

MICROCONTROLLER BASED SYSTEM DESIGN

DEPAERTMENT OF COMPUTER SYSTEM ENGINEERING

UNIVERSITY OF ENGINEERING AND TECHNOLOGY PESHAWAR

NAME: Muhammad Ilyas

REG ID; 21PWCSE2055

SECTION: B

**Lab No 9**

**Design of a Light switch**

**Objective:** Build a light switch controller such that when the push button is pressed the light if “off” turns “on” and if “on” turns “off”.

**Block Diagram:**The top level block diagram is shown in the following figure. The second diagram is a more detailed view of the top level diagram. Synchronizer circuit is used to avoid the metastability problem which arises when synchronous digital system are fed with an asynchronous input. The level to pulse converter is there to convert level input from a push button into a pulse that is high for only one clock cycle. The switch button FSM is actually implementing the state machine for the requirement i-e when the push button is pressed the light if “off” turns “on” and if “on” turns “off”.

**I/O connection:** Connect the button input to a push button on the S3BOARD and light output to LED.

Diagram

Description automatically generated

**Synchronizer Circuit:**

Diagram

Description automatically generated

**Level to Pulse FSM:**

Diagram

Description automatically generated

**Switch Button FSM:**

A screenshot of a computer

Description automatically generated with medium confidence

**Task 01:** Create an FSM having two state. LED will be on in one state and off in other state.

//clock\_divider

module clock\_divider(

input clk\_in,

output reg clk\_out

);

reg [27:0] counter = 28'd0;

parameter divisor = 28'd1000000;

always @(posedge clk\_in) begin

counter <= counter + 28'd1;

if (counter >= (divisor - 1))

counter <= 28'd0;

clk\_out <=(counter < divisor/2)?1'b1:1'b0;

end

endmodule

Clock Divider Module

// Synchronizer

module synchronizer(input clk, input btn, input rst, output synch\_btn);

wire Q1;

D\_FF df1(Q1, btn, clk, rst);

D\_FF df2(synch\_btn, Q1, clk, rst);

endmodule

// D\_FlipFlop

module D\_FF(Q, D, clk, rst);

input D; // Data input

input clk; // Clock input

input rst;

output reg Q; // Output Q

always @(posedge clk or negedge clk) begin

if (rst)

Q = 1'b0;

else

Q = D;

end

endmodule

Synchronizer and D-FlipFlops

module fsm(input btn0,

input clk,

input RST\_BTN,

output reg led);

reg [2:0] presentState, nextState;

parameter s0 = 3'b0, s1 = 3'b1;

always @(posedge clk)

presentState <= nextState;

always @(nextState or btn1 or RST\_BTN) begin

if (RST\_BTN)

nextState <= s0;

else

case (presentState)

s0: nextState <= btn0 ? s1 : s0;

s1: nextState <= btn0 ? s0 : s1;

endcase

end

assign led = (presentState == s1);

endmodule

FSM Module

module level\_to\_pulse(

input synch\_input,

input clk,

input rst,

output pulse

);

wire Q;

D\_FF df(Q, synch\_input, clk, rst);

and a(pulse ,~Q, synch\_input);

endmodule

Level to Pulse Converter Module

module btn\_module(

input btn,

input CLK,

input RST,

output pulse

);

wire synch\_btn;

synchronizer s1(CLK, btn, RST, synch\_btn);

level\_to\_pulse lp1(synch\_btn, CLK, RST, pulse);

endmodule

Module taking input from from Push button

module Top( CLK, RST, RST\_BTN, , BTN, LED, SEVENSEG);

input CLK, RST, BTN, RST\_BTN;

output LED;

wire SLOW\_CLOCK;

wire pulse0, pulse1, RST\_Pulse;

wire [3:0]bcd;

clock\_divider divider(CLK, SLOW\_CLOCK);

fsm my\_fsm( pulse1, SLOW\_CLOCK, RST\_Pulse, LED, bcd);

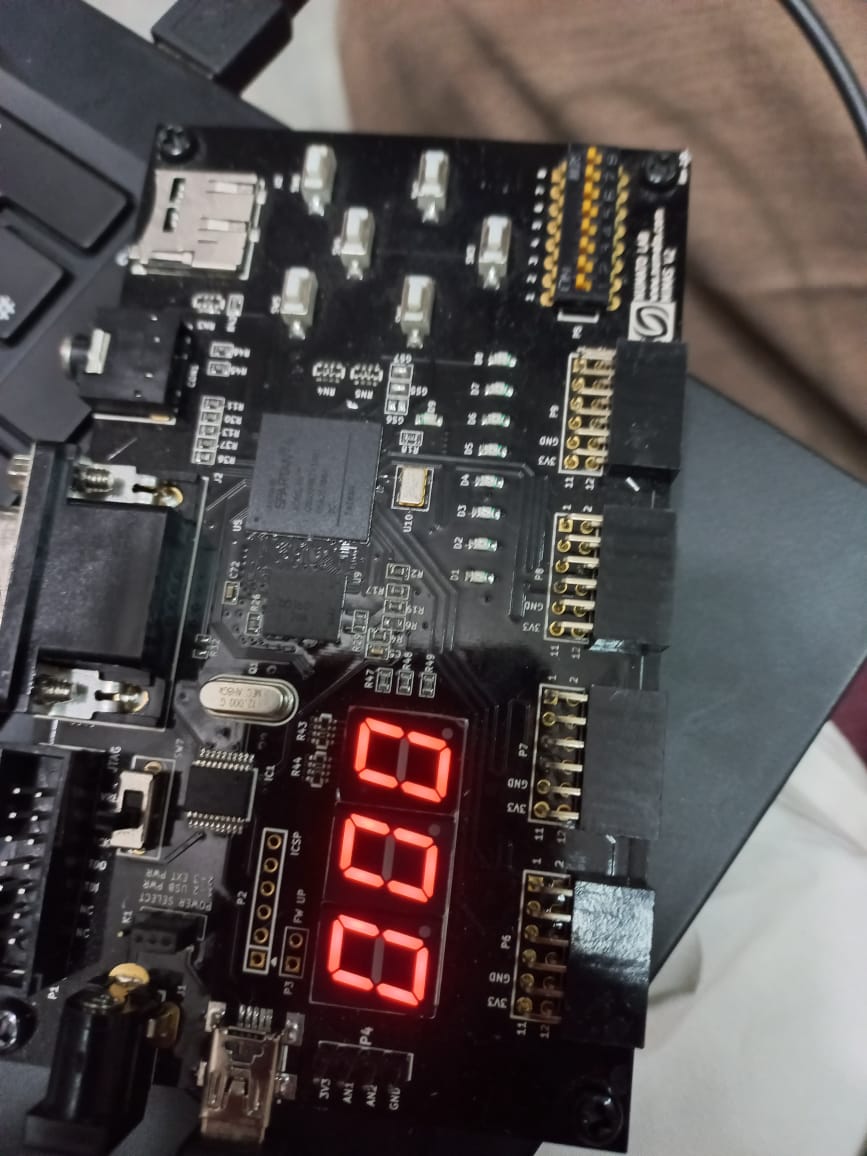
btn\_module b0(BTN, SLOW\_CLOCK, RST, pulse1);

btn\_module b2(RST\_BTN, SLOW\_CLOCK, RST, RST\_Pulse);

endmodule

Top Level Module

----------------------------------------| Output |------------------------------------------



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_