

# Traffic Light Controller (TLC)

Design a simple traffic light controller.

## Tutorial:

1. Write verilog code.

Sample code:

```
module tlc(clk, reset, vehicle_sensor, highway_signal, farm_signal);

    input reset, clk, vehicle_sensor;
    output [1:0] highway_signal, farm_signal;
    parameter HG=2'b00, HY=2'b01, FG=2'b10, FY=2'b11;
    parameter GREEN = 2'b00, YELLOW = 2'b01, RED = 2'b10;
    reg [1:0] highway_signal, farm_signal, cur_state, next_state;
    reg [1:0] highway_signal1, farm_signal1;
    reg reset_timer; reg [3:0] timer; wire long_timeout, short_timeout;

    always @( negedge clk or negedge reset )
        if (reset == 1'b0)
            cur_state <= HG;
        else
            cur_state <= next_state;

    always @( negedge clk or negedge reset )
        if (reset == 1'b0)
            timer <= 4'h0;
        else if ( reset_timer == 1'b1 )
            timer <= 4'h0;
        else if ( timer == 4'hf )
            timer <= 4'b1111;
        else
            timer <= (timer + 1) % 16;

    assign short_timeout = timer[0] ;
    assign long_timeout = ( (timer[1]==1) || (timer[2]==1) ||
(timer[3]==1) ) ? 1'b1 : 1'b0;

    always @(cur_state or long_timeout or vehicle_sensor or
short_timeout)
        begin
            if (cur_state == HG)
                begin
                    highway_signal <= GREEN; farm_signal <= RED;

                    if (vehicle_sensor == 1'b1 && long_timeout == 1'b1 )
                        begin
                            next_state <= HY;reset_timer <= 1'b1;
                        end
                    else
                        begin
```

```


        next_state <= HG; reset_timer <= 1'b0;
    end
end
else if (cur_state == HY)
    begin
        highway_signal <= YELLOW; farm_signal <= RED;
        if ( short_timeout == 1'b1 )
            begin
                next_state <= FG; reset_timer <= 1'b1;
            end
        else
            begin
                next_state <= HY; reset_timer <= 1'b0;
            end
        end
    end
else if (cur_state == FG)
    begin
        highway_signal <= RED; farm_signal <= GREEN;
        if ( long_timeout == 1'b1 )
            begin
                reset_timer <= 1'b1;
                //if ( short_timeout == 1'b1 ) begin
                next_state <= FY; //end
            end
        else
            begin
                next_state <= FG; reset_timer <= 1'b0;
            end
        end
    end
else if (cur_state == FY)
    begin
        highway_signal <= RED; farm_signal <= YELLOW;
        if ( short_timeout == 1'b1 )
            begin
                next_state <= HG; reset_timer <= 1'b1;
            end
        else
            begin
                next_state <= FY; reset_timer <= 1'b0;
            end
        end
    end
else
    begin
        highway_signal <= RED; farm_signal <= RED;
        next_state <= HG; reset_timer <= 1'b1;
    end
end // always
endmodule

```

2. Save this code as seq\_dut.v and save it in a directory.

3. Visit e-prayog webpage: <http://59.181.142.81/>

59.181.142.81

 **e-Prayog** Virtual Labs (Electronics), IIT Bombay  
An Initiative of Ministry of Human Resource Development (MHRD)  
under the National Mission on Education through ICT

Home Devices & Circuits Modern Digital Design Signal Processing Image Processing Microcontrollers People Feedback Workshop Report

**Login for e-Prayog**

Username

Password

Login Free Sign Up

**Downloads**

[Java Runtime](#)

[Labview Runtime](#)

[SEQUEL](#)

[Scilab](#)

[UrJTAG](#)

### Introduction

Laboratories play an important role in engineering education. The laboratory work provides them an opportunity to witness classroom-learned theoretical concepts at work; operate instruments used in the experimental set-up; measure different quantities and thereafter analyze the experimental data; and work collaboratively. However, physical distances and the lack of resources make us unable to perform experiments, especially when they involve sophisticated instruments. Also, trained teachers are always a scarce resource. Rapid development and ever increasing use of Information and Communication Technology (ICT) has revolutionized the education system by eliminating limitations in terms of physical distance, time and access, by providing equal opportunity to everyone, irrespective of place and time.

As a part of e-Prayog, the following electronics lab courses have been developed:

- Electronic Devices and Circuits
- Network Analysis
- Linear Integrated Circuits
- Single and Systems

4. Login on the page if you are registered or sign-up if not registered.

**Login for e-Prayog**

Username

Password

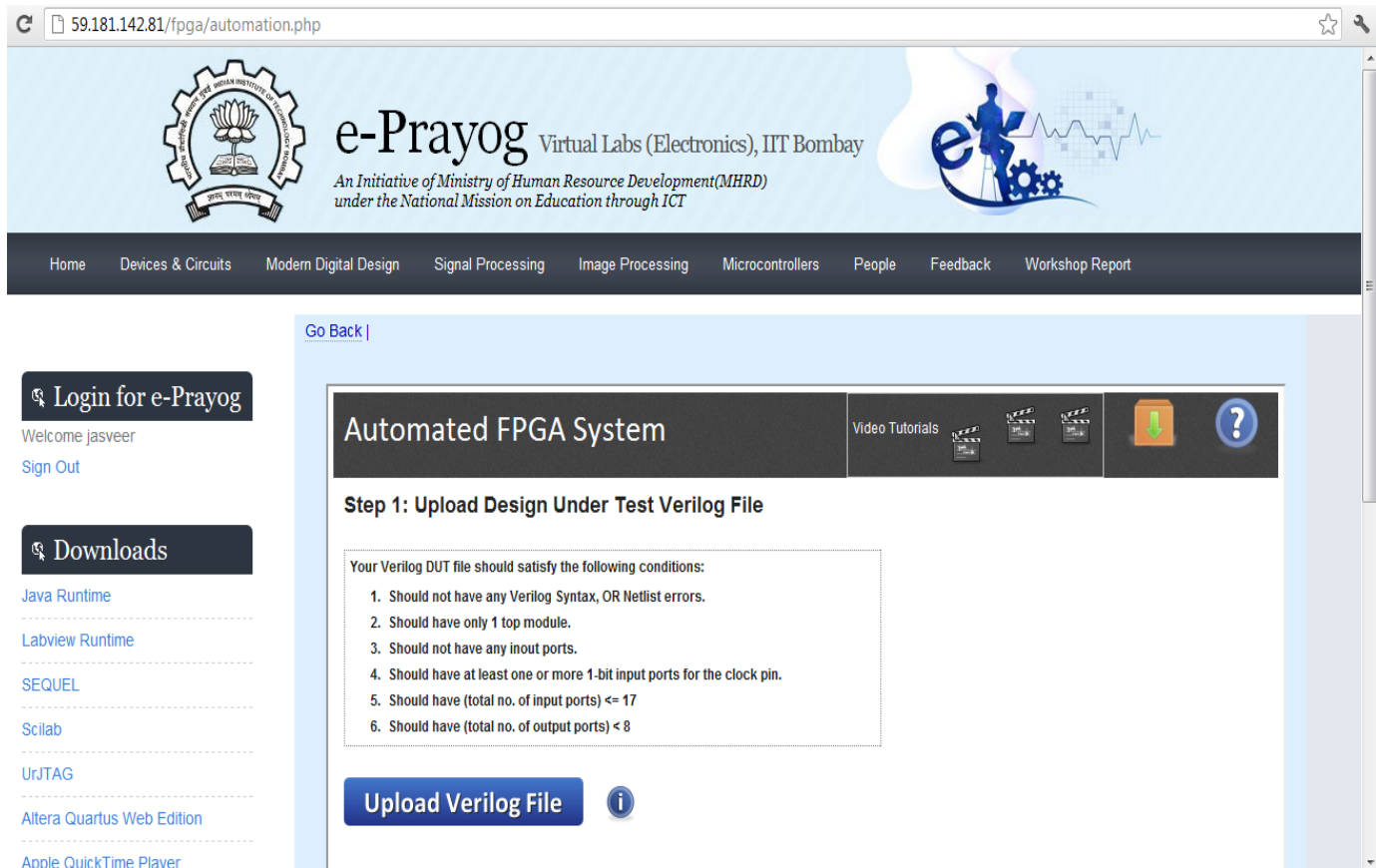
Login Free Sign Up

 **Login for e-Prayog**

Welcome jasveer

[Sign Out](#)



5. Now visit the webpage of Remote Triggered FPGA based Automation system:  
<http://59.181.142.81/fpga/automation.php>



The screenshot shows a web browser window with the address bar displaying [59.181.142.81/fpga/automation.php](http://59.181.142.81/fpga/automation.php). The page header features the IIT Bombay logo and the text "e-Prayog Virtual Labs (Electronics), IIT Bombay". Below the header is a navigation menu with links: Home, Devices & Circuits, Modern Digital Design, Signal Processing, Image Processing, Microcontrollers, People, Feedback, and Workshop Report. The main content area is titled "Automated FPGA System" and includes a "Go Back" link. A sidebar on the left contains a "Login for e-Prayog" button, a "Welcome jasveer" message, a "Sign Out" link, and a "Downloads" section with links to Java Runtime, Labview Runtime, SEQUEL, Scilab, UrJTAG, Altera Quartus Web Edition, and Apple QuickTime Player. The main content area displays "Step 1: Upload Design Under Test Verilog File" and lists conditions for the Verilog DUT file. A blue button labeled "Upload Verilog File" is visible at the bottom of the main content area.

**Login for e-Prayog**  
Welcome jasveer  
[Sign Out](#)


**Downloads**  
[Java Runtime](#)  
[Labview Runtime](#)  
[SEQUEL](#)  
[Scilab](#)  
[UrJTAG](#)  
[Altera Quartus Web Edition](#)  
[Apple QuickTime Player](#)

**Automated FPGA System**  
Video Tutorials  

**Step 1: Upload Design Under Test Verilog File**

Your Verilog DUT file should satisfy the following conditions:



1. Should not have any Verilog Syntax, OR Netlist errors.
2. Should have only 1 top module.
3. Should not have any inout ports.
4. Should have at least one or more 1-bit input ports for the clock pin.
5. Should have (total no. of input ports)  $\leq 17$
6. Should have (total no. of output ports)  $< 8$

**Upload Verilog File** 

6. Click on upload verilog file button and upload your verilog file "tlc.v"
7. After the browser specifies "All checks passed", select the clock pin as "clk".
8. Give the test inputs for clock instances where the input is changing.

Automated FPGA System

Video Tutorials



Step 3: Provide input vectors & Execute Design

Clock Cycle	reset	vehicle_sensor
0	0	0
1	1	
7		1
14		0
25		1
26	0	
6		
7		
8		

Add Row

Delete Row

Set Input Vectors & Execute

Powered by IPintento™

9. Click on “Set input vectors and Execute” - this will compile the design, assign pins according Altera DE2-70 board, synthesize it and execute it on the board with the given test inputs to generate a vcd file output. (this will take about some time from 1-3 mins depending on the design complexity).

10. The log will appear and the output vcd file will be available for download