DIGITAL SYSTEM DESIGN LAB

LAB #07

Spring 2021

CSE308L DSD LAB

Submitted by: **Shah Raza**

Registration No. : **18PWCSE1658**

Class Section: **B**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Student Signature: _______________

Submitted to:

Engr. Madiha Sher

Friday, June 25, 2021

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Objectives:

This lab will enable students to:

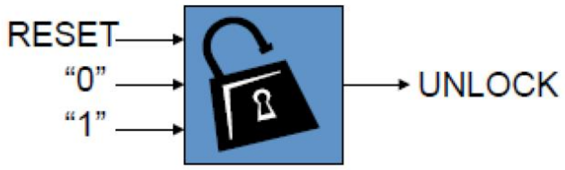
- Code using Behavioral level modeling
- Implement FSM of Sequence Detector

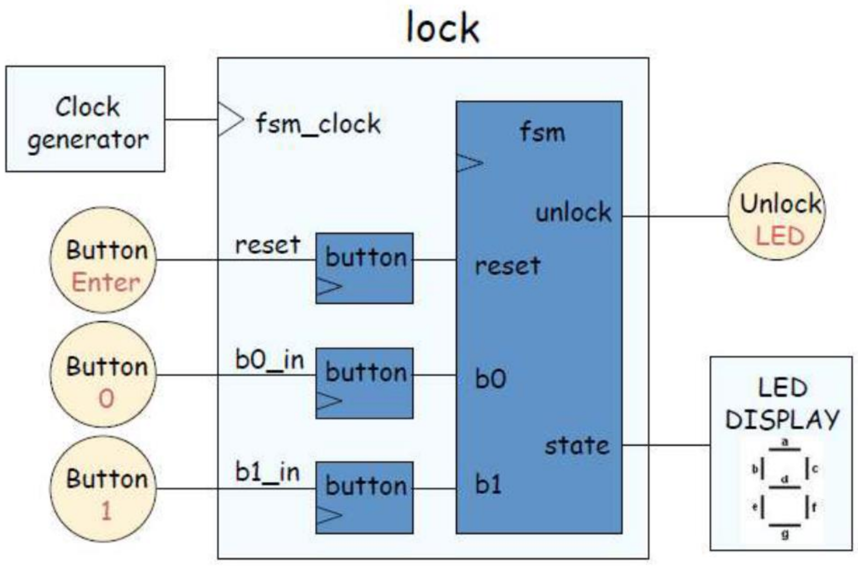
Task # 01:

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

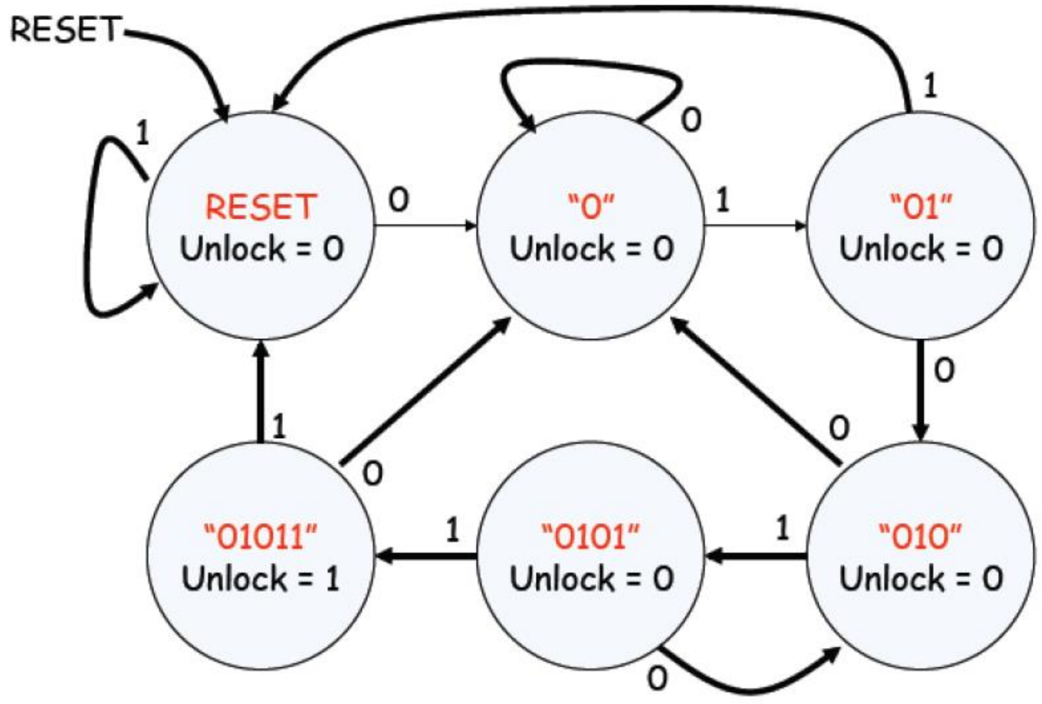
Problem Analysis:

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.





State Diagram:



Code:

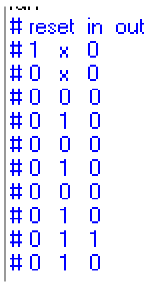
Sequence Detector:

```
module Sequence_detector(in, out, clk, rst);
        input rst,clk,in;
        output out;
        reg out;

        reg [2:0] state, next_state;

        parameter [2:0] A = 3'b000, B = 3'b001, C =3'b010, D = 3'b011, E = 3'b100, F  = 3'b101;

        always @(posedge clk)
                state = next_state;

        always @(state or rst or in)
        begin
                if(rst)
                        next_state = 3'b000;
                else
                begin
                        if(in)
                case(state)
                    A: next_state = A;
                    B: next_state = C;
                    C: next_state = A;
                    D: next_state = E;
                    E: next_state = F;
                    F: next_state = A;
                endcase
            else
                case(state)
                    A: next_state = B;
                    B: next_state = B;
                    C: next_state = D;
                    D: next_state = B;
                    E: next_state = D;
                    F: next_state = B;
                endcase
        end

        if(next_state==F)
            out = 1'b1;
        else
            out = 1'b0;
    end
endmodule
```

TestBench:

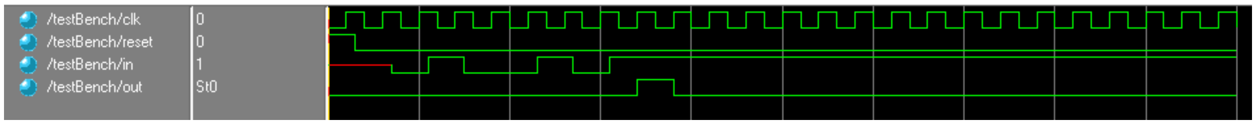
```
module testBench;
    reg clk, reset,in;
    wire out;

    Sequence_detector s1(in, out,clk,reset);

    always
        #2 clk = ~clk;

    initial
    begin
        clk =0;
        reset = 1;
        $display("reset  in  out");
        $monitor("%b   %b   %b",reset,in,out);
        #3
        reset = 0;
        #4
        in = 0;
        #4
        in = 1;
        #4
        in = 0;
        #4
        in = 0;


        #4
        in = 1;
        #4
        in = 0;
        #4
        in = 1;
        #4
        in = 1;
        #200 $stop;
    end
endmodule
```

Output:



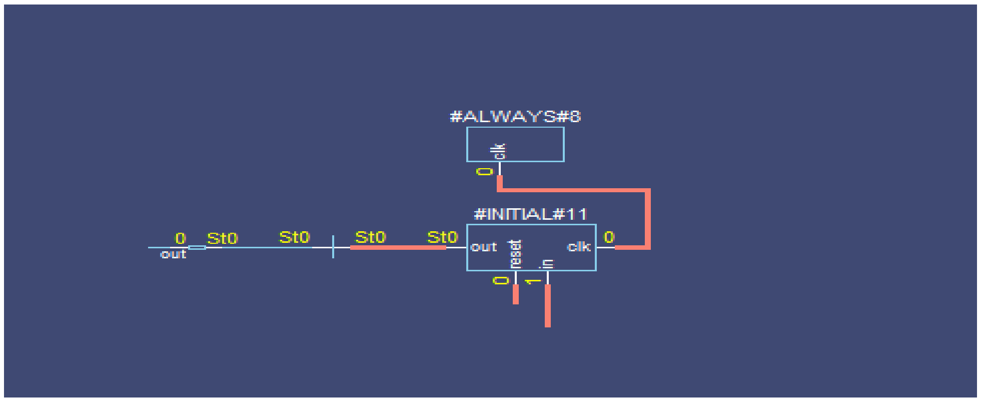
```
# reset  in  out
# 1   x   0
# 0   x   0
# 0   0   0
# 0   1   0
# 0   0   0
# 0   1   0
# 0   0   0
# 0   1   0
# 0   1   1
# 0   1   0
```

Waveform:



```
/testBench/clk    0
/testBench/reset  0
/testBench/in     1
/testBench/out    St0
```

Dataflow:


#ALWAYS#8
clk
0
#INITIAL#11
out
reset
in
clk
0
0
1
0
out
St0
St0
St0
St0