**SELF-PROJECT**

**"T-INTERSECTION ROAD LIGHT CONTROLLER USING VERILOG"**

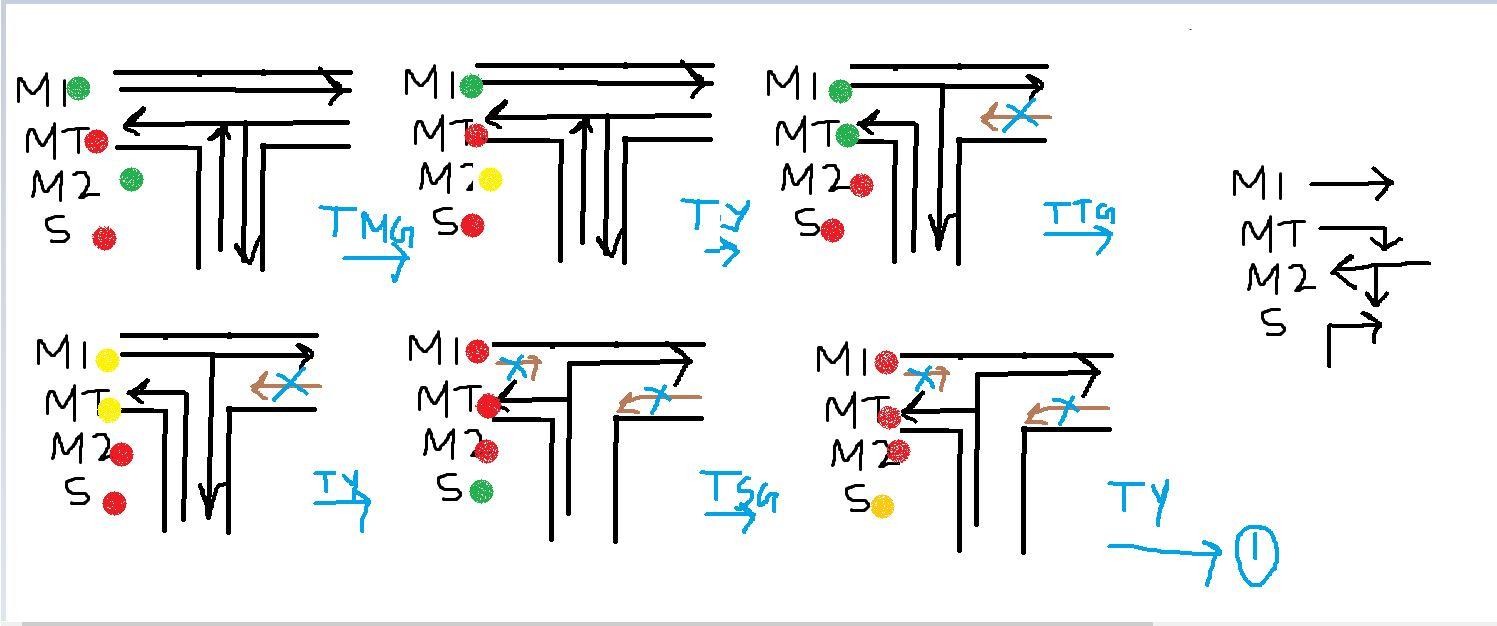
**ANKIT KHANDELWAL**

**Second-Year Undergraduate**

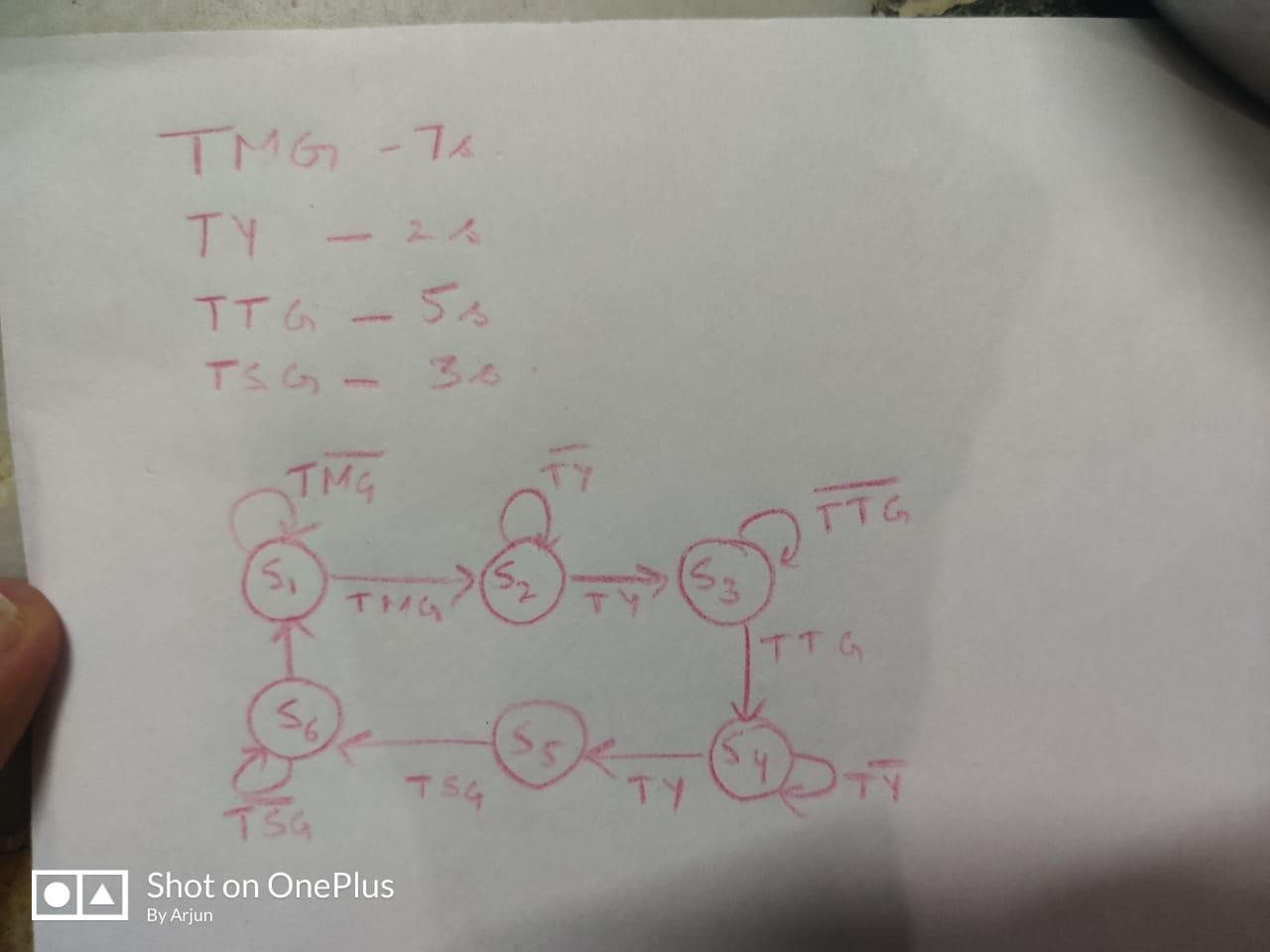
**Department of EE, IIT Kanpur**

**PROBLEM STATEMENT**

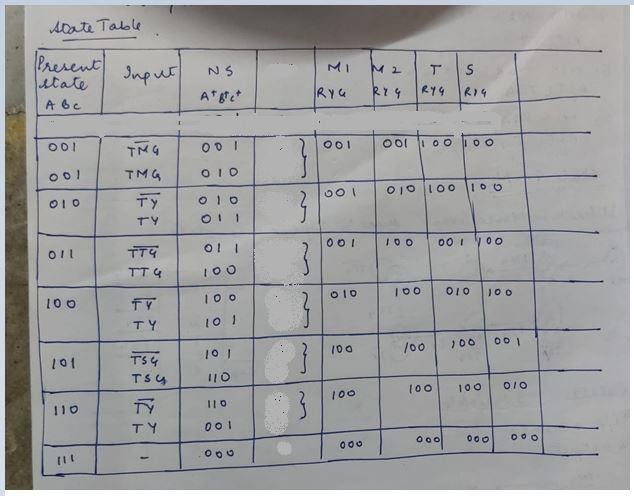
The aim of the project is to design a traffic controller for a T-intersection. Let’s understand the problem statement through the image given below.



The six cases present here eventually turn to the six states . This is the state diagram:



From the state diagram we for the state table:



# VERILOG CODE

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 10.12.2022 12:53:25

// Design Name:

// Module Name: Traffic\_Light\_Controller

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module Traffic\_Light\_Controller(

input clk,rst,

output reg [2:0]light\_M1, output reg [2:0]light\_S, output reg [2:0]light\_MT,

output reg [2:0]light\_M2

);

parameter S1=0, S2=1, S3 =2, S4=3, S5=4,S6=5;

reg [3:0]count; reg[2:0] ps;

parameter sec7=7,sec5=5,sec2=2,sec3=3;

always@(posedge clk or posedge rst) begin

if(rst==1) begin ps<=S1;

count<=0; end

else

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<sec5) 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<sec3) 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

always@(ps) begin

case(ps)

S1:

begin

light\_M1<=3'b001; light\_M2<=3'b001; light\_MT<=3'b100; light\_S<=3'b100;

end S2:

begin

light\_M1<=3'b001; light\_M2<=3'b010; light\_MT<=3'b100; light\_S<=3'b100;

end S3:

begin

light\_M1<=3'b001; light\_M2<=3'b100; light\_MT<=3'b001; light\_S<=3'b100;

end S4:

begin

end

endmodule

light\_M1<=3'b010; light\_M2<=3'b100; light\_MT<=3'b010; light\_S<=3'b100;

end S5:

begin

light\_M1<=3'b100; light\_M2<=3'b100; light\_MT<=3'b100; light\_S<=3'b001;

end S6:

begin

light\_M1<=3'b100; light\_M2<=3'b100; light\_MT<=3'b100; light\_S<=3'b100;

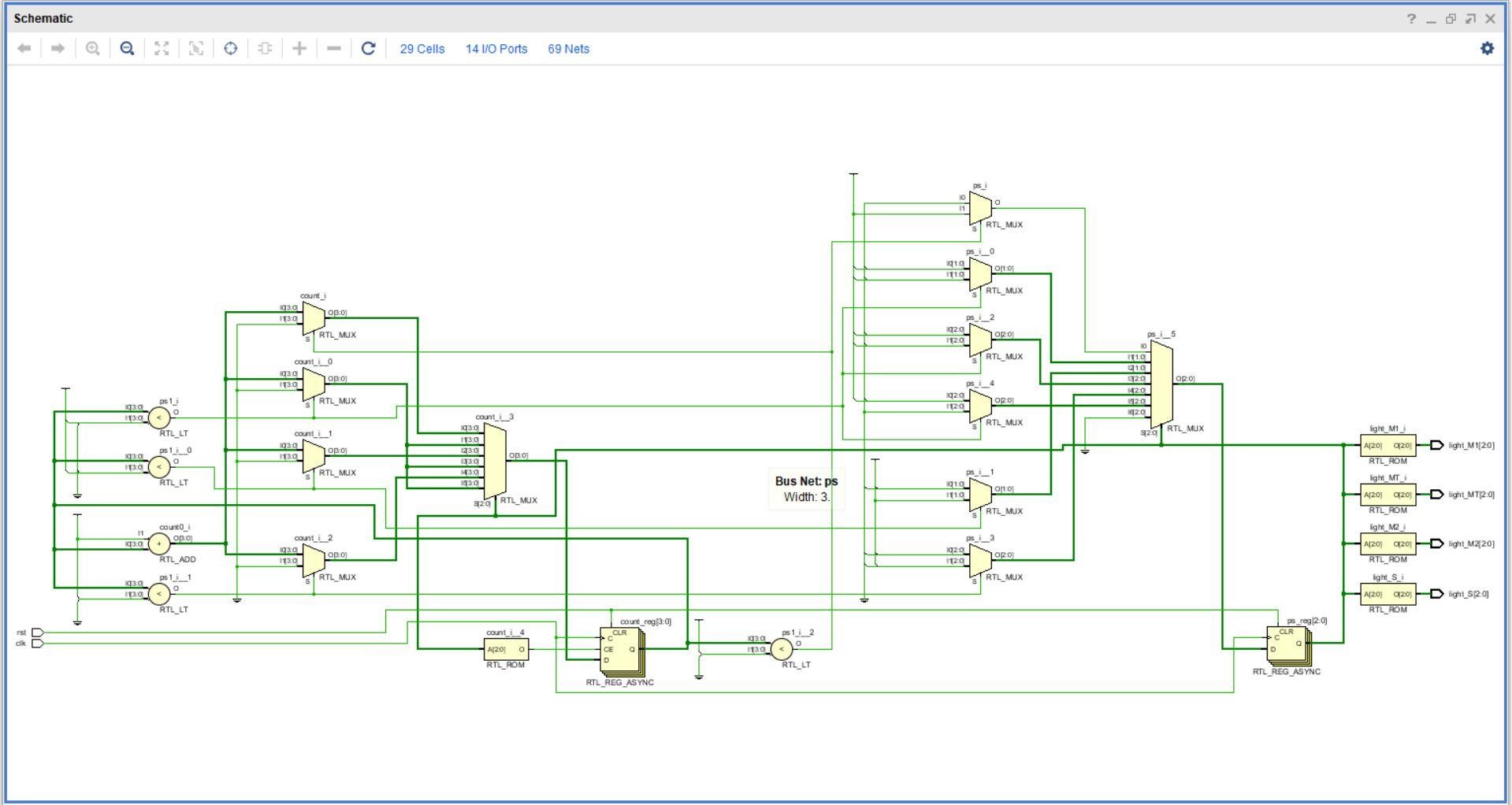
end default:

begin

light\_M1<=3'b000; light\_M2<=3'b000; light\_MT<=3'b000; light\_S<=3'b010;

end endcase

# RTL-SCHEMATIC



**TESTBENCH**

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 16.12.2022 23:44:40

// Design Name:

// Module Name: Traffic\_Light\_Controller\_TB

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module Traffic\_Light\_Controller\_TB; reg clk,rst;

wire [2:0]light\_M1; wire [2:0]light\_S; wire [2:0]light\_MT; wire [2:0]light\_M2;

Traffic\_Light\_Controller dut(.clk(clk) , .rst(rst) , .light\_M1(light\_M1) , .light\_S(light\_S)

,.light\_M2(light\_M2),.light\_MT(light\_MT) ); initial

begin

clk=1'b0;

forever #(1000000000/2) clk=~clk; end

initial begin

rst=0; #1000000000;

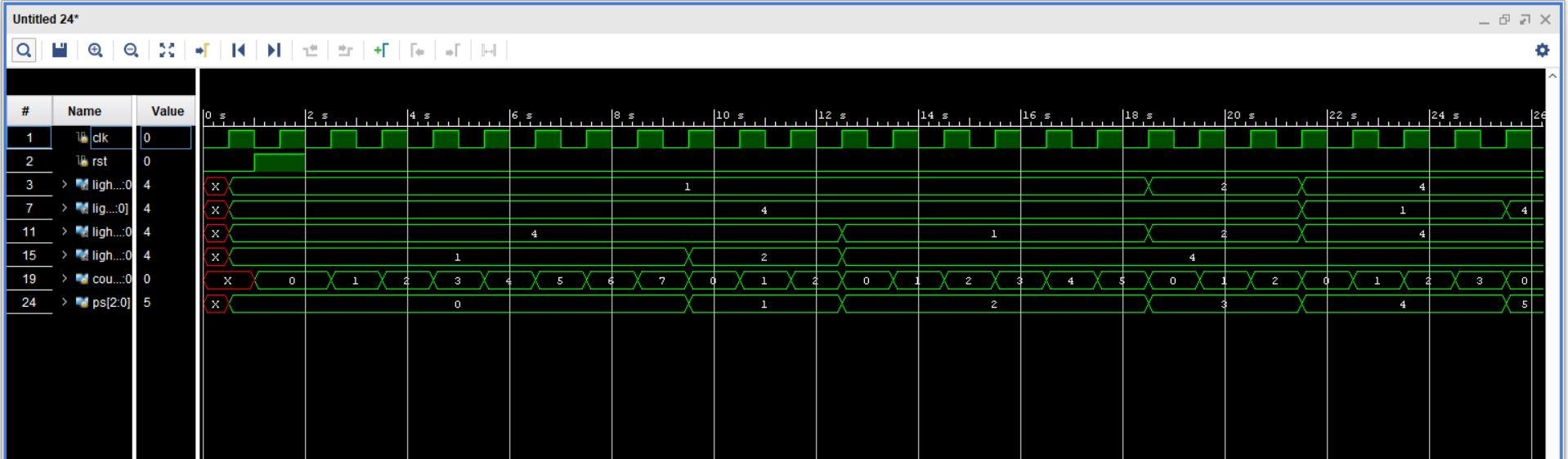
rst=1; #1000000000;

rst=0; #(1000000000\*200);

$finish; end

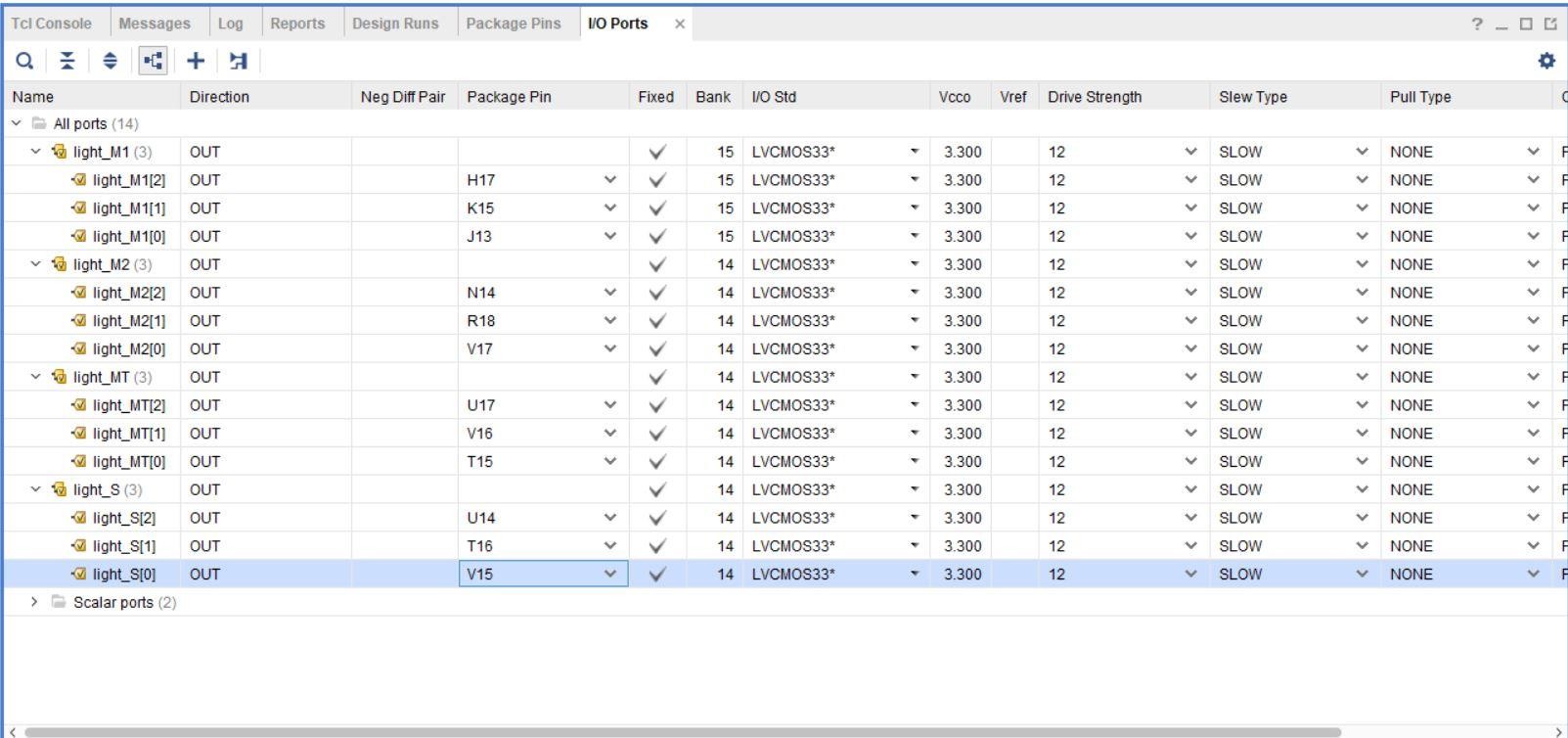
endmodule

# SIMULATED WAVEFORM



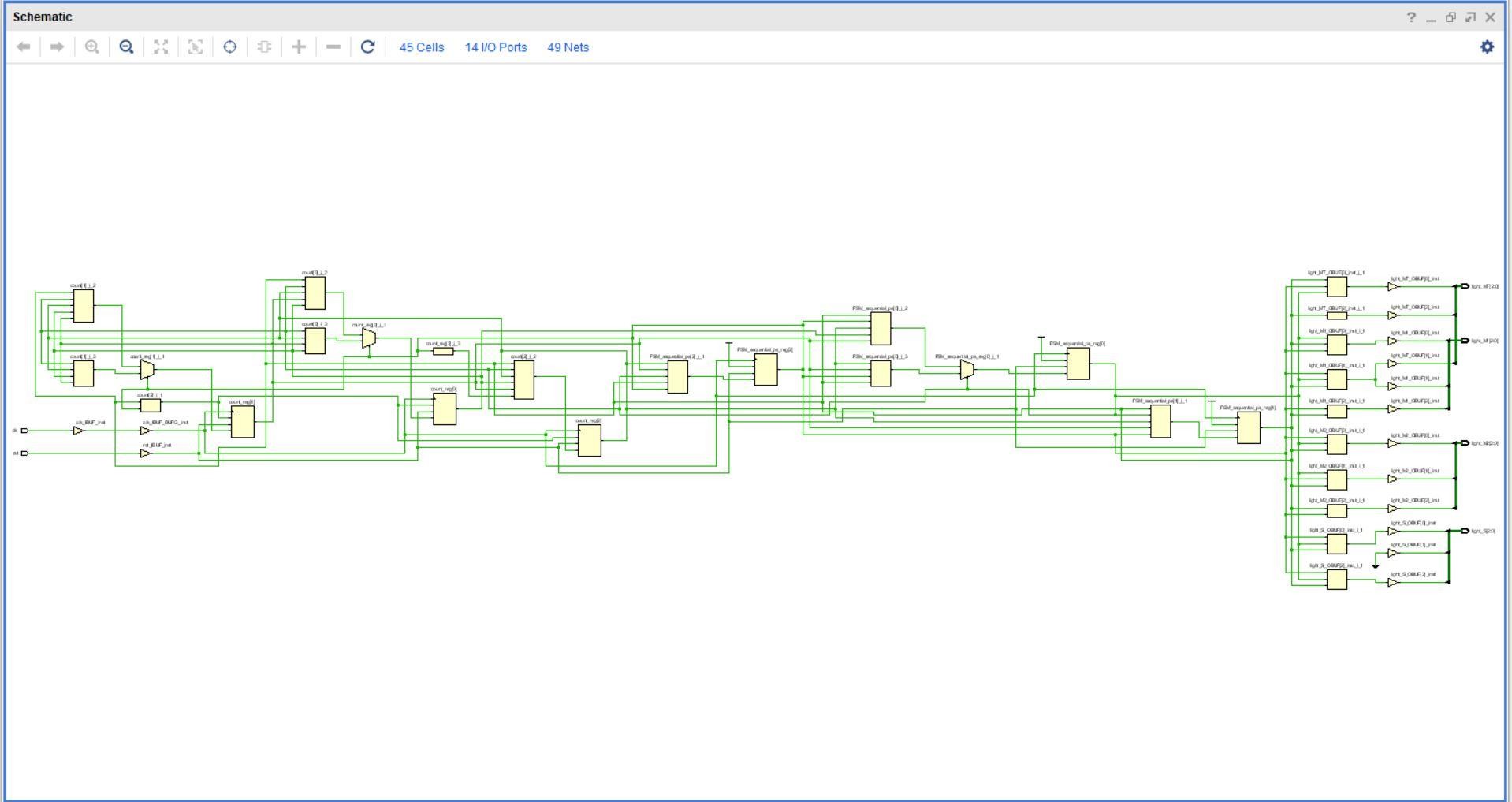
Upon analyzing the waveform we can clearly see that the FSM works perfectly.

# IO PORT ASSIGNMENT



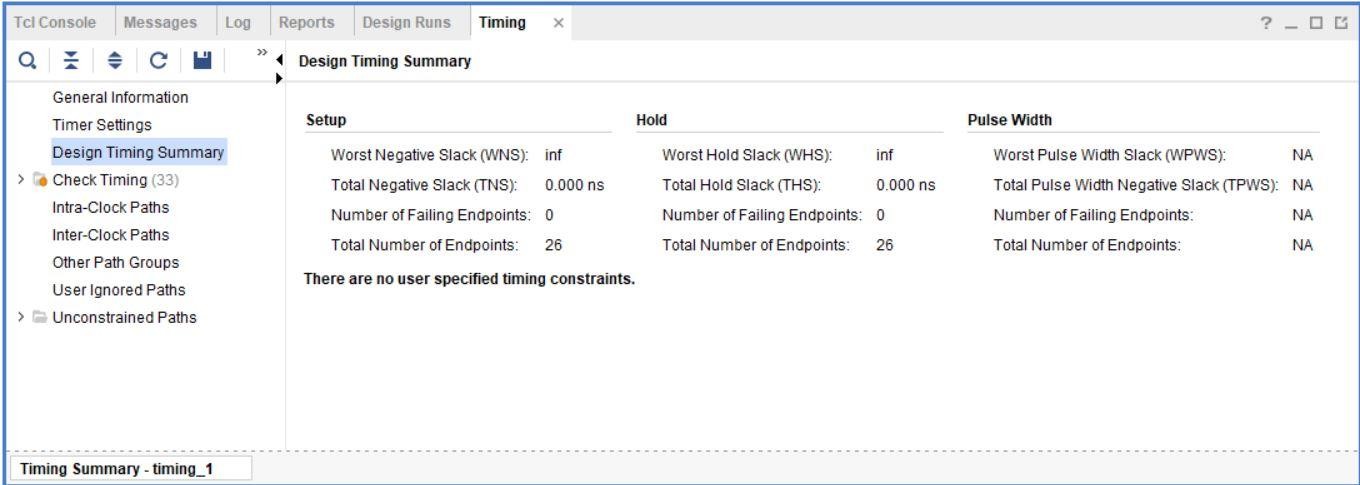
The ports are assigned from the ucf file.

# SCHEMATIC AFTER SYNTHESIS

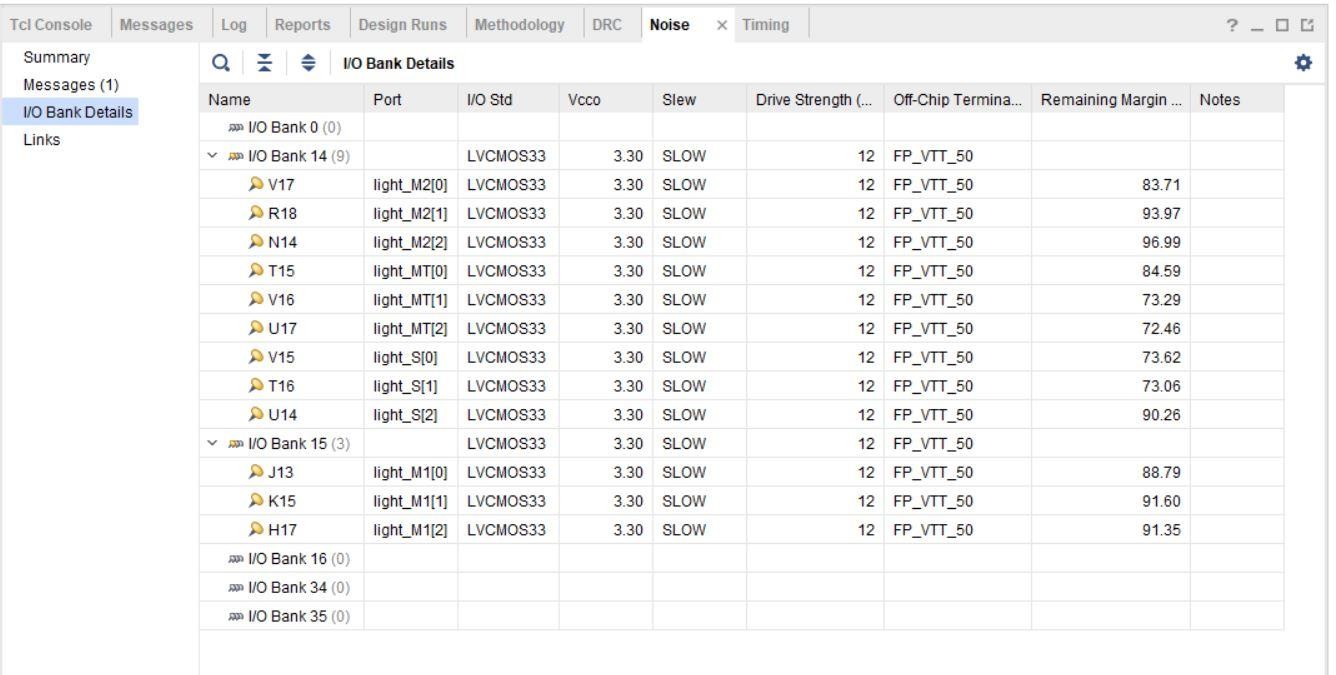


**REPORTS AFTER SYNTHESIS**

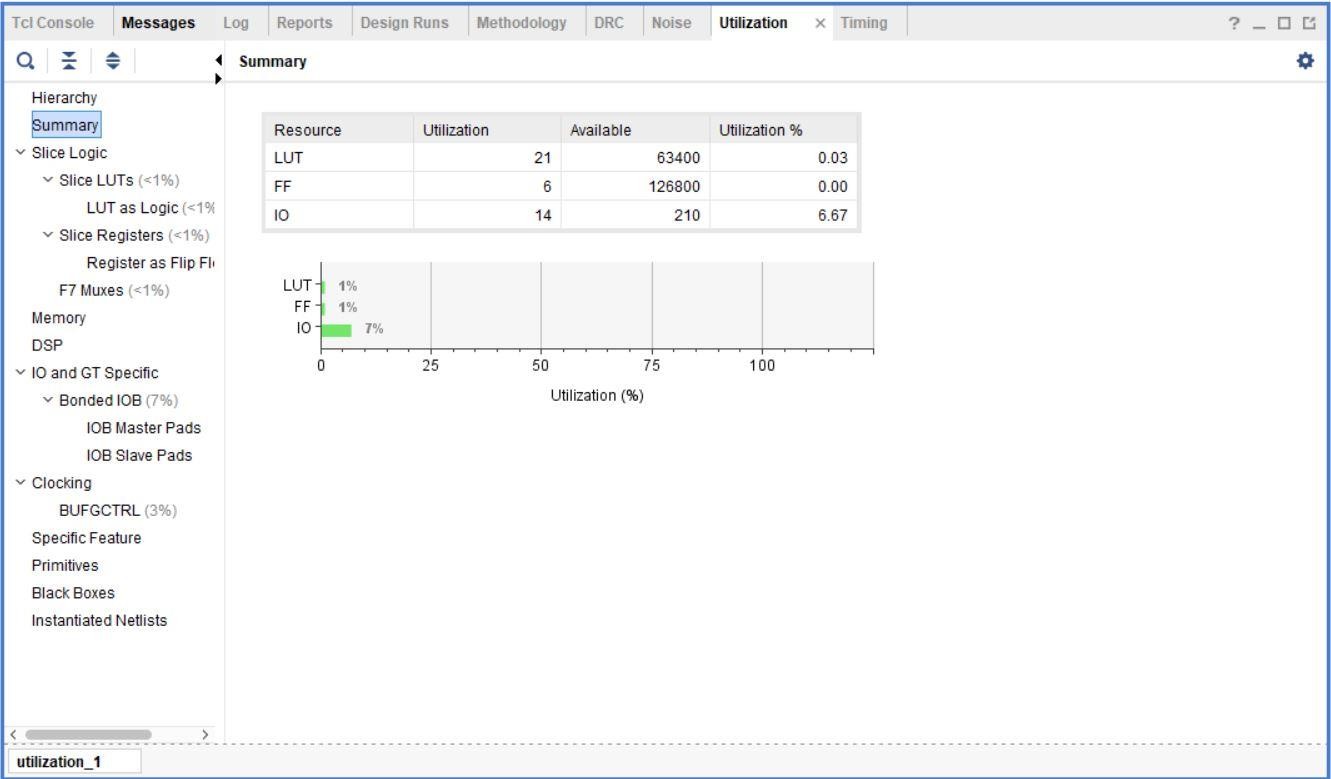
TIMING REPORT



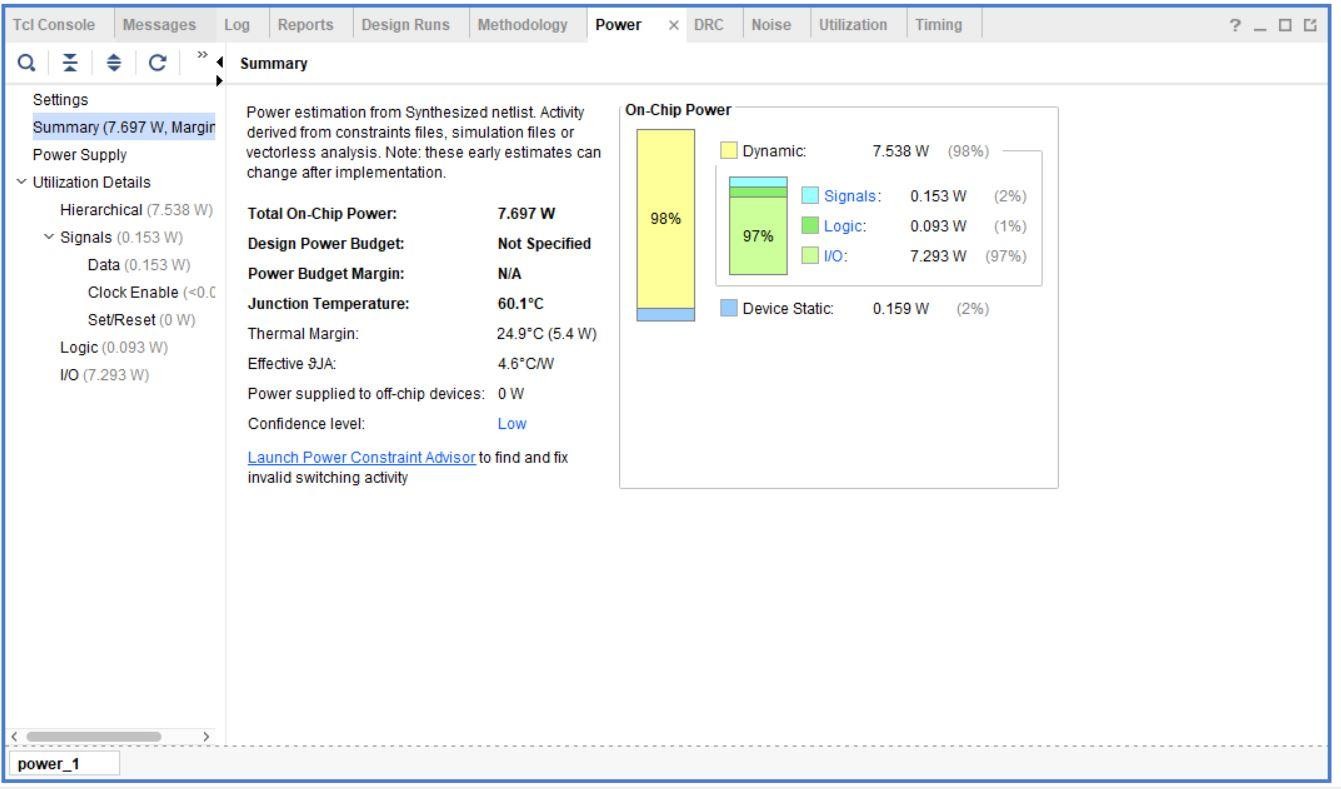
NOISE REPORT



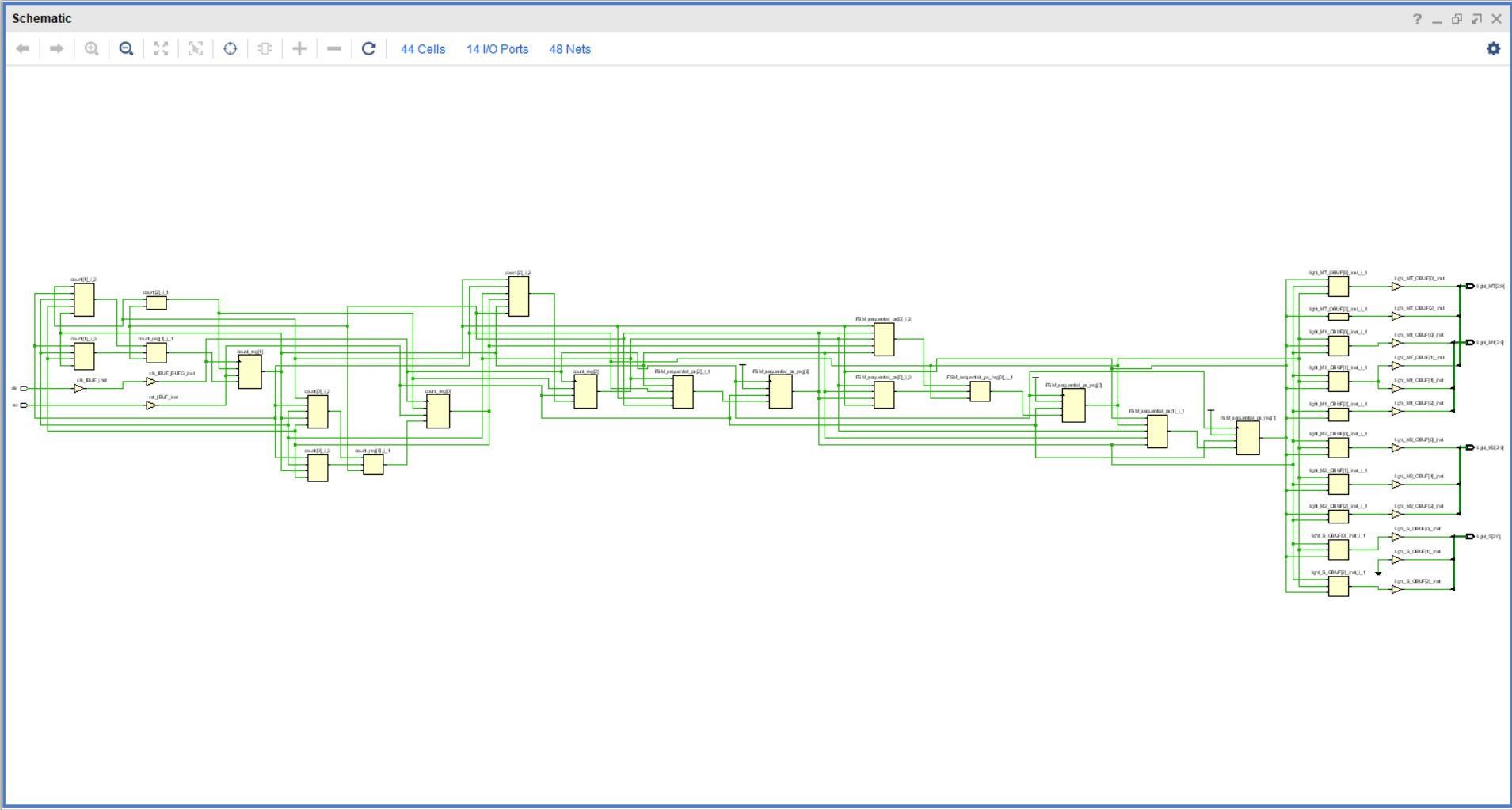
UTILIZATION REPORT



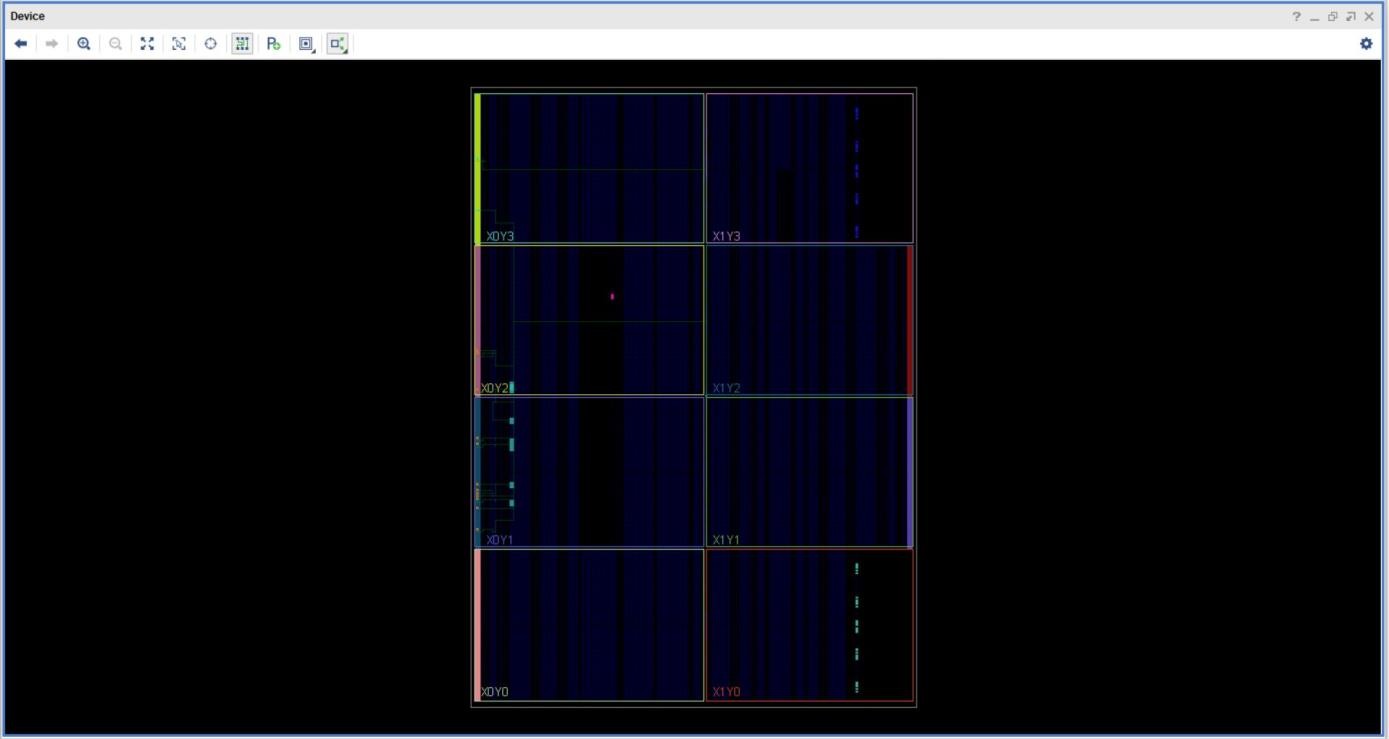
POWER REPORT



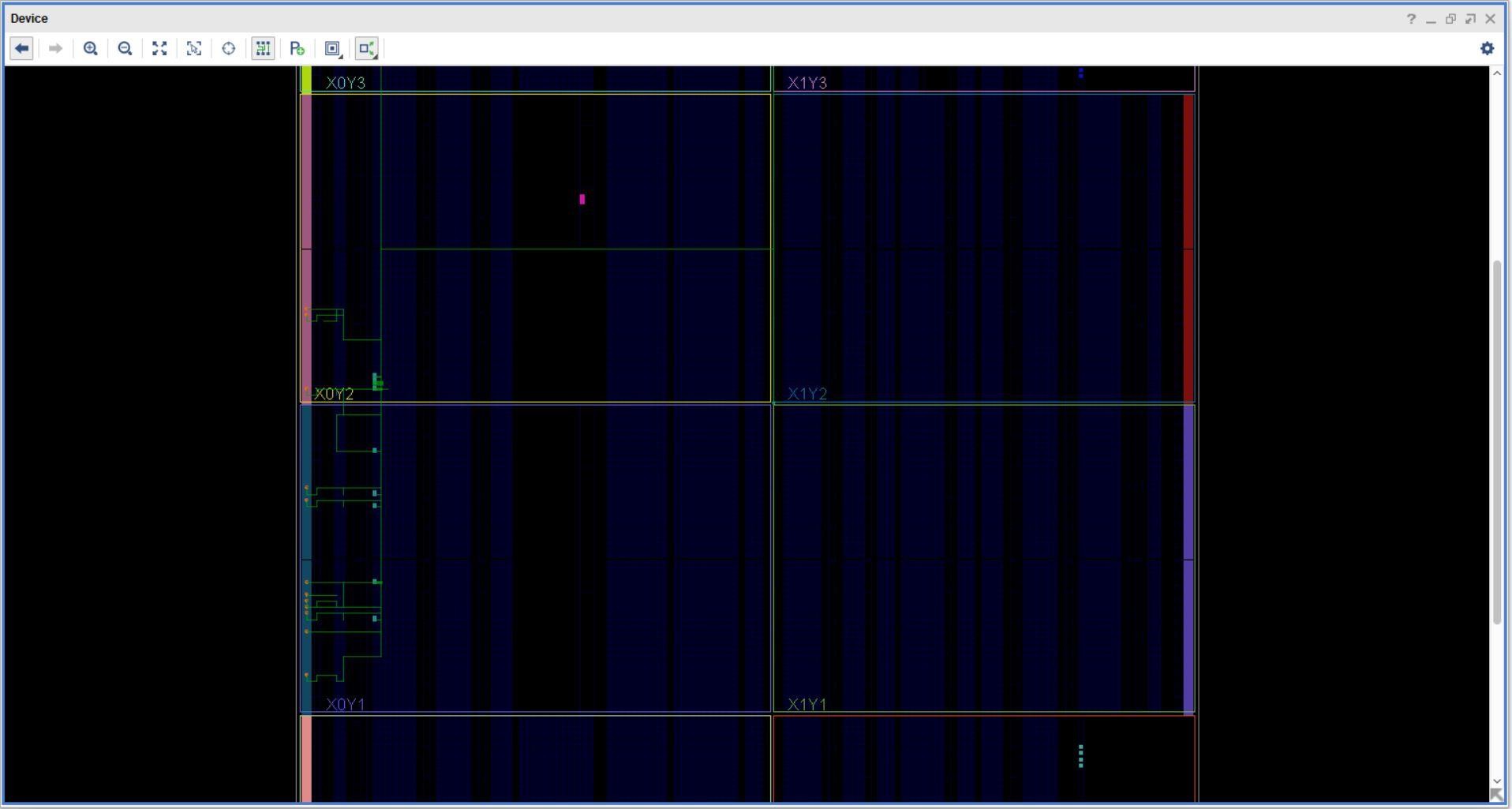
# SCHEMATIC AFTER IMPLEMENTATION



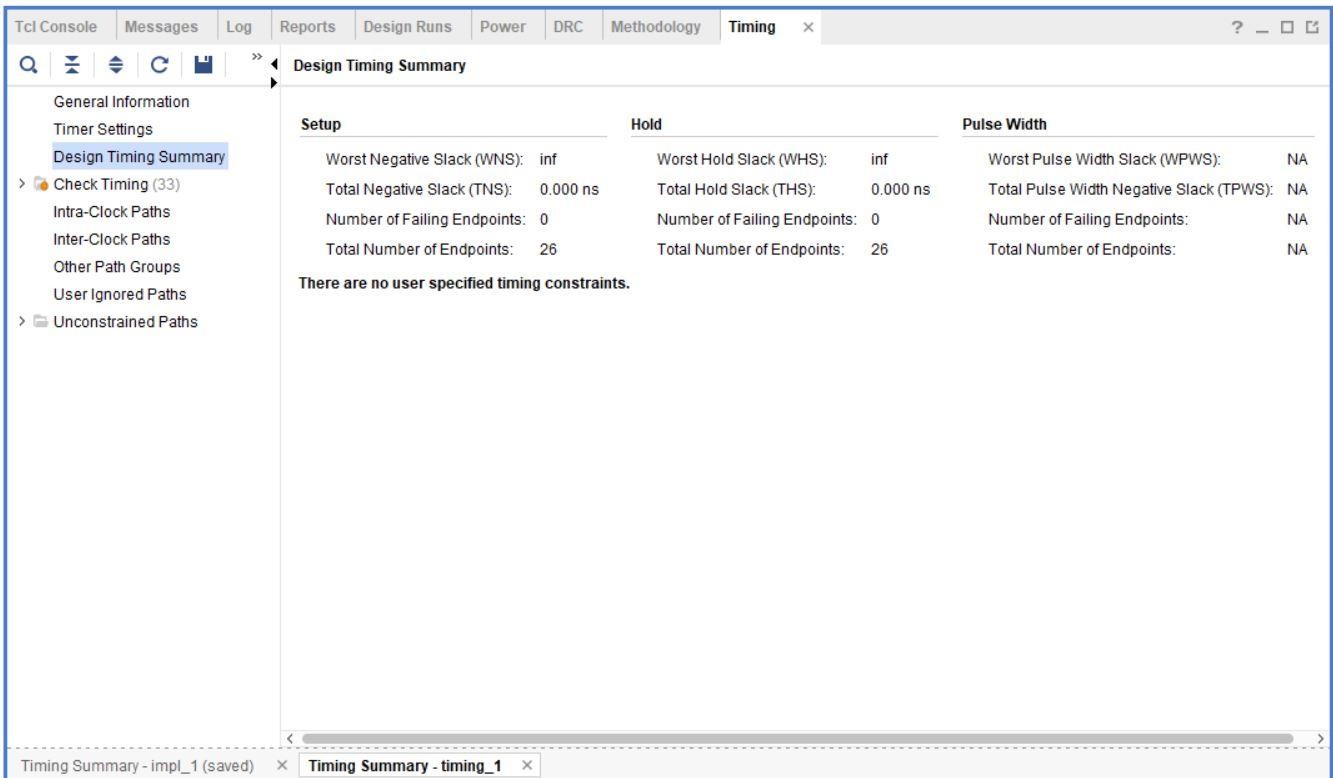
**DEVICE LAYOUT AFTER IMPLEMENTATION**



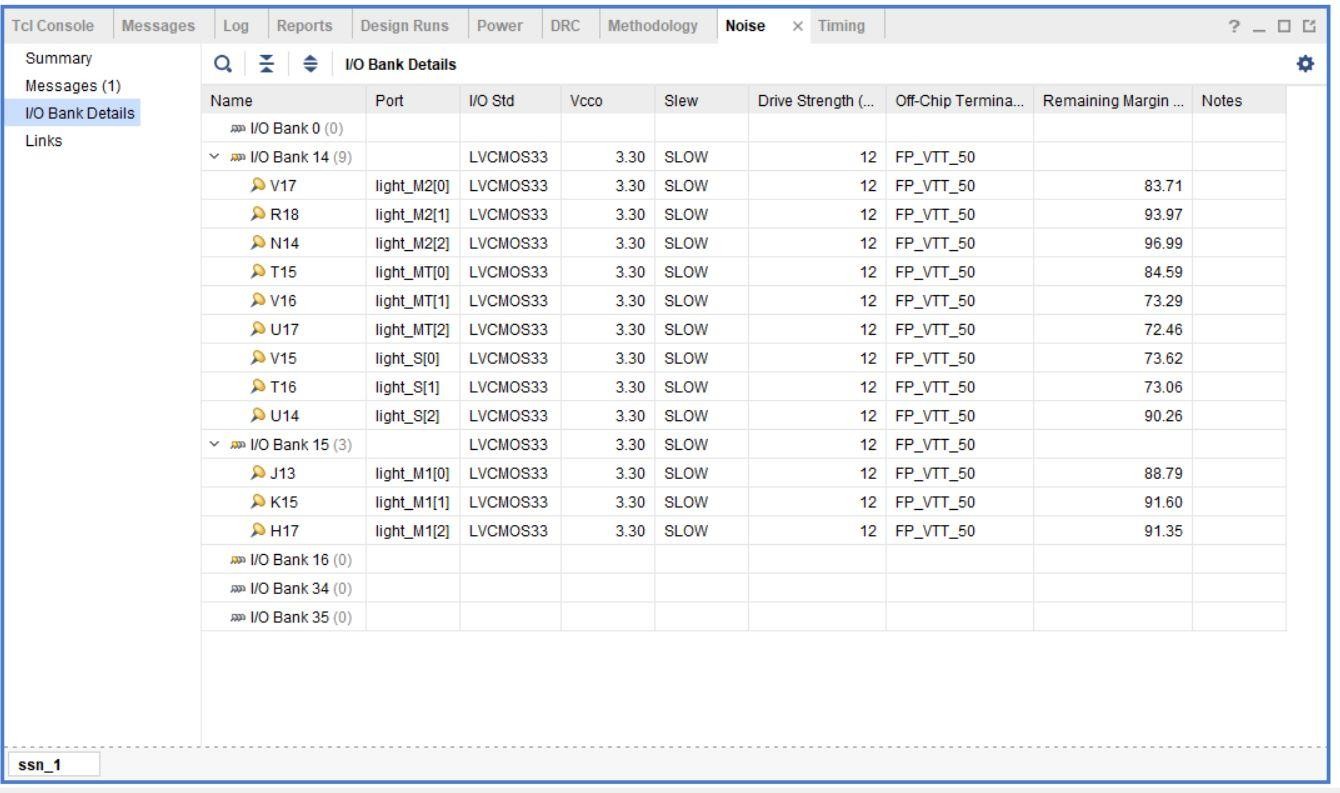
ZOOM IN VIEW



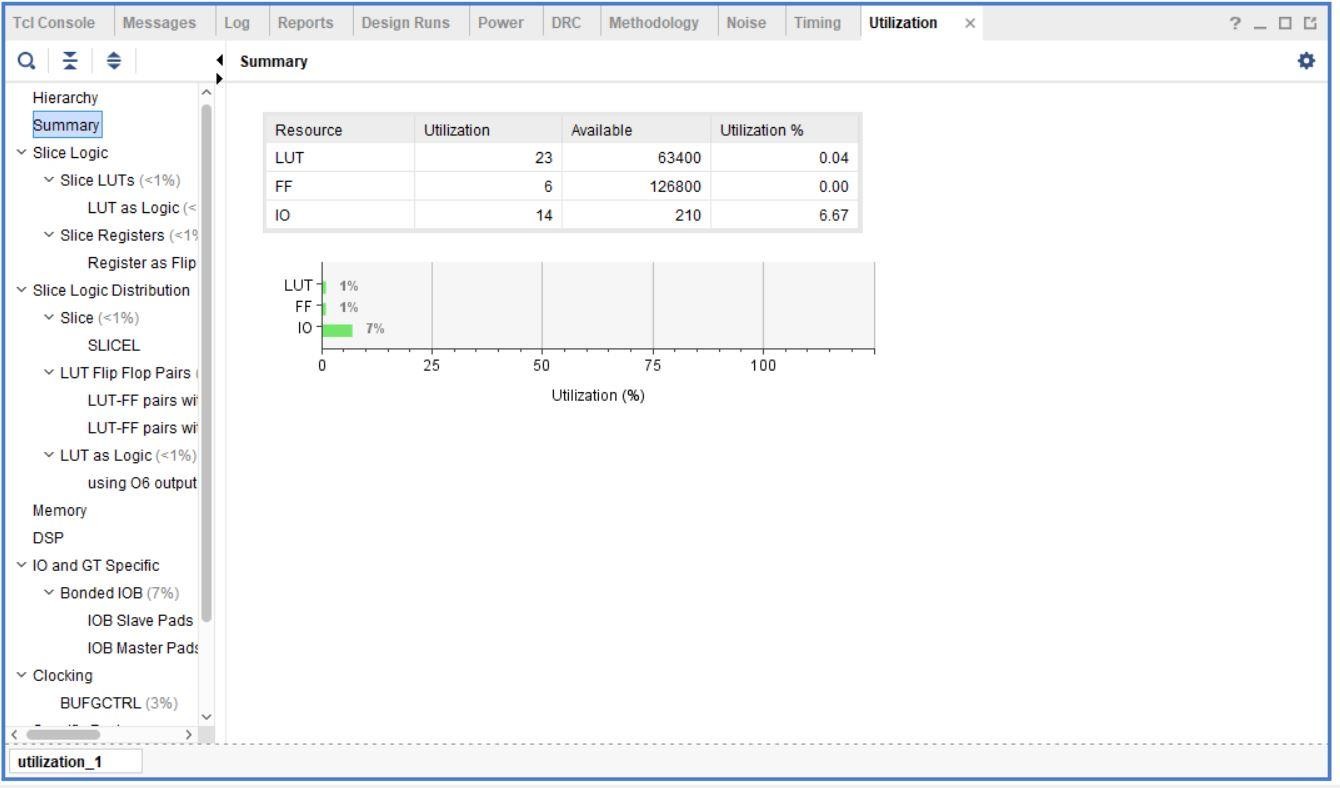
# REPORTS AFTER IMPLEMENTATION

TIMING REPORT

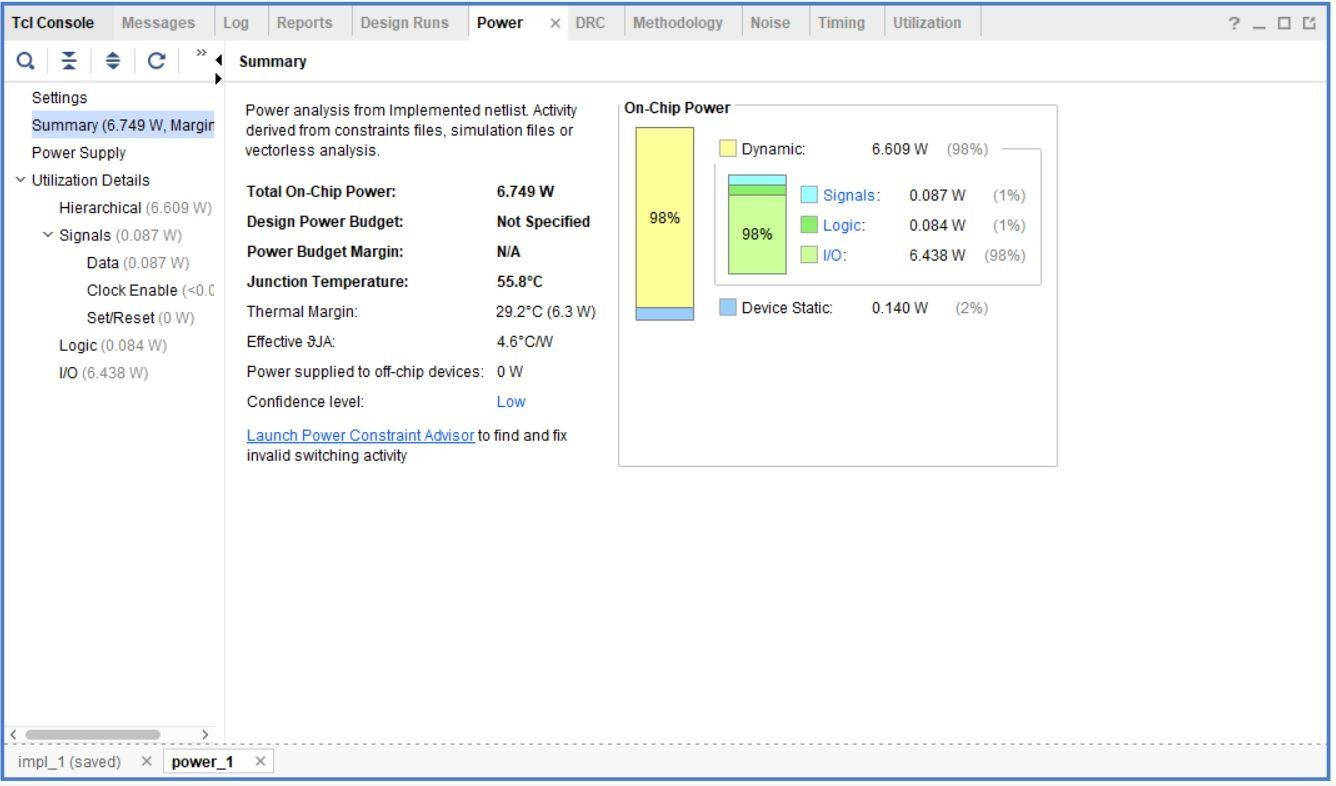
NOISE REPORT



UTILIZATION REPORT



POWER REPORT



.

# REFERENCES:

1. HDLBITS
2. Ucf file for the io assignment.
3. <http://www.asic-world.com/tidbits/verilog_fsm.html>