

MICROCONTROLLER BASED SYSTEM DESIGN

DEPAERTMENT OF COMPUTER SYSTEM ENGINEERING

UNIVERSITY OF ENGINEERING AND TECHNOLOGY PESHAWAR

NAME: Muhammad Ilyas

REG ID; 21PWCSE2055

SECTION: B

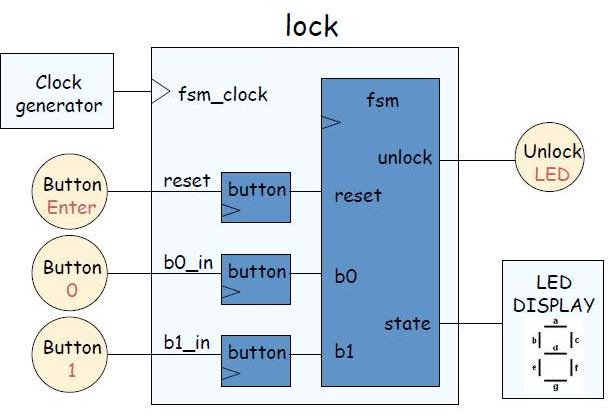
**Lab 10**

**A Digital Lock**

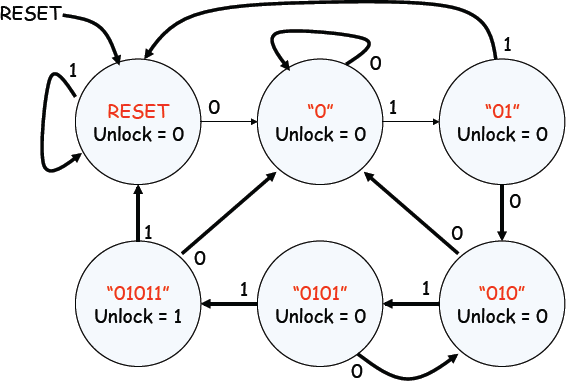
**Objective:** Build an electronic combination lock with reset button, two number buttons (0 and 1), and an unlock output. The combination should be “01011”

**Block Diagram:** A combinational digital lock has three input buttons for Reset, entering a “0” and entering a “1” and output button UNLOCK and state which shows in which state the machine is currently in. The state output is connected to the seven segment display on the S6BOARD. The Module button in the below diagram is an abstraction of the synchronizer and level to pulse converter from the previous lab. The state transition diagram is given in the following figures.





**State Transition Diagram:**

****

**Lab Tasks:**

1. Change the functionality of the lock such that it unlocks on the sequence of 11011.

//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 lock\_fsm(input btn0,

input btn1,

input clk,

input RST\_BTN,

output reg led,

output reg [3:0]bcd);

reg [2:0]presentState, nextState;

parameter s0 = 3'b000,s1 = 3'b001, s2 = 3'b010,s3 = 3'b011,s4 = 3'b100,s5 = 3'b101;

always@(posedge clk)

presentState <= nextState;

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

if(RST\_BTN)

nextState <= s0;

case(state)

s0: nextState <= btn1? s1:s0;

s1: nextState <= btn1? s2:s1;

s2: nextState <= btn1? s0:s3;

s3: nextState <= btn1? s4:s1;

s4: nextState <= btn1? s5:s3;

s5: nextState <= btn1? s0:s1;

endcase

end

assign led = (presentState == s5);

assign bcd = {1'b0, presentState};

endmodule

Lock 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 BCD\_to\_SevenSeg(

input [3:0] bcd,

output reg [7:0] O

);

always @(\*)

case(bcd) //gfedcba

4'b0000: O = 8'b11000000;

4'b0001: O = 8'b11111001;

4'b0010: O = 8'b10100100;

4'b0011: O = 8'b10110000;

4'b0100: O = 8'b10011001;

4'b0101: O = 8'b10010010;

4'b0110: O = 8'b10000010;

4'b0111: O = 8'b11111000;

4'b1000: O = 8'b10000000;

4'b1001: O = 8'b10010000;

default: O = 8'b11111111;

endcase

endmodule

BCD to 7-Segment Display Module

module Top( CLK, RST, RST\_BTN, BTN0, BTN1, LED, SEVENSEG);

input CLK, RST, BTN0, BTN1, RST\_BTN;

output LED;

output [7:0]SEVENSEG;

wire SLOW\_CLOCK;

wire pulse0, pulse1, RST\_Pulse;

wire [3:0]bcd;

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

lock\_fsm2 my\_fsm( pulse0, pulse1, SLOW\_CLOCK, RST\_Pulse, LED, bcd);

btn\_module b0(BTN0, SLOW\_CLOCK, RST, pulse0);

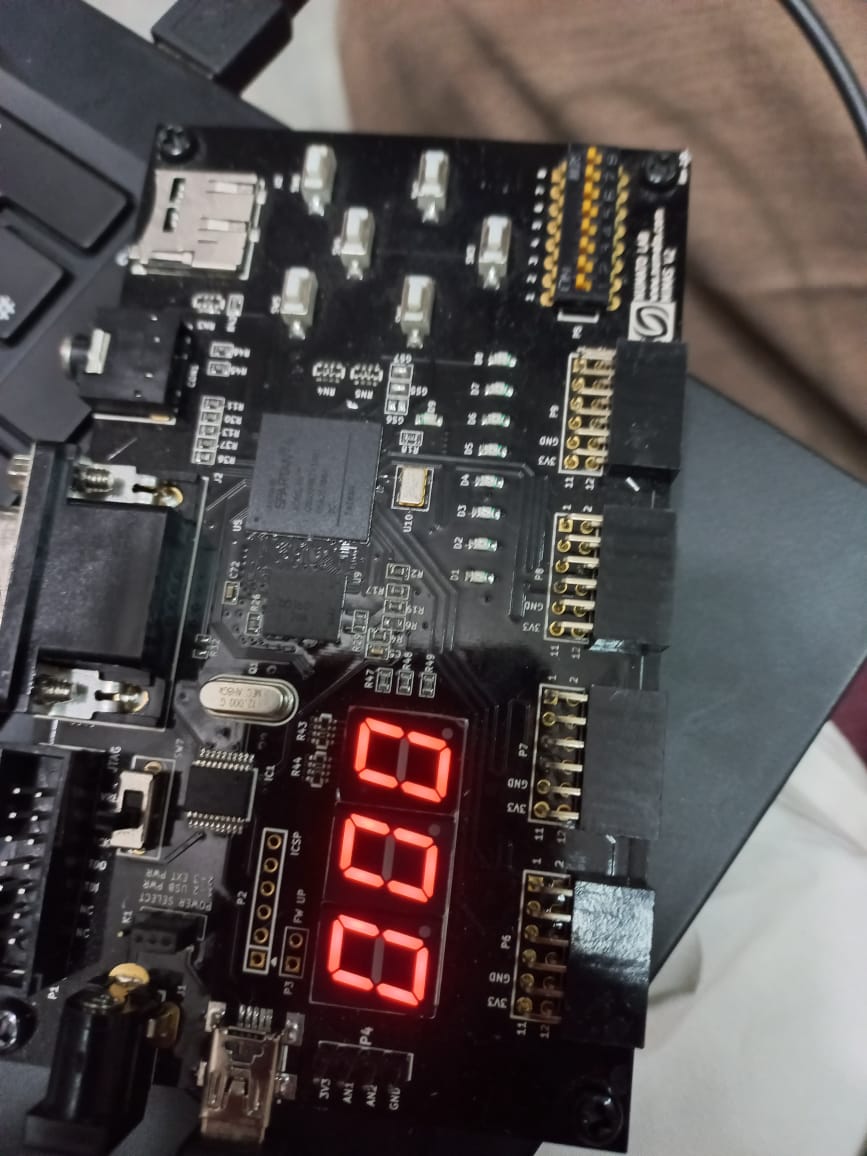
btn\_module b1(BTN1, SLOW\_CLOCK, RST, pulse1);

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

endmodule

Top Level Module

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





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