr 1
ng 0
ny 0
sr 0
sg 1
sy 0
er 1
eg 0
ey 0
wr 1
wy 0
wg 0
clk 1
rst_a 0


42.500 ns
Name
Value
nr
ng
ny
sr
sg
sy
er
eg
ey
wr
wy
wg
clk
rst_a
0
1
0
1
0
0
1
0
0
1
0
0
1
0
0 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns
700 ns
750 ns
800 ns


110.000 ns
Name
Value
0 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns
700 ns
750 ns
800 ns
nr 1
ng 0
ny 0
sr 0
sg 0
sy 1
er 1
eg 0
ey 0
wr 1
wy 0
wg 0
clk 1
rst_a 0