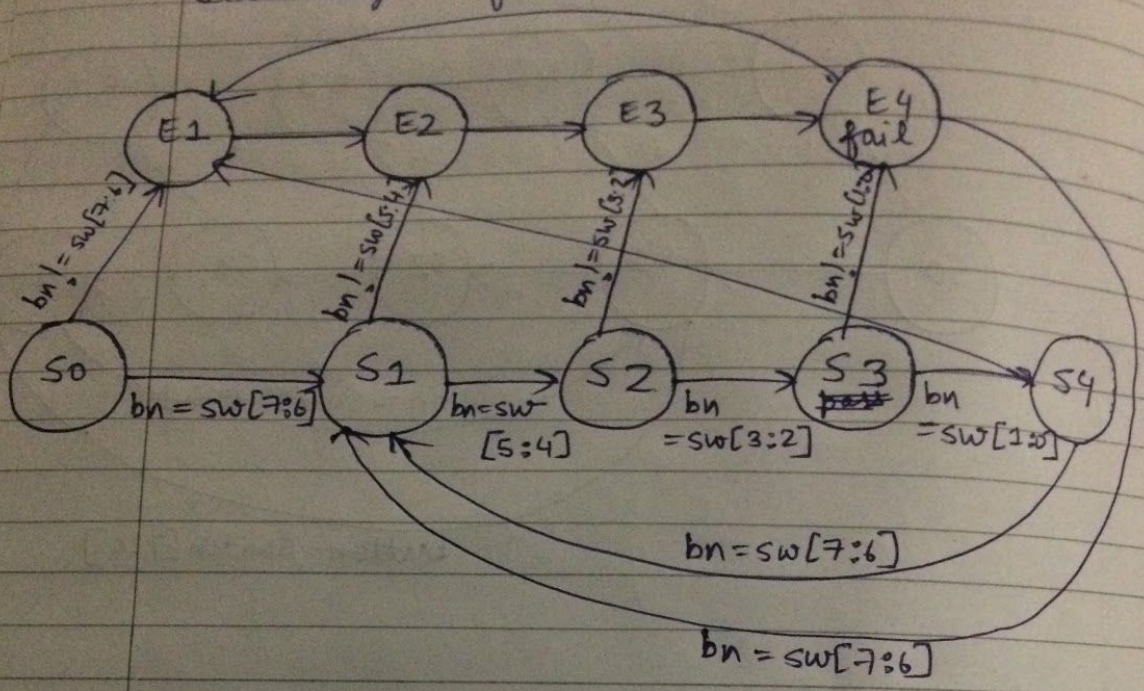
Lab 7
Submitted By :
Snehal Gupta
2016201

DATE:.....

EVERGREEN

State Diagram for Door Lock Code :-



S0 = 4'b0000
S1 = 4'b0001
S2 = 4'b0010
S3 = 4'b0011
S4 = 4'b0100
E1 = 4'b0101
E2 = 4'b0110
E3 = 4'b0111
E4 = 4'b1000

Explanation :
Initially, present state is S0 . If the value corresponding to sw[7:6] is pressed through push buttons , present state becomes S1 . Else, present state moves to E1. Once the present state has switched to E1 , next state switches to E2 irrespective of the input because the first digit entered is wrong . When the present state is at S1, if the value corresponding to sw[5:4] is pressed through push buttons , present state becomes S2.Else, present state moves to E2.Once the present state has switched to E2 , next state switches to E3 irrespective of the input because the second digit entered is wrong . When the present state is at S2, if the value corresponding to sw[3:2] is pressed through push buttons , present state becomes S3.Else, present state moves to E3.Once the present state has switched to E3 , next state switches to E4 irrespective of the input because the third digit entered is wrong.When the present state is at S3, if the value corresponding to sw[1:0] is pressed through push buttons , present state becomes S4.Else, present state moves to E4.When the machine is at state S4 , it implies that the door lock code entered is correct and now next state depends on whether the first digit of code entered is correct or not. If it matches sw[7:6], it switches to S1 else it switches to E1.When the machine is at state E4 , it implies that the door lock code entered is incorrect and now next state depends on whether the first digit of code entered is correct or not. If it matches sw[7:6], it switches to S1 else it switches to E1.

//top.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 04.10.2017 17:15:19
// Design Name:
// Module Name: top
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
```

```
module top(
    input clk,
    input clr,
    input [2:0] btn,
    input [7:0] num,
    output [3:0] states,
    output [1:0] LED
    );
    wire inp;
    wire [1:0] button;
    assign button={btn[2],btn[1]};
    assign inp= btn[0] || btn[1] || btn[2];
    clk_div CD1(.mclk(clk),.clk190(clk_190));
    debounce D1(.clk_in(clk_190),.clr_in(clr),.clr_out(clr_de));
    clock_pulse CP(.inp(inp),.cclk(clk_190),.clr(clr_de),.outp(out_pulse));
    moore M1(.clk(out_pulse),.clr(clr_de),.switch(num),.din(button),.dout(LED),.state(states));
endmodule
```

//clk_div.v

```
`timescale 1ns / 1ps
module clk_div(
input wire mclk,
output wire clk190);
reg [18:0] q;
always @(posedge mclk)
begin
q <= q+1;
end
assign clk190 = q[18];
endmodule
```

//debounce.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 04.10.2017 16:46:57
// Design Name:
// Module Name: debounce
// Project Name:
// Target Devices:
```

```
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module debounce(
    input clk_in,
    input clr_in,
    output clr_out
    );
    reg D1,D2,D3;
    always@(posedge clk_in)
    begin
    D1 <= clr_in;
    D2 <= D1;
    D3 <= D2;
    end
    assign clr_out = D1 && D2 && D3;
endmodule
```

//clock_pulse.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 26.09.2017 18:04:34
// Design Name:
// Module Name: clock_pulse
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
```

```
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module clock_pulse(
input wire inp,
input wire cclk,
input wire clr,
output wire outp);
reg delay1;
reg delay2;
reg delay3;
always@(posedge cclk or posedge clr)
begin
if(clr == 1)
begin
delay1 <= 1'b0;
delay2 <= 1'b0;
delay3 <= 1'b0;
end
else
begin
delay1 <= inp;
delay2 <= delay1;
delay3 <= delay2;
end
end
assign outp = delay1 & delay2 & delay3;
endmodule
```

//moore.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 04.10.2017 16:54:21
// Design Name:
// Module Name: moore
// Project Name:
// Target Devices:
```

```
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module moore(
    input clk,
    input clr,
    input [7:0] switch,
    input [1:0] din,
    output[3:0] state,
    output reg[1:0] dout
    );
    reg[3:0] present_state,next_state;
    parameter S0=4'b0000 , S1=4'b0001 , S2=4'b0010 , S3=4'b0011 , S4=4'b0100 , E1=4'b0101
,E2=4'b0110,E3=4'b0111,E4=4'b1000;
    wire[1:0] button;
    assign button={din[1],din[0]};
    always@(posedge clk or posedge clr)
    begin
    if(clr == 1)
    present_state <= S0;
    else
    present_state <= next_state;
    end
    always@(*)
    begin
    case(present_state)
    S0: if(button==switch[7:6])
            next_state <= S1;
        else
            next_state <= E1;
    S1: if(button==switch[5:4])
            next_state <= S2;
        else
            next_state <= E2;
```

```
    S2: if(button==switch[3:2])
            next_state <= S3;
        else
            next_state <= E3;
    S3: if(button==switch[1:0])
            next_state <= S4;
        else
            next_state <= E4;
    S4: if(button==switch[7:6])
            next_state <= S1;
        else
            next_state <= E1;
    E1: next_state <= E2;
    E2: next_state <= E3;
    E3: next_state <= E4;
    E4: begin
    if(button == switch[7:6])
        next_state<= S1;
    else
        next_state <= E1;
    end
    endcase
    end
    always @(*)
    begin
    if(present_state==S4)
        dout=2'b10;
    else if(present_state==E4)
        dout=2'b01;
    else
        dout=2'b00;
    end
    assign state=present_state;


endmodule
```

//const.xdc

```
set_property PACKAGE_PIN W5 [get_ports {clk}]
set_property IOSTANDARD LVCMOS33 [get_ports {clk}]
set_property PACKAGE_PIN T18 [get_ports {clr}]
set_property IOSTANDARD LVCMOS33 [get_ports {clr}]
set_property PACKAGE_PIN U18 [get_ports {btn[1]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {btn[1]}]
set_property PACKAGE_PIN W19 [get_ports {btn[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn[0]}]
set_property PACKAGE_PIN T17 [get_ports {btn[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn[2]}]
set_property PACKAGE_PIN L1 [get_ports {LED[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED[1]}]
set_property PACKAGE_PIN P1 [get_ports {LED[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED[0]}]
set_property PACKAGE_PIN N3 [get_ports {states[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {states[3]}]
set_property PACKAGE_PIN P3 [get_ports {states[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {states[2]}]
set_property PACKAGE_PIN U3 [get_ports {states[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {states[1]}]
set_property PACKAGE_PIN W3 [get_ports {states[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {states[0]}]
set_property PACKAGE_PIN V17 [get_ports {num[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[0]}]
set_property PACKAGE_PIN V16 [get_ports {num[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[1]}]
set_property PACKAGE_PIN W16 [get_ports {num[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[2]}]
set_property PACKAGE_PIN W17 [get_ports {num[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[3]}]
set_property PACKAGE_PIN W15 [get_ports {num[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[4]}]
set_property PACKAGE_PIN V15 [get_ports {num[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[5]}]
set_property PACKAGE_PIN W14 [get_ports {num[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[6]}]
set_property PACKAGE_PIN W13 [get_ports {num[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {num[7]}]
```