****

**BENGALURU, INDIA**

**HACKTHON PROJECT REPORT**

REVA HACK</>2021

TRAFFIC LIGHTS CONTOLLER USING VLSI CODE

**ICONIC TEAM :**

TEAM MEMBERS:

V.HEMANTH KUMAR (R19CS365)

SRISHTI CHETHAN (R19CS456)

VARSHITHA R(R19CS366)

SWAPNA M(R19CS342)

INDEX

**CONTENT**:

TITLE PAGE

INTRODUCTION

DESCRIPTION

VERILOG CODE

SCHEMATIC DIAGRAM

SIMULATION DIAGRAM

SYNTHESIS AND IMPLEMENTATION

DIAGRAM

**INTRODUCTION**

**HISTORY**

The world's very first traffic lights were invented by JP Knight Installed near London's House Of Commons. Which was on the intersection of George and Bridge Street, in 1868

RED :

Come to complete stop at stop line or before crosswalks or intersection

YELLOW :

Stop if you can do so safely

GREEN :

Go, but only if intersection is clear

Software used:

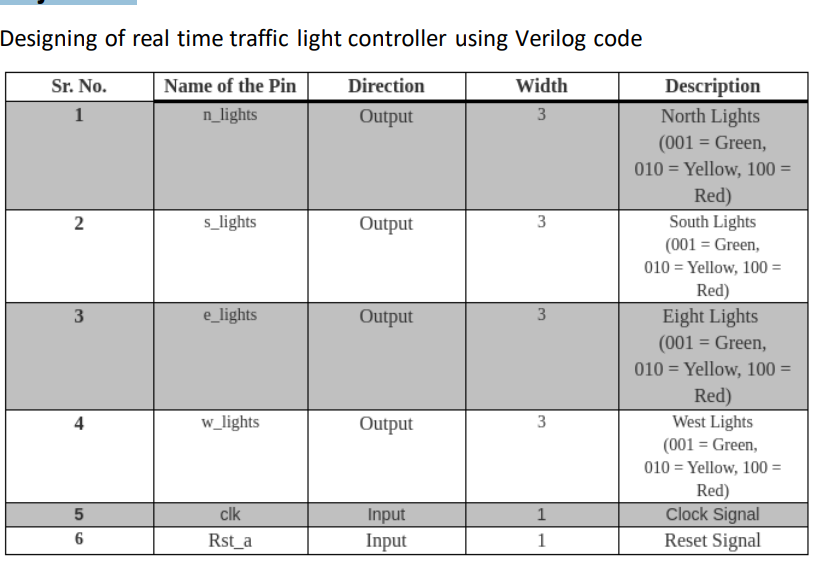
XILINX of version 2020.1

Language used: Verilog

**DESCRIPTION**

Given below code is design code for Traffic Light Controller using Finite State Machine(FSM). In this clk and rst\_a are two input signal and n\_lights, s\_lights, e\_lights and w\_lights and state, count are 3 bit output signal. In output signal, "001’ represents green light, "010" represents Yellow light and "100" represents Red 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.

Objective:



Verilog code:

module traffic\_control(n\_lights,s\_lights,e\_lights,w\_lights,clk,rst\_a);

output reg [2:0]n\_lights,s\_lights,e\_lights,w\_lights;

input clk;

input rst\_a;

reg [2:0] state;

parameter [2:0] north=3'b000; parameter [2:0] north\_y=3'b001; parameter [2:0] south=3'b010; parameter [2:0] south\_y=3'b011; parameter [2:0] east=3'b100; parameter [2:0] east\_y=3'b101; parameter [2:0] west=3'b110; parameter [2:0] west\_y=3'b111;

reg [2:0] count;

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=north\_y; end

else

begin count=count+3'b001; state=north;

end

end

north\_y :

begin if (count==3'b011) begin count=3'b000; state=south; end

else

begin count=count+3'b001; state=north\_y; end

end

south: begin

if(count==3'b111) begin count=3'b0; state=south\_y; end

else

begin count=count+3'b001; state=south;

end end

south\_y: begin

if(count==3'b011) begin

count=3'b0; state=east;

end

else begin count=count+3'b001; state=south\_y; end

end

east: begin

if (count==3'b111) begin count=3'b0; state=east\_y; end

else begin count=count+3'b001; state=east; end

end

east\_y :

begin

if(count==3'b011) begin count=3'b0; state=west;

end

else

begin count=count+3'b001; state=east\_y; end

end

west: begin

if(count==3'b111) begin state=west\_y; count=3'b0; end

else begin

count=count+3'b001; state=west; end

end

west\_y:

begin

if(count==3'b011) begin state=north; count=3'b0;

end

else

begin count=count+3'b001; state=west\_y; end

end

endcase // case(state)

end // always @ (state)

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

north\_y : begin n\_lights = 3'b010; s\_lights = 3'b100; e\_lights = 3'b100; w\_lights = 3'b100; end // case: north\_y south :

begin n\_lights = 3'b100; s\_lights = 3'b001; e\_lights = 3'b100; w\_lights = 3'b100; end // case: south

south\_y : begin n\_lights = 3'b100; s\_lights = 3'b010; e\_lights = 3'b100; w\_lights = 3'b100; end // case: south\_y

west : begin n\_lights = 3'b100; s\_lights = 3'b100; e\_lights = 3'b100; w\_lights = 3'b001; end // case: west west\_y :

begin n\_lights = 3'b100; s\_lights = 3'b100; e\_lights = 3'b100; w\_lights = 3'b010; end // case: west\_y

east : begin n\_lights = 3'b100; s\_lights = 3'b100; e\_lights = 3'b001; w\_lights = 3'b100; end // case: east

east\_y : begin n\_lights = 3'b100; s\_lights = 3'b100; e\_lights = 3'b010; w\_lights = 3'b100; end // case: east\_y endcase // case (state) end // always @ (state) end module

**SCHEMATIC DIAGRAM**

**Diagram, schematic

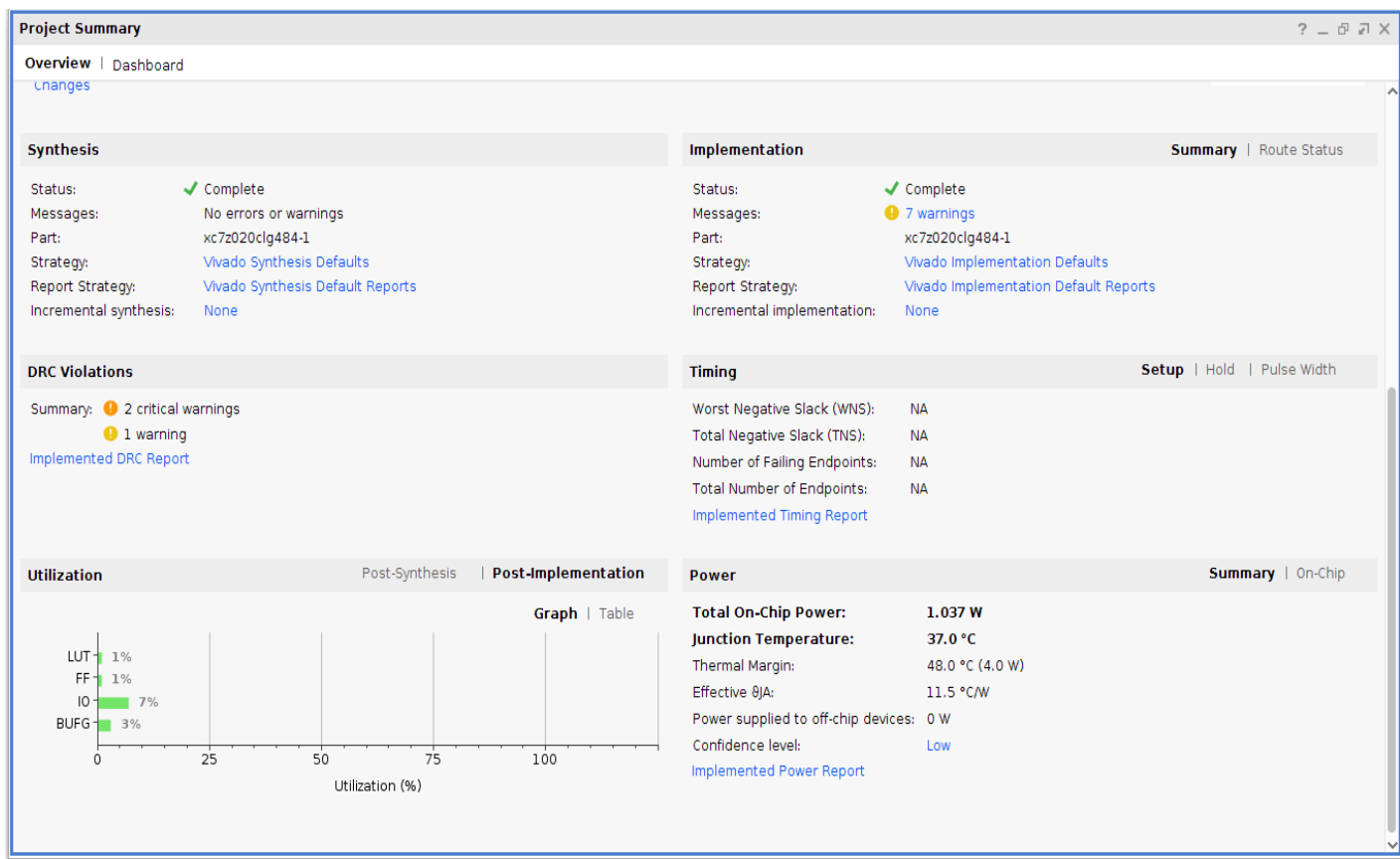
Description automatically generated**

**SIMULATION**

**Graphical user interface

Description automatically generated**

**SYNTHESIS AND IMPLEMENTATION REPORT:**



**THANK YOU**